OpTaliX®

Software for Design, Optimization and Analysis of Optical Systems, Thin Films and Illumination Applications

Reference Manual

Version 10.50

Optenso™
Optical Engineering Software

info@optenso.com
http://www.optenso.com/

August 2020

Copyright

Trademarks

Optϵnso® is a registered trademark of Optenso, Optical Engineering Software.

OpTaliX® is a registered trademark of Optϵnso®.

Microsoft Windows, Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP, Windows Vista, Windows 7, Windows 8 and Windows 10 are trademarks of Microsoft Corporation.

OSLO is a registered trademark of Lambda Research Corp.

ZEMAX is a registered trademark of Zemax, LLC.

CODE V is a registered trademark of Synopsis, Inc.

ProSource is a registered trademark of Radiant Vision Systems

POV-Ray^{TM} is Copyright by the POV-Ray Team^{TM}

Disclaimer

Optϵnso® reserves the right to modify its software and publications with no obligation to notify any person or organization of such modifications. The software is provided as is without any guarantees or warranty. Although Optϵnso® has attempted to find and correct any bugs in the package, in no event shall Optϵnso® be liable for any loss of profit or any commercial damage, included but not limited to special, consequential or other damages.

Optenso, Optical Engineering Software
Herbstweg 9
86859 Igling
Germany
E-mail : info@optenso.com
Web : http://www.optenso.com/

Contents

1

Starting and Exiting OpTaliX

OpTaliX can only be started from within Microsoft Windows. Within Windows, *OpTaliX* can be run by clicking on the *OpTaliX* menu item in the Program Group, double clicking on the *OpTaliX* desktop shortcut icon, double clicking on a lens file in Windows Explorer, or it can be run from a DOS prompt within a DOS window.

1.1 Starting OpTaliX from the Program Group

To start *OpTaliX* in Windows 98/Me/NT/2000/XP, click the **Start** button, click **Programs**, click the *OpTaliX* program group, and then click the *OpTaliX* menu item, as shown in Figure 1.1.

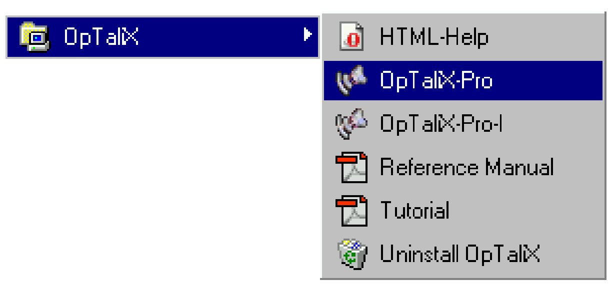


Figure 1.1: *OpTaliX* program group menu.

The *OpTaliX* program group also includes menu items for HTML-Help, Reference Manual, Tutorial and uninstalling *OpTaliX* . Note that two menu items for *OpTaliX* are found: **OpTaliX-Pro** and **OpTaliX-Pro-I**. Both versions, OpTaliX-Pro and OpTaliX-Pro-I, are functionally identical, except for the style of the windows.

1.2 Starting OpTaliX from Windows Explorer

The *OpTaliX* file format has been registered in Windows during program installation. This allows you to launch *OpTaliX* with a specific lens, by double clicking on the file (extension .otx) in Windows Explorer.

1.3 Starting OpTaliX from a DOS Window

Open a DOS Window by clicking on the MS-DOS prompt menu item in the Program Group accessed by using **Start** −> **Programs**. From the DOS prompt start *OpTaliX* by typing

```
C:> c:\programs\optalix\optalixp-i mylens.otx
```

If *OpTaliX* was installed in a different directory than `c:\programs\optalix`, the path to the *OpTaliX* executable must be modified accordingly. Specification of an *OpTaliX* lens file (mylens.otx) is optional. If omitted, *OpTaliX* starts with the recently used lens (i.e. the optical design which was in use during the last session). If specified, *OpTaliX* is launched and ”mylens.otx” is automatically loaded.

1.4 Normal Exit from OpTaliX

- From the File menu, select Exit or click on the close window button  in the upper right corner of the *OpTaliX* main window.
- Select the main window (click on the title bar of the main window) and press the ESC-key.
- In the command line, type `EXI` or `QUIT` and press Return.

In all cases, you will be asked to confirm the exit. After you exit *OpTaliX* , you are returned to the operating system.

1.5 Forced Exit from OpTaliX

Normally an exit request invokes a dialog box asking to confirm exit. Immediate exit by bypassing the confirmation dialog box is accomplished from the command line or from a macro by

`EXI Y`

or

`EXI Yes`

The program is then terminated immediately.

2

Notational Conventions

The following conventions are used throughout this manual:

- In syntax descriptions, `[brackets]` enclose optional items.
- In syntax descriptions, the vertical line | separates optional parameters within an option list.
- The apostrophe ' character encloses character strings which contain blanks. If there is no blank character contained in a string, the apostrophe may be omitted.
- *OpTiX* commands are emphasized by `courier` typeset.
- *ITALICS* refer to menue items of the GUI (graphical user interface)
- An ellipsis, "`...`", following an item indicates that more items of the same form may appear.
- The question mark "`?`" character, used within a command, activates additional dialog box information and/or settings.
- The semicolon "`;`" character separates command entries in the command line, i.e. it allows several command strings in a single line. A detailed description is given in the Macro section.
- The vertical bar "|" is not typed in any command, it means 'or' as in `Yes`|`No`, that is, you type `Yes` **or** `No`.
- The Dollar sign "`$`" followed by a character denotes a short form of a directory path or part of it. These directories are created during installation.

  `$i` is the installation path, i.e. $i may direct to `c:\optalix` or `c:\programs\optalix`
  `$t` is a temporary directory, e.g. `c:\optalix\temp`
  `$c` refers to the directory where coating files are stored, e.g. `c:\optalix\coatings`
  `$g` refers to the directory where glasses are stored, e.g. `c:\optalix\glasses`
- The asterix "`*`" performs wildcard pattern matching in a given string.

3

Program Preferences

Preferences are data associated with the program, not the lens. Change these settings only, if you know what you are doing. In particular, the directories must exist. Changes take effect immediately and it is not required to restart the program.

Preference settings are accessed from the main menu under *File* $-->$ *Preferences*, or in the command line by entering ”`EDI PREF`” (without the quotes). The settings are grouped into several categories, such as defining paths, behavior of the program (operations), windows, colours and other miscellaneous parameters.

3.1 Paths

The path information entered in the preferences section is used as a reference where files are searched first. Fig. 3.1 shows the corresponding dialog box.

POV Render Engine:
$OpTaliX$ provides an interface to the POV-Ray (Persistence of Vision) renderer, which is used to create almost photorealistic images of the optical system. POV-Ray is a separate program, which must be downloaded from http://www.povray.org and must also be separately installed. Once installed, the path where the executable of POV-ray resides must be entered into the path field. Use the ”browse” button in the preferences dialog to select the path.

Glass Catalogues:
This field has been already defined during the installation of $OpTaliX$. It is normally not needed to change this setting, however, should you wish to change the path, make sure that the new directory and the corresponding glass files in that directory exist.

Coatings:
This field has been already defined during the installation of $OpTaliX$. It contains all thin-film coating files.

Temp Dir:
Defines the path to a working directory used by $OpTaliX$ for storage of intermediate data and other purposes. All files in this directory are normally used during runtime of the program only, however, these files are not deleted after program termination.

Macros:
Defines the path to the directory containing the macro files. The default extension is *.mac. If empty, the macros will be stored and loaded by default from the currently active directory (i.e. the directory of the current system).

User defined graphics:

Defines the path to the directory containing the files for *user defined graphics* (UGR). The default extension is *.ugr. If empty, *user defined graphics* (UGR) will be stored and loaded by default from the currently active directory (i.e. the directory of the current system).

3.2 Operations

The settings in the ”operations” tab determine the behaviour of the program (Fig.3.2).

Save current design as default on exit:
When the program is terminated, the current system is automatically stored as the ”default” system. It is restored into memory at the next program start. This preserves design data between subsequent sessions.

Put text output window to foreground ... :
Each time new output is written to the text window it will be raised to the foreground if this option is checked. This is particularly useful if many windows are opened and are obscuring the text window and the output contained in it.

Warn if glasses are obsolete:
Issues a warning message when obsolete glasses are entered. These are glasses, which are no longer produced by a designated glass manufacturer.

Align ray fans horizontally:
Normally transverse ray aberration fans and OPD fans are plotted with the pupil coordinate vertical. It is also possible to plot the pupil coordinate horizontal by checking the appropriate box. Selecting this option is merely a matter of personal preference rather than providing more detailed information.

Refer fan aberrations to the physical coordinates of the stop surface:
When plotting ray aberration fans and OPD fans, the pupil coordinates are referred to the entrance pupil by default, that is where the rays intercept at the (fictitious) entrance pupil. Check this box if you want the plot coordinates to be referred to the physical ray intercept coordinates on the stop surface.

Adjust surface apertures automatically:
It is sometimes required to adjust surface apertures, for example when system parameters (fields, system aperture) have changed or when the optical layout has changed after optimization. Apertures can be set manually on all surfaces as required by the beams going through the optical system using the `SET MHT` command. This task can be performed automatically such that surface apertures are always large enough. The oversize factor determines how much larger the apertures are set. For example, a factor 1.05 will oversize the apertures by 5% in relation to the required apertures.

Blank command lines are mirrored in Text Output Window:
If this check box is enabled, entering a blank (empty) line in one of the two command lines produces a blank line in the text output window. This way, the user input in a command line is mirrored in the text output window, which allows adding extra blank (empty) in the text output window. This option has no effect on the command history window. The default setting of this option is disabled, i.e. blank command lines have no effect on text output.

Selected surfaces in surface editor are highlighted in lens layout plot:

Check this box to highlight surfaces in the lens layout plot according to the focus in the surface editor. That is, clicking into any row (=surface) in the surface editor will show the corresponding surface in the layout plot in a different colour (typically blue). This feature helps identifying surfaces in the surface editor.

3.3 Windows

Save position and size of windows on exit:
As windows can be interactively changed in size, position and can be minimized or maximized, checking this button saves the current settings of all windows if the program is terminated. The window settings will be restored at the next run of the program.

Put text window to foreground when new output is generated:
Optical analyses may generate additional numerical output respectively informational or warning messages in the text window. If this check box is enabled, the text window will be put to foreground to immediately alert the user about a conflicting situation or simply to have additional information readily visible (i.e. in the foreground without needing to click on a particular window).

Close all open windows on restoring a new optical system:
Prior to restoring a new optical system all currently open windows are automatically closed.

3.4 Colours

Graphics window background colour:
This is an option which suits the personal taste of an user. Setting the background colour of **all** graphics windows to a different colour that the default (white) may help to reduce contrast or to make faint colours (like yellow) more visible.

3.5 Miscellaneous

Spot marker size:
Adjusts the size of markers used in spot diagrams. Marker size is defined in plot units (in mm) referred to the size of a standard A4 paper. See also the SPMS command for temporarily changing spot marker size within a session.

Contour Style
Chose between two styles how contour plots are rendered: "*lines only*" or "*lines + area fill*". Since we consider this option a matter of personal preference, it is found in the general preferences rather than adjustable for each plot individually.

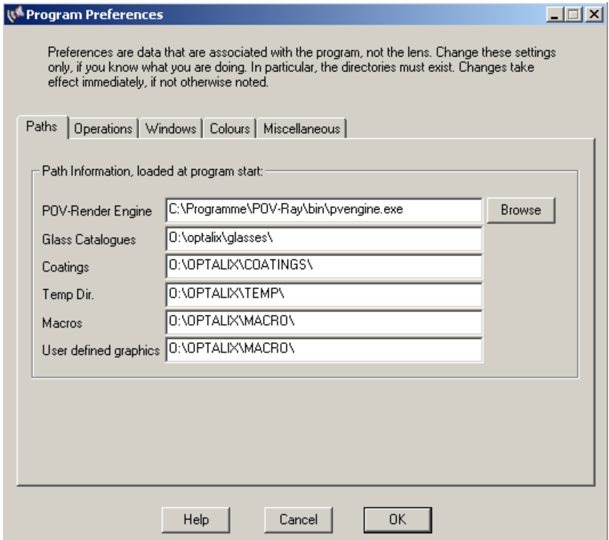


Figure 3.1: Preferences: Program default path settings.

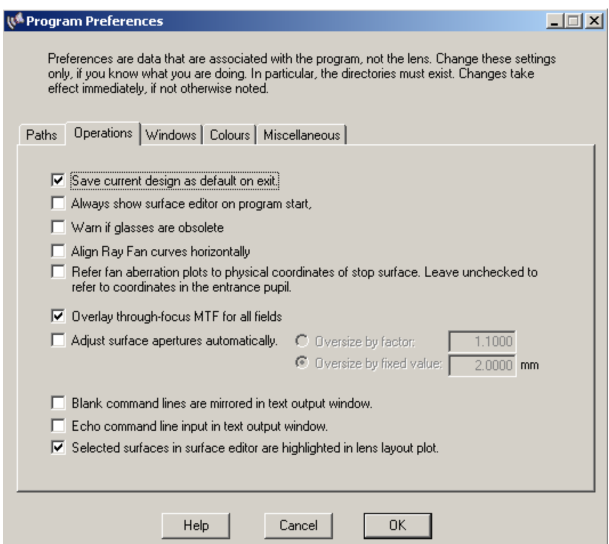


Figure 3.2: Preferences: Operations, determining the behaviour of the program.

4

File Locations

During operation $OpTaliX$ creates intermediate files which are stored in the following directories, specific to each operating system:

4.1 Windows XP

User specific data are stored under Windows XP at:
`c:\Documents and Settings\All Users\Application Data\OpTaliX`.

4.2 Windows Vista / Windows 7 / Windows 10

User specific data are stored under Windows Vista^{TM}, Windows 7^{TM} and Windows 10^{TM} at:
`c:\ProgramData\OpTaliX`

4.3 Description of user-specific Files

In each of the user directories, depending on the operating system, a basic installation of $OpTaliX$ contains the following files:

`default`	Without extension, this files contains the prescription data of the optical system in use after $OpTaliX$ was terminated. Upon restarting $OpTaliX$, this system is automatically reloaded. The file format is ASCII.
`optix.cfg`	$OpTaliX$ configuration file (ASCII format). Stores user-defined preferences as described in sect. 32.1.
`coatp.asc`	Standard ASCII file storing private (user-defined) coating materials. A detailed description of the coating file format is given in sect. 32.3.
`osp_priv.dat`	ASCII file storing private (user-defined) optical spectra (i.e. spectral weights for calculating image performance).

5

Definitions

5.1 Sign Conventions

Conventions are important because they define the frame of reference used for the results. These conventions are applied uniformly throughout the $OpTaliX$ package. It is also important to adhere to strict sign conventions for curvatures and thicknesses (separations), which are determined according to the following rules:

- The radius of curvature of a surface is positive if the center of curvature lies to the right of the surface, otherwise it is negative. This rule is independent on the direction of the light, i.e. if the light travels from left to right (the default condition) or if it travels from right to left (after reflection from a mirror).
- The thickness (separation) of two consecutive surfaces is positive if (in axial direction) the next surface lies to the right of the current surface. If it lies to the left, it is negative.
- In case of tilted and decentered surfaces, the sign conventions apply to the local coordinate system of the current surface.
- A positive tilt means a rotation in counter-clockwise direction, a negative tilt is in clockwise direction.

5.2 Coordinate System(s)

The coordinate system used in $OpTaliX$ is a left-handed system, with the Z-axis being the optical axis in most cases as shown in Fig. 5.1. The vertex of each surface is assumed to lie exactly on the Z-axis. The separation from one surface to the next is along the Z-axis.

5.2.1 Global Coordinate System

The global coordinate system is always located at the vertex of surface 1. Decenter/tilts applied to surface 1 do not change the global coordinate system. Fig. 5.2 illustrates this condition.

5.2.2 Object Coordinate System

The object coordinate system is a derived coordinate system of the global coordinate system. Object points ("fields"), for example, are always referred to the coordinate system defined by the object

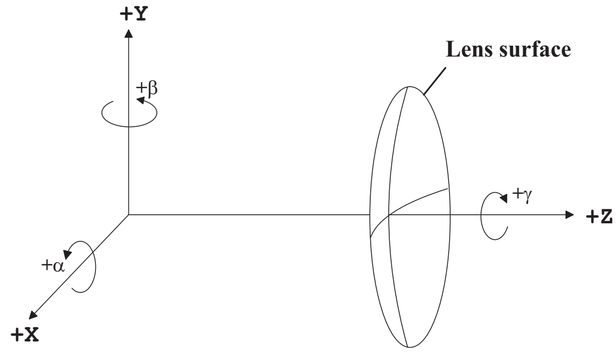


Figure 5.1: Left-handed coordinate system used in $OpTaliX$

surface. In this way, the position and orientation of objects can be altered by changing position and orientation of the object surface (use XDE, YDE, ZDE, ADE, BDE, ZDE commands applied to surface 0).

Using the object coordinate system may also be useful in defining extended sources (as opposed to point-like sources) in illumination calculations.

Note that the object coordinate system may be considered like the *local* coordinate system of any arbitrary surface. It is explained here to emphasize the its meaning for defining illumination sources.

5.2.3 Tilt Angles

The tilt angles in a tilted coordinate system are always given in degree. The sign of the tilt angles follows mathematical convention, i.e. it is positive for counter-clockwise rotation and negative for clockwise rotation. An Euler angle system is used in which each of the three tilt angles α, β, γ takes place in the tilted coordinate system of the preceding tilt. Thus, tilting is non-commutative and undoing tilts must be applied in the reverse order.

Tilts and decenters are always applied to the local coordinate system of a surface.

5.3 Paraxial Conventions

The term paraxial means ”near the axis”. In this region, the linearized version of Snells’ law is used:

$$n' \cdot u' = n \cdot u \tag{5.1}$$

with n = index of refraction and u = angle to the optical axis in radians. The computation of the paraxial entities (e.g. focal length, magnification, etc.) is performed using the ABCD matrix, which is defined as (see also Fig. 5.5):

$$\begin{pmatrix} n'u' \\ h' \end{pmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \cdot \begin{pmatrix} nu \\ h \end{pmatrix} \tag{5.2}$$

There are a few optical components (e.g. gradient index lenses, generalized aspheres) which are not well described by first order theory respectively very complex equations would result. In these cases, $OpTaliX$ uses ”parabasal” rays. These are real rays with very small angles to the optical axis (or the reference ray). The definition of the paraxial entities is:

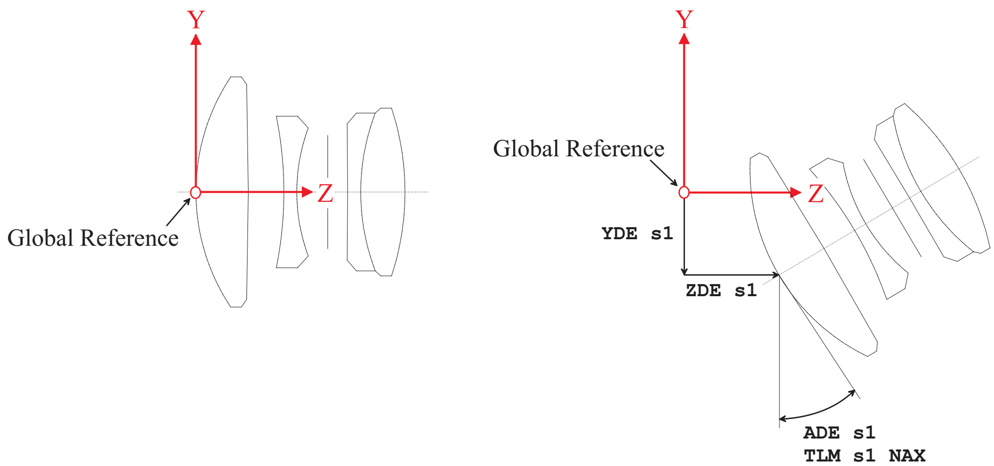


Figure 5.2: The global coordinate system is always referred to the vertex of surface 1. If decenter and/or tilts are applied to surface 1, they are ignored (see right part of this figure).

5.4 Ray Coordinates

Rays are described by unit vectors with a starting point (X,Y,Z) and direction coordinates (CX,CY,CZ). The incidence angle i is always referred to the local surface normal at the ray intersection point.

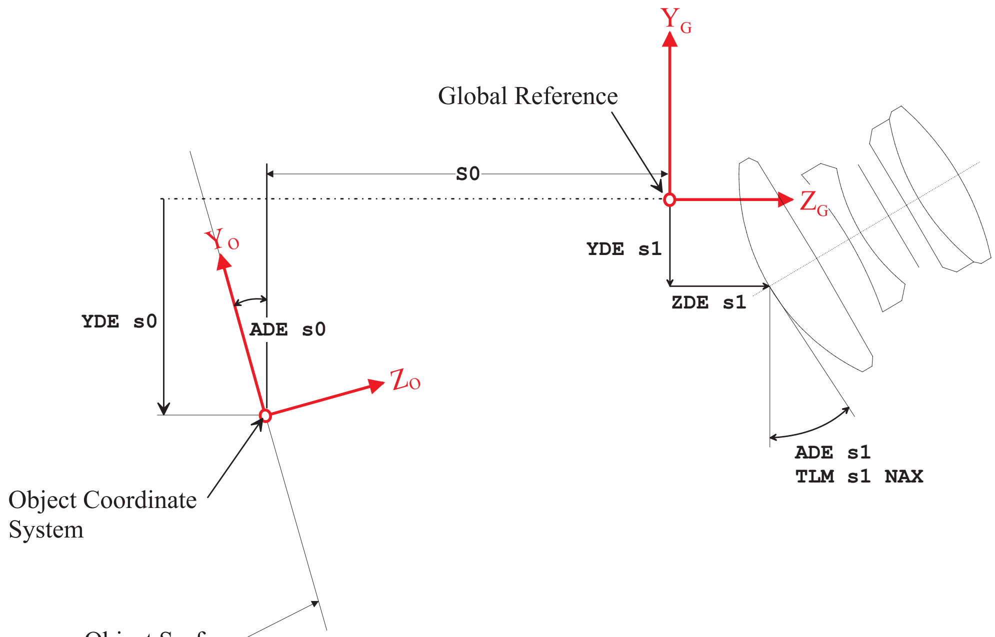


Figure 5.3: Object coordinate system with reference to the global coordinate system.

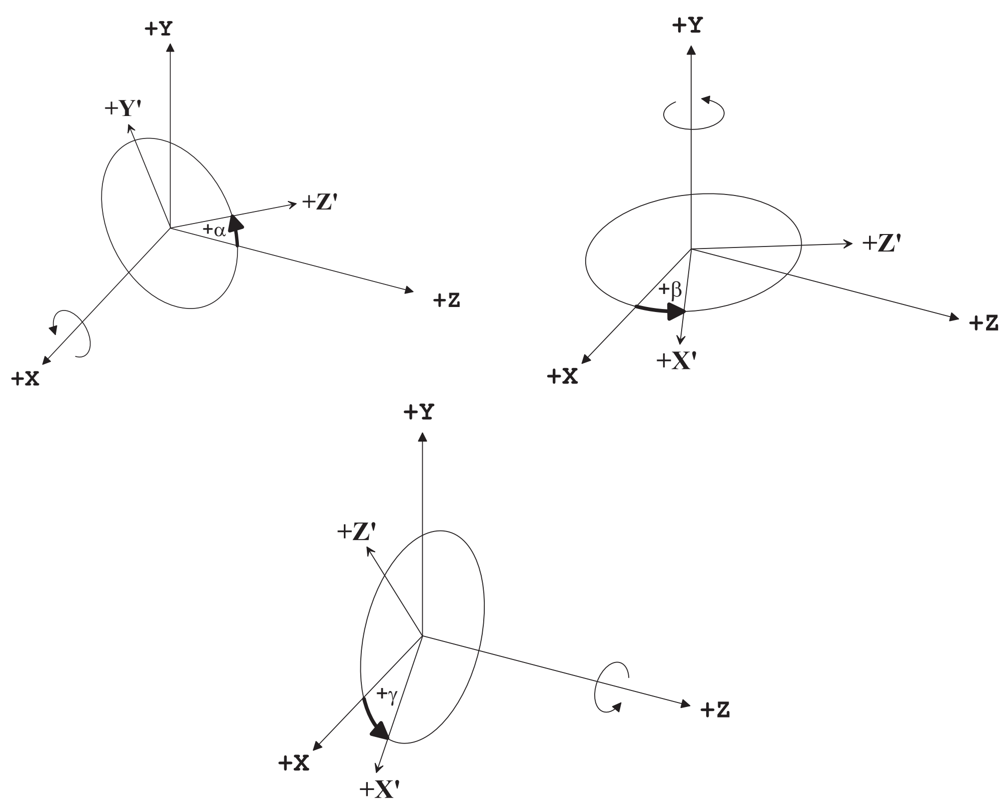


Figure 5.4: Tilt angles and sign conventions for rotations about x-, y- and z-axis.

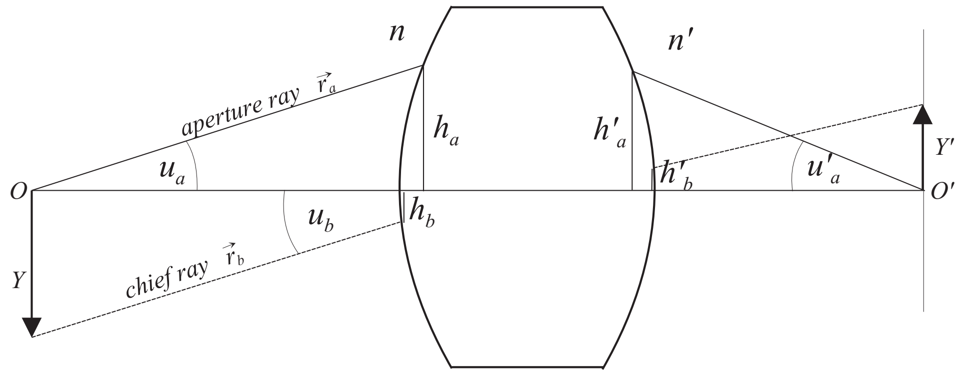


Figure 5.5: Definition of paraxial entities.

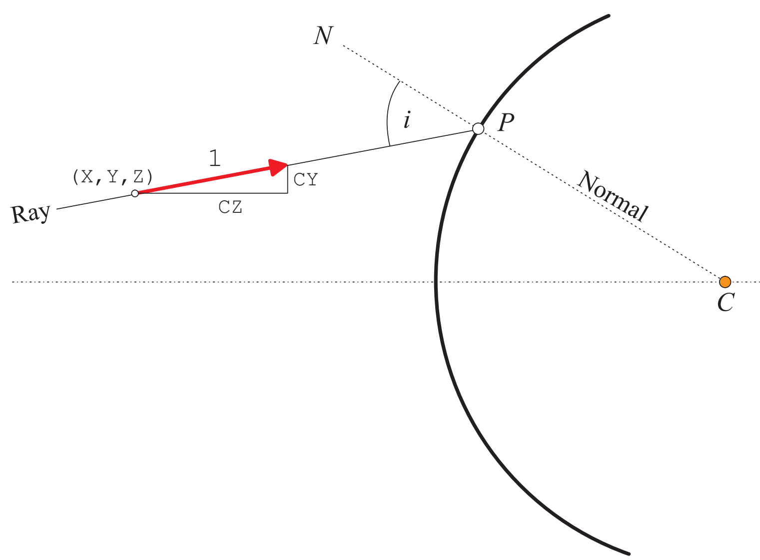


Figure 5.6: Definition of rays.

6

The Command Line

6.1 General

$OpTaliX$ has two modes of operation, either from the menu bar in the main window or from the command line. Although the menu provides an easy to use and easy to learn interface, the command line, which is found underneath the menu bar and in the text (output) window, offers a wider range of options and greater flexibility. All parameters and actions are accessible from the command line.

The syntax of the command line is universal throughout the program, since it is used for program control, for definition of optimization constraints and also in the macro language.

By default commands entered in the command line are reflected in the history window. Commands can also be "echoed" in the text window, if enabled by the "ECHO Y" command.

Any number of commands may appear in the command line, separated by semicolons ";". For example, two simple commands, which list the system data and plot a ray aberration fan, are:

```
lis
fan
```

or, written in a single command line, separated by semicolons ";"

```
lis ; fan
```

6.2 Command Syntax

To a maximum possible extent, the command syntax used in $OpTaliX$ is compatible with CODE-V commands. In addition, there are a few commands not found in CODE-V which describe dedicated $OpTaliX$ features.

6.2.1 Qualifiers

Many of the commands accept parameters for surfaces, field, wavelength, zoom positions, rays, coefficients, pupils, sources, etc. The generic syntax is :

`sk|si..j` Surface (`sk`) or surface range (surfaces i to j), also defines light source number. Distinction between surface number and light source number is made within command context.
`fk|fi..j` Field (`fk`) or field range (field numbers i to j)
`wk|wi..j` Wavelength (`wk`) or wavelength range (color numbers i to j)
`zk|zi..j` Zoom position (`zk`) or zoom range (zoom position i to j)
`ck|ci..j` Coefficient (`ck`) or coefficients range (range i to j, used for holograms (HOE), user-defined surfaces (UDC), and user-defined gradients (UDG).)
`pk|pi..j` Pupil number (`pk`) or range of pupils (= surface aperture) i to j
`lk|li..j` Coating Layer (`lk`) or range of layers i to j
`gi` Global reference surface number i

Thus, surface number, wavelength number, field number, zoom number, pupil number, coating layer, etc. must be preceded by its proper qualifier without spaces (e.g. `s` for surface, `w` for wavelength, `f` for field, `z` for zoom, etc.). A range of either surfaces, fields, wavelengths, rays, coefficients or pupils is specified by two consecutive dots ”`..`”.

If a range is specified on either surface, field, wavelength, zoom position, etc., the parameters are applied to all command items within the given range, e.g.

`rdy s1..3 10.0`	! sets radii of surfaces 1 to 3 to 10.0
`yan f2..4 2.5`	! sets Y-angle of fields 2 to 4 to 2.5 (degree)
`spd f3 w2 z3..4`	! analyzes the (RMS) spot diameter at field 3, wavelength number 2 and zoom positions 3 to 4.
`y s7 f1 w1 g2 0 1`	! Outputs Y coordinate of a ray at surface 7, field 1, wavelength 1, in global coordinates referred to local coordinate system of surface 2

6.2.2 Special Surface Qualifiers

There are special surface qualifiers for object surface, stop surface, image surface and *all* surfaces, which may be specified as

`so` for object surface,
`ss` for stop surface,
`si` for image surface,
`sa` all surfaces.

The following commands are synonymous:

```
thi s0 100        thi so 100
cir s5 12         cir ss 12     ! assuming surface 5 is the stop.
rdy s8 -300       rdy si -300   ! assuming surface 8 is the image.
```

6.2.3 Variable Qualifiers

Qualifiers for surface, field, wavelength or zoom position may also be combined with variables. For example, thickness on surface `s2` may also be defined by

```
$x = 2
thi s$x ...
```

This feature may be understood as concatenating "s" (without the quotes) and the value of `$x`. With the example given above,

`s$x` is interpreted as `s2`
`f$x` is interpreted as `f2`
`w$x` is interpreted as `w2`
`z$x` is interpreted as `z2`

These constructs are available in commands, macros and within lens database items (LDI).

6.2.4 Entering and Changing Data

Entering and changing data is accomplished by a free format command syntax which is similar to CODE-V commands in many (but not all) respects. The main features of the command syntax are:

- It is uniform throughout *OpTaliX* and to a maximum possible extent compatible to CODE-V,
- it is flexible to support future needs,
- it uniformly uses blanks as delimiters,
- the command parameters can be used in any sequence,
- commands can be annotated by semicolon (;) separator.

All commands are case insensitive, i.e. the commands

```
RDY S1 34.5
rdy s1 34.5
Rdy S1 34.5
```

are interpreted in the same manner. All parameters are separated at least by one blank. Multiple blanks are treated as a single blank, i.e. the commands

```
RDY S1 34.5
rdy       S1                     34.5
```

are identical.

6.3 Surface Pointer

As the name implies, a surface pointer directs to a designated surface in the optical system. Use of a surface pointer allows simplified entry of construction data (such as radii of curvatures, thicknesses, etc). The surface pointer is set by the command

`sk`

where `k` denotes a surface number. Thus, `sk` means you should type `s4` or `s17`, where `4` or `17` is the desired surface number. The actual position of the surface pointer is indicated in the prescription listing (see LIS command) by the $>$ character right to the surface number . For example, the commands

```
s3
lis
```

produce the output

```
# TYPE        RADIUS     DISTANCE  GLASS          INDEX    APE-Y AP CP DP TP MP GLB
1 S          31.9354      4.90200  LAK9        1.694019    17.00* C  0  0  0  0   0
2 S          95.0214      0.22600              1.000000    16.36  C  0  0  0  0   0
3>S          18.9471      5.42100  LAK9        1.694019    13.38  C  0  0  0  0   0
4 S          51.7823      2.82700  SF8         1.694169    12.29  C  0  0  0  0   0
5 S          12.8019      6.84900              1.000000     8.58  C  0  0  0  0   0
```

In second and succeeding references to the same surface number the surface qualifier can now be omitted, if desired. For example,

```
s3
rdy 100
thi 5.2
```

is fully equivalent to

```
rdy s3 100
thi s3 5.2
```

That is, in absence of a surface qualifier, the surface specified by a previous `sk` command is used. Note that the surface pointer is set to surface 1 on restoring a new optical system.

The current setting of the surface pointer can be queried by the command

```
s?
```

6.4 Surface Qualifiers and Arithmetic Expressions

Surface qualifiers (NOT field, wavelength, zoom or pupil qualifiers) also accept arithmetic operators, ”+”, ”-”, ”*” and ”/”. This is particularly useful in conjunction with the special qualifiers `so`, `ss` and `si` but also works for regular surface qualifiers, like `s3` or `s16`. The following examples indicate valid usage of arithmetic operations on surface qualifiers:

`si-1`	surface before the image surface,
`ss+1`	surface after the stop surface,
`so+2`	denotes the second surface (object surface = surface 0 plus two surfaces),
`s3..i-1`	denotes a range from surface 3 to the surface before the image surface,
`s2..s+1`	denotes a range from surface 2 to stop surface plus one surface.
`ss-1..s+1`	denotes a range from the surface before the stop surface to the surface after the stop surface.
`ss-1..ss+1`	same as above
`s4..7-2`	surfaces 4 to 5
`s3..s4*2`	invokes multiplication on surfaces, resulting in surfaces 3 to 8.
`s4/2..i-2`	invokes division, resulting in surfaces 2 to image surface less 2.
`s3-2+4`	multiple operators are permitted.
`s3+sqrt(4)`	functions may be used, here resulting in surface 5. Note that only integer value should be used. Float numbers (albeit permitted) may lead to unpredictable results due to rounding effects.

Invalid surface or surface range qualifiers:

`ss+-2`	operator follows operator.
`s3.5`	surface range requires two consecutive dots.

6.5 Functions and Arithmetic Expressions

Numbers entered in the command line can also be arithmetic expressions or functions. In this way, it acts like a pocket calculator. For example, the entries

```
rdy s1 100
rdy s1 2*(40+20)-20
rdy s1 sqrt(10**4)
```

are all equivalent. **Note that blank characters are not allowed in arithmetic expressions, except where enclosed in brackets.** Expressions may also be copied from the clipboard directly to the command line. The functions and operators recognized are shown in table6.1:

Functions	Operators
`cos`	`+`
`sin`	`-`
`tan`	`*`
`exp`	`/`
`log`	`**`
`log10`	`^`
`logn`	
`sqrt`	
`acos`	
`asin`	
`atan`	
`cosh`	
`sinh`	
`tanh`	
`besj0`	
`besj1`	
`besjn`	
`anint`	
`aint`	
`abs`	

Table 6.1: Functions and operators recognized by $OpTaliX$. See also section 26.2

In the command line brackets and correct order of operation are also recognized. In trigonometric functions, the argument must always entered in radians and inverse trigonometric functions report angles in radians. For example to compute $sin(30°)$, it must be entered as `sin(30*3.14159/180)`. This form can be simplified by defining constants or variables and using them in arithmetic expressions

```
#define rad 3.14159/180
sin(30*rad)
```

or

```
@rad == 3.14159/180
sin(30*@rad)
```

Further details are given in chapter 26 (Macro Language).

6.6 Lens Database Items

Lens database items (LDI) are specifications of values which may be retrieved from the current optical system. Virtually anything that can be entered in the command line has a corresponding lens database item (see also chapter 27). All references to lens database items must be enclosed in rectangular brackets [ ], even if there are no qualifiers. Within the brackets, the syntax for database items is identical to the syntax used for command line input.

Examples:

```
thi s2  [EPD]       ! sets thickness s2 equal to entrance pupil diameter
cuy s3 -[cuy s4]    ! curvature on surface 3 is equal to minus the
                    ! curvature on surface 4
```

Database items can be combined with arithmetic operators to form an arithmetic expression anywhere a numeric data entry is expected.

```
fno [EFL]/[EPD]  ! sets F-number
thi s3 2*sqrt(3)*[thi s1]
```

Note that pre-defined functions (sin, tan, sqrt,...) and specification of lens database item references are case insensitive. For example, the following expression given in upper case, lower case or mixed case are valid:

```
thi s3 2*sqrt(3)*[thi s1]
THI S3 2*SQRT(3)*[THI S1]
thi S3 2*SqrT(3)*[thi S1]
```

See also a detailed explanation of the macro capabilities in chapter26 and the lens database reference in chapter 27.

6.7 The Question Mark Symbol (?)

Most of the commands accept the ”question mark” symbol ”?”, which allows a dialog based modification of relevant parameter. For instance, the fan (rim ray) plot may be entered in two ways:

`FAN`	plots the fan (rim ray) aberrations without asking for a scaling parameter (the default or previously applied scaling factor is used).
`Fan ?`	invokes a dialog box to edit the aberration scaling factor prior to plotting the fan aberrations.

6.8 Rules for Command Entry

- Always separate parts of *OpTaliX* instructions with one or more blank characters (blanks).
- Never put spaces between command words, qualifiers, ranges or numbers. For example, `LIS` or `S3` are valid entries, `L IS` or `S 3` (with blanks enclosed) are not.
- Upper and lower case letters can be used. *OpTaliX* ignores cases such as `THI` = `tHi` = `thi`.
- Arithmetic expressions such as `2*3+5` must not contain blanks, except where enclosed in parentheses (). For example, `2*3+5` and `(2*3 + 5)` are equivalent, whereas `2*3 + 5`, (without the parentheses) are interpreted as two separate expressions.
- No spaces are permitted within numerics.
- Numeric input is defined as follows: Integers or floating point values with or without leading sign (+,-) or leading zeros, such as `+0.5`, `.5`, `5E-1`, `-2D-10`, etc. (see also section 26.2).
- Always precede a surface number, field number, zoom number, wavelength number, etc. with its corresponding qualifier prefix (`S` for surface, `W` for wavelength, `Z` for zoom position, etc.), without spaces. For example, `S3`, `W5` are valid entries, `S 3` (with blanks) is not. `O`, `S` and `I` (for object, stop and image) are valid surface numbers. Examples: `SO`, `SI`, `SS`. Addition, subtraction, multiplication and division can be used on surface qualifiers only as in `SI-1`, `SS+4`, `s3*2`, etc.
- Never add additional characters to command or qualifier words. For example, `LIS` is correct, `LIST` is not.
- Strings containing spaces, semi-colons ";" or ampersands "!" must be enclosed in single or double quotes.
- Continuation of commands with the ampersand character "&" is only possible in macros. This feature is not available in the command line.
- Multiple commands within a command line must be separated by the semicolon character ";".

7

Configuration and System Data

In the terminology used throughout the manual, *system or configuration data* are data that pertain to the whole lens or describe its conditions of use. For example, typical system/configuration data, among others, are aperture, field of view and wavelength. These are attached to the lens data and are saved with the surface data.

7.1 Setting up a new lens system

Setting up a new lens system from scratch means that the previous system is deleted from memory, all old lens data is destroyed. An ”empty” system is created which contains only two surfaces, the object surface and the stop surface. Reasonable default values are initialized. The command `LEN` is not necessary prior to restoring a lens from the library. This is done internally by the program. Optical surfaces may be added appropriately by the `INS`-command.

`LEN`	Set up a new lens. Initializes all surface parameter and defaults for a new lens. All old lens data is destroyed.
`DIM I/M`	Dimensional System. `M` = millimeter (default), `I` = inch
`RDM yes/no`	Select radius or curvature mode. Use radii (yes) instead of curvature (no) as the basic shape representation of a surface (default = yes). This option only works in command mode. In the surface spreadsheet editor only radii are accepted.

7.2 Saving and Restoring Lens Data

`RES [file_spec]`	Restore lens data from `file_spec`. Example: `res c:/optix/test.otx`
`SAV [file_spec]`	Save lens data in `file_spec`. The complete path (directory and file name) must be specified. If `file_spec` is omitted, the existing file will be overwritten. Examples: `sav c:/optix/test.otx` `sav` ! overwrites existing file.
`WRL file_spec`	Save lens data in Code V sequential format. See also sect. 30.1.

7.3 General Lens Data (Configuration Data)

General lens data (or configuration data) define the usage of an optical system. These include specifications on fields, wavelengths and aperture, as well as a few special data such as afocal switches or methods of ray aiming.

The commands for editing/defining system configuration data are:

`EDI CNF`, or `EDI CFG`	Edit Configuration Parameter. A dialog box is opened.
`EDI FLD`	Edit Field Parameter. A dialog box is opened.
`EDI LAM`	Edit Wavelength Parameter. A dialog box is opened.
`EDI ZOO`	Edit Zoom Parameter. A spreadsheet is opened.
`AFO yes| no`	Afocal switch. Specifies that this is an afocal system where the exiting beam is nominally parallel (image is at infinity). This model assumes that a perfect lens is placed after the last surface (although the user does not explicitly need to specify this ideal lens, this is automatically done internally). The focal length of the ideal lens is pre-set to 1000mm, i.e. an aberration of 1 mm is equivalent to 1 mrad in image space.
`TIT 'string'`	Enters a title (max. 256 characters). The title is displayed in the lens layout plots and the system prescription.
`RDM yes|no`	Select radius or curvature mode. Use radii (yes) instead of curvature (no) as the basic shape representation of a surface. (default = yes)
`SET MAG mag_value`	Set magnification. Changes the object distance required to satisfy *paraxial* magnification of `mag_value`. This is a static (one-time) adjustment. In order to adjust magnification permanently (dynamically as the system changes), use the RED solve (page 105).

7.3.1 Fields / Object Points

In optical design, the term "fields" describes the entity of object points used for calculating the performance of an optical system. Thus, a "field", or field point, is just the location of an (infinitesimally small) object point defined at the object surface (respectively referred to the object coordinate system (page 27)). For reference see also the object coordinate system.

Another way to specifying objects is by defining extended emitting sources, which are mainly used in illumination analysis. See chapter 15, page 295 for a detailed treatment of this type of sources.

Resorting to **point** objects, the number of field points (objects) is unlimited. Initially, a maximum number of 30 field points is assumed, however, this value can always be increased to any arbitrary value using the `MAXFLD` command. Fields can be specified independently in X- and Y-direction in terms of object height (`XOB`, `YOB`), paraxial image height (`XIM`, `YIM`), real image height (`XRI`, `YRI`) or angles (`XAN`, `YAN`) in the object space. Fig. 7.1 shows the four types of defining fields.

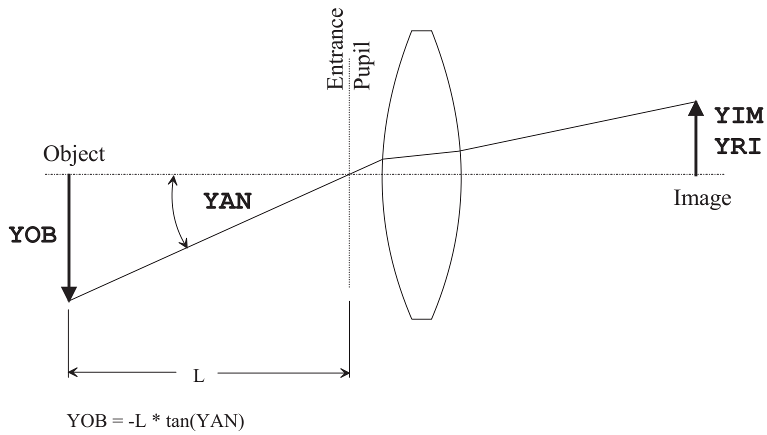


Figure 7.1: Relationships between different forms of field specification. Shown are Y-objects only.

`EDI FLD`	Invokes a dialog box to enter X-field, Y-field, field type, and number of field points. Command line input is given by the commands below.
`NFLD num_fields_used`	Number of field points in use for performance analysis. This command must not confused with `MAXFLD` (see below). Also note that you should set `NFLD` to the maximum number of fields *before* saving the system, otherwise field data larger than `num_fields_used` will be lost.
`MAXFLD max_fields`	Maximum number of field points (objects). This command does not affect the number of fields in use for performance analysis (see `NFLD` command), it merely sets the maximum number of *allocated* fields.
`XAN [fi..j] x_angle1 x_angle2 ... x_angle_n`	Field angle (in degree) in X-direction, referred to Z-Axis. The number of entered field angles also sets the number of fields during performance analysis.
`YAN [fi..j] y_angle1 y_angle2 ... y_angle_n`	Field angle (in degree) in Y-direction, referred to Z-Axis. The number of entered field angles also sets the number of fields during performance analysis.
`XOB [fi..j] x_obj1 x_obj2 ... x_obj_n`	Object coordinates (X) for finite object distances. The number of entered field angles also sets the number of fields during performance analysis. **`XOB` data will be interpreted as X-field angles if the object is at infinity.** See also notes below.
`YOB [fi..j] y_obj1 y_obj2 ... y_obj_n`	Object coordinates (Y) for finite object distances. The number of entered field angles also sets the number of fields during performance analysis. **`YOB` data will be interpreted as Y-field angles if the object is at infinity.** See also notes below.

continued on next page

continued from previous page	
`XIM [fi..j] x_image1 x_image2 ... x_image_n`	Image coordinates (X), defined in the *paraxial* domain. The number of entered fields also sets the number of fields during performance analysis.
`YIM [fi..j] y_image1 y_image2 ... y_image_n`	Image coordinates (Y), defined in the *paraxial* domain. The number of entered fields also sets the number of fields during performance analysis.
`XRI fi..j x_real_img_ht ... n`	Compute X-object height based on real image height. Object heights are continuously adjusted as the lens changes. Ensures that the real chief rays (at the reference wavelength) hit the image surface at the specified image heights. Not applicable in afocal (AFO Y) systems.
`YRI fi..j y_real_img_ht ... n`	Compute Y-object height based on real image height. Object heights are continuously adjusted as the lens changes. Ensures that the real chief rays (at the reference wavelength) hit the image surface at the specified image heights. Not applicable in afocal (AFO Y) systems.
`FTYP field_type`	Field type. This is a complementary command to change the field type specification (i.e. XAN, YAN, XOB,YOB, XIM,YIM). Field type is defined as : 1 = specifies angles (XAN,YAN) 2 = specifies object coordinates (XOB,YOB) 3 = specifies **paraxial** image coordinates (XIM,YIM) 4 = specifies **real** image coordinates (XRI,YRI). Computational intensive!

`FWGT [fi..j] fweight1 fweight2 ...`
or
`WTF [fi..j] fweight1 fweight2 ...`
	Field weight, an integer value between 0 and 100.

`FACT [fi..j] 0/1`	Field activation. A particular field point may be excluded from analysis, i.e. it is not active. 0 = inactive, 1 = active.
	continued on next page

continued from previous page	
CLS FLD [fk|fi..j] [colour...n]	Selects the colour list used for fields in graphical output (e.g. VIE). Input of fewer colours than the number of fields uses the last colour entered for the rest of the fields. With no colours specified, colours are set to default settings. Examples: `cls fld red gre blu !` defines red, green and blue for the first three fields. `cls fld` ! no colours specified, default field colours are selected. `cls fld f3 red` ! change plot colour for field 3 to red. See also names of predefined colours and their definition in sect. 28.1, page 471.

Notes:

- For objects at infinity (i.e. object distance is $\geq 10^{20}$), object coordinates (either entered by XOB,YOB commands or defined by 'FTYP 2' command) are specially handled. Field values are then interpreted as field *angles* instead of real object coordinates. It is obvious that object coordinates must also be very large for infinitely distant objects (i.e. THI s0 is $\geq 1.E20$). For example, an apparent field angle of $30°$ would require an object height (OBY) of $tan(30) * 10^{20} = 5.77E19$. This may lead to a loss of internal computational accuracy and the program therefore interprets field values for infinitely distant objects as field *angles* (in degree).
- Field specifications can be entered in any order. It is not required that they be ascending or descending values.
- If the system is rotationally symmetric, only Y-field specifications should be used, i.e. X-field components are zero. The program checks for symmetry condition about the Y-axis to reduce computing time.
- Object space field specification (XOB/YOB or XAN/YAN) are recommended for systems with decentered surfaces.
- Paraxial image space field specification (XIM/YIM) is useful for zoom systems with constant image size across zoom positions. This eliminates the need to zoom field specifications.
- Real image space field specification (XRI/YRI) is useful when exact image points are desired. Includes effects of distortion, which is particularly useful in zoom systems where distortion can vary across zoom position.

7.3.2 Astigmatic Objects

Simulates an astigmatic shift in the emitted light which some sources, such as laser diodes, have. This option is only available for finite object conjugates.

`ASF delta_f_microns`	Astigmatic focus shift in microns. Shift of sagittal source (i.e. X/Z-plane) from the tangential source (Y/Z-plane). If 0 is entered for `ASF`, the astigmatic shift is disabled. The astigmatic focus is always defined in microns and is always measured along the chief ray.
`ASO angle_degree`	Orientation (in degrees) of astigmatic focus shift. 0 corresponds to shifted source oriented with X-axis.

In gain guided laser diodes, light appears to diverge from different points, depending on the orientation considered. Light perpendicular to the active layer emits from the front face of the diode, whereas light in the plane of the active layer is emitted from a virtual point located between 20μm to 30μm behind the emitting window (in negative Z-direction).

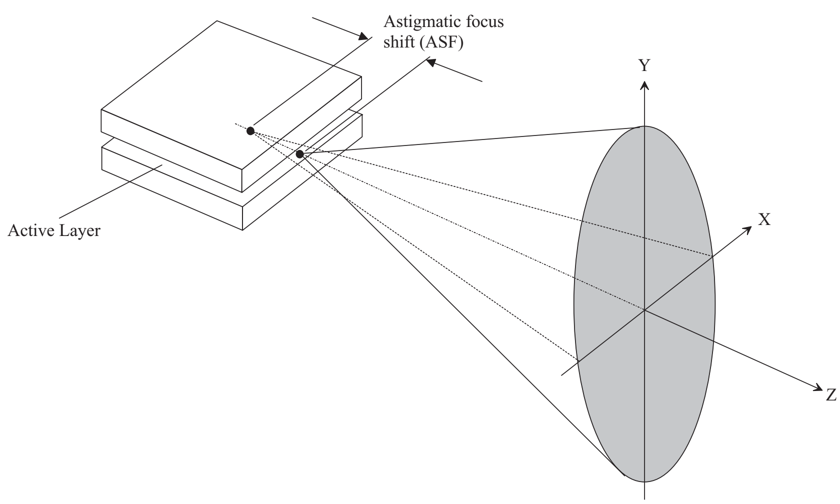


Figure 7.2: Geometry of astigmatic focus shift in a laser diode.

7.3.3 Wavelength Definition

The number of wavelength is limited to 11. The order and sequence of the wavelengths may be arbitrary. There is always one specific wavelength which serves as reference wavelength. It is used to define first order (paraxial) properties, pupil definition, image plane location, etc.

`EDI LAM`	Invokes a dialog box to enter wavelength, weights, number of wavelength and reference wavelength. The dialog box is shown in Fig. 7.3.
`WL lam1 lam2 lam3 ... lam11`	Wavelength definition. Enter up to 11 wavelengths (in μm) in any order. The number of entered wavelength values also sets the number of wavelength during performance analysis. Example: `wl 0.546 0.48 0.7` sets 3 wavelength (colours).
	continued on next page

continued from previous page	
NWL no_of_wavelengths	Sets the number of wavelengths used in the system.
REF ref_w	Sets the reference wavelength. It designates which of the WL wavelengths is to serve as the reference wavelength for all first order properties and monochromatic aberrations. Example: REF 2
WTW weight	Weights for corresponding wavelengths. (Specifies relative spectral intensities). The values given are integer numbers and range from 0 to 100. Example: WTW 50 100 75 Note: the wavelength weights may also be edited in a dialog box using the command EDI LAM (see above).

7.3.4 Optical Spectrum

Rather than enter wavelength/weight pairs explicitly you can store wavelength data as an *optical spectrum*. An optical spectrum is the collection of wavelengths, weights, and reference wavelength stored with a user-definable name for later retrieval. This feature is particularly useful in zoom/multi-configuration systems utilizing different spectral channels. Different optical spectra (i.e. wavelength/weight combinations) may be assigned to each zoom position in a single command.

OSP spectrum_name [?]	Loads a predefined optical spectrum and automatically sets wavelengths, corresponding wavelength weights and reference wavelength. The number of wavelength to be used must be previously set by the NWL command (see above). A list of available optical spectra is given below. Examples: osp photopic ! selects visible (daylight, photopic) spectrum. osp ? ! invokes a dialog box to interactively set the optical spectrum (see Fig. 7.3).
OSP PLANCK temp_degK	Sets the optical spectrum according to the spectral radiance of a black body using Planck's law. A third parameter, the temperature of the black body in Kelvin is expected. This command uses the currently defined wavelengths and only sets wavelength weights! This option is currently only available from the command line. Example: osp planck 6000 ! Sets wavelength weights according to a black body spectrum at 6000K.
SAV OSP spectrum_name	Save optical spectrum (wavelengths, weights and reference wavelength) under spectrum_name. Use OSP command to assign a saved spectrum to the system configuration data.

List of predefined optical spectra:

Spectrum name	Description
Pan	Spectral sensitivity of a typical panchromatic film.
Photopic	Relative sensitivity of the human eye for daylight illumination (photopic vision).
Scotopic	Relative sensitivity of the human eye under conditions of dark adaptation (scotopic vision)
MWIR	Medium wave infrared, 3μm - 5μm waveband
VLAM	Same as ”Photopic”

Dialog based editing of optical spectra:

Wavelengths, weights and reference wavelength can also be edited in a dialog box which is accessed from the main menu *Edit/Configuration* and then selecting the *wavelengths* tab (see Fig. 7.3). The ensemble of wavelengths and corresponding weights constitutes an ”optical spectrum”. It defines the wavelength range and also the relative spectral intensities (weights) within that range. Weights are given by integer numbers, preferably between 0 and 100, but any other positive number is also accepted.

A set of predefined optical spectra may also be directly selected from the combo box in the right part of the dialog. Choosing one of the predefined spectra avoids entering each wavelength/weight pair manually. Once an appropriate spectrum has been selected, pressing the ”Set” button underneath the graphical display of the spectrum will automatically set wavelengths, weights and reference wavelength.

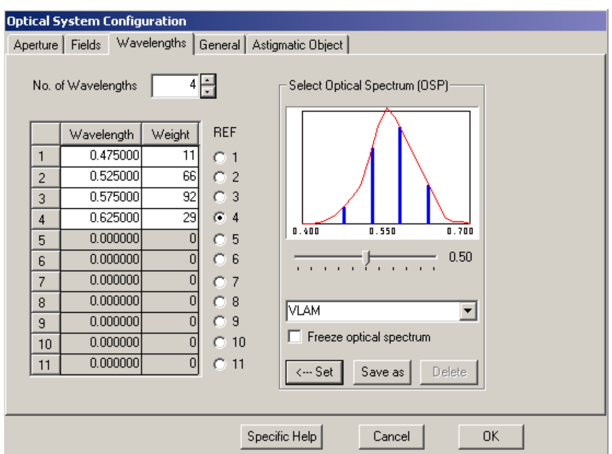


Figure 7.3: Wavelength and optical spectrum editing.

Freeze optical spectrum:

When an optical spectrum is selected and applied to the system configuration, all wavelengths will normally be equidistantly scaled within the spectrum limits. If you wish to apply wavelengths exactly as defined and stored, check the ”Freeze optical spectrum” check box in the wavelengths tab.

7.3.5 System Aperture

The system aperture defines the aperture used for the whole lens. This definition must not be confused with surface apertures (see 8.32 on page 158).

The system aperture may be defined in various ways, for example by

- NA, the numerical aperture in the image space,
- NAO, the numerical aperture in the object space,
- EPD, the entrance pupil diameter,
- FNO, the F-number,
- or by the physical stop semi-diameter.

Fig. 7.4 illustrates these options.

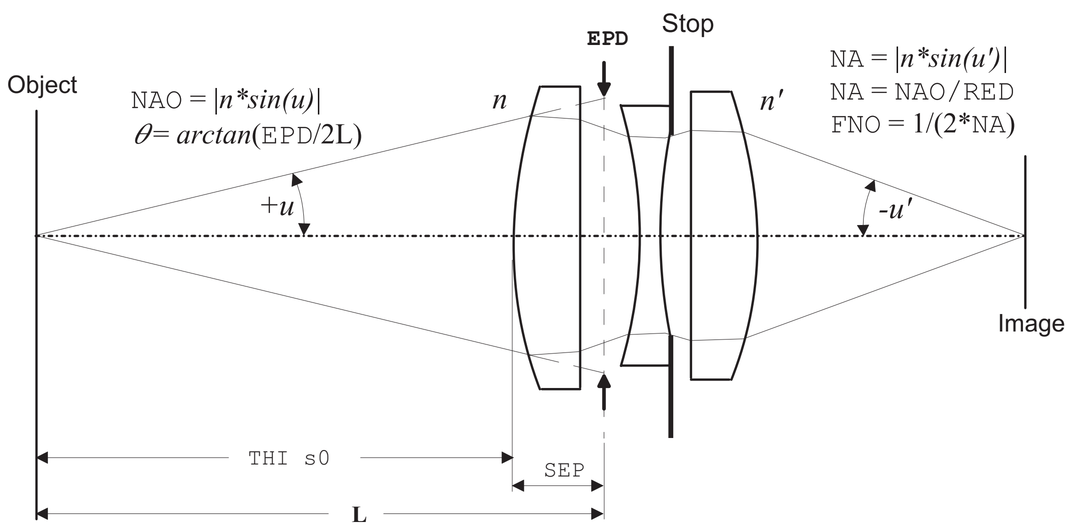


Figure 7.4: Defining system apertures.

Commands:

FNO [zi..j|zk] F_number	Define aperture by F-number in the image space. The stop diameter is adjusted to satisfy the F-number when the lens is changed. Note: The F-number is calculated by definition at magnification = 0 (object at infinity).
DEL FNO	Delete previous F-number setting, so the stop diameter is no longer automatically adjusted.
EPD [zi..j|zk] entrance_pupil_diam	Entrance Pupil Diameter (EPD). This command sets the stop surface aperture dimensions to satisfy the entrance_pupil_diam condition. In case of a rectangular aperture, the EPD is defined as the diagonal of the rectangle, i.e. the surrounding circle. In case of an elliptical aperture, the EPD is the maximum value of the ellipse axes.
DEL EPD	Delete previous EPD (entrance pupil diameter) setting, so there is no subsequent adjustment of the stop diameter.
	continued on next page

continued from previous page	
NA [zi..j|zk] num_aperture_image	Define aperture by numerical aperture in the image space (at working magnification).It adjusts the stop diameter to satisfy the num_aperture_image requirement when the lens is changed.
DEL NA	Delete previous numerical aperture setting, so there is no subsequent stop diameter adjustment.
NAO [zi..j|zk] num_aperture_object	Define aperture by numerical aperture in the object space (at working magnification). It adjusts the stop diameter to satisfy the num_aperture_object requirement when the lens is changed.
DEL NAO	Delete previous numerical aperture setting (in object space)
POF oversize_factor	Increases the dimension of the system aperture by a factor oversize_factor for the ray grid. The default factor is 1. POF only needs to be modified in systems showing significant pupil distortion, for example in wide-angle retrofocus systems.
Related Command	
NRD num_rays_diam	Number of rays across pupil diameter. Defines the size of the (rectangular) ray grid in the entrance pupil. NRD is adjustable in 2^n steps, i.e. the ray grid may have sizes of 4^2, 8^2, 16^2, 32^2, 64^2, 128^2, 256^2, 512^2 and 1024^2. The higher num_rays_diam is, the more accurate the results will be. However, the computing time will increase quadratically with increasing num_rays_diam. Although 1024^2 rays are accepted by the program, practical memory limitations make this option unlikely. Practice has shown, that grid sizes of 64^2 or 128^2 rays are very rarely required and 32 x 32 rays (the default in $OpTaliX$) are the best compromise between accuracy and speed. The ray grid is used in geometrical and diffraction analysis, e.g. spot, wavefront, PSF, MTF, etc.

Note: The aperture definitions (NA,NAO,EPD,FNO) permanently adjust the stop diameter when system parameters change, unless aperture adjustment is deactivated by any of the commands DEL NA, DEL NAO, DEL EPD or DEL FNO. The stop aperture then remains fixed.

In case of non-circular *system* apertures, i.e. rectangular, elliptical or polygon system apertures, specifications of NA, NAO, FNO or EPD are always defined by the surrounding circle of the non-circular system aperture. This convention is illustrated in Fig. 7.5 on the examples of rectangular and polygon system apertures.

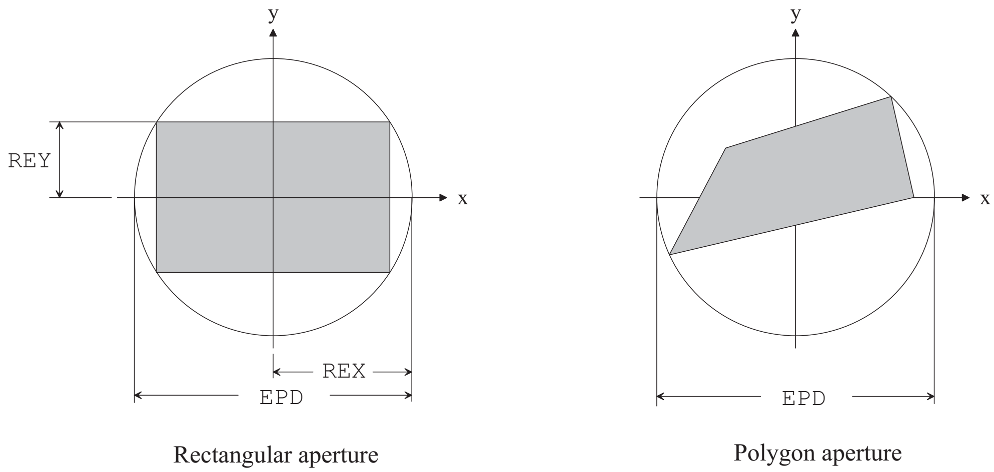


Figure 7.5: Definition of *system* aperture (not surface aperture!). Similarly, this also applies to elliptical apertures. NA, NAO, FNO and EPD are always referred to the surrounding circle of the complex system aperture shape.

7.3.6 Pupil Apodization

Gaussian intensity distribution across the entrance pupil. In most cases, this feature is required to simulate a laser beam which is clipped at a certain level at the paraxial entrance aperture.

`PUI intensity`	Apodization of intensity across the (paraxial) entrance pupil with a gaussian distribution. `intensity` defines the intensity at the relative pupil coordinates of `PUX`,`PUY`. The peak intensity is 1 at the aperture center (`PUX=PUY=0`). The default is `PUI 1.0` which corresponds to a flat (unapodized) intensity distribution.
`PUX rel_ape_radius_X`	Relative X pupil coordinate (normalized to the entrance pupil radius) at which the PUI value is reached. The default is `PUX 1.0`
`PUY rel_ape_radius_Y`	Relative Y pupil coordinate (normalized to the entrance pupil radius) at which the PUI value is reached. The default is `PUY 1.0`

An elliptical intensity distribution may be defined with different values for `PUX` and `PUY` . A gaussian intensity apodization, defined by the commands `PUI , PUX , PUY`, is evaluated by:

$$I(x_p, y_p) = e^{(\ln \text{PUI})\left[\left(\frac{x_p}{X}\right)^2 + \left(\frac{y_p}{Y}\right)^2\right]} \tag{7.1}$$

with

$I(x_p, y_p)$	Intensity
x_p, y_p	entrance pupil coordinate
X	`PUX` * (entrance pupil radius)
Y	`PUY` * (entrance pupil radius)

Eq. 7.1 normalizes the Gaussian apodization to 1 at the center ($x_p = y_p = 0$) and at the value of `PUI` at the elliptical contour defined by `PUX`, `PUY`. Equal values for `PUX` and `PUY` designate a circular

apodization. PUX and PUY may have any value, except 0.

Examples:
A circular gaussian intensity distribution, with intensity 0.135 at the rim of the entrance pupil, is specified as

```
PUI 0.135
PUX  1.
PUY  1.
```

An elliptical gaussian intensity distribution, with intensity 0.5 at relative pupil coordinates X = 1, Y = 0.7 is specified as

```
PUI 0.5
PUX  1.
PUY  0.7
```

Notes on entrance pupil apodization:

- Entrance pupil apodization should be regarded as a property of the incoming beam rather than the lens.
- Apodizing that occurs at surfaces inside the lens should be represented by 'surface intensity filters' stored in INT-files as described in section 8.27.5.
- Entrance pupil (and surface-based INT) apodization is included in all geometrical and diffraction analysis options.
- PUX, PUY are defined on a plane perpendicular to the chief ray at a given field. For an on-axis object point, the apodizing plane is also perpendicular to the optical axis, however, for off-axis field points the apodizing plane tilts in the same direction and by the same amount as the corresponding chief ray for that field.

7.3.7 Defocus

DEF defocus THI si defocus	Defocus value. The defocus defines the offset of the physical image plane from the paraxial focus. A negative value of DEF means that the physical focus is intrafocal (left) from the paraxial focus, and vice versa. **Defocus is only taken into account for "PIM yes". If paraxial image solve is turned off (PIM no), DEF (defocus) has no effect. The distance to the paraxial image, however, is still displayed for information only!** See also Figs. 7.6 and 7.7 for a representation of DEF and the associated data BFL and IMD. Note that the defocus may also be defined as the distance on the image surface (THI si). That way, DEF and THI si are identical.

Typically 'defocus' is used to account for (spherical) aberrations in an optical system for finding the optimum focus. As shown in Fig. 7.6 below, the lens exhibits significant amount of spherical aberration. Selecting the exact paraxial image plane apparently does not yield the optimum focus for

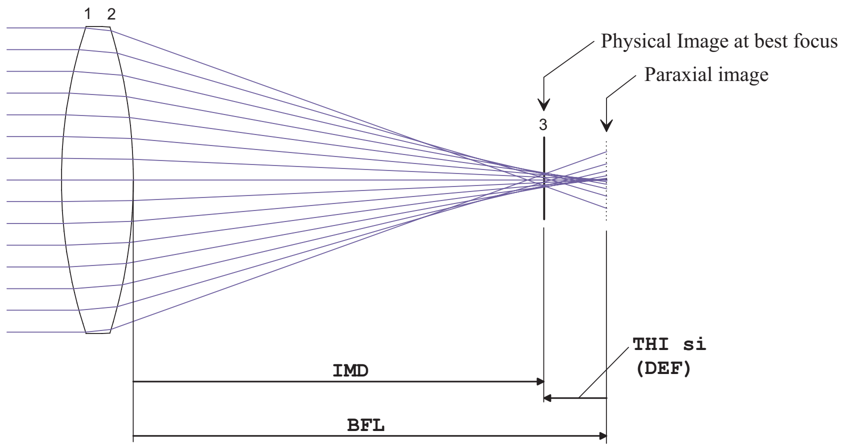


Figure 7.6: Representation of 'defocus' with respect to paraxial image. Defocus (DEF) is always measured from the *paraxial* image to the *physical* image surface at used conjugation. The image distance (IMD) is always measured from the last surface to the physical image surface.

which aberrations are minimized. Introducing an appropriate defocus term moves the *physical* image surface away from the paraxial image surface to the location of minimum circle of confusion.

Image distance (IMD) and defocus (DEF = THI si) are displayed in the surface editor (invoked by EDI SUR) as shown in Fig. 7.7. The defocus value can only be modified if "PIM Y" is set, otherwise (PIM N) defocus settings have no effect.

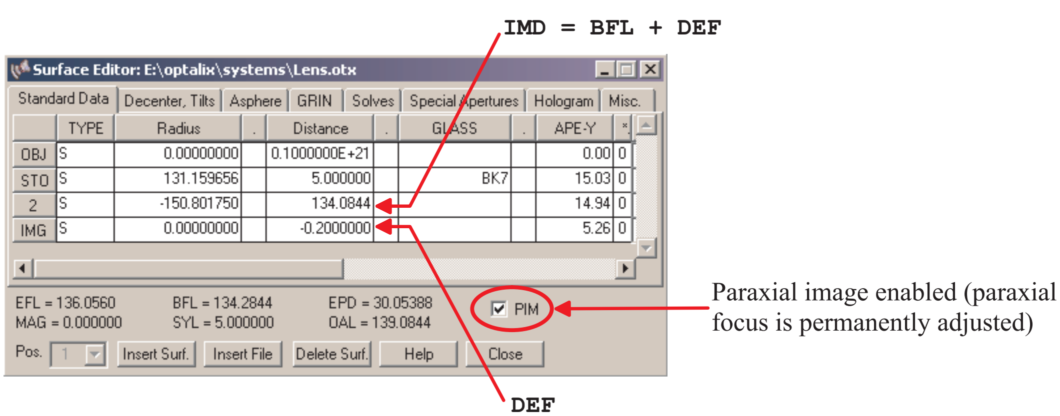


Figure 7.7: Display of image distance (IMD) and defocus (DEF) in the surface editor.

7.3.8 Remarks and Comments

REM	The REM command allows entry of up to 4 lines of text which are stored with the lens system. The comments are displayed with the system data listing and with the lens cross sectional view.
TIT 'string'	A title of the lens system, enclosed in apostrophes, can be entered. Up to 256 characters are allowed for 'string'.
COM si..j comment_string	Enter a descriptive text (up to 80 characters) per surface(s) si..j.
SLB si..j comment_string	As above, enter a descriptive text (comment) containing up to 80 characters per surface(s) si..j. This command is equivalent to the COM command, but has been added for Code V compatibility.

7.4 Ray Aiming Methods

Ray aiming is the method of determining start coordinates for selected fields. Ray aiming can be controlled by three parameters, RAIM, RAIT and RAIS. The RAIO command is obsolete (though still available) but use is discouraged. In general, the default settings for these three parameters need not be altered, but may accelerate ray generation in a few special cases.

RAIM [ENP|STO|TEL|OMN]	Ray aiming modes: ENP Rays are aimed at the paraxial entrance pupil. STO Rays are aimed to the physical stop surface. This is the default mode. TEL Telecentric ray aiming. OMN Omni-directional, i.e. rays are launched from a point source into arbitrary directions within 4π directional space. See also the commands OMN MIN and OMN MAX below. A detailed description on ray aiming methods is given below.
RAIT tolerance	Ray aiming tolerance. Only applicable for RAIM STO. The default ray aiming tolerance is 0.001 and is understood as a fraction of the aperture radius. For example, RAIT 0.001 on a 5mm aperture terminates ray iteration if the error on the desired ray coordinate is $< 0.001 \cdot 5mm$, i.e. $< 0.005mm$.
RAIS max_search_step	Ray aiming maximum step. Limits the step size during iteration for finding the start coordinates of a ray. max_search_step is defined in fractions of the entrance aperture, i.e. 1.0 corresponds to a step equal to the entrance pupil radius. Smaller values improve the probability of successful ray finding, in particular for systems with large pupil aberrations (for example wide-angle systems), however, speed of convergence may be reduced. Larger values accelerate ray iteration speed but ray aiming may fail on unusual systems. Reduce RAIS in such cases. The default value of RAIS is 5.

continued on next page

continued from previous page	
`RAIO 0|1`	Ray aiming option, now obsolete (but still available). RAIO 1 is equivalent to RAIS 0.2. This allows switching between normal ray aiming mode (RAIO 0) and a more accurate (but significantly slower) mode (RAIO 1). The default setting is RAIO 0. The mode RAIO 1 should **only** be enabled if the 'normal' ray iteration mode fails, which is *very* rarely the case. RAIO 1 does a finer search and also checks for false convergence conditions. For example, in some wide-angle systems, it may be advisable to switch to RAIO 1. Use this switch with care! This setting is saved with the prescription data.
`OMN MIN|MAX angle_deg`	Specifies minimum (MIN) and maximum (MAX) angles in degrees at which rays can be launched in the omni-directional ray aiming mode. Requires that `RAIM OMN` is set, otherwise this command has no effect. Examples: `OMN MIN -80` ! minimum omni-directional angle is -80° `OMN MAX 130 !` maximum omni-directional angle is 130°

The ray aiming mode determines the generation of the start rays in the object space. By default, ray aiming is performed for all wavelengths in use. Because ray aiming for all wavelengths is time consuming, an option to confine ray aiming to the reference wavelength is given in the configuration dialog. Select *Edit - Configuration* from the main menu. In the *Aperture* tab, disable the check box "Ray aiming at ALL wavelengths". Ray aiming is then performed at the reference wavelength only.

Currently there are four modes available to define start rays from an object point towards the pupil of a system:

7.4.1 `ENP`: Paraxial entrance pupil mode:

Rays are aimed to the *paraxial* entrance pupil. This mode does not account for pupil aberrations and is independent on tilted and decentered surfaces in the system. Since only paraxial quantities are used, it is the fasted mode. However, paraxial ray aiming may fail in systems with noticeable pupil aberrations, such as in wide angle systems or systems with large numerical aperture. If this occurs, use the `STO` ray aiming method described in the next section.

7.4.2 `STO`: Stop Surface Mode

Rays are aimed to the physical boundaries of the stop surface, independent of its shape (circular, elliptical, rectangular, etc.). This is an iterative process and therefore consumes more time. It also takes tilted and decentered surfaces and apertures into account, as well as vignetting caused by undersized surface apertures.
The effect of ray aiming mode "`STO`" is neatly observed with wide-angle lenses which exhibit strong pupil distortion. Fig.7.8 gives an example of this effect. If rays are aimed to the paraxial entrance pupil, i.e. `RAIM ENP`, they will not hit the real stop surface at all for some field angles. This is due to the fact that the axial position of the entrance pupil varies strongly with field angle. Since paraxial quantities do not account for field dependent effects, solely aiming to the *paraxial* entrance pupil will

fail in most wide-angle systems.

Therefore, in using `RAIM STO`, the correct start coordinates of the rays are exactly traced in an iterative process, such that size and position of the stop are always exactly found.

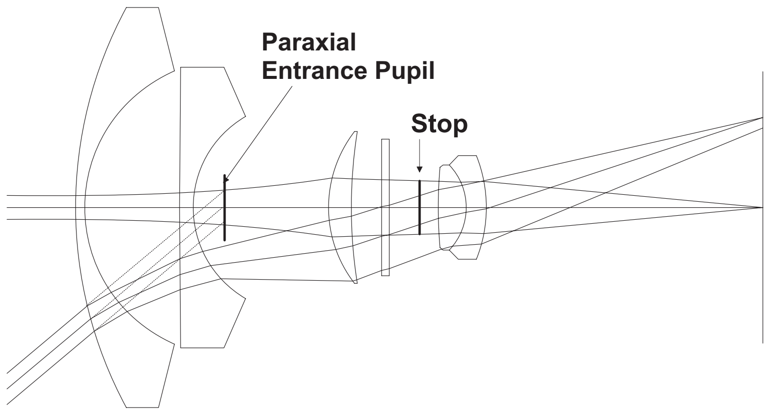


Figure 7.8: Ray aiming methods. Rays aiming to the paraxial entrance pupil (`RAIM ENP`) will not hit the stop surface at the corresponding coordinates. `RAIM STO` takes account for pupil aberrations in centered and decentered systems by iterating for the exact start coordinates.

7.4.3 `TEL`: Telecentric Mode

Systems having an infinitely distant entrance pupil are best modelled in the telecentric mode. The initial direction of chief rays in the object space is always parallel to the optical axis. The telecentric mode requires systems with a *finite* object distance and the angular subtense of the beam emerging the object must be defined by the numerical aperture (see `NAO` command).

Note, that telecentric beams do not necessarily go through the center of the stop. Since the stop surface is always limiting the beams (independent of the `FHY` setting on the stop surface), it may be likely that the stop surface truncates the beams in an unwanted manner. The aperture dimensions of the stop should be appropriately oversized if such effects are not wanted.

7.4.4 `OMN`: Omni-directional Mode

In some systems it is necessary to launch rays into arbitrary directions, irrespective of stop position or definition of the system aperture (such as NA, EPD, FNO, etc). This can be a valuable option, for example in condensor systems or illumination systems in which sources irradiate into the full 4π angular space.

For example, Fig. 7.9 shows an elliptical reflector where rays are launched from a point object at angles greater than $\pm 90°$, i.e. rays also exit the source in opposite direction to the positive Z-axis. This is normally not possible with the standard ray aiming (generation) methodsENP, STO, and TEL as described above.

The only parameters required for defining an omni-directional beam are the minimum and maximum angles (referred to the global coordinate system) at which rays can be launched from a point source. Fig. 7.10 illustrates an arbitrary condition. The allowable range of minimum and maximum source ray angles is from $0°$ to $\pm 180°$.

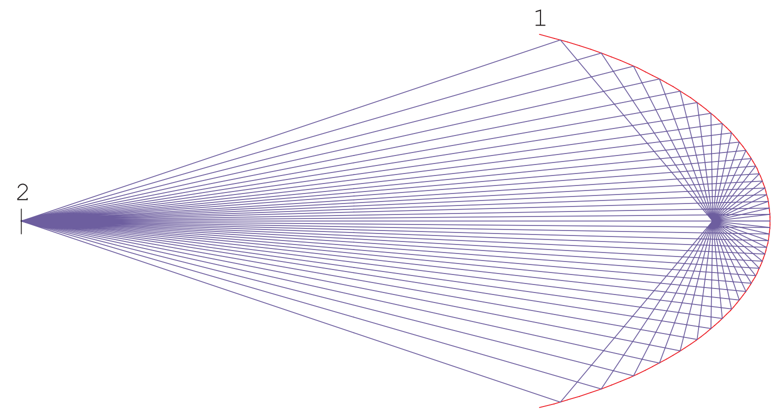


Figure 7.9: Example of omni-directional ray aiming. See examples directory `\examples\mirror\ellipsoid_1.otx`

In omni-directional mode, rays are generated such that their intersections with a sphere are equidistant, like with the degrees of longitude and latitude on the globe. This imposes some difficulties with some kinds of analysis plots. For example the results of ray intersection plots or illumination plots are always referred to the tangent plane at a given surface. Since it is impossible to convert a coordinate system based on spherical coordinates to a plane, distortion of a regular ray grid emitted from a point source is always distorted on a plane.

7.5 Afocal Systems

In an afocal system the principal points and focal points are at infinity, which does not imply that the object and image are at infinity. This condition requires special procedures to be used in ray tracing because tracing to infinity would create numerical problems. We will distinguish between afocal in the object space and afocal in the image space. While *afocal in the object space* is quite normal in many systems, *afocal in image space* is handled by *angular* ray aberrations instead of transverse ray

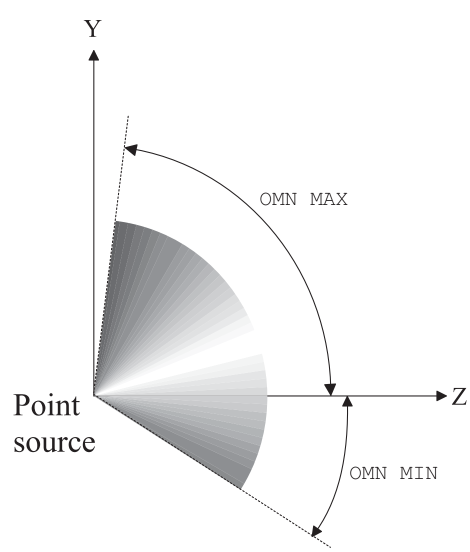


Figure 7.10: Definition of beam angles emitted from a point source in omni-directional ray aiming.

aberrations in a finite image plane. To illustrate the concept of angular measures, we will consider a simple Fraunhofer-type telescope as shown in Fig. ...

A rim ray exits the system at an angle α to the optical axis due to inherent aberrations in the system. Since the image is assumed at infinity (afocal in image space), the transverse aberration of the ray would also be infinity. At this point we will introduce the concept of a virtual "ideal" lens, which is placed at the exit of the system and helps us to convert the angular aberration of the ray to a finite measure. For simplicity, the focal length of the ideal lens is assumed 1000mm, thus converting an angle $\alpha = 1mrad$ to a transverse aberration $y' = 1mm$.

The beauty of the "ideal lens" concept is, that we do not need to leave our world of transverse aberrations. If the system is afocal in image space, 1mm aberration in the focal plane of the assumed "ideal" lens corresponds to 1mrad angular ray deviation.

If the system is afocal (in image space), $OpTaliX$ automatically does this conversion internally. It is not necessary to add an ideal lens after the optical system. The only command required to make a system afocal is

```
AFO yes
```

irrespective whether the focus is actually at infinity or not. All performance analyses (Spot, Fan, MTF, PSF, etc.) will then be given in angular aberrations (mrad) instead of transverse aberration (mm).

Optical path differences (OPD) will be referred to a plane wave in the exit pupil of the system. Since the focal length of the (internally used) ideal lens is always 1000mm, field sags are reported in diopters.

7.6 Vignetting

Vignetting in optical systems is defined by the shape and dimensions of the stop surface and by hard limiting (fixed) apertures on other surfaces using the FHY command. There can be as many fixed apertures as there are surfaces in the optical system. Fixed apertures are indicated in the system listing (see LIS command) by an asterisk (*) character immediately following the aperture value.

`SET VIG`	Calculates vignetting factors `VUX, VLX, VUY, VLY` in accordance to the setting of fixed (hard limiting) surface apertures. Included for Code V compatibility. See also notes below.
`DEL VIG [fi..j]`	Delete vignetting factors for fields `i` to `j`.

For related commands, SET MHT and FHY see section 8.32.3 on page 163.

Notes on SET VIG Command:

Modelling of ray bundles in $OpTaliX$ is solely based on hard-limiting (fixed) apertures on surfaces. Even though vignetting factors can be evaluated (SET VIG), they are reported for information only and do not have any impact on size and shape of light beams.

Since light beams are always calculated using real apertures, there is no risk of inconsistency and $OpTaliX$ will always calculate the correct beam. In particular, rays shown in the lens layout plot actually represent the beam limits used for all performance analysis options.

A typical output of the SET VIG command is as follows:

```
VIGNETTING FACTORS:

Field        VUX        VLX        VUY        VLY      UX   LX   UY   LY
```

```
  1    -0.00011    -0.00011    -0.00011    -0.00011       6    6    6    6
  2    -0.00002    -0.00002    -0.00003    -0.00010       6    6    6    6
  3     0.00043     0.00043     0.17753     0.13093       6    6   11    1
```

Vignetting factors are given for each field separately. The UX, LX, UY, LY columns denote the surfaces which limit the beam. On the example given above, at field 3, surface 11 limits the upper Y-portion (UY) of the beam whereas surface 1 limits the lower Y-portion of the beam.

8

Surface Data

Surface data include the typical lens prescription items such as radius of curvature, thickness (axial separation), glasses, etc. The numbering sequence starts with 0 for the object surface. The first surface of the optical system is surface 1 and, in a normal (sequential) system, the surface numbers increase monotonically in the order that rays strike them.

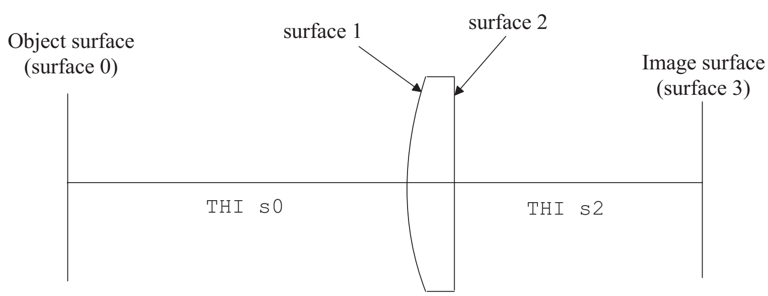


Figure 8.1: Surface numbering.

Note that in systems with reflectors, the thickness is usually negative to the next surface. This is because after a reflector, the next surface in the optical path is usually located in the negative Z direction from the reflecting surface. Thus, the thickness associated with a surface should not be thought of as an optical distance, but rather as what is the location on the Z axis of the next surface relative to that of the current surface.

The thickness associated with the image surface (`THI SI`) is unique. The actual image distance from the surface prior to the image surface (`SI-1`) to the image surface (`SI`) is the sum of the paraxial image distance and defocus term (`THI SI`). This is to accommodate the use of a paraxial image solve (`PIM`) plus a defocusing term. If the paraxial image solve is not used, the image surface thickness (`THI SI`) is automatically updated to show the difference to the paraxial focus.

There are two ways to enter and modify surface data. The first is the surface spreadsheet editor, which can be invoked from the *Edit –> Surfaces* menu or from the appropriate toolbar icon. The second is from the command line, which exists twice, under the main menu and as a floating dialog that can be placed anywhere on the screen.

8.1 Surface Editor

The surface editor is a tabbed dialog which contains several spreadsheets for editing surface parameter from the graphical user interface (GUI). This allows entering surface (prescription) parameters solely from the GUI, as an alternative to entering data on the command line (sect. 6). The surface editor is invoked from the main menu *Edit –> Surface Data* or by clicking on the  icon in the toolbar or by entering `EDI SUR` in the command line. The surface parameter are grouped in several tabs as shown in Fig. 8.2):

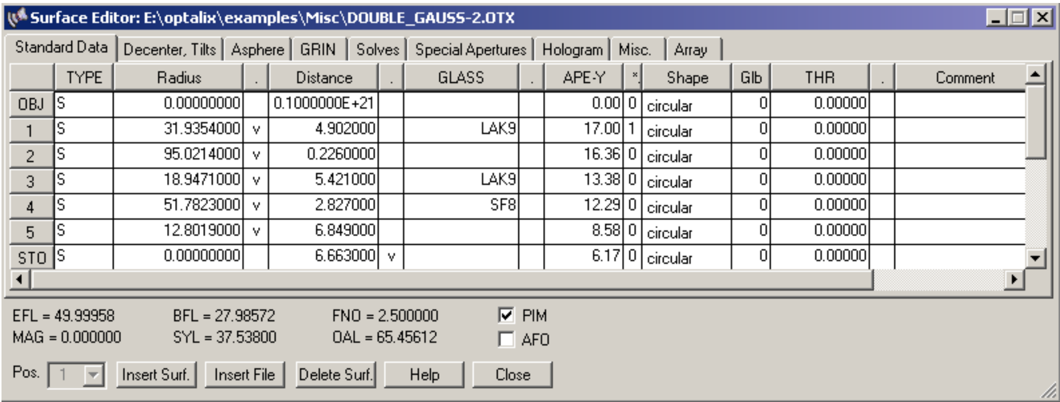


Figure 8.2: Surface spreadsheet editor, invoked by the command `EDI SUR`.

8.2 Surface Parameters

`S rad thi gla`	Shorthand entry, inserts a new surface at the current surface pointer. See also section 8.4 for a detailed explanation.
`ASP [si..j]`	defines aspheric surface
`SPH [si..j]`	defines spherical surface
`NOR [si..j]`	defines "no-raytrace" surface
`K [si..j] value`	conic constant
`A [si..j] value`	4^{th} order aspheric constant as defined in equation 8.1.
`B [si..j] value`	6^{th} order aspheric constant as defined in equation 8.1.
`C [si..j] value`	8^{th} order aspheric constant as defined in equation 8.1.
`D [si..j] value`	10^{th} order aspheric constant as defined in equation 8.1.
`E [si..j] value`	12^{th} order aspheric constant as defined in equation 8.1.
`F [si..j] value`	14^{th} order aspheric constant as defined in equation 8.1.
`G [si..j] value`	16^{th} order aspheric constant as defined in equation 8.1.
`H [si..j] value`	18^{th} order aspheric constant as defined in equation 8.1.
`CON [si..j]`	defines conic surface
`TOR`	defines toric surface
`STO si` `STO i`	Makes surface `i` stop surface. The "s" qualifier is not mandatory. The following examples are equally valid: `STO s3`, `STO 3`

continued on next page

continued from previous page	
SUT [si..j] ABCD	Surface type defined by a string, up to 6-characters long "cccccc". Examples: SUT s1 AD : surface 1 is aspheric and decentered, SUT s2..3 si : surfaces 2 to 3 are spherical and gradient index. See also the list on available surface type qualifiers below (page 65).
CPI si..j sx	Curvature pickup. Pick surfaces si..j to surface sx. A negative sign for sx picks the surface with opposite curvature. Example: CPI s5 -3 : curvature 5 is picked from surface 3 with opposite sign.
DPI si..j sx	Distance pickup. Pick surfaces si..j to surface sx. A negative sign for sx picks the surface with opposite distance. Example: DPI s5 -3 : distance 5 is picked from surface 3 with opposite sign.
MPI si..j sx	Material pickup. The material properties of surface sx are picked up (copied) to surfaces si..j.
TPI si..j sx	Tilt and decenter pickup. The tilt and/or decenter values are picked up from surfaces si..j. Thus, surfaces si..j are tilted/decentered by the same amount than surface sx.
TPF si factor	Tilt/decenter pick-up factor. If factor is not 1.0, picked values for tilts and decenters will be multiplied by factor
CUX [si..j] curvature_x	Curvature in X/Z plane. This parameter is effective only for toric surfaces and requires the surface type "A" (aspheric).
CUY [si..j] curvature_y	Curvature in Y/Z plane. This is the default for spherical surfaces. See also the command RDY which specifies the radius instead of curvature.
CIY [si..j] curvature_incr	Increment Y-curvature (CUY) immediately. Convenient for a power change to an unknown curvature value.
RDX [si..j] radius_x	Radius in X/Z plane. This parameter is effective only for toric surfaces and requires the surface type "A" (aspheric).
RDY [si..j] curvature_y	Radius in Y/Z plane. This is the default for spherical surfaces. See also the command CUY which specifies curvature instead of radius. Note: A radius value of 0 is not physically possible, and is therefore interpreted as a curvature of 0 (a flat surface).
THI [si..j] [zi..j|zk] thickness	Axial thickness (separation) from actual surface vertex to subsequent surface.
TIN [si..j] thickness_incr	Increment distance (THI) immediately. Convenient for a change to an unknown thickness value.
THM [si..j|sk] mirr_thickness	Center thickness to back surface of first-surface mirror at surface sk respectively surfaces si..j. Value is always positive.
	continued on next page

continued from previous page	
THR [si..j] reference_thickness	Axial separation of surface(s) i..j to "referenced" surface. Used in conjunction with global referencing. This command must not be confused with THI (axial thickness). THR is referred to a *preceding* surface whereas THI always refers to the *subsequent* surface. Thus, a referencing surface can have both THI and THR parameters. See also section 8.21 for a detailed explanation of the concept of global referencing. Note: Specify the referenced surface by the command GLB si..j k
GRO [si..j] ival	Grating order, an integer value. This command is obsolete, HOR should be used instead.
HOR [si..j] ival	Hologram diffraction order, an integer value.
GRX [si..j] grating_freq_x	Grating frequency in grooves/mm (grooves parallel to X-axis)
GRY [si..j] grating_freq_y	Grating frequency in grooves/mm (grooves parallel to Y-axis)
NSS [si..j]	Make the surface(s) si..j non-sequential.
MXH [si..j] n_hits	Maximum number of allowable ray hits at non-sequential surface (default : n_hits = 10)
REFL [si..j]	Reflect all rays (mirror surface).
REFR [si..j]	Refract all rays. Total internal reflection (TIR) is a failure.
TIR [si..j]	Total internal reflection. This surface acts like a mirror surface (REFL) except that rays that do not satisfy TIR condition are reported as failure.
RMD [si..j] REFR|REFL|TIR	Refractive/reflective mode. Available modes are REFR = refract all rays at surface(s) si..j = default mode. REFL = reflect all rays at surface(s) si..j TIR = only reflects rays obeying TIR condition This command complements the explicit commands REFR, REFL and TIR as given above.
MFL si module_efl	Module focal length. si is the first surface of the module range.
SPG [si..j|sk] spec_gravity	Specific gravity in g/cm^3. Value is taken from glass catalogue but may be overwritten by the SPG command.

8.3 Infinity Values

Because infinity values cannot be accurately represented in computers, the following conventions apply:

Distances, Separations: Any distance greater than 10^{10} is considered as an infinite value. This convention particularly applies to object distances at infinity. Make sure that the object distance (THI s0) is $> 10^{10}$ to ensure infinitely distant objects.

Radius, Curvature: Any radius greater 10^{10} is considered as infinity, that is, the surface is assumed perfectly flat. A special case is the surface radius 0, for example RDY sk 0. This command automatically defaults to a flat surface with infinite surface radius (curvature = 0, i.e. CUY sk 0).

8.4 Surface Shorthand Entry

A shorthand entry of a spherical surface is obtained by the command:

`S rad_curv thickness glassname`

where

`rad_curv`	is the radius or curvature in Y-direction. Radius or curvature entry is defined by the `RDM` command (see section 7.1 page 41),
`thickness`	is the axial separation right of the surface vertex
`glassname`	is the glass manufacturer's designation

The default surface type on surface shorthand entry is spherical.

8.5 Surface Type

Surface types are characterized by six-character strings which are assigned to each surface. The surface type is defined by the following command:

`SUT si..j cccccc`

where `cccccc` is an arbitrary sequence of surface descriptors (a character). Surface types are categorized into obligatory and optional ones, according to table8.5.

One of the obligatory surface types (”A”, ”S”, ”X”, ”U” or ”L”) must always be specified. ”A” and ”S” describe the base surface (aspheric or spherical). Surface type ”L” (lens module) does not specify a base surface, since it only has transformational properties. ”L” is also an exception of the rule, because no optional surface types are allowed in addition to the ”L” character.

Optional surface descriptors may be arbitrarily combined in order to build complex surfaces. For example,

`SUT s1..3 DAM`	sets the surface type of the surfaces 1 - 3 to D = decentered, A = aspheric, M = mirror

The order of surface type qualifiers does not matter, i.e.

```
SUT s1..3 DAM
SUT s1..3 AMD
SUT s1..3 MDA
```

are equivalent.

Note: Gradient index surfaces and step index fibers require two qualifiers, one to define the surface type and a second one for the material properties (GRIN or step index). For example,

`SI`	denotes a spherical surface with gradient index material attached,
`SP`	is a spherical surface with step index properties.

8.6 Aspheric Surfaces

Aspheric surfaces are commonly defined by polynomial expressions in one dimension which are then rotated about the local Z-axis to form the surface. The following types of polynomial aspheres are available:

- even power polynomial asphere, up to 18^{th} order,
- odd power polynomial asphere, up to 9^{th} order,
- odd power special polynomial asphere, up to 30^{th} order,
- XY polynomial surface, up to 10^{th} order.
- anamorphic (biconic) surface, up to 10^{th} order,
- toroidal surface,
- cylindrical surface.
- Qcon polynomial
- Qbf polynomial

Aspheric surfaces are defined by a type designator command `ASP sk` or by changing thesurface type to "A". The surface form is further defined by coefficients of various types.

Aspheric surfaces command overview:

`ASP si..j|sk EVEN|ODD9|ODD30|XYP|AAS|CYL|QCN|QBF`	Converts surface(s) `si..j|sk` to type aspheric. Any corresponding coefficients are appropriately converted. A warning message is issued if the order of coefficients does not match. For example, an `ASP EVEN` type asphere can be converted to an `ASP ODD30` asphere, whereas the inverse conversion (`ASP ODD30` to `ASP EVEN`) may result in loss of coefficients because odd power coefficients cannot be modelled in the `ASP EVEN` type surface. See also the `ATY` command below.
`ATY si..j|sk EVEN|ODD9|ODD30|XYP|AAS|CYL|QCN|QBF`	Only changes asphere type without converting coefficients. The type of of higher order polynomials is defined as: `EVEN` = only even power polynomial according to Eq. 8.1, `ODD9` = mixed odd and even powers according to Eq. 8.5. `ODD30` = extended odd and even powers, `XYP` = XY polynomial up to 10^{th} order, `AAS` = anamorphic asphere (biconic in absence of higher-order coefficients). `CYL` = cylindrical surface. Note that the coefficients for the `even` and `odd9` asphere types are entered by the `A,B,C,D,E,F,G,H` commands, whereas the coefficients for the `ODD30` and `XYP` asphere types **must** be entered using the `SCO` command (see below). Alternatively, a dialog-based entry is provided by the `EDI SPS` command.
Code V compatibility commands	
	continued on next page

continued from previous page	
`SPS ODD|XYP|QCN|QBF si..j|sk`	Change surface profile to ODD, XYP, QCN or QBF special aspheric surface. Automatically sets surface type to ”A” and asphere type according to the equivalences `ATY odd30` for `SPS ODD` `ATY xyp` for `SPS XYP`. `SPS` surface profile is determined by the curvature (`RDY` or `CUY`) and the `SCO` coefficients. If the surface is changed from an aspheric surface of kind ”EVEN” or ”ODD9” to an SPS surface, then any corresponding surface parameters are retained and stored in the appropriate SCO coefficients. All other SCO coefficients are set to zero.
`SCO si..j|sk ci coefficient`	Coefficients for describing the `SPS ODD|XYP|QCN|QBF` surface(s) `si..j|sk`. The coefficients differ in meaning for each `ODD|XYP` type as described in sections 8.6.4 and 8.6.5 respectively.
`YTO si..j|sk`	Defines a Y-toroid. The surface can be an ODD9 or EVEN power asphere in the Y-plane but is always assumed spherical in the X-plane. The Y-toroid degenerates to a sphere for `CUX` = `CUY` (respectively `CUX` = 0) and K = A = B = C = D = E = F = G = H = 0.
`CYL si..j|sk`	Defines a cylinder. For details see sect. 8.6.7 (page 74).
`IC sk|si..j Yes|No`	Intersection direction. As there may be more than one intersection of a ray with a surface, this option allows choosing the alternate intersection point from the one normally used. This option is normally not needed except when rays are at high angle to the local surface axis. `IC Yes` = default, `IC No` = selects alternate intersection point. In the surface editor, `IC` can be set in the ”Misc” tab. See also the notes on alternate intersection points in sect. 8.7.

Note that aspheric surfaces always require the surface type (SUT) ”A”, which must replace the surface types ”S”, ”L”, ”U” or ”X”. For example, simultaneous specification of surface types ”SA”, ”LA” or ”XA” is not permitted. See also a detailed description of surface types in section8.5 on page 65.

8.6.1 ”EVEN” Power Asphere

The ”EVEN” power polynomial aspheric surface is defined as

$$z = \frac{ch^2}{1+\sqrt{1-(K+1)c^2h^2}} + A\cdot h^4 + B\cdot h^6 + C\cdot h^8 + D\cdot h^{10} + E\cdot h^{12} + F\cdot h^{14} + G\cdot h^{16} + H\cdot h^{18} \quad (8.1)$$

where:

c	= vertex curvature (in mm^-1)
K	= conic constant
A, B, C, D, E, F, G, H	= asph. coefficients
h^2	= $x^2 + y^2$ (in mm)
x,y	= surface coordinates (in mm)

The EVEN power asphere is a rotationally symmetric surface, that is, the conic/polynomial profile defined in Eq. 8.1 is rotated about the local Z-axis.

The conic constant K describes surfaces of conic sections:

A different variant of equation 8.1 is occasionally in use:

$$z = \rho h^2 / \left(1 + \sqrt{1 + (1 - \kappa \rho^2 h^2)}\right) + A \cdots h^4 + B \cdots h^6 + \cdots \tag{8.2}$$

Since both, K and κ, are termed conic constants and both equations are of similar form, they can be easily confused. For the sake of clarity, equation 8.1 is used consistently in $OpTaliX$.

The numerical eccentricity ε and the conic constant k are then related by:

$$K = -\varepsilon^2 \quad \text{ellipse at major axis} \tag{8.3}$$

$$\frac{K}{K+1} = \varepsilon^2 \quad \text{ellipse at minor axis} \tag{8.4}$$

Equation 8.3 is also valid for a hyperbola.

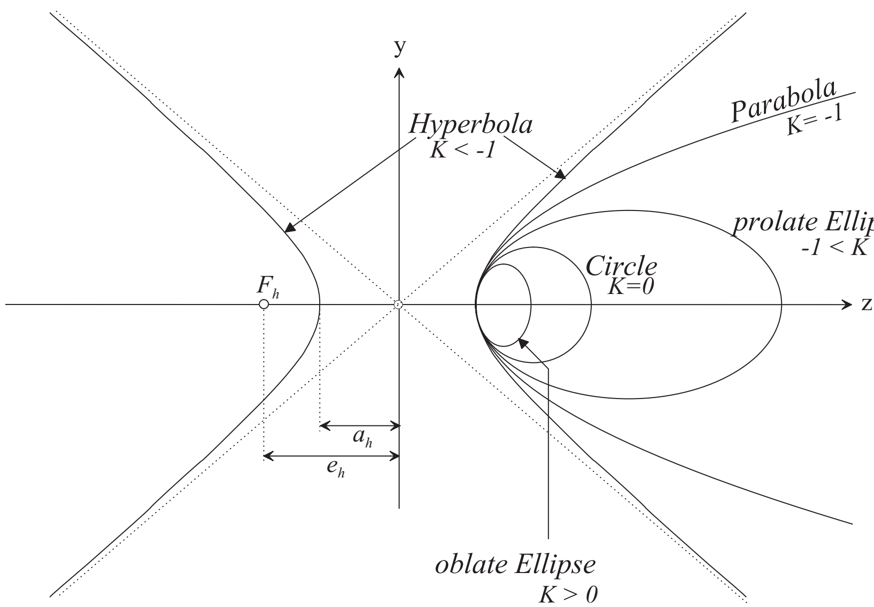


Figure 8.3: Conic sections of aspheric surfaces.

8.6.2 ”ODD9” Power Asphere

The difference between this surface and the ”EVEN” power polynomial asphere defined in the previous section is the form of the expansion polynomial, which includes both the odd and even powers of radial distance up to 9^{th} order. In addition, the terms start at power 2 instead at power 4.

$$z = \frac{ch^2}{1 + \sqrt{1 - (K+1)c^2 h^2}} + A \cdot h^2 + B \cdot h^3 + C \cdot h^4 + D \cdot h^5 + E \cdot h^6 + F \cdot h^7 + G \cdot h^8 + H \cdot h^9 \tag{8.5}$$

The $A \cdot h^2$ term is taken into account in paraxial calculations. The quadratic term describes a parabola with vertex curvature $2 \cdot A$. Thus, the effective curvature used in paraxial analysis is $c = c_0 + 2 \cdot A$.

The ODD power asphere is a rotationally symmetric surface, that is, the conic/polynomial profile defined in Eq. 8.5 is rotated about the local Z-axis.

8.6.3 Ellipse at major or minor Axis in the EVEN and ODD9 Asphere Models

The terminology ”*ellipse at major respectively minor axis*” as used in the previous sections often leads to confusion. The EVEN and ODD9 asphere surfaces are primarily *rotationally symmetric* surfaces, if we assume c_x = `CUX` = 0 (special case of toric surface). That is, the surface is generated by rotating a 2-dimensional curve (conic or polynomial) in the Y/Z-plane about the local Z-axis.

This concept is important to understanding how elliptical surfaces are formed in the `EVEN` and `ODD9` asphere models. Eqs. 8.1 and 8.5 only define the sag in the Y/Z-plane. Rotating these curves about the local Z-axis describes an ellipsoid for $-1 < K < 0$ (ellipse at major axis), however, it does NOT for elliptical sections at the minor axis ($K > 0$).

Figures 8.4 and 8.5 illustrate the difference.

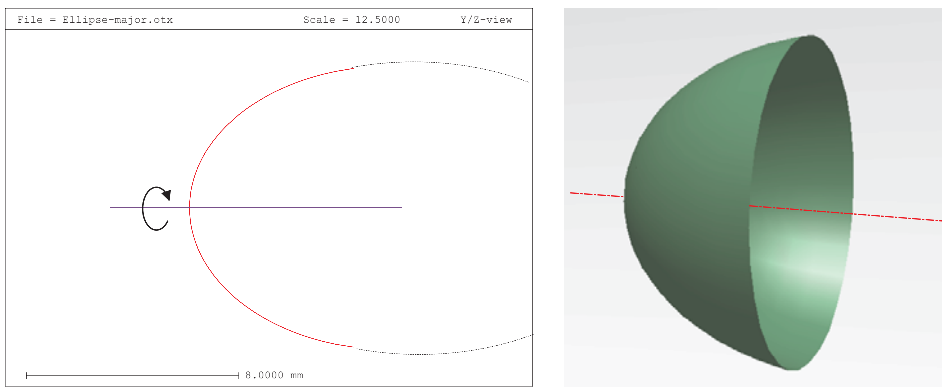


Figure 8.4: Definition of an elliptical section at the major axis ($-1 < K < 0$). Left: Section of the ellipse. Right: Perspective view showing the resulting surface.

Figure 8.5: Definition of an elliptical section at the minor axis ($K > 0$). Left: Section of the ellipse. Right: Perspective view showing the resulting surface.

Thus, an elliptical section defined at the minor axis does not describe a ”true” ellipsoid with its minor axis aligned with the local Z-axis. If you need to model a true ellipsoid aligned at the minor axis, use the anamorphic (biconic) surface model as described in section 8.6.6.

8.6.4 ”ODD30” Power Asphere

The ”ODD30” asphere is an extension of the ”ODD9” surface to 30^{th} order including both odd and even powers of radial distance. It is a purely rotationally symmetric surface. Due to the larger number of coefficients accepted, it is handled as a *special* aspheric surface respectively `SPS` in the Code V lingo. Basically, a *special* surface (`SPS`) is handled like a ”user defined surface” (`UDS`) because it uses the same domain of coefficients. The only difference between the two variants is that special surface coefficients are entered by the `SCO` command and user defined surface coefficients are entered by the `UCO` command. User defined surfaces and special surfaces are distinguished by the surface type

- `A` for special surfaces (of kind EVEN, ODD9, ODD30, XYP)
- `U` for user defined surfaces

$$z = \frac{ch^2}{1+\sqrt{1-(K+1)c^2h^2}} + C_2 \cdot h + C_3 \cdot h^2 + C_4 \cdot h^3 + C_5 \cdot h^4 + C_6 \cdot h^5 + \cdots + C_{31} \cdot h^{30} \quad (8.6)$$

where:
- c = vertex curvature (in mm^-1)
- K = conic constant
- C_i = coefficient of h^{i-1}, for $2 \leq i \leq 31$
- h^2 = $x^2 + y^2$ (in mm)
- x,y = surface coordinates (in mm)

If all C_i coefficients are zero (the default), a pure conic surface results. The maximum number of terms to use in the expansion can be specified with coefficient C_{32} (C32) in order to speed up computation. If C32 is 0, then all 31 coefficients are used.

The table below gives the coefficient numbers for the surface parameters of the `ATY ODD30` asphere type (use alternatively `SPS ODD` command).

Coefficient	Definition
C1	Conic constant
C2	1^{st} order aspheric coefficient
C3	2^{nd} order aspheric coefficient
C4	3^{rd} order aspheric coefficient
C5	4^{th} order aspheric coefficient
C6	5^{th} order aspheric coefficient
C7	6^{th} order aspheric coefficient
C8	7^{th} order aspheric coefficient
C9	8^{th} order aspheric coefficient
C10	9^{th} order aspheric coefficient
C11	10^{th} order aspheric coefficient
C12	11^{th} order aspheric coefficient
C13	12^{th} order aspheric coefficient
C14	13^{th} order aspheric coefficient
C15	14^{th} order aspheric coefficient
⋮	⋮
C31	30^{th} order aspheric coefficient
C32	Number of terms to use in the expansion

Entering coefficients C1 to C32 is accomplished by the `SCO` command explained on page79.

In the surface editor the `SPS ODD` surface is selected from the 'Asph.Type' column in the 'Asphere' tab. Use the pull-down menu to define the proper asphere type, as shown in Fig. 8.6.

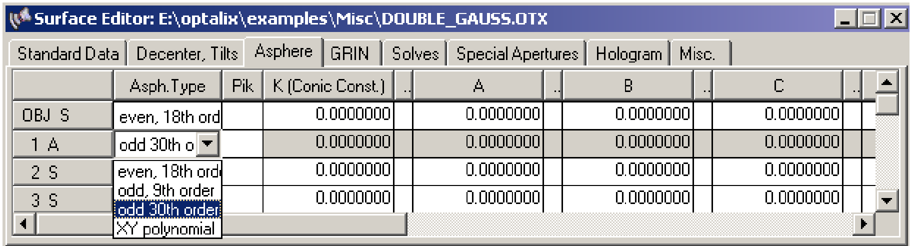


Figure 8.6: Defining SPS ODD aspheric surfaces.

Note that the K, A, B, C, ... columns are greyed out as they have no meaning for `SPS ODD` surfaces. Instead, invoke the SPS/UDS editor to edit ODD/ODD30 coefficients. This is performed from the main menu *Edit -> SPS/UDS Coefficients* or from the command line by entering `EDI UDS`

8.6.5 "XY" Polynomial Asphere

The XY polynomial asphere is a 10^{th} order polynomial surface added to a base conic. The polynomial is expanded into monomials of $x^m y^n$, where $m + n \leq 10$. The equation is

$$z = \frac{ch^2}{1 + \sqrt{1 - (K+1)c^2h^2}} + \sum_{i=2}^{66} C_i x^m y^n \qquad (8.7)$$

where:
- c = vertex curvature (in mm^-1)
- K = conic constant
- C_i = coefficient of the monomial $x^m y^n$
- h^2 = $x^2 + y^2$ (in mm)
- x,y = surface coordinates (in mm)

The maximum number of terms used in the expansion can be specified with C67, which speeds up computation. If C67 is 0, all 66 terms are used.

Coefficient	Definition	Coefficient	Definition
C1	Conic constant	C34	x^2y^5
C2	x	C35	xy^6
C3	y	C36	y^7
C4	x^2	C37	x^8
C5	xy	C38	x^7y
C6	y^2	C39	x^6y^2
C7	x^3	C40	x^5y^3
C8	x^2y	C41	x^4y^4
C9	xy^2	C42	x^3y^5
C10	y^3	C43	x^2y^6
C11	x^4	C44	xy^7
C12	x^3y	C45	y^8
C13	x^2y^2	C46	x^9

continued on next page

C14	xy^3	C47	x^8y
C15	y^4	C48	x^7y^2
C16	x^5	C49	x^6y^3
C17	x^4y	C50	x^5y^4
C18	x^3y^2	C51	x^4y^5
C19	x^2y^3	C52	x^3y^6
C20	xy^4	C53	x^2y^7
C21	y^5	C54	xy^8
C22	x^6	C55	y^9
C23	x^5y	C56	x^{10}
C24	x^4y^2	C57	x^9y
C25	x^3y^3	C58	x^8y^2
C26	x^2y^4	C59	x^7y^3
C27	xy^5	C60	x^6y^4
C28	y^6	C61	x^5y^5
C29	x^7	C62	x^4y^6
C30	x^6y	C63	x^3y^7
C31	x^5y^2	C64	x^2y^8
C32	x^4y^3	C65	xy^9
C33	x^3y^4	C66	y^{10}
		C67	Number of terms

Entering coefficients C1 to C67 is accomplished by the `SCO` command explained on page 79.

In the surface editor the `SPS XYP` surface is selected from the 'Asph.Type' column in the 'Asphere' tab. Use the pull-down menu to define the proper asphere type, as shown in Fig. 8.7.

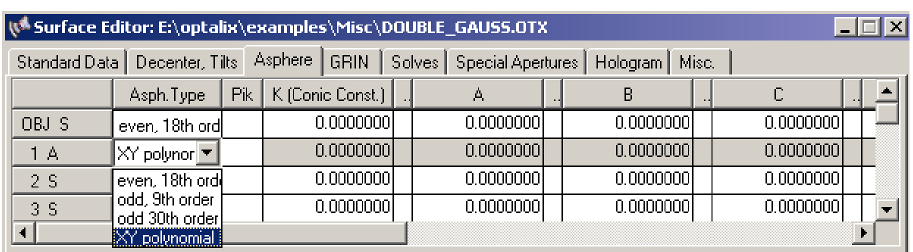


Figure 8.7: Defining SPS ODD or SPS XYP aspheric surfaces. Note that the K, A, B, C, ... coefficients are greyed out and cannot be edited in the surface editor. For editing `SPS ODD` or `SPS XYP` coefficients, use the `EDI UDS` command.

Note that the K, A, B, C, ... columns are greyed out as they have no meaning for `SPS ODD` or `SPS XYP` surfaces. Instead, invoke the SPS/UDS editor to edit XYP coefficients. This is performed from the main menu *Edit -> SPS/UDS Coefficients* or from the command line by entering `EDI UDS`

8.6.6 Anamorphic (Biconic) Asphere

The anamorphic asphere surface exhibits bilateral symmetry in both sections X and Y. The equation is:

$$
\begin{aligned}
z = & \frac{c_x x^2 + c_y y^2}{1 + \sqrt{1 - (1+K_x)c_x^2 x^2 - (1+K_y)c_y^2 y^2}} \\
& + A_R \left[(1 - A_P)x^2 + (1 + A_P)y^2\right]^2 \\
& + B_R \left[(1 - B_P)x^2 + (1 + B_P)y^2\right]^3 \\
& + C_R \left[(1 - C_P)x^2 + (1 + C_P)y^2\right]^4 \\
& + D_R \left[(1 - D_P)x^2 + (1 + D_P)y^2\right]^5
\end{aligned}
\tag{8.8}
$$

where:

Variable	**Command**	**Description**
z	SAG	the sag of the surface at the local surface coordinates
c_x, c_y	CUX, CUY	the curvatures in X and Y
K_x, K_y	KX, KY	conic constants in X and Y. The definition of K_y is equivalent to the conic constant K as given in table 8.4 (page 169).
A_R	AR	rotationally symmetric coefficient, $4^{th} order$
B_R	BR	rotationally symmetric coefficient, $6^{th} order$
C_R	CR	rotationally symmetric coefficient, $8^{th} order$
D_R	DR	rotationally symmetric coefficient, $10^{th} order$
A_P	AP	non-rotationally symmetric coefficient, $4^{th} order$
B_P	BP	non-rotationally symmetric coefficient, $6^{th} order$
C_P	CP	non-rotationally symmetric coefficient, $8^{th} order$
D_P	DP	non-rotationally symmetric coefficient, $10^{th} order$

Note that the anamorphic surface reduces to the standard EVEN power asphere (see sect. 8.6.1) when

Variables	Commands
$c_x = c_y$	CUX = CUY
$k_x = k_y$	KX = KY
$A_P = B_P = C_P = D_P = 0$	AP = BP = CP = DP = 0

Commands:

`AAS sk|si..j`	Specifies anamorphic asphere. Parameters are X-Curvature/X-Radius (CUX/RDX), Y-Curvature/Y-Radius (CUY/RDY), X-conic constant (KX), Y-conic constant (KY), $4^{th} - 10^{th}$ order rotationally symmetric coefficients (AR, BR, CR, DR), $4^{th} - 10^{th}$ order non-rotationally symmetric coefficients (AP, BP, CP, DP).
`ATY sk|si..j AAP`	as above, sets asphere type (ATY) to anamorphic asphere
`KX sk|si..j X_conic_const`	X-conic coefficient

continued on next page

continued from previous page	
KY sk|si..j Y_conic_const	Y-conic coefficient, identical with K
AR sk|si..j coeff	4^{th} order rotational symmetric coefficient
BR sk|si..j coeff	6^{th} order rotational symmetric coefficient
CR sk|si..j coeff	8^{th} order rotational symmetric coefficient
DR sk|si..j coeff	10^{th} order rotational symmetric coefficient
AP sk|si..j coeff	4^{th} order non-rotational symmetric coefficient
BP sk|si..j coeff	6^{th} order non-rotational symmetric coefficient
CP sk|si..j coeff	8^{th} order non-rotational symmetric coefficient
DP sk|si..j coeff	10^{th} order non-rotational symmetric coefficient

8.6.7 Cylindrical Surfaces

A cylinder surface is defined by `CUX/RDX` or `CUY/RDY`, depending on the orientation of the cylinder. By default, the axis of the cylinder is assumed along the X-axis (that is, `CUY/RDY` $\neq 0$, `CUX/RDX` = 0). For arbitrary orientations of the cylinder axis use γ-rotation (`CDE`).

CYL sk|si..j	Defines cylinder surface. By default, the cylinder axis is assumed along the local X-axis, i.e. `CUY/RDY` $\neq 0$, `CUX/RDX` = 0. The profile in the local Y/Z-section can be a sphere or an EVEN asphere whereas in the local X/Z-plane only spherical sections are allowed (See also toroidal surfaces, page 75 with the cylinder surface as special case). Use γ-rotation (`CDE`) for arbitrary orientation of the cylinder axis.
ASP CYL sk|si..j	As above. Complementary syntax.

Notes:

- Cylinder surfaces may also be defined using the regular EVEN or ODD9 asphere types. In this case, CUX/RDY $\neq 0$ defines a toroidal surface, which, for very large radii (RDX $> 10^{10}$), very well approximates a plane section in X.
- In the Y/Z-section any profile according to the EVEN asphere type (see Eq. 8.1, page 67) is allowed, whereas in the X/Z-section the profile is a straight line. Use γ-rotation (`CDE`) for any other orientation of the cylinder axis.

Examples:

Cylinder axis along X-axis:
```
CYL s1
RDY s1 100
```

Cylinder axis along Y-axis:
```
CYL s1
RDX s1 100
```

Arbitrary cylinder orientation :
```
CYL s1
RDY s1 100
CDE s1 45   ! γ-rotation 45°
```

Notice that cylinder surfaces may also be defined using the regular `EVEN` or `ODD9` asphere types (see sect. 8.6.1 and 8.6.2). In this case, CUX/RDX $\neq 0$ defines a toroidal surface, which, for very large radii (RDX $\geq 10^{10}$), very well approximates a plane section in X.

8.6.8 Toroidal Surfaces

Toroidal surfaces exhibit different radii/curvatures in X- and Y-direction. A toroidal surface is a subset of the general aspheric surface (type EVEN or ODD9, see sections 8.6.1 and 8.6.2) and is distinguished from a rotationally symmetric asphere by a non-zero X-curvature (`CUX` $\neq 0$). Toroidal surfaces must be of surface type "A" (asphere). Commands for entering curvatures in X-plane and Y-plane are:

`CUX`	`si..j`	`curv`	! curvature in X-direction
`RDX`	`si..j`	`radius`	! radius in X-direction
`CUY`	`si..j`	`curv`	! curvature in Y-direction
`RDY`	`si..j`	`radius`	! radius in Y-direction

Toroidal surfaces are described by the following extension to the aspheric equation8.1 :

$$z = F(y) + \frac{c_x}{2}\left(x^2 + z^2 - F(y)^2\right) \tag{8.9}$$

where c_x is the curvature in the X/Z plane and $F(y)$ is equivalent to equation 8.1 respectively 8.5. Equation 8.9 can be transformed to the normal form by:

$$0 = x^2 - \left(F(y)^2 - \frac{2}{c_x}F(y)\right) + z^2 - \frac{2}{c_x}z + \frac{1}{c_x^2} - \frac{1}{c_x^2} \tag{8.10}$$

$$0 = x^2 - \left(F(y) - \frac{1}{c_x}\right)^2 + \left(z - \frac{1}{c_x}\right)^2 \tag{8.11}$$

thus, the toric deformation of the aspheric surface in the X/Z plane can be a sphere only. The aspheric deformations in the Y/Z plane remain as described in equations 8.1 and 8.5.

The cylinder surface is a special case of the toroidal surface with $\rho_x = 10^{-10}$. While the EVEN/ODD surface is more general, there is a special asphere type "CYLINDER" (page 74) which simplifies data input for this special surface/asphere type.

8.6.9 Q-Type Polynomials

Aspheric surfaces using Q-type polynomials as described by G.Forbes[**?**, **?**] offer several advantages over the classical monomials as given in sect. 8.6.1. Major advantages are:

- The coefficients have a physical meaning. In particular, Q-type polynomials for aspheric surfaces have units of length and their value directly expresses their contribution to the surface departure.
- The polynomial terms form a descending series giving a clear indication as to when a coefficient becomes irrelevant.
- Q-type polynomial coefficients can be given meaningful tolerances for the fabricator.

- The aspheric terms are orthogonal (within a normalization radius). Each term is unique and simplifies tolerancing.
- Easier definition of slope constraints for improvement of manufacturability.
- Fewer digits of precision are required. This simplifies the numerical burden for transferring asphere prescription data to optical fabrication.
- Helps to reduce the number of terms.

Two Q-type polynomial descriptions are available:

- The Qbfs (”best fit”) polynomial form is characterized by an RMS slope departure from a best-fit sphere. The RMS slope of the departure provides a sensible metric of the testability of the surface. It can easily be calculated from the Qbfs coefficients, and it is proportional to mean fringe density. Typically it is intended for use with ”mild” aspheres.
- The Qcon (”conic”) form is characterized by the sag departure from a base conic.

8.6.10 Qbfs Polynomial (SPS QBF)

The SPS QBF surface describes symmetrical aspheres using Qbfs polynomials up to 30^{th} order. The aspheric deviation is defined an the basis of a best-fit sphere. The surface sagitta is defined by

$$z = \frac{c_{bfs} r^2}{1 + \sqrt{1 - c_{bfs}^2 \cdot r^2}} + \frac{u^2(1-u^2)}{\sqrt{1 - c_{bfs}^2 r_n^2 u^2}} \sum_{m=0}^{13} a_m Q_m^{bfs}(u^2) \tag{8.12}$$

with

z	sag of the surface perpendicular to the vertex tangent plane (parallel to the local z-axis)
c_{bfs}	curvature of best-fit sphere
r	$= \sqrt{x^2 + y^2}$ radial distance from vertex
r_n	normalization radius.
u	$= r/r_n$
a_m	$m^{th} Q^{bfs}$ coefficient
Q_m^{bfs}	the Q^{bfs} polynomial of order m.

Given the relation $u^4 \cdot u^{2m} = u^{2m+4}$, the order of the Q^{bfs} polynomial is $2m + 4$. The range 0-13 for m yields orders 4-30.

In explicit notation, the first six Qbfs basis elements are:

Term	Qbfs polynomial expression
1	1
2	$\frac{1}{\sqrt{19}}\left(13 - 16u^2\right)$
3	$\sqrt{\frac{2}{95}}\left[29 - 4x\left(25 - 19u^2\right)\right]$
4	$\sqrt{\frac{2}{2545}}\left\{207 - 4u^2\left[315 - u^2\left(577 - 320u^2\right)\right]\right\}$
5	$\frac{1}{3\sqrt{131831}}\left(7737 - 16u^2\left\{4653 - 2u^2\left[7381 - 8u^2\left(1168 - 509u^2\right)\right]\right\}\right)$
6	$\frac{1}{3\sqrt{6632213}}\left[66657 - 32u^2\left(28338 - u^2\left\{135325 - 8u^2\left[35884 - u^2\left(34661 - 12432u^2\right)\right]\right\}\right)\right]$

The table below lists the coefficient numbers for the surface parameters of the `SPS QBF` asphere type. (use alternatively `ATY QBF` command).

Coefficient	Definition
C1	Conic constant
C2	Normalization radius (NRAD). If a normalization radius is not defined, the clear Y semi-aperture (e.g. CIR, REY, etc.) is used instead.
C3	4^{th} order Qbfs coefficient (a_0)
C4	6^{th} order Qbfs coefficient (a_1)
C5	8^{th} order Qbfs coefficient (a_2)
C6	10^{th} order Qbfs coefficient (a_3)
C7	12^{th} order Qbfs coefficient (a_4)
C8	14^{th} order Qbfs coefficient (a_5)
C9	16^{th} order Qbfs coefficient (a_6)
C10	18^{th} order Qbfs coefficient (a_7)
C11	20^{th} order Qbfs coefficient (a_8)
C12	22^{th} order Qbfs coefficient (a_9)
C13	24^{th} order Qbfs coefficient (a_{10})
C14	26^{th} order Qbfs coefficient (a_{11})
C15	28^{th} order Qbfs coefficient (a_{12})
C16	30^{th} order Qbfs coefficient (a_{13})
...	...
C32	Number of terms to use in the expansion (¿2, ¡13). If zero (0), $OpTaliX$ automatically determines the number of terms by searching for the highest order non-zero coefficient.

Entering coefficients C1 to C32 is accomplished by the `SCO` command, explained in general on page 79. Specifically for Qbfs surfaces the necessary commands are:

`SPS QBF si..j|sk`	Change surface profile to QBF special aspheric surface.
`SCO si..j sk ci..j coefficient(s)`	Defines coefficients for `SPS QBF` surface(s) `si..j|sk`. If more than one coefficient is entered, all coefficients must be specified on the same command line. Example: `SCO s3 c3..5 0.1 0.2 0.3`

8.6.11 Qcon Polynomial (SPS QCN)

The SPS QCN surface describes symmetrical aspheres using Qcon polynomials as described by Forbes cite Forbes1 up to 30^{th} order. The aspheric deviation is defined an the basis of a base conic. The surface sagitta is defined by

$$z = \frac{c \cdot r^2}{1+\sqrt{1-(1+k)c^2r^2}} + u^4 \sum_{m=0}^{13} a_m Q_m^{con}(u^2) \tag{8.13}$$

with

z	sag of the surface perpendicular to the vertex tangent plane (parallel to the local z-axis)
c	vertex curvature (CUY)
k	conic constant
r	$= \sqrt{x^2+y^2}$ radial distance from vertex
r_n	normalization radius (NRAD).
u	$= r/r_n$
a_m	$m^{th} Q^{con}$ coefficient
Q_m^{con}	the Q^{con} polynomial of order m.

The Q vector at a particular $x = u^2$ is calculated by the following recurrance relationship:

$$Q(0,x) = 1$$
$$Q(1,x) = 6x - 5$$

$$Q(n,x) = \frac{(2*n+3)((n+1)(n+2)(2x-1)-4)Q(n-1,x)-(n-1)(n+2)(n+3)Q(n-2,x))}{n(n+1)(n+4)} \tag{8.14}$$

In explicit notation, the first six Qcon basis elements are:

Term	Qcon polynomial
1	1
2	$6u^2 - 5$
3	$15 - 14u^2(3 - 3u^2)$
4	$-\left\{35 - 12u^2\left[14 - u^2\left(21 - 10u^2\right)\right]\right\}$
5	$70 - 3u^2\left\{168 - 5u^2\left[84 - 11u^2\left(8 - 3u^2\right)\right]\right\}$
6	$-\left[126 - u^2\left(1260 - 11u^2\left\{420 - u^2\left[720 - 13u^2\left(45 - 14u^2\right)\right]\right\}\right)\right]$

The table below lists the coefficient numbers for the surface parameters of the `SPS QCN` asphere type. (use alternatively `ATY QCN` command).

Coefficient	Definition
C1	Conic constant
C2	Normalization radius (NRAD). If a normalization radius is not defined, the clear Y semi-aperture (e.g. CIR, REY, etc.) is used instead.
C3	4^{th} order Qcon coefficient (a_0)
C4	6^{th} order Qcon coefficient (a_1)
C5	8^{th} order Qcon coefficient (a_2)
C6	10^{th} order Qcon coefficient (a_3)
C7	12^{th} order Qcon coefficient (a_4)
C8	14^{th} order Qcon coefficient (a_5)
C9	16^{th} order Qcon coefficient (a_6)
C10	18^{th} order Qcon coefficient (a_7)
C11	20^{th} order Qcon coefficient (a_8)
C12	22^{th} order Qcon coefficient (a_9)
C13	24^{th} order Qcon coefficient (a_{10})
C14	26^{th} order Qcon coefficient (a_{11})

continued on next page

C15	28^{th} order Qcon coefficient (a_{12})
C16	30^{th} order Qcon coefficient (a_{13})
...	...
C32	Number of terms to use in the expansion (¿2, ¡13). If zero (0), $OpTaliX$ automatically determines the number of terms by searching for the highest order non-zero coefficient.

Entering coefficients C1 to C32 is accomplished by the `SCO` command, explained in general on page 79. Specifically for Qcon surfaces the necessary commands are:

`SPS QCN si..j|sk`	Change surface profile to QCN special aspheric surface.
`SCO si..j|sk ci..j coefficient(s)`	Defines coefficients for `SPS QCN` surface(s) `si..j|sk`.

A test case, the cartesian oval, is given by Forbes [63]. This system is found in the $OpTaliX$ examples library at `$i\examples\misc\qcon_cartesian_oval.otx`. The corresponding layout is shown in Fig. 8.8.

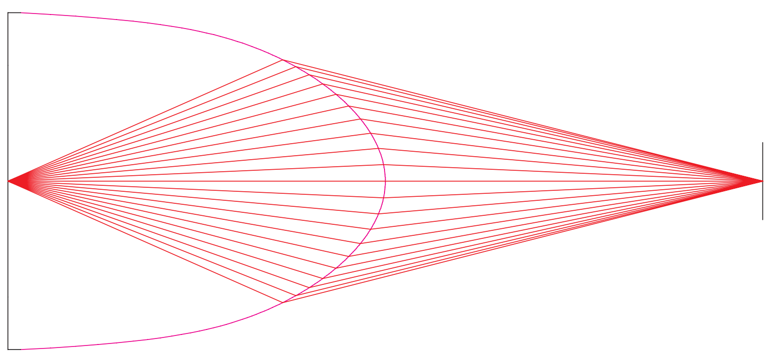


Figure 8.8: Cartesian oval using Qcon parameters.

8.7 Alternate Intersection Point

It is not always possible to predict the intersection point of a ray with a surface, in particular if the ray is at a high angle to the local surface axis. For example, consider the following case of a conic surface (parabola), where two intersection points are found (Fig. 8.9). Normally, the intersection point at P_1 would be selected by the program which is correct. If the ray originates from 'inside' of the parabola, however, the `IC` command allows selecting the alternate intersection point P_2 which would be more appropriate.

8.8 Axicon

Axicon surfaces are rotationally symmetric about the Z-axis and are like a cone, with the tip of the cone at the vertex of the surface. Axicons are modelled by an aspheric surface (surface type "A"). The following examples show the definition of an axicon surface by using the "EVEN" power polynomial asphere respectively the "ODD30" power (30^{th} order) polynomial asphere.

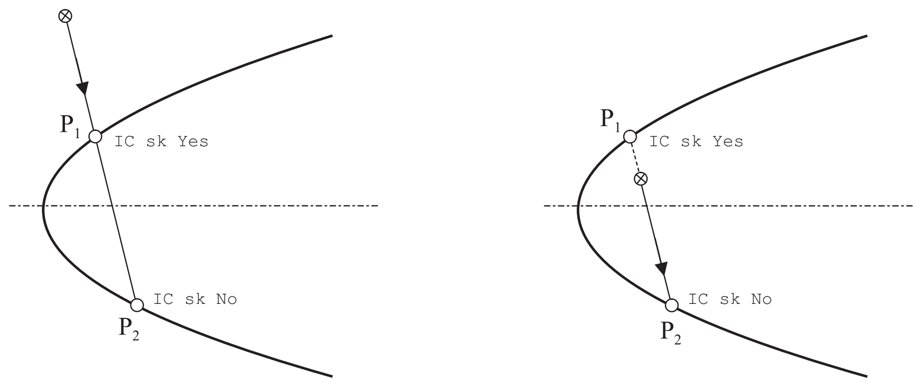


Figure 8.9: Selection of alternate intersection point and geometrical meaning of IC code. Left: ray starting 'outside' surface, right: ray starting 'inside' surface.

8.8.1 Axicon modelled by "EVEN" Power Asphere

In the "EVEN" power polynomial asphere only the radius radius of curvature and the conic constant K need to be defined. The radius of curvature is set to a small value, the conic constant is -2 (hyperbola). As a guideline, the radius of curvature should be at least one order of magnitude smaller than the smallest radial aperture of the surface. Make sure that the radius of curvature is NOT zero!

Due to the non-zero radius of curvature there is a small deviation of the slope to that of an axicon near the tip of the cone. This deviation can be made arbitrarily small by selecting a small enough radius of curvature.

From a practical point of view, the cone angle is the most interesting parameter and the only one needed. The cone angle θ is defined as the angle between the vertex tangent plane (i.e. the plane perpendicular to the Z-axis) and the axicon surface. This angle can be easily converted to the conic constant K by taking the limit case of the standard asphere sag (Eq. 8.1) as the radius of curvature approaches zero (curvature goes to infinity):

$$K = -\left(\frac{1}{tan^2\theta} + 1\right) \tag{8.15}$$

Example command input:

sut s2 a	! defines aspheric surface
rdy s2 0.1	! radius of curvature should be small (but must be non-zero)
k s2 -2	! Conic constant (hyperbola)

8.8.2 Axicon modelled by "ODD30" Power Asphere

An alternative way of defining an axicon surface is by using the odd power special asphere (see Eq. 8.6) which accepts coefficients up to 30^{th} order. Its advantage is that the tip of the axicon is exactly modelled because the ODD30 asphere also includes a linear term (slope).

sps odd s2	! defines odd (30^{th} order) aspheric surface
sco s2 c2 0.2	! sets special surface coefficient C2

The cone angle θ is related to the coefficient C_2 by the relation

$$tan(\theta) = C_2 \tag{8.16}$$

8.9 Hologram Surface

The optical properties of a holographic surface are based on diffraction at the effective grating spacing seen at the local intersection point of a ray. Commonly, holographic surfaces are also denoted as *diffractive* surfaces. A diffractive lens behaves like an ideal, thin refractive lens with an infinite number of focal lengths given by

$$f(\lambda) = \frac{\lambda_0 f_0}{m\lambda} \tag{8.17}$$

where f_0 is the focal length at the design wavelength λ_0 and m is the diffraction order. This result reveals the highly dispersive nature of a diffractive lens. To model these effects, several types of diffractive surfaces are available in $OpTaliX$.

- Linear grating (section 8.10),
- Variable linear spacing (VLS) grating (section 8.10.1),
- Optical hologram, formed by interfering two beams of light (section 8.9.4),
- Computer-generated holograms (CGH) with a user specified radial symmetric phase distribution (section 8.9.2),
- Computer-generated holograms (CGH) with a user specified asymmetric (two-dimensional) phase distribution (section 8.9.1),
- ”Sweatt” model (section 8.9.3).

Diffractive surfaces, which are represented by phase distributions $\Phi(x, y)$, add a phase to a ray when it strikes the diffractive surface. The direction cosines K, L, M of an impinging ray changes according to the classical grating equation, if the vectors are resolved in a rectangular coordinate system oriented with its Z-axis along the local surface normal

$$K' = K + m \cdot \frac{\lambda}{2\pi} \cdot \frac{\partial\Phi(x,y)}{\partial x} \tag{8.18}$$

$$L' = L + m \cdot \frac{\lambda}{2\pi} \cdot \frac{\partial\Phi(x,y)}{\partial y} \tag{8.19}$$

$$M' = \sqrt{1 - (K'^2 + L'^2)} \tag{8.20}$$

where λ is the wavelength and m is the diffraction order. The partial derivatives of the function $\Phi(x, y)$ are proportional to the local grating frequencies ν_x, ν_y

$$\nu_x = \frac{\Phi(x,y)}{x}, \qquad \nu_y = \frac{\Phi(x,y)}{y} \tag{8.21}$$

and we have

$$K' = K + m \cdot \frac{\lambda}{2\pi} \cdot \nu_x \tag{8.22}$$

$$L' = L + m \cdot \frac{\lambda}{2\pi} \cdot \nu_y \tag{8.23}$$

Note, that the phase function Φ is expressed in terms of the **absolute** optical path difference (OPD), i.e. in lens units. A more detailed treatment of vector ray tracing through general holograms is given by Welford [58].

Some other programs define the phase in units of the reference/design wavelength. For such cases the hologram coefficients must be normalized to the design wavelength first before they can be used in $OpTaliX$. This is accomplished by the relation

$$c_i(OpTaliX\ ) = \frac{c_i(other)}{\lambda_0} = \frac{c_i(other)}{\texttt{HWL}} \tag{8.24}$$

with λ_0 = `HWL` given in μm.

Note also that diffractive structures (holograms, grating, etc.) exhibit a significant variation of diffracted energy depending on wavelength, incidence/diffraction angle, diffraction order and on the grating structure. This effect is accounted for in transmission analysis (page 325). A detailed description of the relevant theory is given in sect. 8.10.3 (page 90).

Hologram Data Entry:

The nomenclature for hologram surfaces is uniform throughout all types of holograms, including linear (straight-line ruled) gratings.

`HCO si..j ci..j coeff`	Hologram coefficients `ci..j` on surface(s) `si..j`

`HCi si..j coeff`	Alternative form of entering HOE-coefficients, where "`i`" denotes a coefficient number. For example, `HC12` is coefficient no. 12. This form is particularly useful for defining coefficients as variables in optimization.
	The following commands are synonymous :
	`HC7 s4 0.1234e-3`
	`HCO s4 c7 0.1234e-3`

`HOT [si..j] htype`	Hologram type, designating which phase function is used. `htype` = 0 : linear grating, see section 8.10, `htype` = 1 : symmetrical phase function as defined in Eq. 8.25, `htype` = 2 : asymmetrical (2d) phase function as defined in section 8.9.1. `htype` = 3 : two-point hologram defined by object and reference point source. `htype` = 4 : VLS-grating (see section 8.10.1).
`HWL sk design_wavel`	Hologram design wavelength at surface `sk`, in micrometers.
`HOR [sk|si..j] order`	Hologram order, an integer value. Note that the sign of the hologram order must be changed if the orientation of the HOE changes between setups and the local surface normal points in the opposite sense.
`GRO [sk|si..j] order`	Grating order, an integer value. This command is obsolete, but still available. Use HOR instead.
`GRX [sk|si..j] grooves_per_mm_X`	Grooves per mm, the diffraction is seen in the X-direction.
`GRY [sk|si..j] grooves_per_mm_Y`	Grooves per mm, the diffraction is seen in the Y-direction.

continued on next page

continued from previous page	
`HX1 si..j obj_source_x`	X-coordinate of object point source for holographic surface. `obj_source_x` is given relative to the local coordinate system of the hologram surface.
`HY1 si..j obj_source_y`	Y-coordinate of object point source for holographic surface. `obj_source_y` is given relative to the local coordinate system of the hologram surface.
`HZ1 si..j obj_source_z`	Z-coordinate of object point source for holographic surface. `obj_source_y` is given relative to the local coordinate system of the hologram surface.
`HX2 si..j ref_source_x`	X-coordinate of reference point source for holographic surface. `ref_source_x` is given relative to the local coordinate system of the hologram surface.
`HY2 si..j ref_source_y`	Y-coordinate of reference point source for holographic surface. `ref_source_y` is given relative to the local coordinate system of the hologram surface.
`HZ2 si..j ref_source_z`	Y-coordinate of reference point source for holographic surface. `ref_source_y` is given relative to the local coordinate system of the hologram surface.
`HV1 si..j rea|vir`	Defines object point source as real (REA, diverging beam) or virtual (VIR, converging beam) for the designated surface(s).
`HV2 si..j rea|vir`	Defines reference point source as real (REA, diverging beam) or virtual (VIR, converging beam) for the designated surface(s).

`SUT [si..j] SG`
 	
`SUT [si..j] SH`
	Set surface type to put a grating on a (spherical) base surface as given in the example command to the left. See also the full description of the SUT command, page 63.
	Set surface type to put a general hologram (including grating) on a spherical base surface as given in the example command to the left. See also the full description of the SUT command, page 63.

`BLD [sk|si..j] depth`	Blaze depth in mm. Required in transmission/efficiency calculations.
`BLT [sk|si..j] [IDL|KIN|STE|SIN]`	Blaze type used for diffraction efficiency calculations. Specifies the diffractive structure as `IDL` : 100% of the diffracted energy is directed into the specified grating order (GRO/HOR) at all wavelengths. `KIN` : Kinoform (sawtooth) profile, `STE` : step approximation of the Kinoform profile, `SIN` : sinusoidal profile.
`BLN [sk|si..j] levels`	Number of discrete levels in the step approximation of a Kinoform diffracting profile.
`HPH [sk|si..j]`	Plot hologram phase.
`HZO [sk|si..j]`	Calculate zones in radial holograms.
	continued on next page

continued from previous page	
`VLS [si..j] c_3 c_4 .... c_10`	Adds properties of a variable linear spacing (VLS) grating to a surface, i.e. converts a surface to a VLS grating. Surface type and hologram type are automatically set and do not require any further user interaction. The coefficients `c_3` to `c_10` are defined in Eqs. 8.33 and 8.34 respectively. For example, `c_3` defines the constant grating frequency in grooves/mm.

8.9.1 Asymmetric Phase Function

The function for a generally asymmetric phase is defined by a polynomial function of up to 28 coefficients:

$$\begin{array}{lccccccccccccc}
\Phi(x,y) = & a_1 \\
& a_2x & + & a_3y \\
& a_4x^2 & + & a_5xy & + & a_6y^2 \\
& a_7x^3 & + & a_8x^2y & + & a_9xy^2 & + & a_{10}y^3 \\
& a_{11}x^4 & + & a_{12}x^3y & + & a_{13}x^2y^2 & + & a_{14}xy^3 & + & a_{15}y^4 \\
& a_{16}x^5 & + & a_{17}x^4y & + & a_{18}x^3y^2 & + & a_{19}x^2y^3 & + & a_{20}xy^4 & + & a_{21}y^5 \\
& a_{22}x^6 & + & a_{23}x^5y & + & a_{24}x^4y^2 & + & a_{25}x^3y^3 & + & a_{26}x^2y^4 & + & a_{27}xy^5 & + & a_{28}y^6
\end{array}$$

Note that the phase is a function of x and y and not z, and thus is independent of the substrate shape. Individual coefficients a_i are entered by the commands `HCi` or `HOC` (see also section8.9 for a complete description of the commands).

Also note that the phase is defined in absolute (lens) units (i.e. typically in mm).

Example:

`sut s2 SH`	! base surface is spherical with superimposed hologram
`HC3 s1 0.123`	! Hologram coefficient c3 (a_3 term) on surface 1 is 0.123
`HOC s1 c3 0.123`	! As above

8.9.2 Symmetric Phase Function

The phase function of a symmetric hologram takes the absolute value of a power series expansion in the radial coordinate h.

$$\Phi(x,y) = a_1 + a_2h + a_3h^2 + a_4h^3 + a_5h^4 + a_6h^5 + ... \tag{8.25}$$

where $h = \sqrt{x^2 + y^2}$

In the paraxial domain the properties of a lens are completely described by the a_3 term and the diffractive lens power φ_{diff} is given by

$$\varphi_{diff} = \frac{1}{f} = -2ma_3\lambda \tag{8.26}$$

where m is the diffraction order.

The blaze depth d, i.e. the sagitta of the radial groove profile, is then calculated by [62],

$$d = \frac{\lambda_0}{n_0 - 1} \tag{8.27}$$

where λ_0 is the reference wavelength, and n_0 is the refractive index at the reference wavelength. See also sect. 23.3 about manufacturing aspects and calculation of diffraction zones related to diffractive structures. This section also describes conversion of hologram coefficients to other programs.

8.9.3 Sweatt Model

An alternative to the phase models described in the previous sections is to using the so-called *Sweatt model*. It has been shown by Sweatt [52, 53] and Kleinhans [26] that a diffractive lens is mathematically equivalent to a thin refractive lens, provided the index of refraction goes to infinity. For practical cases a very high refractive index ($n = 10000$) is used. This reduces the lens thickness profile and introduces an appreciable shape over a relatively small physical path length. The advantage of this method is, that it allows the use of existing ray tracing routines for designing diffractive lenses. The chromatic properties of the diffractive lens are modelled by

$$n_s(\lambda, m) = m\frac{\lambda}{\lambda_0}\left[n_s(\lambda_0) - 1\right] + 1 \tag{8.28}$$

where the subscript s refers to the "Sweatt" model and λ_0 is the design wavelength. The refractive index is proportional to the wavelength. It is implicitly assumed that the design order is the first order.

The lens curvatures of the equivalent "Sweatt" model for a given lens power φ at the design wavelength are given by

$$c_{1,2} = c_s \pm \frac{\varphi_0}{2\left[n_s(\lambda_0) - 1\right]} \tag{8.29}$$

where c_s is the curvature of the diffractive substrate. Higher order terms in the diffractive surface phase polynomial are modelled by asphierization of the base surface.

To simplify the set up of the "Sweatt" model, a material (glass) `SWEATT` is available. Enter `gla sk sweatt` in the command line to convert a surface `sk` to the "Sweatt" model. Alternatively, enter the material (glass) name in the appropriate row/column of the surface spreadsheet editor.

Example:

`sut s2 S`	! Base surface is spherical. Note, that the surface type "H" is not required in the Sweatt model
`gla s2 sweatt`	! Defines the high-index glass "SWEATT"
`hwl s2 0.633`	! Design wavelength used in the Sweatt model is 0.633 μm

8.9.4 Two-Point Hologram

This type of holographic surface describes the interference pattern of two point sources, i.e. two spherical waves, which includes plane wavefronts as the limiting case. The local grating frequency is determined by the location and orientation of the resultant interference fringes. To model a two-point hologram, the location of the two sources and the wavelength of the source beams must be given. The sources used to record the hologram are specified by X-, Y- and Z-coordinates relative to the local coordinate system of the holographic surface. The parameters are `HX1,HY1,HZ1` for the object point source and `HX2,HY2,HZ2` for the reference point source.

The parameters HV1 and HV2 define from which side each beam is directed during construction. Point sources are considered *real* if the beam is diverging from the source, or *virtual* if the beam is converging toward the source.

Tracing a ray through a holographic surface makes use of the information about the geometry of formation of the hologram. Unlike to phase models, the local fringe spacing is not explicitly computed . Holograms can be applied to surfaces of any arbitrary shape.

We follow the notation by Welford [58] and let n be a unit vector along the local normal to the hologram surface (see Fig. 8.10). The hologram is recorded by two spherical wavefronts emerging from the object point source and the reference point source, represented by the vectors r_o and r_r. The unit vectors r'_o and r'_r represent the reconstruction and image rays at the intersection point P. The image ray r'_r is obtained by the equation

$$n \times \left(r'_o - r'_r\right) = \frac{m\lambda'}{\lambda} n \times \left(r_o - r_r\right) \tag{8.30}$$

where m is the order of diffraction, λ is the recording wavelength (design wavelengthHWL) and λ' is the reconstruction wavelength.

In a coordinate system oriented with its Z-axis to the local surface normal at P the vectors are resolved into two components

$$K'_0 - K'_r = \frac{m\lambda'}{\lambda}(K_0 - K_r) \tag{8.31}$$

$$L'_0 - L'_r = \frac{m\lambda'}{\lambda}(L_0 - L_r) \tag{8.32}$$

of a typical unit vector (K,L,M).

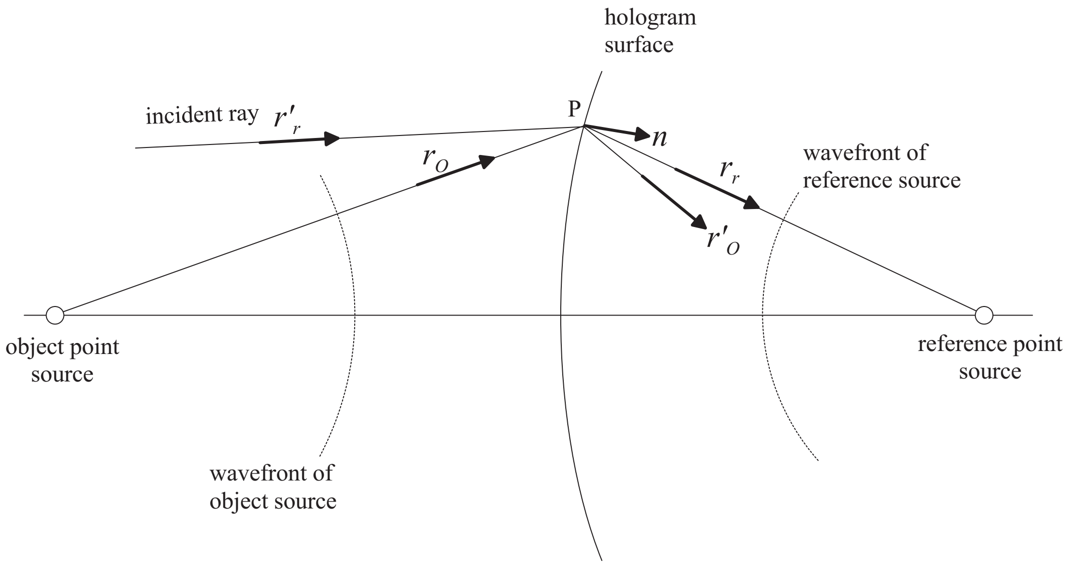


Figure 8.10: Notation for ray tracing at a holographic surface.

Example using a two-point model:

```
sut s2 SH        ! base surface is spherical with superimposed hologram
hot s2 3         ! Hologram type specifies "two-point" hologram
hz1 s2 -1.e20    ! Object point source is at infinity, object wavefront is flat.
hz2 s2 50        ! Reference point source is at +50 mm with respect to surface vertex.
hv1 rea          ! Real object point source.
hv2 rea          ! Real reference point source.
```

Note, that all other point source parameters (HX1,HY1, HX2,HY2) are initially zero.

Design Example:

An example holographic lens is found in the directory `$i\examples\diffractive\two-point-hoe.otx`. The diffractive optical element (DOE) is recorded with a He-Ne laser at a wavelength 0.6328μm. The location of the point sources are specified in the local coordinate system of the holographic optical element (HOE).

We also note the hologram construction parameters as shown in the surface listing (see `LIS` command):

```
#   Hologram coefficients :

1 HOT  3      HOR  -1      HWL  0.63300
  HX1    0.0000             HX2    0.0000
  HY1    0.0000             HY2    0.0000
  HZ1 -0.10000E+21          HZ2    50.000
  HV1   REA                 HV2   REA
```

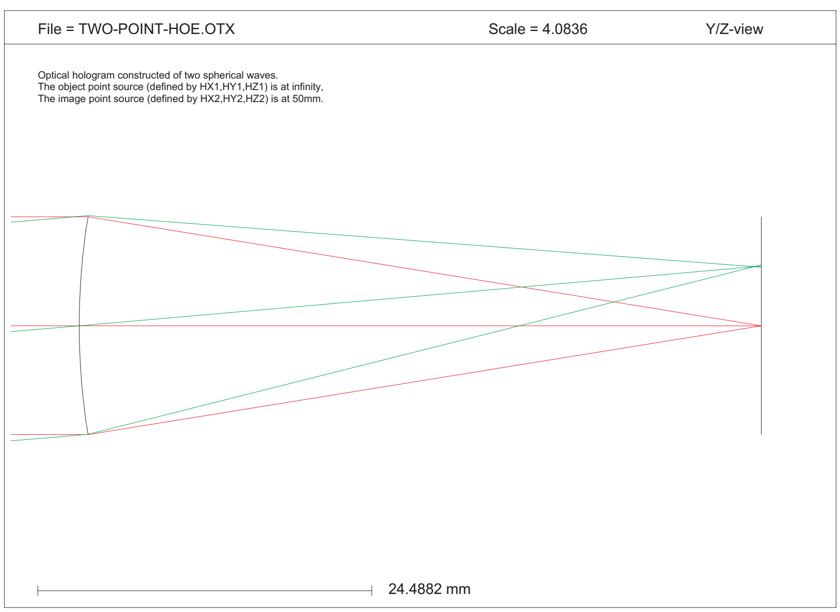


Figure 8.11: Two-point hologram on curved substrate. See example file at `$i\examples\diffractive\two-point-hoe.otx`

Since this is an on-axis lens, the location of the point sources of the recording laser beams are at `HX1` = `HY1` = 0, and `HX2` = `HY2` = 0. Point source 1 is at infinity (`HZ1 0`), so it is actually a plane wave at the hologram surface. Point source 2 is located at the focal point, which is 50mm to the right of the HOE (`HZ2 50.0`). Based on elementary holography theory, the plane wave incident to the hologram will be diffracted into a spherical (on-axis) wave converging to the focal point and thus constructing a perfect image.

We also note the curvature of the hologram surface. For on-axis imaging it does not make any difference whether the hologram surface is curved or not, since the hologram is recorded by two (perfect) point sources located on the axis. In this case the reconstruction geometry is identical to the recording geometry. For off-axis imaging, however, a curved hologram substrate is analogous to ”bending” of a thin lens and yields coma-free and aplanatic imaging.

8.10 Diffraction Grating Surface

Diffraction gratings are a subset of holographic surfaces and are used to model straight-line ruled gratings. This simplifies data entry without the need to fully specify complex holograms. However, gratings may also be specified by an asymmetric hologram surface (see section 8.9.1, in which the linear coefficients $a2, a3$ directly give the grating frequency in X- and Y-direction. The straight rules may have any orientation with respect to the base surface (respectively the local coordinate system). The orientation is defined by proper setting of the grating frequency in X- and Y-direction (`GRX`, `GRY`). The grating frequency is always defined on the surface tangent plane in lines (grooves) per millimeter.

`GRX [si..j] grooves_per_mm_X`	Grooves per mm, the diffraction is seen in the X-direction.
`GRY [si..j] grooves_per_mm_Y`	Grooves per mm, the diffraction is seen in the Y-direction.
`HOR [si..j] order`	Hologram diffraction order, an integer value.
`SUT [si..j] SG`	Set surface type to put a grating on a (spherical) base surface as given in the example command to the left. See also the full description of `SUT` command (page 65).

Example:

```
sut s2 SG     ! base surface is spherical with grating additive
hor s2 1      ! Diffraction order is +1
gry s2 100    ! Grating frequency is seen in Y-direction at 100 Lines/mm.
grx s2 55     ! Grating frequency is seen in X-direction at 55 Lines/mm.
```

8.10.1 Variable Line Spacing (VLS) Grating Surface

A linear variable spacing grating (VLS-grating) is a special form of a straight-line ruled grating (see previous section). The phase is described by a polynomial function

$$\Phi(y) = a_3y + a_4y^2 + a_5y^3 + a_6y^4 + a_7y^5 + a_8y^6 + a_9y^7 + a_{10}y^8 \tag{8.33}$$

The grating frequency ν_y is the first derivative of Φ

$$\nu_y = a_3 + 2a_4y + 3a_5y^2 + 4a_6y^3 + 5a_7y^4 + 6a_8y^5 + 7a_9y^6 + 8a_{10}y^7 \tag{8.34}$$

Note that a VLS-grating is only defined in the Y-direction. Arbitrary orientations of the grooves can be simulated by applying a Z-rotation to the surface (see CDE command). Also note that the coefficients numbering starts at 3, which ensures consistency with the definitions of the conventional grating (sect. 8.10) and the asymmetric phase function (sect. 8.9.1).

The grating frequency ν_y is always defined on the tangent plane of a surface. If only a_3 is specified, the VLS-grating behaves like a straight-line ruled gratings with constant groove spacing (grating frequency = a_3 in grooves/mm).

A VLS-grating is traced in $OpTaliX$ similarly to an asymmetric phase hologram. Therefore the surface type must be ”H”.

Example:

`sut s2 SH`	! base surface is spherical plus hologram
`hor s2 1`	! Diffraction order is +1
`hot s2 4`	! Hologram type is VLS-grating
`hco s2 c3 100`	! Hologram coefficient 3 (equivalent to the grating frequency = 100 grooves/mm).
`hc3 s2 100`	! as above.

A simplified form of entering/defining VLS gratings is provided by the following command:

`VLS [si..j] c_3 c_4 .... c_10`	Adds properties of a variable linear spacing (VLS) grating to a surface, i.e. converts a surface to a VLS grating. Surface type and hologram type are automatically set and do not require any further user interaction. The coefficients `c_3` to `c_10` are defined in Eqs. 8.33 and 8.34 respectively. For example, `c_3` defines the constant grating frequency in grooves/mm.

8.10.2 Conversion of Coefficients for a VLS Grating

A different form of describing VLS-gratings on a curved substrate is occasionally used. It is given by Kita et.al. [25]

$$\sigma = \frac{\sigma_0}{\left(1 + \frac{2b_2 w}{R} + \frac{3b_3 w^2}{R^2} + \frac{4b_4 w^3}{R^3}\right)} \tag{8.35}$$

where the groove spacing σ is defined as a function of the local coordinate w measured from the center of the grating and the radius of curvature R of the concave grating surface. The coefficients b_2, b_3, b_4 are easily converted to the form used in $OpTaliX$ (Eq. 8.34)

In the Kita paper, the groove spacing σ is defined as a function of the local coordinate w measured from the center of the grating and the radius of curvature R of the concave grating surface, whereas in $OpTaliX$ the groove spacing is expressed by the grating frequency ν

$$\nu_y = a_3 + 2a_4 y + 3a_5 y^2 + 4a_6 y^3 + ... \tag{8.36}$$

Groove spacing and (local) grating frequency are related by $\nu = 1/\sigma$. Inserting into Eq. 8.35 and rearranging yields

$$\nu = \nu_0 + \frac{2\nu_0 b_2}{R} y + \frac{3\nu_0 b_3}{R^2} y^2 + \frac{4\nu_0 b_4}{R^3} y^3 \tag{8.37}$$

A deeper analysis indicates that the conventions of the coordinate axes used in the paper by Kita and those used in $OpTaliX$ are different. Obviously $w = -y$. Thus, we modify Eq. 8.37 accordingly

$$\nu = \nu_0 - \frac{2\nu_0 b_2}{R} y + \frac{3\nu_0 b_3}{R^2} y^2 - \frac{4\nu_0 b_4}{R^3} y^3 \qquad (8.38)$$

Comparing Eqs. 8.34 and 8.38, the conversion formulas are directly obtained as

$$\begin{aligned} a_3 &= \nu_0 &&= 1/\sigma_0 \\ a_4 &= -\frac{\nu_0 b_2}{R} &&= -\frac{b_2}{\sigma_0 R} \\ a_5 &= \frac{\nu_0 b_3}{R^2} &&= \frac{b_3}{\sigma_0 R^2} \\ a_6 &= -\frac{\nu_0 b_4}{R^3} &&= -\frac{b_4}{\sigma_0 R^3} \end{aligned} \qquad (8.39)$$

Numerical Example:

We use the data given in the paper by Kita 8.35: $R = 5649mm$, $\sigma_0 = 1/1200mm$, $b_2 = -20$, $b_3 = 4.558 \cdot 10^2$, $b_4 = -1.184 \cdot 10^4$. The following table shows the analytically converted coefficients.

$OpTaliX$ Coeff.	calculated from Eq. 8.39
a_3	1200
a_4	4.2485
a_5	$1.714 \cdot 10^{-2}$
a_6	$7.882 \cdot 10^{-5}$

8.10.3 Diffraction Efficiency Calculation

$OpTaliX$ calculates the *scalar* diffraction efficiency on surfaces that contain diffractive structures (hologram, grating). Diffraction efficiency describes the amount of energy associated to a ray when passing a diffractive structure. Diffraction efficiency depends on wavelength, incidence angle, diffraction order and on the profile of the diffractive structure. The scalar model implemented in $OpTaliX$ currently does not include variations due to polarization state.

The results of diffraction efficiency calculations are included in transmission analyses (requires settings `TRA Y` and `POL Y`).

The following profiles of diffractive structures are currently available:

- Sawtooth Profile (Kinoform Blaze Type)
- Sawtooth Step Approximation
- Sinusoidal Profile

8.10.3.1 Sawtooth Profile (Kinoform)

The diffraction efficiency into the m^{th} diffracted order of a sawtooth (Kinoform) profile (Fig. 8.12) is approximated by

$$\eta(m) = \left(\frac{sin\left[\pi(\alpha - m)\right]}{\pi(\alpha - m)} \right)^2 \qquad (8.40)$$

with:

α	=	$\dfrac{d_1\,(n_1 \cdot cos\theta_1 - n_2 \cdot cos\theta_2)}{\lambda}$
m	=	diffracted order (GRO or HOR)
d_1	=	blaze depth (BLD)
n_1	=	refractive index before surface
n_2	=	refractive index after surface
λ	=	wavelength
θ_1	=	local incidence angle of ray
θ_2	=	local diffraction angle of ray

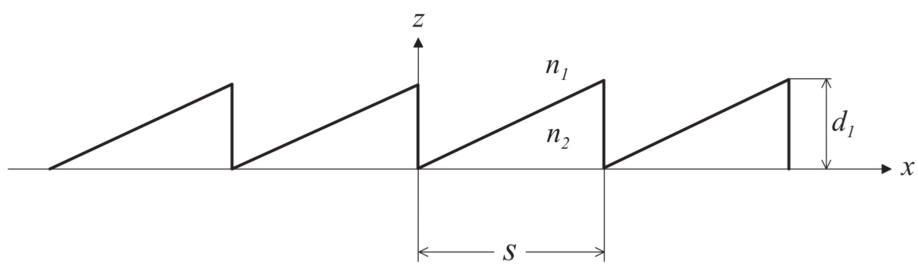


Figure 8.12: Sawtooth profile of a diffracting structure (Kinoform structure)

Within each period, the profile is a linear function of the spatial coordinate x. The blaze depth d_1 (BLD command) of the local grating structure is always measured to the local surface normal.

8.10.3.2 Sinusoidal Profile

The diffraction efficiency into the m^{th} diffracted order of a sinusoidal profile (Fig. 8.13) is approximated by

$$\eta(m) = [J_m(\pi \cdot \alpha)]^2 \tag{8.41}$$

where $\alpha = \dfrac{d_1\,(n_1 \cdot cos\theta_1 - n_2 \cdot cos\theta_2)}{\lambda}$, and J_m is the Bessel function of first kind, order m.

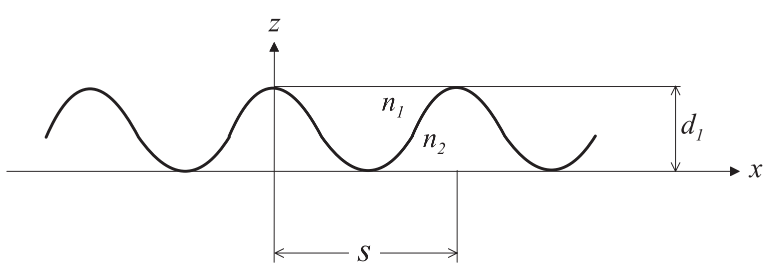


Figure 8.13: Sinusoidal profile of a diffracting structure.

8.10.3.3 Step Approximation

The step approximation of a Kinoform profile is specified by the BLT STE command. The diffraction efficiency into the m^{th} diffracted order of a step approximation of a Kinoform profile (Fig. 8.14) is approximated by

$$\eta(m) = \left[\frac{sin(m\pi/N)}{m\pi}\right]^2 \cdot \left[\frac{sin(\pi(\alpha - m))}{sin(\pi(\alpha - m)/N)}\right]^2 \tag{8.42}$$

where:

N	=	number of discrete levels in each grating period (BLN command).
α	=	$\dfrac{d_1\,(n_1 \cdot cos\theta_1 - n_2 \cdot cos\theta_2)}{\lambda}$
m	=	diffracted order (commands `GRO` or `HOR`)
d_1	=	blaze depth (`BLD` command)
n_1	=	refractive index before surface
n_2	=	refractive index after surface
λ	=	wavelength
θ_1	=	local incidence angle of ray
θ_2	=	local diffraction angle of ray

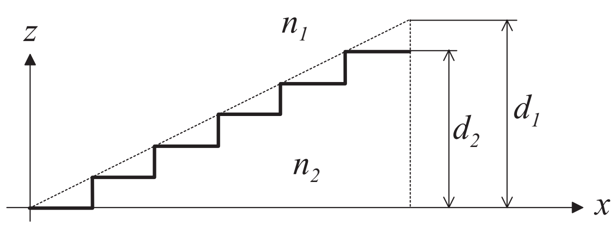


Figure 8.14: Step approximation profile of a Kinoform diffracting structure.

8.10.3.4 Diffraction Efficiency Example

The effect of diffraction efficiency at diffractive structures (hologram, grating, etc.) can be best demonstrated with transmission analysis vs. wavelength. Load the example file
`$i\examples\spectrometer\rowland-grating.otx`.
The optical system, as shown in Fig. 8.15 contains a linear grating on a curved surface. The necessary parameters required to analyze diffraction efficiency effects are blaze type (BLT) and blaze depth (BLD):

`BLT s1 KIN`	Blaze type is Kinoform
`BLD s1 0.00027`	Blaze depth is 0.00027 mm

Transmission analysis vs. wavelength is then accomplished by the command:

```
TRA LAM
```

See Fig. 8.16 for the corresponding transmission curve.

8.11 Fresnel Surface

In a Fresnel lens the curved surface of a lens is collapsed in annular zones to a thin plate. As shown in Fig. 8.17, this has the refracting effect of the lens without its thickness or weight. Such lenses are often used as condensors in overhead projectors, spotlights and signal lamps.

A Fresnel lens is defined by the radius of curvature R of the refracting surface (as it would be defined for a conventional lens) and the depth d of the annular zones (see Fig. 8.17).

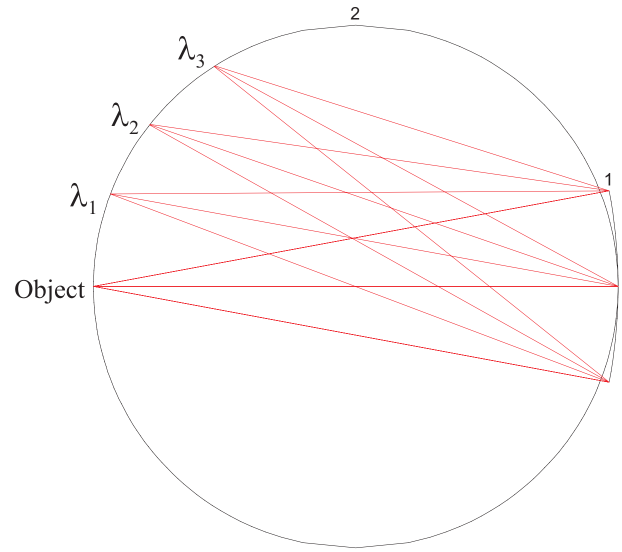


Figure 8.15: Rowland grating.

`FTH fresnel_depth`	Fresnel thickness, that is the depth or thickness of the annular rings. Smaller values for `FTH` result in a finer radial spacing of the annular zones. This option is currently only available in the command line. It cannot be set from the menu. Note that the surface type (SUT) must be "**F**" in conjunction with the "**S**" or "**A**" qualifier for the surface shape (S = spherical, A = aspherical).

Note, that "shadowing" effects due to the finite thickness of the structure are not taken into account during ray tracing.

Example input:

```
sut s1 SF   ! defines a Fresnel surface with spherical base curvature
rdy s1 30   ! defines base radius, which controls refraction
fth s1 1    ! depth of annular zones
```

8.12 Total Internal Reflection (TIR) Surface

Total internal reflection (TIR) occurs on glass-air interfaces when the angle of incidence in the medium of higher index exceeds the critical angle θ_c. Under that condition there can be no refracted light and every ray undergoes total reflection as shown in Fig. 8.18.

The critical angle is calculated by

$$sin(\theta_c) = \frac{n}{n'} \tag{8.43}$$

A TIR surface always behaves like a mirror surface, except that TIR condition is calculated to determine whether a ray is valid or is blocked. Thus, rays that hurt the TIR condition (i.e. the angle of incidence is less than θ_c are blocked whereas rays at $\theta > \theta_c$ is reflected.

A TIR surface is defined by the following command:

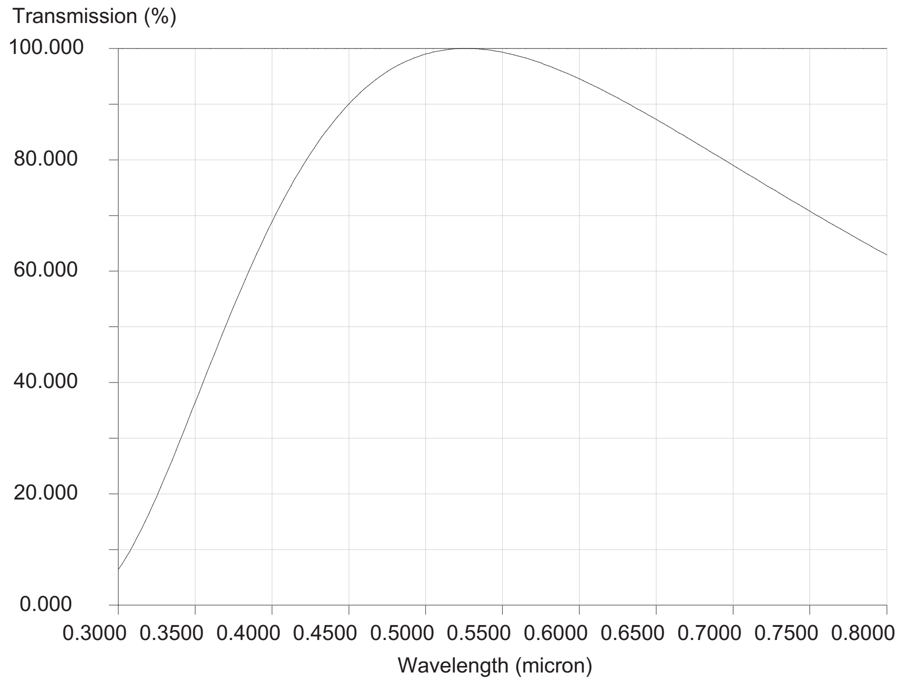


Figure 8.16: Diffraction efficiency calculation on a Rowland grating with a ”Kinoform” profile.

`TIR sk|si..j` or `RMD TIR sk|si..j`	Defines total internal reflecting surface (TIR). Adds ”T” to surface type. A TIR surface behaves like a mirror surface except that rays only pass if TIR condition is fulfilled. See also RMD TIR, respectively REFL and REFR to convert a surface to reflecting or refracting mode. Calculating TIR condition requires proper definition of both materials, `GL1` **and** `GL2`, where, according to Eq. 8.43, n = index of `GL1` and n' = index of `GL2`. By default, $n' = 1$. The TIR flag is ignored at non-sequential surfaces as the TIR condition is *always* checked and the corresponding ray direction is automatically chosen.

Light is totally reflected, i.e. $R = 1$, if the TIR condition according to Eq. 8.43 is fulfilled, however, there is a phase change on reflection which depends on incidence angle, wavelength and which is different for S- and P-components (polarized light). The phase changes are calculated by [4]

$$tan\frac{\delta_1}{2} = -\frac{\sqrt{sin^2\theta_i - n^2}}{n^2 cos\theta_i} \tag{8.44}$$

$$tan\frac{\delta_2}{2} = -\frac{\sqrt{sin^2\theta_i - n^2}}{cos\theta_i} \tag{8.45}$$

where the subscript (1) means S-polarization (German: *senkrecht*) and (2) means P-polarization (German: *parallel*).

Although there is no loss of light at TIR, the wavefront (i.e. phase) is altered according to Eqs. 8.44 and 8.45. For unpolarized light, the impact on wavefront Δw is given by

$$\Delta w = \frac{(\delta_1 - \delta_2)\,\lambda}{2\pi} \tag{8.46}$$

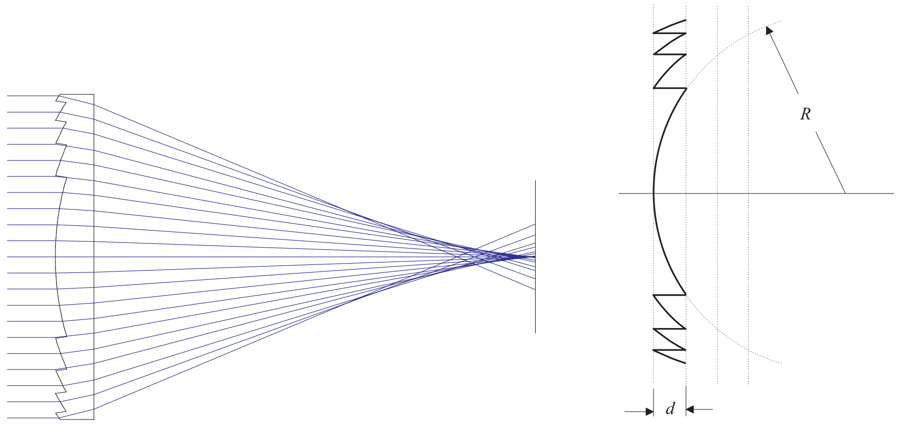


Figure 8.17: Fresnel lens and construction method of annular zones.

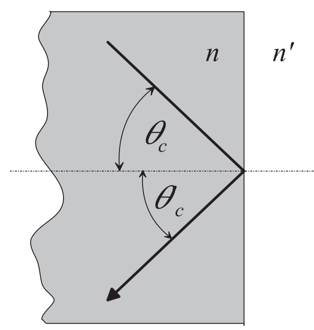


Figure 8.18: Total internal reflection (TIR) condition.

The phase change is **always** applied, irrespectively of whether polarization ray trace is enabled or not (see POL).

An example showing the effect on wavefront is provided in `$i\examples\misc\tir.otx`. The results are shown in Fig. 8.19

Even though the aspheric lens should provide a near perfect image, the coma-like tail appearing on the PSF in Fig. 8.19 is caused by wavefront (phase) variation as a function of incidence angle variation across the pupil, in particular by those rays striking the TIR surface in the neighborhood of the critical angle θ_c. Note that the focussed spot of Fig. 8.19 is not centered on the optical axis but is shifted. This shift is known as the Goos-Hanchen effect. Similarly, we may explain this effect in the language of Fourier-Transform theory by multiplying a function (the wavefront) by a linear phase factor. See also Mansuripur [36] for a more thorough explanation of this effect.

8.13 Non-Sequential Surface

Non sequential surfaces (NSS) are a special subset of the total lens, where the sequence of the surfaces, which are hit by a ray, is determined by the light ray itself. This means that the program automatically determines which surface is hit next.

Command Overview:

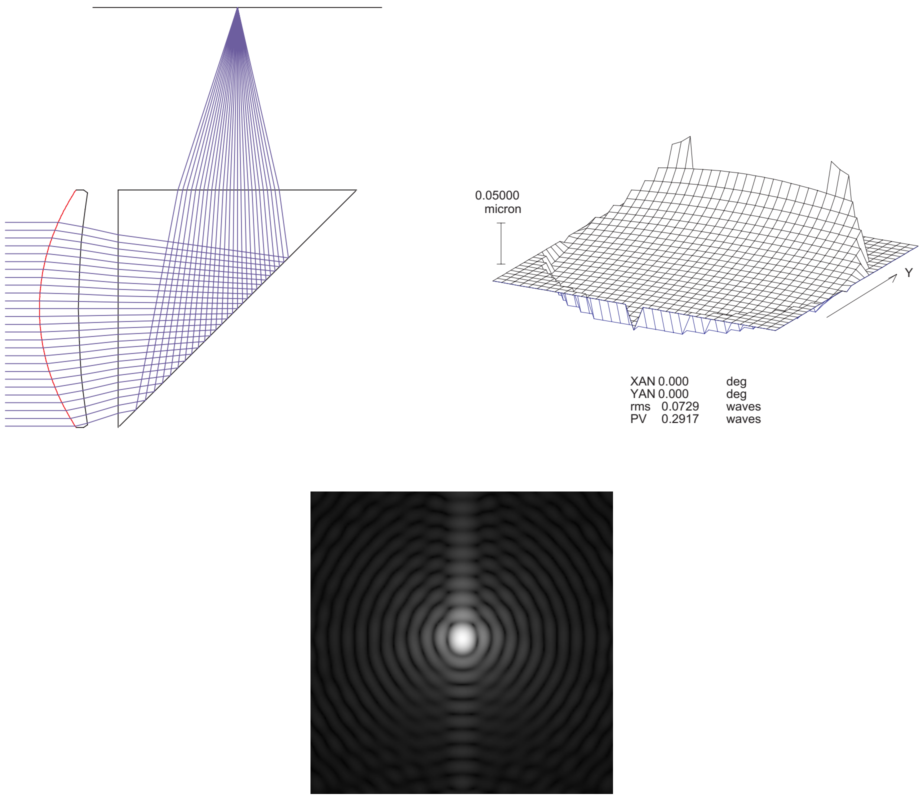


Figure 8.19: Total internal reflection example. See `$i\examples\misc\tir.otx`. Shows optical layout (left), wavefront (right) and point spread function (underneath).

`NSS si..j`	Converts a group of (previously entered) sequential surfaces into an equivalent NSS-range. The command automatically sets the correct tilt types on entrance port and exit port. The non-sequential surface range may also include the object surface (e.g. `NSS s0..8`), however, ray aiming is unlikely to work properly in this case. The `NSS s0..k` option is mainly useful in illumination applications with predefined rays (see also source rays, page 302).
`DEL NSS si..j`	Converts a group of non-sequential surfaces into sequential surfaces. Tilts and decenters are appropriately changed to reflect the sequential model. If there is more than one NSS range in an optical system, each range must be separately converted. Thus, it is not allowed to convert the whole surface range spanning the NSS sub-ranges.
`GL1 si..j glass-name`	Define glass on the "left side" (i.e. the side with negative local Z-axis) of the surfaces si..j
`GL2 si..j gl-name`	Define glass on the "right side" (i.e. the side with positive local Z-axis) of the surfaces si..j
`MXH si..j max_hits`	Maximum number of hits allowed for each surface in a NSS-range before declaring a ray failure. Note that each non-sequential surface may be assigned a different value for MXH. Ray tracing may also be terminated if a surface with absorbing (obstructing) property is hit.

Add "N" to the surface type (SUT) to specify a non-sequential surface. In *OpTaliX* non-sequential surfaces are always handled as decentered surfaces, even where all decenter/tilt data on a designated surface are zero. Thus, the surface type qualifier "D" must always be specified in conjunction with non-sequential surfaces. Consecutive non-sequential surfaces are defined in a NSS-range. The number of NSS-ranges within an optical system is unlimited. Fig. 8.20 shows the definition of non-sequential surfaces within the environment of sequential surfaces. A NSS-range is defined by an entrance port surface and an exit port surface. The entrance port surface is sequential, since it is the last surface of the sequential range. The exit port surface is non-sequential, since it is the last surface of the NSS-range. All surfaces entered between the entrance- and exit- port surface are non-sequential. Within a specified NSS-range, they may be entered in any order and may be arbitrarily tilted and decentered. The entrance port and exit port surfaces must have the tilt mode NAX, whereas for all other surfaces within a NSS-range the tilt mode `DAR` must be selected. NAX and BEN tilt modes are not allowed in a NSS-range!

8.13.1 Converting Sequential Surfaces to Non-sequential Surfaces

A range of sequential surfaces is converted to non-sequential surfaces by the command `NSS si..j`. This conversion automatically performs the following steps:

- set the glasses GL1 and GL2,
- set the tilt modes (TLM) of all surfaces inside the NSS-range to DAR,
- set the tilt modes (TLM) of entrance port and exit port to NAX,
- freezes all apertures (i.e. all apertures of surfaces inside the NSS-range are checked if a ray hits the surface inside the aperture (valid) or outside (invalid),

- refer all non-sequential surface vertex coordinates locally to the entrance port.

Also note that all surfaces in the range must be sequential surfaces. Ranges containing both sequential and non-sequential surfaces (before conversion is attempted) may lead to unexpected results, because they cannot be unambiguously converted.

8.13.2 Non-Sequential Coordinate System

The entrance port surface defines a new (local) coordinate system for all subsequent surfaces within a NSS-range. The origin is at the vertex of the entrance port surface. All non-sequential surfaces in a given NSS-range are entered by specifying their X, Y and Z decenters (XDE,YDE,ZDE) and their Euler rotation angles (ADE,BDE,CDE) with respect to this (local) coordinate system. Note that the separation (`THI` - command) has no meaning for NSS and is (must) therefore set to zero for all non-sequential surfaces. The `THI`-values are ignored within a NSS range. To specify the Z-location of a non-sequential surface relative to the entrance port coordinate system, use theZDE command instead.

The exit port surface, being of type non-sequential, defines a new coordinate system for the following sequential surfaces. The origin is at the vertex of the exit port surface.The entrance port surface and the exit port surface must not be mirror surfaces. The image surface must be sequential. NSS-ranges must not overlap.

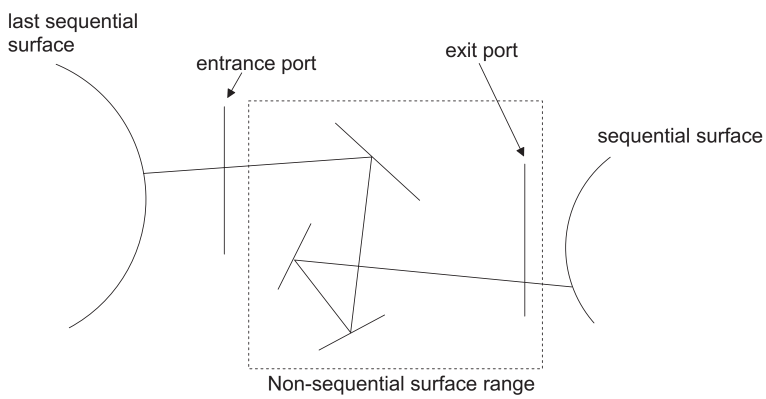


Figure 8.20: Definition of non-sequential surface range.

8.13.3 Glass Specification for Non-Sequential Surfaces

With a NSS-range, two glasses must be specified for each non sequential surface: The GL1 command specifies the glass on the ”left side” of the surfaces (the side containing the negative local Z-axis). GL2 specifies the glass for the opposite side (positive local Z-axis).

8.13.4 Transfer between Non-Sequential Surfaces

At a given surface, the program traces the intersection points of a ray with all other surfaces within a NSS-range. On the basis of this information, the transfer of a ray from one NSS to the next NSS is determined by the following criteria:

The optical path difference (OPD) must always be positive. ”Virtual” ray trace within a NSS-range is not allowed. If more than one surface with positive OPD exist, the surface with the smallest OPD is selected. It is not possible to ignore aperture violations (i.e. a ray falls outside of the valid aperture definition). The ray intersection point must always be within the valid aperture definition. A ray can hit the same surface two or more times in succession without having to transfer to another surface.

Entrance port surface :	The surface is always a sequential surface (since it is the last surface of the sequential range) and of type ”SD” or ”AD”. It defines a new axis and a new origin for all subsequent surfaces in the NSS-range. The tilt modus (TLM) is 1 (NAX), defining a new coordinate system, with its origin at the vertex of the entrance port surface.
Exit port surface :	The surface is always a non-sequential surface (since it is the last surface in the NSS-range) and decentered. (TLM = 1).
All other surfaces in NSS-range :	Surfaces are always referred to the origin (local vertex coordinates) of the entrance port surface.

8.13.5 Absorbing (obstructing) Surface Property

An absorbing property may be assigned to a non-sequential surface by declaring the primary aperture (pupil) p1 on a surface obstructing. For example,

```
cir s3 obs
```

sets the aperture type (property) of a circular aperture to obstructing. A ray which hits an absorbing (obstructing) surface is terminated on that surface.

8.13.6 General Notes on Non-Sequential Ray Tracing

The object surface and the image surface cannot be included in a non-sequential range.

It is possible to set up non-sequential ranges such that a ray that enters cannot exit. To avoid infinite ray trace loops, a maximum of hits on a given surface can be specified. See the MXH command, which provides a means to terminate non-sequential ray tracing after a certain number of surface hits.

Pupil finding may be unpredictable whenever the stop is a non-sequential surface or follows a non-sequential surface. It is recommended that the stop is placed ahead of any non-sequential range whenever possible.

8.14 Pickup Surfaces

The parameters of a surface can be made dependent on the setting of another surface. This is particularly useful in double pass or symmetrical systems where surface parameters, such as curvature, thickness, tilt/decenter, material, aspheric coefficients, are specified by a linear relationship with parameters on a preceding surface. In the simplest case, the value of a parameter can be directly copied (picked up) from another (preceding) surface, however, its value may also be negated or scaled by a factor.

A pickup is used to specify a particular surface parameter (such as a radius) by the value of another surface parameter of the same kind (e.g. another radius). The parameter to be picked up is an *independent* parameter, as its value can be independently specified. The parameter defined at the pickup surface is the *dependent* parameter as its value is permanently updated on changes of the independent parameter.

Pickups can be applied to a group of surface parameters, for example to *all* tilt/decenters (XDE, YDE, ZDE, ADE, BDE, CDE) as a whole, or may be individually specified for single parameters (for example XDE only).

Surface pickups are specified by the commands:

`PIK CUY sk|si..j pik_surf`
or
`CPI sk|si..j pik_surf`
	Curvature pickup. The curvatures of surface(s) i to j are picked up from surface `pik_surf`. Note that `pik_surf` is an integer number. Negative values of `pik_surf` pick up curvature with opposite sign. Pick-up offset is applied with the command `CPO` (see examples below). If the surface is aspheric, also the aspheric coefficients `A,B,C,D,E,F,G,H` and the conic constant `K` are picked up initially. However, you may change the aspheric coefficients pickup using the `API` command.

`CPO sk offset`	Pickup offset on curvature on surface `sk`.

`PIK XXX sk|si..j pik_surf`
or
`API [XXX] si..j pik_surf`
	Individual or group pickup of aspheric coefficients. `XXX` is optional with the API command. `XXX` specifies an individual pickup of one of the aspheric coefficients `K,A,B,C,D,E,F,G,H`, `CUX` and tilt/decenter pickups `XDE, YDE, ZDE`, `ADE, BDE, CDE`.
	If `XXX` is omitted, defines a group pickup on the designated surface(s). That is, all aspheric coefficients, including `CUX/RDX`, respectively tilt/decenter parameters are picked up from `pik_surf`. For example, `PIK ASP sk` define a group pickup on asphere coefficients, `PIK DEC sk` defines a group pickup of tilt/decenter values.
	
	Negative values of `pik_surf` pick up aspheric coefficients (except conic constant K) with opposite sign.
	
	Tilts and decenters may also be picked up by a multiplying factor, see `TPF` below.
	
	The `PIK` command requires explicit specification of the pickup parameter `XXX`. That is, `PIK` only allows individual aspheric pickups. Use the `API` form for defining aspheric group pickups.

`TPF [XXX] sk|si..j factor`	Individual or group tilt/decenter pickup factor. `XXX` is optional and specifies individual pickup of one of the tilt/decenter parameter `XDE, YDE, ZDE, ADE, BDE, CDE`. If `XXX` is omitted, the pickup factor is applied to all tilt/decenter values on the designated surface(s) `sk|si..j`.
	continued on next page

continued from previous page	
PIK THI si..j pik_surf or DPI sk|si..j pik_surf	Distance pickup.The distance (separation) of surface(s) i to j are picked up from surface pik_surf. Negative values of pik_surf pick up distance with opposite sign.
DPO sk|si..j offset	Pickup offset on distance on surface sk.
PIK GLA si..j pik_surf or MPI si..j pik_surf	Material (glass) pickup from surface pik_surf.
LIS PIK [sk|si..j] or PKL [sk|si..j]	List pickups.

Notes:

- The pickup commands CPI, DPI, API, TPI, and MPI are obsolete but are retained for backwards compatibility. It is recommended to use the PIK XXX forms instead.
- If the dependant surface is not already decentered, it is automatically converted to a decentered surface (i.e. adds the "D" qualifier to the surface type, see sect. 8.5).
- If the dependant surface is not already an aspheric surface, it is automatically converted to an aspheric surface (i.e. adds the "A" qualifier to the surface type, see sect. 8.5).

Pickups may be entered in any order and pickups can be chained. That is, a dependent parameter can become the independent parameter of an other pickup. For example, the independent pickups

```
PIK CUY s3 1
PIK CUY s5 1
```

are equivalent to chaining pickups

```
PIK CUY s3 1
PIK CUY s5 3
```

Pickups may also be defined in reverse order. For example,

```
PIK THI s3 4
```

Circular pickups are not allowed. For example,

```
PIK CUY s3 2
PIK CUY s2 3
```

More Examples:

`PIK CUY s5 4`	The curvature of surface 5 is picked up from surface 4.
`CPO s5 .001`	Curvature pickup offset = 0.001 at surface 5.
`DPI s3 -2`	The distance of surface 3 is picked up from surface 2 with opposite sign of surface 2.
`PIK THI s3 -2`	Same as above. The distance of surface 3 is picked up from surface 2 with opposite sign of surface 2.
`PIK ASP s3 1`	All aspheric coefficients A,B,C,D,E,F,G,H and the X-radius of curvature (except conic constant K) of surface 3 are picked up from surface 1. This is a group pickup, i.e. all aspheric coefficients (including CUX) are picked up from the designated surface (surface 1).
`PIK D s3 1`	Pick up aspheric coefficient `D` only. Disables group pickup on s3, if previously enabled.
`TPF ADE s3 1.23`	Tilt pickup factor for `ADE` tilt only. ADE tilts on surface 3 are multiplied by factor 1.23
`MPI s4 1`	Material properties of surface 5 are picked up from surface 1
`PIK GLA s4 1`	As above, material properties of surface 5 are picked up from surface 1
`PIK MAT s4 1`	As above, `GLA` and `MAT` are synonymous in material/glass pickups.

8.14.1 Group Pickups

Individual pickups may be grouped together as a single entity. This holds for tilt/decenter pickups and asphere pickups only. Group pickups are entered in the command line by

`TPI s3 1`	Pickup all decenter/tilt values at surface 3 from surface 1 (group pickup)
`PIK DEC s3 1`	As above, but with command syntax similar to Code V
`API s4 2`	Pickup all aspheric coefficients at surface 3 from surface 1 (group pickup)

In the surface editor, group tilt/decenter pickups are specified by selecting the "Decenter, Tilts" tab and entering the pickup surface in the "Pik" column, as shown in Fig. 8.21.

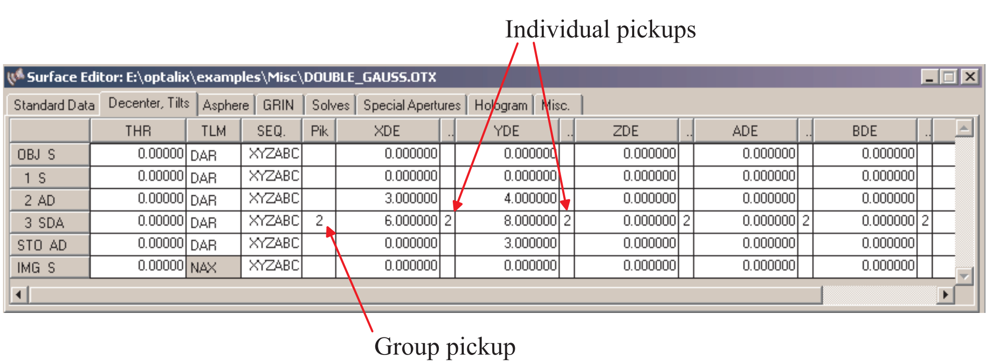


Figure 8.21: Defining group pickups for tilt/decenter parameter.

Note that individual pickups (shown in the columns right to each parameter column) reflect the setting of the group pickup. Specifying an individual pickup (see sect. 8.14.4) will automatically remove the group pickup on that particular surface.

8.14.2 Individual Pickups

Individual pickups are applicable only for tilt/decenter parameters and aspheric parameters. An individual pickup specifies a pickup for a single parameter only. For example,

`TPI YDE s3 1`	Pickup *only* `YDE` decenter value at surface 3 from surface 1 (individual pickup)
`PIK YDE s3 1`	As above, but with command syntax similar to Code V

Entering an individual pickup will automatically remove the group pickup on that particular surface.

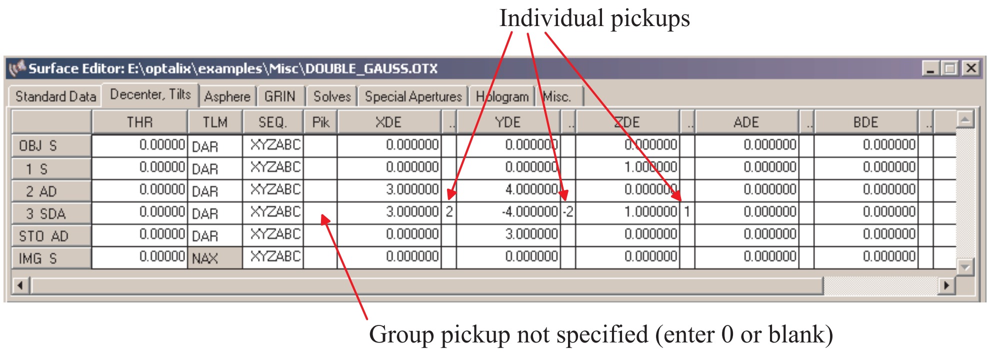


Figure 8.22: Defining individual pickups for tilt/decenter parameters.

8.14.3 Deleting Pickups

In the command line pickups are deleted by specifying ”0” (without quotation marks) as independent surface. For example,

`TPI s3 2`	Picks up tilt parameter at surface 3 from surface 2 (group pickup)
`TPI s3 0`	Deletes the (group) pickup defined above

In the surface editor, enter ”0” (without quotation marks) or a blank character in the appropriate column.

If a group pickup is deleted (”Pik” column in the surface editor, ”*Decenter, Tilt*” tab), the individual pickups will also be deleted.

8.14.4 Pickups and Solves

Pickups are evaluated prior to solves. That is, a solve on the same surface affecting the pickup parameter will override the pickup value. Consider the following example:

```
cpi s3 1
sol umy s3 -0.1
```

The first command `cpi s3 1` picks the curvature on surface 3 from surface 1. The second command, however, alters (solves) the curvature on surface 3 such that the paraxial marginal ray angle on surface 3 is -0.1. The pickup on surface 3 will be ineffective.

Note that aperture data cannot be picked up. This is due to multiple apertures being allowed on a surface.

8.14.5 Listing Pickups

Listing pickups is accomplished by the command `LIS PIK`. Here is a sample output:

```
PICKUPS :
  2  PIK  DEC    3   1.0000
  3  PIK  CUY    2   0.0000
  3  PIK  ASP    2
  3  PIK  THI    1   0.0000
  3  PIK  GLA    1
```

8.15 Solves

In contrast to linked (pick-up) surfaces, which only affect surface parameters, solves allow control of paraxial properties. Conditions for specifying a solve are, for example, holding the paraxial ray angle, the paraxial ray height or a certain paraxial ray incidence angle to a specified value. Solves will keep these requirements satisfied. For example, a paraxial ray angle solve at a surface will change its radius of curvature to maintain the specified ray angle. It is to be noted, that solves only apply to *paraxial* quantities. In optimization, this also makes it possible to reduce the number of independent variables.

`SOL sk solve_type param1 param2`	Sets a solve at surface `sk`. `solve_type` can be any 3-character string of
	`UMX` solve x-curvature on `sk` to produce a ray exit angle of `param1`
	`UMY` solve y-curvature on `sk` to produce a ray exit angle of `param1`
	`HMX` solve axial separation/thickness on `sk` to produce a paraxial height `param1` in the X/Z-plane at surface `sk+1`.
	`HMY` solve axial separation/thickness on `sk` to produce a paraxial height `param1` in the Y/Z-plane at surface `sk+1`.
	`UCY` Solve paraxial direction angle (in radians) of the chief ray at surface(s) si..j and zoom position zi..j with reference to a nominal input field angle of 1.0 radians.
	`HCY` Solve paraxial height of the chief ray at surface(s) si..j and zoom position zi..j
	`AMY` solve Y-curvature on `sk` to make it aplanatic to the paraxial marginal ray.
	`IMY` solve Y-curvature on `sk` for an angle of incidence (`param1`) of the marginal ray. (`param2`) is not used.
	`ET` solve axial thickness on `sk` for an edge thickness (`param1`) at semi-diameter `param2`.

`DEL SOL sk solve_type`	Delete solve of `solve_type` at surface `sk`. Example : `DEL SOL S4 UMY`
`LIS SOL [si..j]`	List solves
`PIM yes|no`	Paraxial image solve. `yes` adjusts the back focal distance to the *paraxial* image location, `no` keeps the back focus fixed.
`RED reduction_ratio`	Reduction ratio solve. Dynamically (i.e. as the optical system changes) set the paraxial object distance required to satisfy $$RED = \frac{ImageHeight}{-ObjectHeight} = -m \quad (8.47)$$ where m is the optical magnification. For an object at infinity $m = 0$, any other value establishes a finite conjugate system. See also the SET MAG command on page 42, which adjusts magnification statically (i.e. one-time adjustment) and the notes below.
`DEL RED`	Delete solve on reduction ratio. Leaves object distances unsolved.

Examples:

`sol umy s3 -0.1`	Solve curvature at surface 3 to produce a marginal ray angle of -0.1 (radians).
`sol s3 et 0.1 15`	Solve axial thickness at s3 such that an edge thickness of 0.1mm is achieved at a radial surface height of 15 mm.
`sol et s4 0 15`	Solve axial thickness at surface 4 for 0mm edge contact at a semi-diameter 15mm.
`red 2.0`	Solves for object distance to satisfy optical magnification -2.0.
`pim y`	Solves for paraxial image.

Notes:

- In zoomed systems, solves only apply to the first zoom position. The resulting value is then used in all zoom positions.
- A paraxial height solve (`HMY`) at the last surface (in order to hold the back focus) must not be used in conjunction with `PIM`, as `PIM` always sets the image surface to the paraxial focus, thus overriding the `HMY` solve.
- A paraxial height solve (`HMY`) should not be used in conjunction with a distance pick-up `DPI`. The height solve will always override the corresponding distance pick-up.
- A paraxial angle solve (`UMY`) should not be used in conjunction with a curvature pick-up `CPI`. The angle solve will always override the corresponding curvature pick-up.
- In optimization, solve parameter must not be used as a constraint. For example, a `UMY` solve and a `UMY` constraint at the same surface will add to the computing load and the constraint will be ignored.
- A `RED` solve is not accepted if paraxial ray solves are simultaneously set in the system. Exception: `ET` solve (edge thickness).

Solves will be updated each time a paraxial ray trace is required. The selected parameters (curvature, separation, ...) are forced to be dependent variables on system parameters, which are solved directly. No iteration is required. Referring to the paraxial quantities in Fig5.5, the relevant equations are for paraxial marginal ray angle ($\texttt{UMY} = u'$), solving for curvature c,

$$c = -\frac{u' - u}{(n' - n)h_a} \tag{8.48}$$

for paraxial marginal ray height at the subsequent surface ($\texttt{HMY} = h'$), solving for axial separation d,

$$d = \frac{h' - h}{u} \tag{8.49}$$

for aplanatic condition (`AMY`), solving for curvature c

$$c = \frac{(\frac{1 + n'}{n}) \cdot u}{h} \tag{8.50}$$

for angle of incidence ($\texttt{IMY} = i$), solving for curvature c

$$c = -\frac{i + u}{n \cdot h} \tag{8.51}$$

8.16 Tilted and Decentered Surfaces

The default condition is a centered system in which all surfaces are aligned along the optical axis. However, optical surfaces can be positioned arbitrarily in 3-D space. This is accomplished by tilting and/or decentering the coordinate system, in which the surface is described. The position of this coordinate system is specified by the XDE, YDE and ZDE parameters, its orientation is specified by the ADE, BDE and CDE parameters. By default, the positions/orientations of the (local) surface coordinate systems are always defined with respect to the global coordinate system (see DAR surface, section 8.17.1). Other forms of defining the local coordinate systems of subsequent surfaces areNAX (new axis) and BEN (bend at mirror). Tilt values are understood in a mathematical sense, i.e. positive tilts are counter clockwise (see also section 5.2.3 for a detailed definition of tilt orientation).

Tilts and decenter are non-commutative operations, i.e. tilting, then decentering results in a different coordinate system from decentering and then tilting. It is therefore important to specify the order in which tilts and decenter are applied to surfaces. The default condition is decenter first and then tilt.

ADE [si..j|sk] [zi..j|zk] alpha_tilt	Tilt angle (in degree) around X-axis . Positive tilts are counter clockwise.
BDE [si..j] [zi..j|zk] beta_tilt	Tilt angle (in degree) around Y-axis. Positive tilts are counter clockwise.
CDE [si..j] [zi..j|zk] gamma_tilt	Tilt angle (in degree) around Z-axis. Positive tilts are counter clockwise.
XDE [si..j] [zi..j|zk] x_dec	X-decenter
YDE [si..j] [zi..j|zk] y_dec	Y-decenter
ZDE [si..j] [zi..j|zk] z_dec	Z-decenter
GADE [si..j]	GRIN tilt around X-axis (This is an "ADE"-tilt of the GRIN material axis with respect to the surface vertex).
GBDE [si..j]	GRIN tilt around Y-axis (This is a "BDE"-tilt of the GRIN material axis with respect to the surface vertex).
GCDE [si..j]	GRIN tilt around Z-axis (This is a "CDE"-tilt of the GRIN material axis with respect to the surface vertex).
TLT si..j	Tilt surface range si..j. This command tilts a group of surfaces. The tilt angles and reference points are requested in a dialog box.
TLM [si..j] mode|DAR|NAX|BEN	Tilt mode, describes how the optical axis is defined after surface(s) si..j: mode = 0 : local decenter, (decenter and return, see DAR below.) mode = 1 : surface normal defines new optical axis, see NAX mode = 2 : optical axis follows law of reflection at mirror (see BEN) Alternatively, the tilt mode may be entered by the corresponding acronyms. For example, TLM s4 NAX TLM s4 BEN, etc.

continued on next page

continued from previous page	
`TSEQ [si..j] sequence`	Tilt sequence (order in which the decenter/tilt operations are applied). `sequence` is a character string of up to 6 characters. The permitted characters are: X = decenter-X Y = decenter-Y Z = decenter-Z A = tilt about X-axis B = tilt about Y-axis C = tilt about Z-axis The sequence of tilt/decenter operations is specified by the sequence of the characters. For example, BX performs tilt about Y-axis first, then decenter in X-direction. XYZABC is the default setting (i.e. decenter first, then tilts).
`TMAT si..j|sk glb_ref param1..12`	Define surface decenter and tilt by a transformation matrix $M_{i,j}$. The coordinate transformation may be referred to the coordinate system of a previous surface defined by `glb_ref`. Enter 0 for reference to the immediately preceding surface. Twelve parameters `param1..12` define the elements of the transformation matrix $M_{i,j}$. The matrix elements $m_{i,j}$ are entered row wise. An example is given in sect. 8.19.1. For a detailed description of transformation matrices see also section 8.19, page 112. Hint: Global transformation matrices defined in the system may be listed by the GSM command (page 177).
`DAR [si..j]`	Surface decenter and return (equivalent command is `TLM 0`).
`BEN [si..j]`	Surface bend, the optical axis follows the law of reflection at mirror (equivalent command is `TLM 2`).
`NAX [si..j]`	New optical axis. The surface normal defines the new optical axis for all subsequent surfaces (equivalent command is `TLM 1`).

Notes:

Surface decenter and/or tilts only take effect if a surface type qualifier ”`D`” is specified to the surface type. For example, a spherical tilted/decentered surface is set by the command `SUT s3 SD`. See also section 8.5 on page 65 for further details on surface types.

Consequently, tilts and/or decenter are deactivated for a particular surface by removing the ”`D`” qualifier from the surface type string.

Unlike CODE V, `DAR` is the default tilt mode in $OpTaliX$.

Paraxial analysis may not be correct for non-symmetric systems, since the paraxial ray trace (by definition) does not account for decenters and tilts.

8.16.1 Sign convention for tilted surfaces:

The tilt angles `ADE`, `BDE`, `CDE` are referred to rotations around the X-, Y- and Z-axis respectively. The sign of the tilts follows the mathematical convention, i.e. a positive sign means a counter-clockwise rotation, a negative sign is a clockwise rotation (see Fig. 5.1 on page 28).

8.17 Tilt Modes

The method of tilting and decentering surfaces is specified by the tilt mode. Three types of decentered and tilted surfaces are provided. They can be specified by the following commands:

`TLM si..j tilt_mode`	Define the tilt-mode of surface (surface range) `si..j`, where `tilt_mode` = 0 : The optical axis is not changed (see also `DAR` command), `tilt_mode` = 1 : The new optical axis is the surface normal of the actual surface (see `NAX` command), `tilt_mode` = 2 The new optical axis follows the light path on reflection on a mirror surface, without requiring an additional tilted dummy surface. (see `BEN` command). To be used only for mirror surfaces !!
`BEN si..j`	Bended surfaces. The new axis follows the law of reflection. See detailed description in section 8.17.3
`DAR si..j` `NAX si..j`	Decenter and Return. See detailed description in section 8.17.1 New axis. See detailed description in section 8.17.2

The following sections give a more detailed explanation on the definition of tilt modes.

8.17.1 Tilt Modus 0 : Decenter and Return (DAR)

The ”decenter and return” surface (Tilt modus = 0) is the default for tilted and decentered surfaces in *OpTaliX* . This option means that if a decentered surface is specified (either by `DAR` or `TLM` command), the subsequent surfaces refer to the coordinate system of the surface of the last `TLM = 1` or `TLM = 2` specifier. Example (Fig. 8.23):

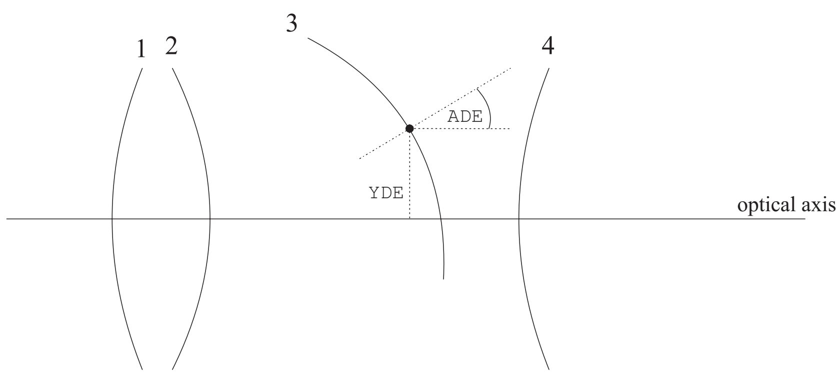


Figure 8.23: Definition of tilted/decentered surface with tilt mode (`TLM`) = 0

Surface 3 is decentered and tilted by the following command sequence:

`SUT S3 SD`	! surface type is spheric and decentered
`TLM s3 0`	! Tilt modus is 0 (not initially required because TLM 0 is the default, however, if the surface is in a different tilt mode (1 or 2), then this command must be explicitly given to set the surface to this mode).
`DAR s3`	Decenter and return surface. This command is synonymous to "`TLM s3 0`" as given above.
`YDE s3 2.5`	! Y-decenter of surface 3 is +2.5mm
`ADE s3 30.`	! Tilt around X-axis is 30 deg (counter clockwise since tilt is positive).

The subsequent surface 4 lies on the optical axis again, since surface 3 does not alter the optical axis. If a previous surface (for example surface 2) is a surface with TLM=1 or TLM=2, surface 4 (in the example of Fig. 8.23) refers to the previous surface surface 2). `DAR`-surfaces ("decenter and return") need not to be initially specified (since they are the default) but they may be explicitly forced by :

```
TLM si..j 0      or
DAR si..j
```

8.17.2 Tilt Modus 1 : Surface Normal defines new Axis (NAX)

The tilt modus 1 (see `TLM` command) applied to a surface s_x sets the coordinate system for all subsequent surfaces to the local coordinate system of the surface s_x. The new optical axis coincides with the normal of surface s_x. The command sequence to generate the configuration of Fig. 8.24 is:

`SUT S3 SD`	! surface type is spheric and decentered
`TLM s3 1`	! Tilt modus is 1 (axis follows normal of preceding surface)
`YDE s3 2.5`	! Y-decenter of surface 3 is +2.5mm
`ADE s3 30.`	! Tilt around X-axis is 30 deg (counter clockwise since tilt is positive).

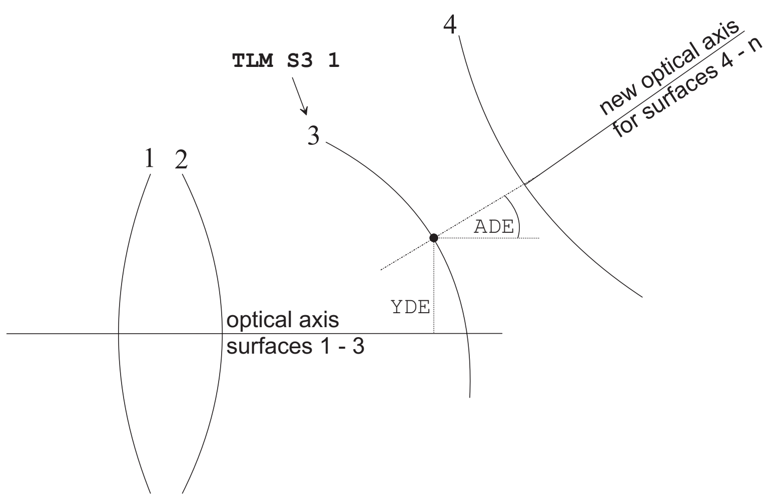


Figure 8.24: Definition of tilted/decentered surface with tilt mode (`TLM`) = 1, i.e. the optical axis follows surface normal of the preceding surface.

8.17.3 Tilt Modus 2 : Bend Surface (BEN)

The optical axis follows the reflection by a mirror. The `ADE, BDE` tilts are applied a second time after reflection in order to generate the new optical axis (see Fig. 8.25).

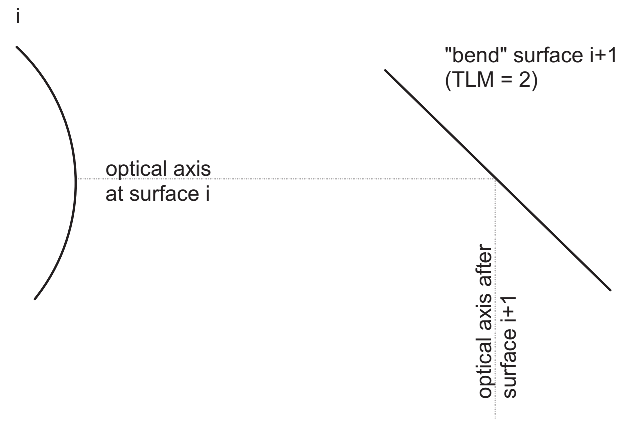


Figure 8.25: Definition of tilted/decentered surface with tilt mode (TLM) = 2, i.e. optical axis follows law of reflection.

8.17.4 Compound Tilts on a BENd Surface

A CDE tilt is automatically applied to compound tilts (ADE and BDE) on BEN type surfaces to keep the coordinate system properly applied. This rotates the system following a BEN surface so that a meridional ray will remain a meridional ray in the surfaces following the BEN surface. $OpTaliX$ generates the CDE, it cannot be entered manually. The relationship between CDE and (ADE,BDE) is

$$cos(CDE) = \frac{cos(ADE) + cos(BDE)}{1 + cos(ADE)cos(BDE)} \tag{8.52}$$

The calculated CDE is reported in the prescription data (see LIS command). If CDE is explicitly required on a BEN surface (for example in non-rotationally symmetric systems), BEN should be removed from this surface and the corresponding decenters/rotations should be applied to an extra dummy surface.

8.17.5 Reverse Decenter and Tilts (REV)

The REV command takes the decenter/tilt information on a surface and applies the inverse.

REV [si..j]	Takes decenter and tilts and applies it with inverse sign and reverse order.

8.18 Tilt Sequence

Any sequence of tilts and decenter may be specified. The default sequence is given in table8.21.

The tilt sequence is specified by a 6-character string, describing the sequence of decenter/tilts. For the default sequence, the tilt sequence would be ”XYZABC”, which corresponds to decenters $\Delta x, \Delta y, \Delta z$ and the Euler tilt angles α, β, γ. This means, that decenters are applied before tilts. The tilt/decenter sequence is entered by the command

TSEQ [si..j] string	Tilt sequence. Specify the sequence of tilts or decenters by a 6-character string. The default sequence is XYZABC.

Unlike in other optical design programs, an arbitrary sequence not only allows changing the order of tilts and decenter (e.g. decenter-after-tilt or tilt-after-decenter), it also permits arbitrary sequences within tilts or decenters (e.g. first around Z-axis, second around X-axis, third around Z-axis) and even mixed sequences of decenters and tilts.

It is important to note, that the order of tilts and decenters matters. The tilt sequence α, β, γ does not provide the same result as the tilt sequence β, α, γ or $-\alpha, -\beta, -\gamma$ with the same tilt/rotation angles, or any other arbitrary combination.

Tilting is performed internally by successive matrix multiplications, applied in the specified sequence. For example, the default tilt sequence (i.e. first tilt around X-axis, second around Y-axis, third around Y-axis) results in the following matrix multiplication (from right to left)[1]

$$M_z \cdot M_y \cdot M_x = \underbrace{\begin{bmatrix} \cos\gamma & \sin\gamma & 0 & 0 \\ -\sin\gamma & \cos\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{rotation\ around\ Z} \cdot \underbrace{\begin{bmatrix} \cos\beta & 0 & -\sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{rotation\ around\ Y} \cdot \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha & 0 \\ 0 & -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{rotation\ around\ X} \tag{8.53}$$

In case of uncertainties, it is always possible to spread the tilts out over several dummy surfaces.

8.19 Transformation Matrix

Surface tilts and decenters may also be defined by so-called transformation matrices. A transformation matrix gives a unique representation of location and orientation of a surface with respect to another surface or to a global coordinate system. Surface matrices can be entered by the TMAT command. Before entering transformation matrices we shall be concerned with the definition of a transformation matrix which is a 3x4 matrix of the form

$$M_{i,j} = \begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} & m_{1,4} \\ m_{2,1} & m_{2,2} & m_{2,3} & m_{2,4} \\ m_{3,1} & m_{3,2} & m_{3,3} & m_{3,4} \end{bmatrix} \tag{8.54}$$

A transformation matrix describes tilts and decenters of the vertex normals (i.e. the local coordinate system) of a surface with respect to another coordinate system which can be the coordinate system of a previous surface or of a global coordinate system.

Coordinate transformations are performed by tilts about the local X-axis (α), Y-axis (β), Z-axis (γ) and decenters (X, Y, Z). See also the definition of (local or global) coordinate systems in section 5.2, page 27. We also note that tilts are not commutative, that is, the order of tilts matters.

Tilt of a surface about the X-axis:

$$M_{i,j} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha & 0 \\ 0 & -\sin\alpha & \cos\alpha & 0 \end{bmatrix} \tag{8.55}$$

Tilt of a surface about the Y-axis:

$$M_{i,j} = \begin{bmatrix} \cos\beta & 0 & -\sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\beta & 0 & \cos\beta & 0 \end{bmatrix} \tag{8.56}$$

[1] Only 3x4 matrices are needed to fully describe surface tilt and decenters. In $OpTaliX$ these matrices are extended to 4x4 matrices. This is a marginal overhead but greatly simplifies matrix operations in a form suited for computers.

Tilt of a surface about the Z-axis:

$$M_{i,j} = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 & 0 \\ -\sin\gamma & \cos\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{8.57}$$

Lateral shift (decenter):

$$M_{i,j} = \begin{bmatrix} 1 & 0 & 0 & -X \\ 0 & 1 & 0 & -Y \\ 0 & 0 & 1 & -Z \end{bmatrix} \tag{8.58}$$

Example:
A 20° tilt about the X-axis plus a 5mm decenter in Y-direction results in the transformation matrix

$$M_{i,j} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.8660254 & 0.5 & -5 \\ 0 & 0.5 & 0.8660254 & 0 \end{bmatrix} \tag{8.59}$$

8.19.1 Entering Transformation Matrices:

A 20° tilt about the X-axis plus a 5mm decenter in Y-direction is entered as follows:

```
tmat s4 0  1 0 0 0 0 0.8660254 0.5 -5 0 -0.5 0.8660254 0
```

This is a very cryptic form of entering a transformation matrix. So, it is advisable putting this command in a macro file which allows arrangement of the data in a matrix-like fashion for better readability. We define the following text in a file, say `tmat.mac`

```
tmat s4 0    1.0000000     0.0000000     0.0000000      0.000000 &
             0.0000000     0.8660254     0.5000000     -5.000000 &
             0.0000000    -0.5000000     0.8660254      0.000000
```

and execute the macro from the command line with

`run tmat.mac`

Note the operator for line continuation (&) in the macro example above.
Hint: Global transformation matrices defined in the system may also be listed/controlled by theGSM command (page 177).

8.20 Tilting GRIN Material Properties

The alignment of the refractive index profile of GRIN materials is defined by the tilt mode of the surface, which specifies the GRIN material properties. By default, the GRIN profile is aligned along the optical axis, but it may be laterally and axially displaced using the `GXDE`, `GYDE`, `GZDE` commands or may be differently oriented using `GADE`, `GBDE`, `GCDE` commands. In addition, the tilt mode (`DAR` or `NAX`) of the surface holding the GRIN material properties also affects the orientation of GRIN media. The combination of *surface* tilts/decenters and *GRIN* tilts/decenters can be a complicated process. Figs. 8.26 and 8.27 illustrate the absolute orientation of GRIN profiles for various tilt modes.

Note that `BEN` (bend) surfaces are not allowed in conjunction with GRIN media. If the bend function is explicitly required inside GRIN media, it should be applied to an extra dummy surface.

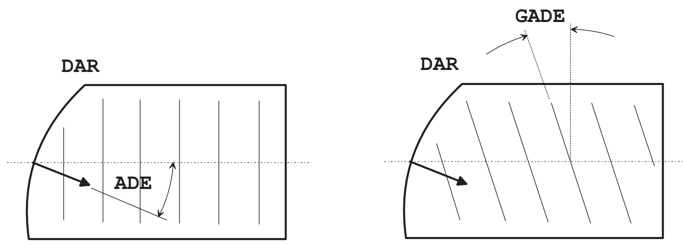


Figure 8.26: Orientation of GRIN profiles with DAR surfaces. Left: Since a DAR surface does not alter the optical axis, the index of refraction profile of the GRIN medium is also aligned along the optical axis. Right: Use `GADE`, `GBDE`, `GCDE` to tilt the GRIN profile with respect to the optical axis.

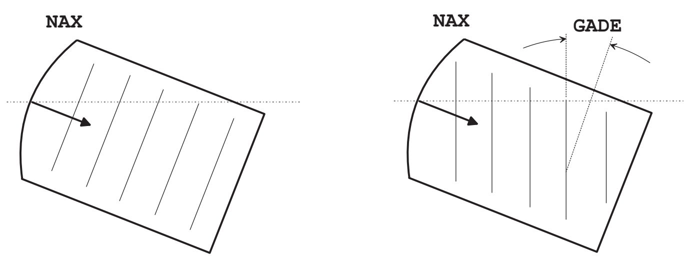


Figure 8.27: Orientation of GRIN profiles with NAX surfaces. Left: The vertex normal of a NAX surface defines the new optical axis. Thus, the profile of the GRIN medium is also aligned along the *new* optical axis. Right: Use `GADE`, `GBDE`, `GCDE` to additionally tilt the GRIN profile with respect to the *new* optical axis.

8.21 Global Referencing

Any surface may be referenced to the local coordinate system of a *previous* surface. In this manner it is possible to break the strict sequential order of surfaces (where the local coordinate system of a surface refers to its preceding surface), even though the ray trace is still sequential.

Referenced surfaces must always be `NAX`-surfaces, which means that a subsequent surface is referred to the local coordinate system of the referenced surface. On entering a surface reference, the tilt mode is automatically set to 1 (see `NAX`, `TLM`).

`GLB Si..j k`	Global surface reference. Coordinate data (`XDE`, `YDE`, `ZDE`, `ADE`, `BDE`, `CDE`) are interpreted for surface(s) `i..j` with respect to the coordinate system of a *preceding* surface k. Tilts and decentrations are recalculated to retain the physical position of the surface. A surface which is already globally referenced may be referenced to another surface by simply reapplying the `GLB` command with the new (preceding) surface number. Global referencing can be removed by `GLB si..j 0`
`REF Si..j k`	Specifies a global reference for surfaces `i..j` with respect to surface `k`. The difference to the `GLB` command is that thickness/tilt/decentration data are not altered. This may result in a change of the optical layout. Warning: The "`REF si..j`" command must not be confused with the command "`REF ref_w`" which changes the reference wavelength. Distinction is made by the surface qualifier `si..j` wether REF means a reference to another surface or the reference wavelength.
`THR si..k ref_thi`	Reference thickness of surface(s) i..j to surface k is `ref_thi`. The reference thickness is measured from the referenced surface (`k`) to the referencing surface (`i..j`). The referenced surface `k` must have a lower number than the referencing surface `i`.

To explain the concept of global referencing, let us consider a simple system with a moveable lens (see Fig. 8.28). Here, the image surface (surface 7) is referred to the local coordinate system of surface 1 instead of being referenced to its previous surface (surface 6), as would be expected in a strict sequential model. In this example, surface 7 is the *referencing* surface, surface 1 is the *referenced* surface. This is accomplished by two commands:

`GLB s7 1`	! Surface 7 is referenced to surface 1
`THR s7 194.7`	! The reference thickness of surface 7 to surface 1 is 194.7mm, i.e. surface 7 is 194.7mm separated from the local vertex of surface 1

Thickness 6 can no longer be freely altered by the user because it has become a *dependent* variable. Its value is computed from the thicknesses 1 to 5 and from the absolute position of surface 7 (the referencing surface). In the surface spreadsheet editor, the field for thickness 6 is greyed out.

We note,

- The position of a globally referenced surface is solely determined by the THR value on this surface,
- THR is an *independent* variable and is always specified as the separation *before* the referencing surface,
- the thickness before a globally referenced surface is always a *dependent* variable (greyed out in the surface editor).

We also note that specifying the reference thickness `THR` as the separation *before* the referencing surface is in contrast to the convention used in $OpTaliX$ (separations are always defined as distance from the local surface to the subsequent surface. Using this method, it is straightforward to change the separation between the doublet and the negative lens (thickness 4) without affecting the position of the image surface (as it would be in a model of strictly consecutive surface separations). Thus, we now have an elegant way to keep the overall length of the system constant without compromising or

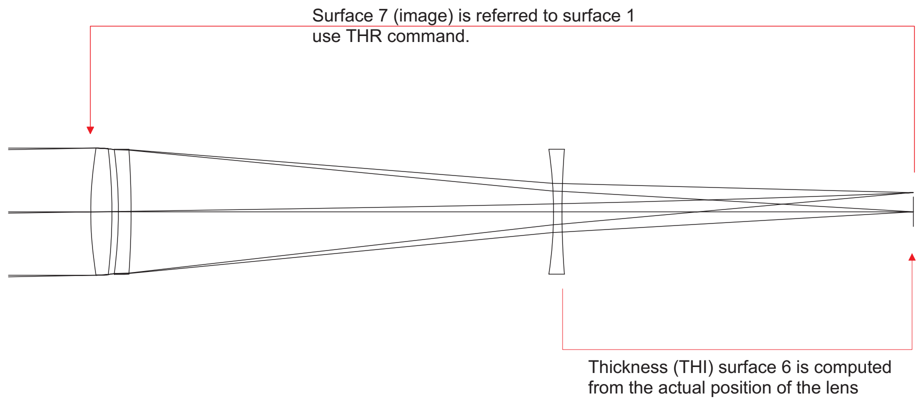


Figure 8.28: Definition of surface references.

altering other system parameters. Such a feature is particularly useful in zoomed (multi-configuration) systems where only one parameter needs to be controlled, instead of two (the separation before and after a lens group). We will now move the negative lens by changing thickness 4: The position of the lens relative to surface 4 has changed while the image plane position remains the same, because it is referred to the vertex of surface 1 which has not changed (Fig. 8.29).

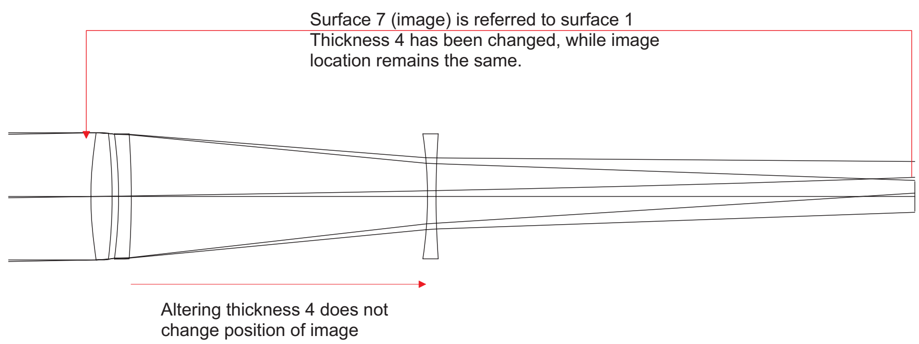


Figure 8.29: Definition of surface references.

From these considerations it is now evident, that a *referencing* surface has two axial thicknesses, `THR` and `THI`. While `THR` refers the vertex of a surface to the vertex of another (*previous*) surface, `THI` defines the thickness to the subsequent surface.

8.22 ”No-Raytrace” (NOR) Surface

A ”no-raytrace” (NOR) surface is a special surface that only transforms surface and ray coordinates, but does not actually trace rays to this surface. NOR surfaces are particularly useful for optical systems that contain tilts and decenters, however, they may also be favourably used in centered systems. NOR surfaces can be used to define non-optical reference points such as mechanical interfaces (flanges, polygon scanner rotation axis, etc) and refer optical surfaces and components to these points.

NOR surfaces require the surface type (SUT) ”X”, which is obligatory. The surface type qualifiers ”S”, ”A” or ”L” must not be contained in the surface type definition. The command

`NOR si..j`

does all the necessary actions to convert a surface to a ”no-raytrace” (NOR) surface. NOR surfaces can be centered or decentered. Thus, NOR surfaces are only defined by the surface types ”X” or ”XD”. Other surface types (such as the optional qualifiers M,I,H,G, ...) are allowed but have no effect on the ray trace.

Note that NOR surfaces do not return ray intersection data – for example as displayed in ray intersection plots (SPO RIS), single ray trace analysis (RSI) or in footprint analysis (FOO), because rays are not actually traced to the designated surface (only coordinates are transformed). Therefore, ray intersection coordinates cannot be made available on NOR-surfaces!

NOR surfaces, together with globally referenced surfaces, provide a powerful means for modelling opto-mechanical effects. Their use is explained on the example of a polygon scanner as shown in Fig. 8.30. We will use both global referencing and NOR surfaces to achieve the desired effect of moving polygon facets. In this model, surface 1 (the first surface of the $F\theta$ - lens) is globally referenced to surface 1, the stop surface. Since the $F\theta$ lens is tilted by 90° with respect to the entrance beam at surface 1, the desired position is accomplished by the commands

`glo s5 1`	! global reference of surface 5 to surface 1. Surface 5 is automatically converted to decentered type with tilt mode NAX.
`ade s5 90`	! tilt surface 5 by 90°
`yde s5 50`	! Y-vertex position of surface 5
`thr s5 25`	! reference thickness is 25mm, that is the Z-separation of the vertex of surface 5 from surface 1.

Surface 2 is located at the polygon’s rotation axis. The Z-position (along the optical axis) is defined by `THI s1`, the Y-position is entered by a `YDE s2` command. Surface 2 is of decenter type NAX, thus surfaces 3 and 4 refer to surface 2. Surface 3 is not really needed, it is only used in this example to better visualize the polygon center by plotting a cross. Surface 4 represents one mirror facet of the polygon. Its tilt and decenter values are appropriately set with reference to surface 2.

Note that the global decenter type on surface 5 avoids the need to apply a second tilt angle on a dummy surface to keep the geometry fixed.

Surfaces 2 and 3 are made NOR surfaces by the command `NOR s2..3`, thus avoiding that rays are apparently plotted ”through” the polygon facet mirror (surface 4) to surfaces 2,3. Surfaces 2 and 3 are solely used for transformational purposes and need not to be traced by real rays.

8.23 Gradient Index Surface

In inhomogeneous or *gradient-index* materials, rays no longer propagate in straight lines. The index of refraction changes as a function of the position of the ray in the medium. A gradient in the direction of the optical axis is called an axial gradient, a gradient perpendicular to the optical axis is called a radial gradient. Of course, there are mixed gradients possible, in which the index of refraction is a function of axial and radial position in the material.

A complete specification of a gradient surface must take into account the surface properties as well as the material properties. The qualifier ”`I`” must be added to the surface type to tell the program how refraction into the gradient-index material shall be performed. The material properties may be defined by either specifying a predefined gradient-index glass (e.g. `G14SFN` for Gradium^{TM} glass) or by entering gradient coefficients for each of the defined wavelengths.

The numerical solution of finding the exact ray path involves the choice of a step size ds. Choosing small values for ds will improve numerical accuracy, however, will also increase computing time.

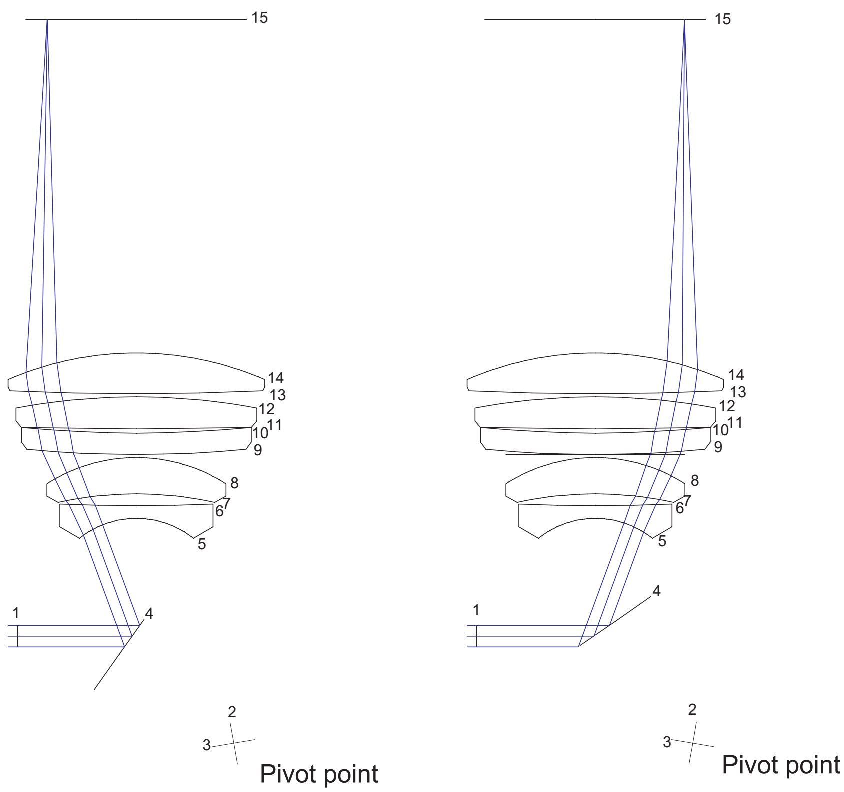


Figure 8.30: Use of global coordinates and NOR surfaces for modelling of a polygon scanner.

`SUT si..j string`	Surface Type (`SUT`) of surface(s) i..j is ”`string`”. Note that the surface type must contain at least a `S` (for spherical surfaces) or `A` (for aspheric surfaces) within string. Example: `sut s3 ai` (aspheric + GRIN)
`GLA si..j name`	Glass name. The specification of the glass name takes precedence over the base index specification . It automatically causes proper setting of the base index and the gradient index coefficients for all specified wavelengths. If glass name is omitted, at least the base refractive index (i.e. refractive index at the optical axis) must be given. There are predefined glasses for the gradient types `LPT`, `NSG` and `GLC` (see `GIT` command below). For all other types of gradients where the index profile is defined by manual entry of coefficients (`GIC`), the generic glass ”`GRIN`” must be used. Examples: `gla s2 g41sfn` (LightPath Gradium^{TM}-glass) `gla s2 grin` (generic GRIN-glass, enter coefficients with `GIC` command)
continued on next page	

continued from previous page	
`GIC si..j ci..j val`	Gradient index profile coefficients. The definition of the coefficients `c1, c2, c3, ...` in dependence on the GRIN-type (`GIT`) is given in table 8.23.16. In order to take effect, the glass type (GLA) must be `GRIN`. Other gradient index glasses (for example G51SFN from LightPath or SLW18 from Nippon Sheet Glass Corp., etc.) have predefined profile coefficients, which cannot be changed.
`GDISP sk disp_name`	Gradient index dispersion name. Defines which user defined dispersion characteristics is assigned to a gradient index material on surface `sk`. Note that the glass type (see GLA command) on surface `sk` must be `GRIN`. This command does not work with predefined gradient index materials. The dispersion coefficients are defined in the file `grindisp.asc` in the GLASSES directory and are then globally available. See also section 32.9 for a definition of the file format. Currently only LPT, URN, SEL, GLC and GRT dispersion models may be selected. If `disp_name` is left blank, dispersion properties are removed from the GRIN material on surface `sk`.
`GIS si..j step`	Gradient step size ds. The parameter step is the integration step along the ray path. See also the note at the end of this table.
`GZO si..j val`	Gradient Z-offset, for axial gradients only. Describes the axial offset of the vertex of the entrance surface from the zero-point of the axial index function.
`GADE [si..j] val`	GRIN tilt around X-axis (This is an "`ADE`"-tilt of the GRIN material axis with respect to the preceding surface).
`GBDE [si..j] val`	GRIN tilt around Y-axis (This is a "`BDE`"-tilt of the GRIN material axis with respect to the preceding surface).
`GCDE [si..j] val`	GRIN tilt around Z-axis (This is a "`CDE`"-tilt of the GRIN material axis with respect to the preceding surface).
`GIT si..j string`	Gradient Index Type. The following types of gradient index profiles are available: SEL : SELFOC gradient GLC : Gradient Lens Corporation Gradient (EndoGRIN^{TM}) GRT : Radial gradient from Grintech, Jena LPT : LightPath GRADIUM axial gradient AXG : Linear axial gradient URN : University of Rochester gradient LUN : Luneberg Lens SPG : Spherical gradient MAX : Maxwell's Fisheye Example: `git s3 lpt` ! LightPath Gradium^{TM}-glass

continued on next page

continued from previous page	
MXG si..j|sk max_grin_iterations	Maximum number of iteration steps in the GRIN medium defined on surface(s) si..j|sk. Gradient index ray trace may loop infinitely if improper coefficients are specified, in particular for user defined profiles. Note that each gradient index surface may be assigned a different value for MXG. Setting MXG to values other than 0 provides a means to prematurely terminate ray tracing. MXG si..j|sk 0 disables limit checking on that particular surface(s).

Note on optimal gradient-index step (GIS): The accuracy and speed of gradient-index ray tracing is determined by the choice of step length. The default step size in $OpTaliX$ is set to 0.1 mm, which is a good compromise for various gradients. It is recommended to test the step size until an acceptable accuracy is achieved for a particular system and, if required, to be reduced accordingly. As a guideline, the step size may be as large as 1mm for weak gradients without the need to sacrifice accuracy in geometrical analysis. For diffraction analysis, however, typically smaller step sizes are required for acceptable accuracy. In cases, where a large step size (> 0.1mm) is selected, the program automatically reduces step size to 0.1mm in all diffraction analyses and restores the user selected step size afterwards.

Aperture checking for gradient index surfaces may be accomplished by assigning the fixed aperture flag FHY (see section 8.32.3) on the first surface of a GRIN lens. Rays inside the gradient material are blocked if their radial coordinate exceeds the aperture of the entrance surface.

Example Commands:

SUT s3 AI	! surface type of surface 3 is AI (aspheric, gradient index)
GLA s3 SLN20	! glass type at surface 3 is SLN20
GIT s3 SEL	! gradient index type at surface 3 is SEL (=SELFOC lens)
GIC s3 c4 0.42	! gradient index coefficient No.4 = 0.42 for all wavelengths
GZO s3 1.2	! gradient z-offset = 1.2 mm
MXG s3 200	! Limit number of iterations in GRIN medium defined on surface 3 to 200.

Example 1: Setting up a LightPath GRADIUM^{TM} gradient:

Defining LightPath GRADIUM^{TM} gradients only requires specification of the LightPath glass name, e.g.

```
GLA s2 G14SFN
```

All other parameters (gradient index type, surface type) are automatically determined. In addition, when switching back from a LightPath GRADIUM glass to a homogeneous glass, the gradient index type and the surface type are automatically reset.

Example 2: Defining gradient material with coefficients:

If a predefined gradient material does not exist or if a user profile shall be simulated, the index profile may be defined by entering profile coefficients directly. The coefficients depend on the gradient type chosen, as explained in Eq's. 8.66 to 8.85 and in table 8.23.16 (page 128).
For example, a "University of Rochester (URN)" gradient consists of axial and radial coefficients, thus allowing definition of a mixed gradient.

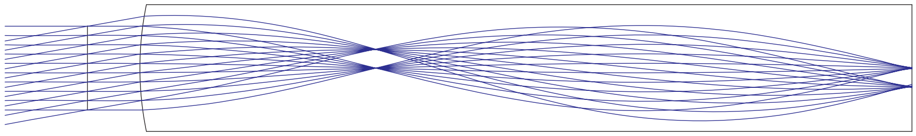


Figure 8.31: Gradient index raytrace, shown for a radial index profile.

`gic s3 c1 1.65`	defines 1st profile coefficient (the base index n_{00})
`gic s3 c2 -0.035`	defines 2nd profile coefficient (the linear axial slope n_{01})

8.23.1 Editing GRIN Coefficients on a Surface

In addition to selecting own GRIN dispersion models via the GDISP command, coefficients may also conveniently edited in a dialog called from the surface editor. The major difference to the GDISP option is that the GRIN material is only defined on a particular surface in a lens and is therefore not globally available as with predefined GRIN materials.

In order to enable this option, the glass name on the surface must be 'GRIN'. No other name is allowed. Then select the GRIN-tab in the surface editor and click on the appropriate button in the 'Coeff' column. This opens a dialog as shown in Fig. 8.32. You may now select a predefined dispersion characteristics (as defined in '`$ i\glasses\grin.asc`' for catalogue GRIN's or in '`$ i\glasses\grindisp.asc`' for user defined dispersions) or you may select the 'USER' option in the list box. If 'USER' is selected, the dispersion coefficients can be edited, otherwise (for predefined dispersions) the coefficients field is disabled (greyed out). The name 'USER' in the list box may be changed at wish.

'User' defined profiles and dispersions always pertain to the particular surface from which the dialog was called. The 'USER' definitions are stored with the optical system and are therefore only 'locally' available within that particular optical system.

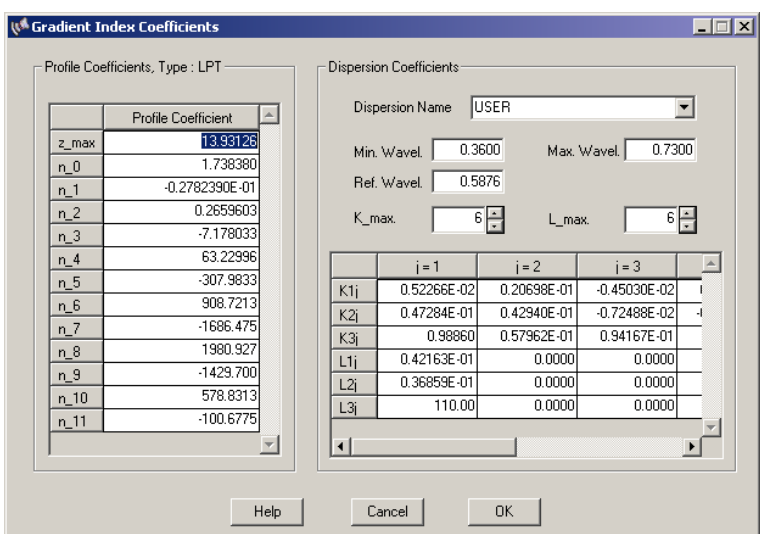


Figure 8.32: Editing GRIN coefficients on a particular surface.

Warning: Altering GRIN coefficients should be done with great care. In case of improper data, the program may hang in an infinite loop because no exit surface is found. It is prudent to reduce the maximum allowable number of GRIN steps on a surface before testing or experimenting with new profiles. See the MXG command.

8.23.2 Ray-Tracing Method

Tracing rays in inhomogeneous (gradient) index material is obtained by solving the ray equation [49] :

$$\frac{d^2\mathbf{r}}{dt^2} = n\nabla n \tag{8.60}$$

with

$$t = \int \frac{ds}{n}; \qquad dt = \frac{ds}{n} \tag{8.61}$$

where $\mathbf{r}$ is the position vector of a point on the ray, ds is an element of the arc along the ray. Equation 8.60 has three components which can be solved simultaneously by using three-element arrays:

$$\mathcal{R} \equiv \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{8.62}$$

$$\mathcal{T} = \begin{pmatrix} T_x \\ T_y \\ T_z \end{pmatrix} = n \begin{pmatrix} dx/ds \\ dy/ds \\ dz/ds \end{pmatrix} \tag{8.63}$$

and

$$\mathcal{D} = n \begin{pmatrix} \partial n/\partial x \\ \partial n/\partial y \\ \partial n/\partial z \end{pmatrix} \tag{8.64}$$

It is obvious that the components of the vector $\mathcal{T}$ are the three optical direction cosines α, β, γ of a ray. Equation 8.60 can be written as the following matrix equation:

$$\frac{d^2R}{dt^2} = \mathcal{D}(\mathcal{R}) \tag{8.65}$$

Equation 8.65 is solved by the Sharma method [49] with the initial condition that at $\mathcal{R} = R_0(x_0, y_0, z_0), \mathcal{T} = T_0$ which is a known quantity. Starting from the known point (R_0, T_0), one can generate successively $(R_1, T_1), (R_2, T_2), \cdots (R_n, T_n)$, i.e., one can trace a ray through the medium using the *Runge-Kutta* algorithm.

8.23.3 SELFOC^{TM} Lens (`SEL`)

The radial gradient of SELFOC^{TM} lenses is given by:

$$n(r) = n_0 \left(1 - \frac{A}{2} r^a\right) \tag{8.66}$$

with

$$\begin{aligned} a &= 2 \\ A &= \frac{2 \cdot \Delta n}{n_0 \cdot r_k^a} \end{aligned} \tag{8.67}$$

In SELFOC^{TM} material the refractive index decreases *parabolically*, which is defined by $a = 2$ in eq. 8.66. Substituting eq. 8.67 into eq. 8.66, we obtain, after some simple manipulations, the more general form

$$n(r) = n_0 - \underbrace{\frac{2 \cdot \Delta n}{n_0 \cdot r_k^a}}_{A} \cdot \frac{n_0 r^a}{2} \tag{8.68}$$

See also section 13.6.4 for a list of available GRIN profiles from NSG.

The wavelength dependency (dispersion) of SELFOC^{TM} glasses is given by the equations [39]

$$n_0(\lambda) = c_1 + \frac{c_2}{\lambda^2} \tag{8.69}$$

$$\sqrt{A}(\lambda) = k_{11} + \frac{k_{12}}{\lambda^2} + \frac{k_{13}}{\lambda^4} \tag{8.70}$$

8.23.4 Gradient Lens Corporation (GLC)

The radial gradient of ”EndoGRIN” rod lenses provided by ”Gradient Lens Corporation” is:

$$n\left(r\right) = n_{00} + n_{10} r^2 + n_{20} r^4 \tag{8.71}$$

where $r^2 = x^2 + y^2$.

The coefficients n_{00}, n_{10}, n_{20} are wavelength dependent:

$$n_{ij}\left(\lambda\right) = A + B\lambda^2 + \frac{C}{\lambda^2} + \frac{D}{\lambda^4} \tag{8.72}$$

where λ must be given in nm. For each n_{00}, n_{10}, n_{20} there exist a separate set of parameters A, B, C, D. See also section 13.6.4 for a list of available GRIN profiles from Gradient Lens Corp.

8.23.5 Grintech Radial Gradient (GRT)

The radial gradient profile of rod lenses manufactured by Grintech, Jena (Germany) is defined as

$$n(r) = n_0 \cdot sech(gr) = \frac{n_0}{cosh(gr)} \tag{8.73}$$

where $r^2 = x^2 + y^2$ and g is a material constant. The dispersion of n_0 is modelled with good accuracy by

$$n_0(\lambda) = 1.61189 + \frac{7614[nm^2]}{\lambda^2} \tag{8.74}$$

See also section 13.6.4 (page 225) for a list of available GRIN profiles from Grintech.

8.23.6 Grintech Cylindrical Gradient (GRC)

The gradient profile of cylindrical lenses manufactured by Grintech, Jena (Germany) is defined as

$$n(y) = n_0 \cdot sech(g \cdot y) = \frac{n_0}{cosh(g \cdot y)} \tag{8.75}$$

where y is the height in Y-direction and g is a material constant. In the X-direction, the g-coefficient is assumed zero and the index of refraction is n_0. The dispersion of n_0 is modelled with good accuracy by

$$n_0(\lambda) = 1.61189 + \frac{7614[nm^2]}{\lambda^2} \tag{8.76}$$

See also section 13.6.4 (page 225) for a list of available GRIN profiles from Grintech.

8.23.7 Linear Axial Gradient (AXG)

The refractive index is a linear function of the axial distance z:

$$n(z) \;=\; n_0 + a \cdot z \tag{8.77}$$

with : n_0 = base index at the optical axis
a = linear axial coefficient

8.23.8 LightPath Technologies Gradient (LPT)

LightPath Technologies, Inc. are using a 11^{th} order axial profile for their proprietary GRADIUM^{TM} glasses:

$$n\left(z\right) = \sum_{i=0}^{11} n_i \left(\frac{z}{z_m}\right) = n_0 + n_1\left(\frac{z}{z_m}\right)^1 + n_2\left(\frac{z}{z_m}\right)^2 + n_3\left(\frac{z}{z_m}\right)^3 + n_4\left(\frac{z}{z_m}\right)^4 + \ldots\ldots + n_{11}\left(\frac{z}{z_m}\right)^{11} \tag{8.78}$$

where the coefficients n_0 to n_{11} are given in ascending order at the wavelength $\lambda_{ref} = 587.6nm$. z is the distance into the blank from either the high index or low index surface. The value of z ranges from 0 to the maximum value z_m.
The wavelength dependence is modelled by a modified Sellmeier formula

$$n(\lambda)^2 - n(\lambda_{ref})^2 = \sum_i \frac{K_i \lambda^2}{\lambda^2 - L_i} \tag{8.79}$$

where $n(\lambda_{ref})$ is the index at the reference wavelength and the constants are functions of n

$$K_i = \sum_{j=1}^{k} K_{ij} \left[n(z, \lambda_0)\right]^{j-1} \tag{8.80}$$

and

$$L_i = \sum_{j=1}^{k} L_{ij} \left[n(z, \lambda_0)\right]^{j-1} \tag{8.81}$$

The wavelength λ is given in microns. See also section 13.6.4 for a list of available GRIN profiles from LightPath Inc.

8.23.9 University of Rochester Gradient (URN)

$$n(r,z) = n_{00} + n_{01}z + n_{02}z^2 + n_{03}z^3 + n_{04}z^4 + n_{10}r^2 + n_{20}r^4 + n_{30}r^6 + n_{40}r^8 \quad (8.82)$$

with :
$r(x,y)^2 = x^2 + y^2$
n_{00} = base index
n_{0i} = axial coefficients
n_{i0} = radial coefficients

Dispersion properties can be assigned to URN gradient index profiles by specifying a *dispersion name* as provided in the GDISP command. The same set of dispersion coefficients as for the LightPath material is used. In particular Eqs. 8.79 to 8.81 apply. Dispersion coefficients must be stored in the file `grindisp.asc` in the GLASSES directory.

Example for setting up a generic URN profile with dispersion modelling:

```
gla s1 GRIN          ! generic name for gradient index glass
git s1 URN           ! gradient index type is URN
gic s1 c1 1.678      ! first profile coefficient
gic s1 c2 0.00345    ! second profile coefficient
gic ...              ! repeat coefficients entry if required
gdisp s1 GLAK        ! the dispersion name is GLAK (must exist in file grindisp.asc).
```

8.23.10 Luneberg Gradient (LUN)

$$n^2(p) = n_0^2 \left(2 - \frac{p^2}{a^2}\right) \quad (8.83)$$

with: $p^2 = x^2 + y^2 + (z-r)^2$

8.23.11 Spherical Gradient (SPG)

$$n(p) = n_0 + n_1(r-p) + n_2(r-p)^2 + n_3(r-p)^3 + n_4(r-p)^4 \quad (8.84)$$

with: $p^2 = x^2 + y^2 + (z-r)^2$

8.23.12 Maxwells's Fisheye (MAX)

$$n(p) = \frac{n_0}{1 + \frac{p^2}{a^2}} \quad (8.85)$$

with: $p^2 = x^2 + y^2 + (z-r)^2$

8.23.13 User-Defined Gradient Index (UDG)

User-defined gradient index profiles can be programmed in FORTRAN or C in a user-written subroutine. The default name for a user-defined gradient index profile is ”usergrn”.

The usergrn subroutine must compute the refractive index at any point (x,y,z) in the glass, i.e., $n = n(x, y, z)$. The subroutine must also explicitly evaluate the derivatives of the index, dn/dx, dn/dy, and dn/dz.

Coefficients of a user-defined gradient are specified by the UDG command:

UDG si..j|sk ci..j|ck coeff_1 coeff_2 ...	Enter user-defined coefficients c..j on surface(s) si..j, respectively surface sk. Requires surface type ”I” (for gradient **I**ndex) on that surface.

OpTaliX provides a sample subroutine in both FORTRAN and C programming languages. It is found in the directories

\optalix\usergrn\Fortran	for FORTRAN
\optalix\usergrn\C	for C/C++

with appropriate subdirectories for Lahey/Fujitsu FORTRAN, Intel FORTRAN, Compaq Visual FORTRAN and Microsoft Visual C compilers. Note that the subroutine name must be exactly ”usergrn” in small characters and no other name is permitted. The usergrn subroutine can also, if needed, call other subroutines or read data files. The usergrn subroutine that you write in FORTRAN or C must have the following parameters:

```
usergrn((isur,sdata,x,y,z,wvl,rindx,gx,gy,gz,i_err)
```

where:

isur	Current surface number for which the index function and the derivatives are to be evaluated. This is an input parameter which may be used to distinguish between various algorithms on different surfaces. If only one UDG type surface is used, this parameter is normally not needed. See also the note below.
sdata	Data array with 91 elements for passing data between *OpTaliX* and the usergrn subroutine. The elements of data correspond to the UDG coefficients C1 to C91.
x,y,z	Coordinates at a point along the ray, with z along the optical axis.
wvl	Wavelength, in microns.
rindx	The calculated index of refraction at the point (x,y,z).
gx,gy,gz	A three-element output vector with the x, y, and z components of $\nabla(n)$ at the point (x,y,z).
i_err	Error flag. It should be set to 0 if there is no error generated and set to 1 otherwise.

Notes:

- Only one usergrn subroutine can be linked to *OpTaliX* at one time. Therefore all user-defined gradients in the optical system must use the same usergrn subroutine. However, it is possible to program more than one UDG description with different coefficients in the same usergrn subroutine. The parameter isur designates the surface number currently in use for evaluating index of refraction and derivatives.

- If the user-defined gradient has any axial (z) dependence, then the value of "brind" will be negative after a reflector.

8.23.14 Default usergrn Subroutine

The default UDG in *OpTaliX* is the "University of Rochester" type gradient index. The index profile is given by Eq. 8.82 on page 125. The FORTRAN source code of the usergrn subroutine is as follows:

```
      subroutine usergrn(isur,sdata,x,y,z,wvl,rindx,gx,gy,gz,i_err)
!
!     Evaluate the function and its derivatives of a user defined GRIN surface
!     The function is of the form n(x,y,z) where (x,y,z) are the cartesian
!     coordinates of a point in the gradient.
!
!     The example GRIN profile is the "University of Rochester" gradient:
!     rindx = sdata(1) + sdata(2)*z + sdata(3)*z^2 + sdata(4)*z^3 + sdata(5)*z^4 +
!             sdata(6)*r^2 + sdata(7)*r^4 + sdata(8)*r^6 + sdata(9)*r^8
!     where r^2 = x^2 + y^2
!
!     Parameters:
!     ----------
!     isur       : surface number                                    (input)
!     sdata(91)  : Array containing the user-defined GRIN parameters (input)
!                  For example, sdata(1) is the value entered with the
!                  command UCO C1.
!     x,y,z      : Coordinates of the current position of the ray with
!                  respect to the origin of the surface              (input)
!     wvl        : wavelength (in microns)                           (input)
!     rindx      : The calculated index of refraction at (x,y,z)     (output)
!     gx,gy,gz   : Gradient (derivatives) at coordinates (x,y,z)     (output)
!                  i.e. dn/dx, dn/dy, dn/dz
!     i_err      : Error flag (0 = no error, 1 = error)              (output)
!                  Note: The error flag must be properly set by the user
!
!     Notes:
!     -----
!     The user will typically substitute his own FORTRAN code for a
!     particular surface.
!
!     More than one surface description can be programmed in this subroutine.
!     Use the "isur" parameter to distinguish between surfaces and
!     determine the interpretation of the coefficients stored in "sdata"
!
      dll_export usergrn
      integer          :: i_err,isur
      double precision :: x,y,z,gx,gy,gz,rindx,wvl,sdata(91)
      double precision :: rad2,t1,t2,tabl
!
      i_err = 0
!                            University of Rochester Gradient
      rad2 = x*x + y*y
!
! Evaluate index of refraction:
      t1 = z *(z  *(z  *(z  *sdata(5)+sdata(4))+sdata(3))+sdata(2))
      t2 = rad2*(rad2*(rad2*(rad2*sdata(9)+sdata(8))+sdata(7))+sdata(6))
      rindx = sdata(1) + T1 + T2
      if(rindx.lt.1.0d0) then
         i_err = 1
         rindx = 1.0d0
      endif
!
! Evaluate gradient :
      t1 = rad2*(rad2*(rad2*8.d0*sdata(9) + 6.d0*sdata(8)) + 4.d0*sdata(7))
      tabl = t1 + 2.d0*sdata(6)
      gx = tabl * x
      gy = tabl * y
      gz = z*(z*(z*4.d0*sdata(5) + 3.d0*sdata(4)) + 2.d0*sdata(3)) + sdata(2)
```

```
!
      return
      end
```

8.23.15 Compiling and Linking `usergrn`

$OpTaliX$ supports the Lahey/Fujitsu FORTRAN, Compaq Visual FORTRAN, Intel FORTRAN and the Microsoft Visual C++ compilers. All supported compilers are 32 bit versions. The 16 bit versions are not supported. All compilers must have version numbers equal or higher as listed below. References to compiler specific instructions are given in the last column.

Manufacturer	**Compiler Version**	**See Section**
Lahey Fujitsu	FORTRAN-95, version 5.7 or later	8.30.3
Compaq	Visual FORTRAN, version 6.6 or later	8.30.4
Intel	FORTRAN-95, version 7.1 or later	8.30.4
Microsoft	Visual C/C++, version 5.0 or later	8.30.6

8.23.16 GRIN - Coefficients Overview

The parameter `C1` to `C10` are the coefficients which describe the index *profile* of a gradient index material. To be used in conjunction with the GIC command. The meaning of each profile coefficient depends on the GRIN-type and is defined as follows:

Type	**Equation**	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
`SEL`	$n_0(\lambda) = c_1 + \frac{c_2}{\lambda^2}$ $\sqrt{A}(\lambda) = k_{11} + \frac{k_{12}}{\lambda^2} + \frac{k_{13}}{\lambda^4}$	c_1	c_2	k_{11}	k_{12}	k_{13}					
`GLC`	$n(r) = n_0 + n_1 r^2 + n_2 r^4$	n_0	n_1	n_2							
`GRT`	$n(r) = n_0 \cdot sech(gr)$	n_0	g								
`GRC`	$n(y) = n_0 \cdot sech(gy)$	n_0	g								
`AXG`	$n(z) = n_0 + a \cdot z$	n_0	a								

continued on next page

continued from previous page											
LPT	$n(z) = n_0 + n_1 \left(\frac{z}{z_m}\right)^1 + n_2 \left(\frac{z}{z_m}\right)^2 + n_3 \left(\frac{z}{z_m}\right)^3 + \ldots + n_{11} \left(\frac{z}{z_m}\right)^{11}$	z_m	n_0	n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8
		n_9	n_{10}	n_{11}							
URN	$n(r,z) = n_{00} + n_{01}z + n_{02}z^2 + n_{03}z^4 + n_{04}z^4 + n_{10}r^2 + n_{20}r^4 + n_{30}r^6 + n_{40}r^8$	n_{00}	n_{01}	n_{02}	n_{03}	n_{04}	n_{10}	n_{20}	n_{30}	n_{40}	
LUN	$n^2(p) = n_0^2 \left(2 - \frac{p^2}{a^2}\right)$ with $p^2 = x^2 + y^2 + (z-r)^2$	n_0	a	r							
SPG	$n(p) = n_0 + n_1(r-p) + n_2(r-p)^2 + n_3(r-p)^3 + n_4(r-p)^4$	n_0	n_1	n_2	n_3	n_4					
MAX	$n(p) = \frac{n_0}{\left(1+\frac{p^2}{a^2}\right)}$ with $p^2 = x^2 + y^2 + (z-r)^2$	n_0	a	r							

8.24 Light Pipe, Step Index Fiber

Light pipes and step index fibers are handled in an identical manner. Rays enter a tube (being either solid or hollow) and reflect from the walls an indeterminate number of times until they emerge. Circular and rectangular cross sections are supported. Both end surfaces may have any form (spherical, aspheric, with grating, with surface deformation, etc) and may also be arbitrarily tilted.

Fibers and light pipes are formed by extruded surfaces. The aperture boundary of the entrance surface defines the diameter (= 2*aperture radius) of the tube and the axial separation to the next surface (the end surface) defines the length of the tube. Thus, the rod conforms to the aperture shape (circular or rectangular) of the entrance surface. In addition, two materials (glasses) must be provided at the entrance surface for core and cladding (use GLA and GL2 commands). The only difference between a light pipe and a step index fiber is in the material for the cladding. In a light pipe, the index of refraction of the cladding is 1, whereas for a step index fiber it is > 1.

The entrance surface of light pipes must have the surface type "P" in addition to the "S" (spherical) or "A" (aspheric) base shape. Example command: `sut s3 sp`

In a tapered fiber, the cone angle is defined by the semi-diameters of entrance surface and exit surface

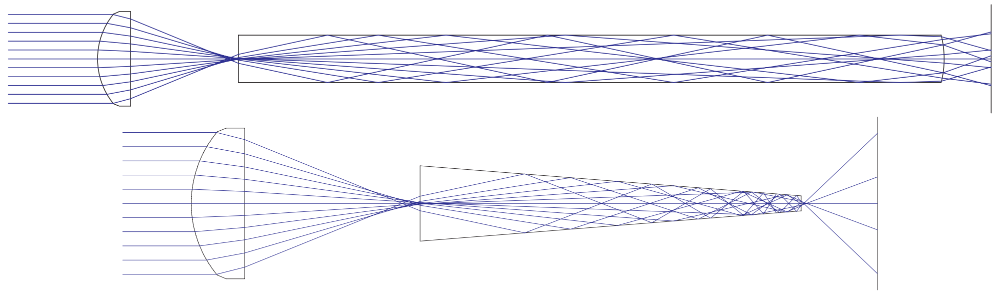


Figure 8.33: Light pipe (top) and tapered fiber (bottom).

respectively. In case of rectangular apertures, X- and Y-cross sections of the rod are tapered separately.

Hollow light pipes may be simulated by defining a mirror on the outside walls (not on the end surfaces), which bypasses checking of total internal reflection (TIR). This is accomplished by the command

`PMI si..j yes|no`	Pipe Mirror. Enables (`yes`) or disables (`no`) reflective properties on the outer walls. If enabled, TIR condition will be ignored and rays will always reflect at the outer walls.

Examples:

Step index fibers respectively light pipes are completely defined by the following command sequence (supposed, the rod/fiber entrance is at surface 3):

`sut s3 SP`	makes surface spherical and defines light pipe respectively fiber
`gla s3 sf6`	defines core material
`gl2 s3 bk7`	defines cladding material (`gl2 s3 air` is a fiber without cladding)
`thi s3 100`	length of fiber/pipe is 100mm
`cir s3..4 2.5`	diameter of rod is 5mm (=2*aperture radius)

Tapered fibers with circular apertures use the same commands, except that the semi-apertures on entrance surface and exit surface are different:

`sut s3 SP`	makes surface spherical and defines fiber/pipe
`thi s3 100`	length of fiber/pipe is 100mm
`cir s3 2.5`	diameter of entrance aperture is 5mm
`cir s4 1.0`	diameter of exit aperture is 2mm. Since the exit diameter differs from the entrance diameter, the pipe/fiber is tapered.

The semi cone angle ϑ of the tapered fiber in the second example above is then $\vartheta = tan^{-1}[(2.5 - 1.0)/100]$.

Rectangular (tapered) light pipes have rectangular apertures on both end surfaces. They are defined by the commands:

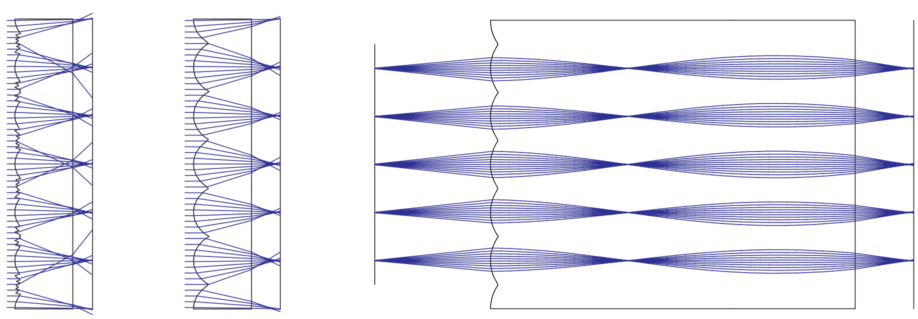
`sut s3 SP`	makes surface spherical and defines fiber/pipe
`thi s3 100`	length of fiber/pipe is 100mm
`rex s3 2.5`	rectangular aperture, entrance aperture X-diameter is 5mm
`rey s3 2.5`	rectangular aperture, entrance aperture Y-diameter is 5mm
`rex s4 1.0`	rectangular aperture, exit aperture X-diameter is 2mm
`rey s4 1.0`	rectangular aperture, exit aperture Y-diameter is 2mm. Since the exit aperture dimensions differ from the entrance aperture dimensions, the pipe/fiber is of pyramidal shape.

Sheared rectangular light pipe:
The end surface apertures may also be sheared (laterally displaced) at rectangular light pipes. This is accomplished by aperture offsets (see commands ADX, ADY) on the end surfaces. The side walls will automatically be adjusted. Note that shearing of end surface apertures does not shift the optical axis. Aperture offsets are ignored on cylindrical light pipes.

8.25 Array Element

The array surface arranges optical elements (surfaces) in a regular grid, i.e. they are repeated many times at specified X/Y locations with respect to the local coordinate of a surface, denoted hereafter as *array cells* or *channel surface*.

The individual lens or surface assemblies may be regarded as *cells* or *channels*. The channel surface encompasses all of the channels in the array. The aperture limits of the array surface are defined by the `AMX`, `AMY` parameters. Depending on the aperture dimensions and the cell/channel spacings (`ARX`, `ARY`) some channels (array elements) may be truncated. Individual channels are distributed in a uniform grid over the channel surface. The channel centers are located at (local) X/Y coordinates defined by the X-spacing (`ARX`) and Y-spacing (`ARY`).



a b c

Figure 8.34: Examples of array elements, a) fresnel lens array, b) spherical lens array, c) GRIN rod array. The corresponding example files can be found in the `$i\examples\array` directory as `sphere-array.otx`, `fresnel-array.otx` and `selfoc-array.otx`.

Array surfaces are defined by the surface type qualifier ”R” in addition to any other qualifier describing the shape of the surface (e.g. ”S” or ”A”) to be repeated.

ARR si..j x_spacing y_spacing x_offset y_offset max_x max_y	
	Convert surface(s) si..j to an array, using a regular grid pattern of channels. The channel coordinates (centerlines) are determined by `x_spacing` Grid spacing in X-direction between channel centers. `Y_spacing` Grid spacing in Y-direction between channel centers. `x_offset` Offset of center channel from surface vertex in X-direction. `y_offset` Offset of center channel from surface vertex in Y-direction. `max_x` ± limit for grid in X-direction `max_y` ± limit for grid in Y-direction

ARH sk|si..j Y|N	Array hexagonal arrangement. ARH sk Y : hexagonal cells arrangement (Fig. 8.36), ARH sk N : cells arranged in rectangular grid (Fig. 8.35).
ARX sk|si..j x_spacing	X-spacing of array channels.
ARY sk|si..j Y_spacing	Y-spacing of array channels.
ARXO sk|si..j X_offset	X-offset of entity of array channels with respect to local surface coordinate system.
ARYO sk|si..j Y_offset	Y-offset of entity of array channels with respect to local surface coordinate system.
AMX sk|si..j max_x	± limit for grid in X-direction
AMY sk|si..j max_y	± limit for grid in Y-direction
AADE sk|si..j angle_deg	α-tilt angle (in degree) of each array cell.
ABDE sk|si..j angle_deg	β-tilt angle (in degree) of each array cell.
ACDE sk|si..j angle_deg	γ-tilt angle (in degree) of each array cell.

Array properties can be combined with any type of surface, i.e. spherical, aspheric, Fresnel, GRIN and so on. For example, the following commands define various valid combinations of array surfaces:

`sut s1 SR`	Defines surface type for an array of spherical surfaces
`sut s1 AR`	Defines surface type for an array of aspheric surfaces
`sut s1 SFR`	Defines surface type for an array of Fresnel surfaces with spherical base curvature
`sut s1 SIR`	Defines surface type for an array of GRIN surfaces with spherical base curvature

There can be as many arrays as are surfaces in the optical system. Lens arrays, which span more than one surface (i.e. elements) can be generated by repeating the array parameters from previous surfaces. The apertures of the array channels are defined by the surface apertures (seeCIR, REX, REY, ELX, ELY commands).

If both, `x_spacing` and `y_spacing` are zero on a given surface, the array property is ignored and the lens behaves like a continuous (non-array) surface.

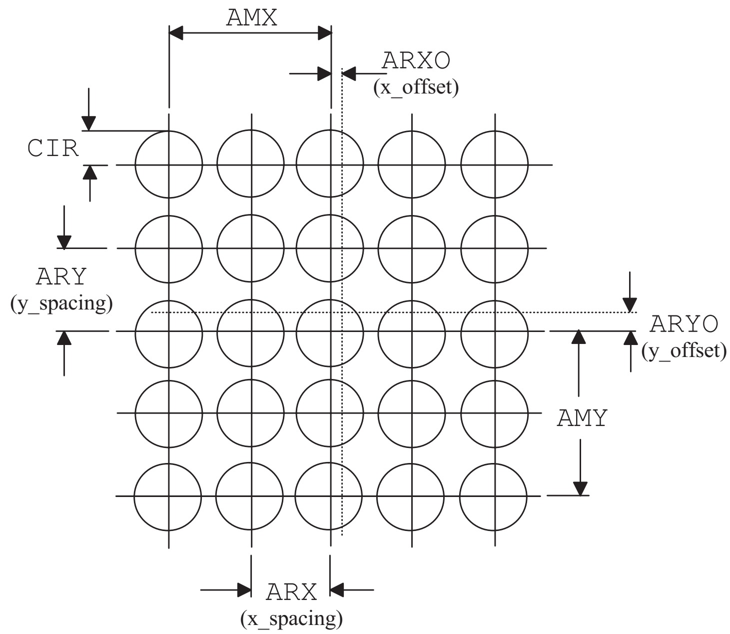


Figure 8.35: Definition of array parameter shown for a square regular grid. The dashed lines indicate the vertex of the base surface.

Restrictions:

1. Array parameters may not be zoomed. Parameters of the channel surface such as curvature, thickness, etc may be zoomed.
2. Array parameters may not be used in optimization.

Example:

An array of spherical channel surfaces as shown in Fig. 8.34(b) is best created when starting from a plano-convex lens. The first surface of the lens is converted to an array by

```
arr s1 5 5 0 0 15 15
```

where the spacings of the channel centerlines are 5mm in X- and Y-direction. The qualifier ”R” is correspondingly added to the surface type without requiring user interaction. The X- and Y-offsets are zero. This aligns the center channel on the vertex of the base surface. The extent of the array is given by the ± data pair (15 15). We may also enter the ARR command by discrete commands:

```
sut s1 sr
arx s1 5
ary s1 5
arxo s1 0
aryo s1 0
amx s1 15
amy s1 15
```

Next we will reduce the radius of curvature of surface 1 to pronounce the effect.

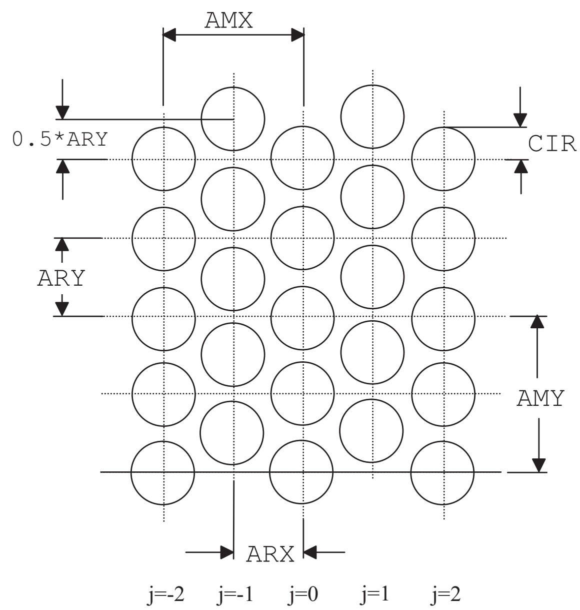


Figure 8.36: Hexagonal arrangement of array cells. All odd numbered columns are shifted (staggered) in Y-direction by 0.5*ARY. Optimal packaging of cells is then accomplished with ARX = cos(30°)*ARY = 0.866*ARY.

```
rdy s1 3
```

and will also define a fan of 31 rays along the Y-direction in order to better visualize refraction of rays in the lens layout plot (see also `VIE` command).

```
set fan y 31
```

The output should be as shown in Fig. 8.34(b).

8.26 Radial Spline Deformation Surfaces

The radial spline deformation surface is rotationally symmetric about the vertex of the base surface. The radial spline is defined by deformation points in radial direction, starting from the vertex to the outer rim of the surface. Each deformation point is described by a pair of two values, the radial distance (`SPLR`) from the vertex and the deformation value (`SPLZ`) perpendicular to the base surface. The base surface can be any of the surface types available in $OpTaliX$, for example a sphere or asphere. Since the spline function is added to the base surface, the surface type (`SUT`) must be composed of two letters, e.g.

SC = spherical base surface + spline
AC = aspherical base surface + spline

Up to 20 radial deformation points are supported per surface. There may be as many spline surfaces as are surfaces in the current system. The deformation points are then fitted by a ”Spline” interpolation method to obtain a continuous radial function across the surface. It should be noted that the deformation points are simulated exactly while all intermediate coordinates may exhibit ”overshooting” effects which are generally not desired. Since spline interpolation attempts to generate ”smooth” curves (i.e. first and second derivative of two adjacent segments match), there is no direct control of the surface slope. This behaviour is inherent to the Spline fitting method and does not constitute an implementation fault. A finer (smaller) sampling interval should be chosen in such cases. It is also good practice to provide additional sampling points outside the active area (if available) to avoid boundary effects. In some cases, when the spline deformation is very steep, a ray passing the exact surface vertex at exact normal incidence of the local surface may be deviated. This is also a boundary effect which may be reduced (or eliminated in most cases) by adding an extra sampling point close to the vertex point of the surface. This forces a zero slope at this point.

`SPLN si..j n_spline_points`	Number of (radial) spline deformation points at surface(s) `si..j`
`SPLR si..j ci..j rad_dist1 ... rad_dist_n`	Radial distance from the vertex of the surface(s) `si..j`. The radial distances are measured along the vertex tangent plane. Example: `splr s3 c1..5 0 2 4 7 13` where the deformation points are located at 0,2,4,7 and 13mm from the surface vertex.
`SPLZ si..j ci..j def_1 def_2 ... def_n`	Deformation from the base surface, measured perpendicular to the normal of the base surface. Example: `splz s3 c1..5 0.0 0.001 -0.002 0.003 -0.004`
`SPL si..j file file_spec`	Load Spline deformations from file "`file_spec`". A detailed description of the radial Spline file format is given in section 32.5. Example: `spl s4 file c:/temp/spline_def.dat`

Example:
We will apply a periodic deformation of roughly sinusoidal shape for easy visualization of the effects. First, we will enter the data manually in the command line and later on will learn about importing (loading) the spline deformation stored in a file. Assuming 6 sampling points, the command sequence is (without entering the exclamation mark and the text right to it)

```
spln 6                                   ! define number of sample points
splr s1 c1..6 0 0.001 10 20 30 40        ! define the radial distances
splz s1 c1..6 0 0 .001 -.001 .001 -.001  ! define the deformation
```

Note the second sampling point, which has been set very close to the first sampling point. This forces a zero slope at the vertex in the spline interpolation.

Alternatively, we could edit the data in a separate text (ASCII) editor outside of *OpTaliX* and store it in a file. It is then loaded with a single command. Using the demonstration example above, the file would look like (with comments included)

```
! Spline deformation file
0 0
0.001 0  ! this is an extra data point
10  0.001
20 -0.001
30  0.001
40 -0.001
! end of file
```

See also section 32.5 for a detailed description of the radial Spline file format. The file is loaded with the command `SPL s1 file 'c:\optalix\my-spline-data.spl'`. Path and file-name must be adjusted accordingly.

8.27 Two-Dimensional Interferometric Deformation on Surfaces

Interferometric deformations are specified as two-dimensional gridded data. Using this method, non-rotationally symmetric deformations can be modelled. Typically, such data is obtained from interferometric measurements of lens surfaces or complete optical systems or from external programs that generate appropriate data files. The surface type (SUT) must have the qualifier ”W” in order to make 2-dimensional deformation/apodization data active.

The data in an interferogram file can represent either surface deformation, wavefront perturbation data or intensity apodization data:

- **Surface deformation data** is added to whatever surface shape is defined with the lens. Deformation data is always measured normal to the nominal surface. During ray tracing, both ray aberrations and wave aberrations will be properly modified. Surface deformation data are always associated with refractive or reflective surfaces, they have no effect on dummy surfaces (same medium on both sides of a surface).
- **Wavefront perturbation data** modify the ray deviations and optical path difference (OPD) but has no effect on surface shape, even though it is associated with a (refracting/reflecting) surface.
- **Intensity apodization data** modify the transmission characteristics of an optical system but do not alter surface shape and ray directions.

Interferometric deformations can be scaled in deformation (`ISF`) and its origin can be placed at a particular X,Y location on the surface (`INX` and `INY` commands).

A file interface is provided that allows reading (importing) two-dimensional data sets. This data (surface deformation, wavefront perturbation or filter) is then assigned to a surface.

`INT sk file int_file_name`	Assign surface deformation data given in the file `int_file_name` to surface `sk`. No particular extension of the file name is required, however, ”`.int`” is recommended. The file format must obey a specific structure, which is specified in section 32.11.
`ORB sk file orb_file_name`	This command is functionally equivalent to the ”INT” command above, except that it expects surface deformation data in a form provided by the ”Orbscan II” topography system from Bausch & Lomb used in surgical treatments of the human eye. The data must have been exported in cartesian form (gridded data) using the ”Recorder” option. The surface deformation data in the file `orb_file_name` is then attached to surface `sk`.

continued on next page

continued from previous page	
`ISF si..j scale_factor`	Scales the measured deformation by a specified scale factor. For example, a scale factor 0.5 is often used for scaling of surface data obtained in a double-pass interferometric setup. A scaling factor -1.0 also allows flipping the deformation data from "bump" to "dent".
`INX sk x_offset`	X-coordinate on surface `sk` where the center of the deformation data is placed.
`INY sk x_offset`	Y-coordinate on surface `sk` where the center of the deformation data is placed.
`IRX sk x_extension`	Physical extension of the deformation array in X-direction on surface `sk`. Extension is meant as ± value from the center of the deformation data.
`IRY sk y_extension`	Physical extension of the deformation array in Y-direction on surface `sk`. Extension is meant as ± value from the center of the deformation data.
`PLO INT [sk]`	Plots two-dimensional deformation assigned to surface `sk`. See also sect. 8.27.8.
`RAW2INT file raw_file`	Convert two-dimensional gridded data in "raw" format to INT format. This is a utility command which is useful when only "raw" data are available. The file `raw_file` must be provided in ASCII format with full path specification. The parameter "file" is mandatory. The data in the RAW format may be separated by blank characters, comma, tabs or by quote characters ". One line in the ASCII file corresponds to one row in the data grid. Thus, there are as many lines in the the file as are rows in the data array. The file must not contain any header or comment lines. The array size is extracted from the data itself. **Example:** `raw2int file c:\mydata.txt` The converted data are then written in a separate file in the same directory with the extension `.int` appended. From the example above, the output (converted) file is then `c:\mydata.txt.int`

8.27.1 Saving Deformation Data

Deformation data associated to surfaces in the current optical system can be saved in two variants:

a) The deformation data are kept in the original file and only a "link" to the file containing the data is saved with the prescription data. This method allows small prescription files, however, an absolute path is stored. However, absolute paths cannot be updated when your computer configuration changes. For example, if you change the location of the deformation file (move it) or send your prescription file to anybody else (via Internet/Intranet) who most likely has a different directory structure on his computer, $OpTaliX$ will not be able to find the deformation file. Only in cases where you can relay on a stable and consistent file structure, saving links is recommended.

b) The second option, which is independent on file structure, saves the deformation data as an integral part of the prescription data. Large file sizes may result, depending on the number of surfaces that have deformations associated and on the array sizes of the deformation data itself.

Saving deformation data is controlled by from the command line by

`ILN Yes|No`	Save interferometric deformation, wavefront or filter data as link to a file. On saving or restoring an optical system, the data are retrieved from the original file (`ILN YES`) or are stored along with the description data (`ILN NO`). There are specific advantages/disadvantages in choosing either method: `ILN YES` : Only stores a link to the file containing the data (INT, BMP, PCX or PNG file). On restoring the optical system, the file must exist, i.e. accessible by path and file name. Moving files may result in loss of data due to inaccessible files. `ILN NO` : Saves all data with the prescription data. The corresponding *OpTaliX* file may become VERY large, depending on the amount of data involved in describing the perturbation or filter characteristics. This way, perturbation data will always be available, however, it cannot be changed except by reloading new data.

or from the configuration dialog invoked from the main menu by *Edit − > Configuration Data*. In the *General* tab, check the option "*Store 2-dim deformation with prescription data*", as shown in Fig. 8.37.

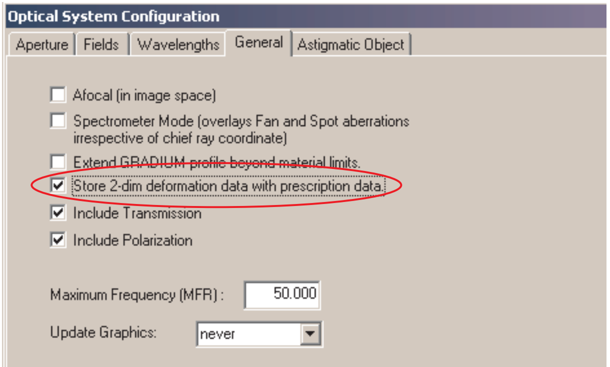


Figure 8.37: Option for saving interferometric deformation data, wavefront or filter data. Check if data are to be saved with prescription data, leave unchecked if data are maintained in separate file, accessed by a link.

Caution: Once 2-dimensional deformation data are stored with the prescription data and the appropriate check box in the configuration dialog has been checked, it is not recommended to uncheck it. If unchecked, the program does not know where to store the deformation data, since it cannot create the original files, and the data will be lost. That is, the program provides two methods of handling and storing deformation data, however, the storing method should not be changed after a selection has been made.

8.27.2 Sign Conventions

A positive deformation in the data file(s) is in the direction of the local Z-axis for the surface, regardless of the direction of light. Thus, the physical meaning depends on which side of an optical element is considered. For a singlet lens, for example, a positive deformation on the first surface is a concave increment (”dent”) to the surface while a positive deformation on the second (”rear”) surface is a convex increment (”bump”) to the surface.

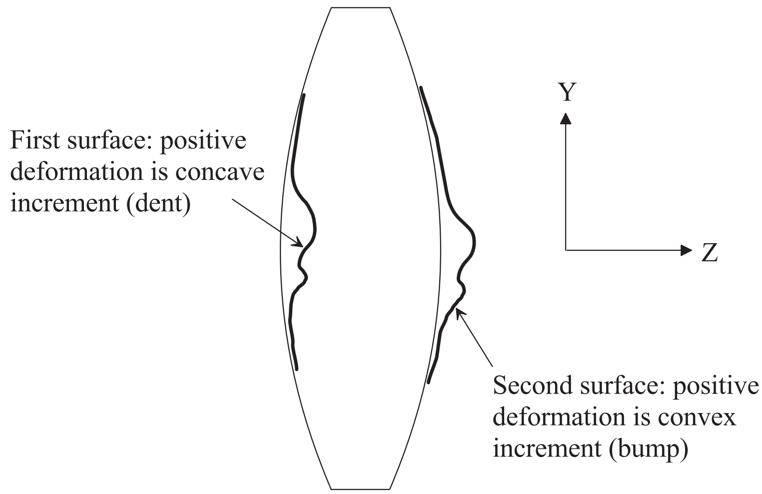


Figure 8.38: Sign convention for two-dimensional deformations on surfaces.

It is generally a good idea to test the correct orientation of coordinate axes (X,Y) of deformation data with marked pieces. A plot of the deformation data as shown in Fig. 8.39 is helpful to visualize the data in the $OpTaliX$ coordinate system. This plot is generated by the command (on the example of surface 3)

```
plo int s3
```

or from the menu: *Display –> Show 2-dim. Surface Deformation*

8.27.3 Interferometric Deformation Data

Surface deformations obtained from interferometric measurements or from other external programs (e.g. NASTRAN deformations) are read in by the INT command. The file format is identical to the Code V INT-files and is specified in section 32.11.

Due to the inherent structure of Code-V INT files, no provision for specifying the lateral X- and Y-extensions of the data, respectively the coordinates of the X/Y sample points, is foreseen. Thus, the connection of the unit length of the file data to the physical length on the surface must be specified separately. To control the correct X/Y-extensions on a specific surface use thePLO INT command.

In $OpTaliX$ mapping of the file data to the surface aperture is queried at the time of loading/assigning deformation data as shown in Fig. 8.40.

8.27.4 Wavefront Perturbations

Wavefront perturbation data must be provided in the INT file-format (see section32.11 on page 504) as defined in Code V. This means that Code V INT files can be directly read in and associated to surfaces without modification.

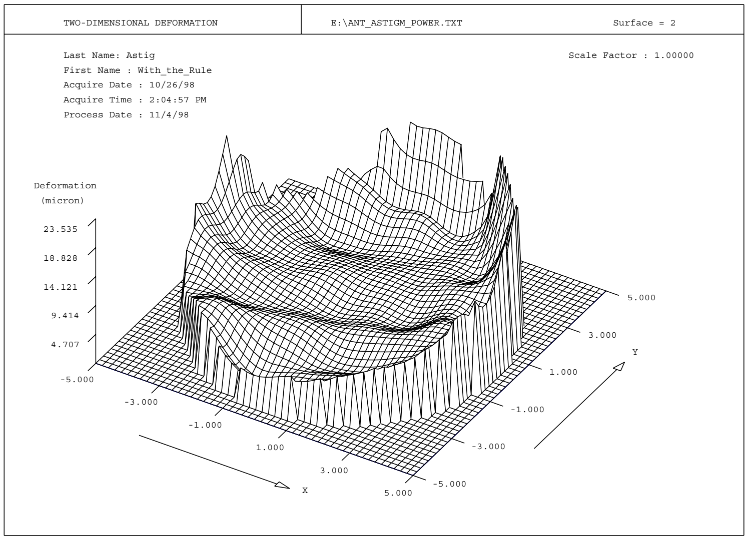


Figure 8.39: Plot of two-dimensional surface deformation in the $OpTaliX$ coordinate system. The deformation is always shown in the direction of the positive Z-axis. For systems having no mirrors or tilted components, the positive Z-direction is identical to the direction of light (from left to right in the lens layout plot).

Wavefront perturbations modify the ray directions and the optical path difference (OPD) but there is no effect on surface shape, even though it is associated to a surface. Wavefront perturbations are usually placed on dummy surfaces. Wavefront perturbation data can be viewed using the PLO INT command.

8.27.5 Surface Intensity Apodization (Intensity Filter)

Intensity apodization data are read in from an INT-file or a bitmap file (BMP, PCX or PNG) and are associated to a specific surface. Surface based apodization only modifies the intensity transmission along a ray path and thus can be understood as a spatial intensity filter. There is no effect on surface shape and direction of rays. By default, rays are not blocked, except in regions where data is missing (see sect. 8.27.7). In addition, rays can be blocked in regions of zero intensity if the `IBZ` attribute is assigned to a filter (sect. 8.27.7).

Intensity apodization can be associated to any surface (except object and image surface), however, they are typically associated to dummy surfaces. The effect of the apodization on the beam profile depends upon the region of the surface that is hit by the beam.

Apodization filter data in INT-files or BMP/PCX/PNG files are transmission and can have any value grater than 0. See a detailed description of the INT file format in section 32.11. Apodization filters can also be defined in a bitmap file (BMP, PCX or PNG) in which transmission is grey-coded in grey levels between 0 (no transmission) and 255 (full transmission = 1.0).

Apodization filters can be placed on surfaces with X- and Y-offsets using the INX and INY commands. Inversion and scaling of intensity data is not possible. Use the PLO INT command to control correct placement and scaling of apodization data on surfaces. The effect of intensity apodization on system transmittance can be plotted by the pupil intensity map (PMA) option as described in section 14.1.9.

It is not required to activate transmission analysis (`TRA yes|no`) or polarization analysis (`POL yes|no`) to see the effects of intensity apodization filters on performance. Once attached to a surface, intensity

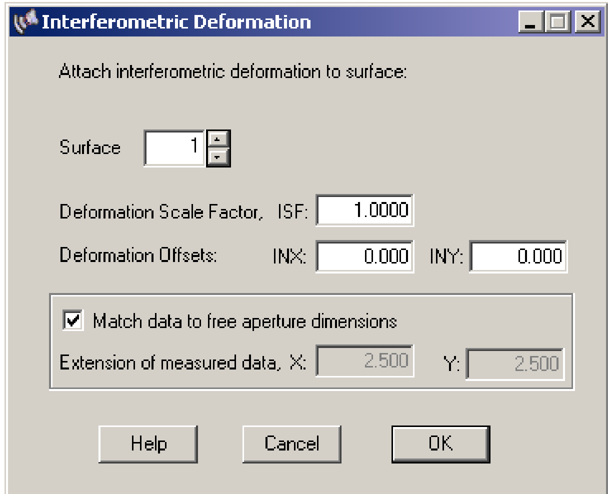


Figure 8.40: Assigning two-dimensional deformations from Code V compatible INT files to surfaces and specifying scaling factor and X/Y offsets. The connection of the unit length (maximum array size) to the physical extension on the surface can be accomplished by matching the data to the clear surface aperture (default) or by explicitly specifying X/Y extensions of the interferogram data.

apodization filters are always active.

8.27.6 Deformations from Orbscan II Topography System

Surface deformation data obtained from the ”Orbscan II” topography system from Bausch & Lomb are assigned to surfaces using the `ORB` command. It is functionally equivalent to the `INT` command, except that a different file format is expected.

The Orbscan II data must be provided in cartesian form (gridded data) using the ”Recorder” option (see the Orbscan manual). This option writes a readable ASCII file. Orbscan topographic data can be read in and assigned to optical surfaces from the command line or by selecting menus. For example, importing Orbscan II deformation data is accomplished in the command line by

```
orb s3 file c:\temp\def_data.txt
```

The file may have any extension. Note the use of the expression ”`file`” in the command. It is required to identify the subsequent string as a path and file specification. Using menu items, the same file is assigned to surface 3 by clicking

File –> Import –> Orbscan Map Data

Select the file containing the deformation data from the file selection box. The surface association is performed in a subsequent dialog box as shown in Fig. 8.41. It also allows definition of the (interferogram) scaling factor `ISF`, which is used to change the sign of the deformation data, as well as X- and Y-offsets (INX, INY) where the deformation is placed on the surface.

Orbscan map data are defined and stored in a left-handed coordinate system. Since the coordinate system used in $OpTaliX$ is also left-handed, no special precautions such as inverting or mirroring data is required. In particular, ISF should be +1.0.

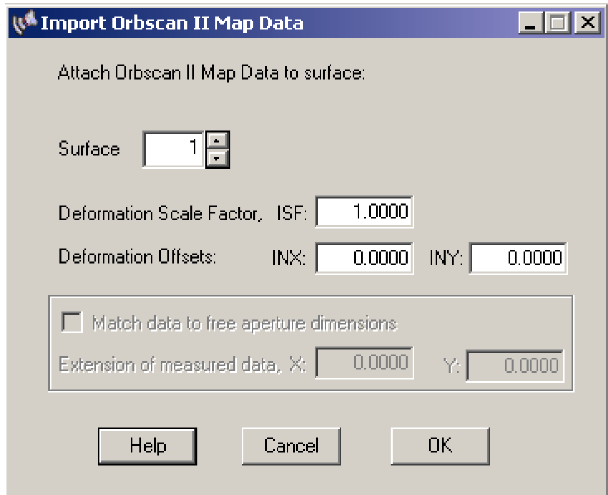


Figure 8.41: Assigning Orbscan map data (two-dimensional deformations) to surfaces and specifying scaling factor and X/Y offsets. The lateral X/Y extensions are greyed out, because these are explicitly provided with Orbscan files and need not specified.

8.27.7 Behaviour of Rays in Regions of No Data

Interferogram or filter data can have regions of missing data. Possible reasons may be clipping by the edge or obscuration of the piece being tested, noise or too weak signal in the interferometer detector, or other reasons. Missing data are indicated in the files according to the value associated with the NDA file entry.

Rays which hit ”no data” regions will be blocked, irrespectively whether the surface aperture is checked (fixed aperture) or not.

Optionally rays can also be blocked on surfaces with intensity filters if the intensity reaches zero. The IBZ flag controls behaviour of rays in such regions:

`IBZ si..j|sk Yes|No`	Block rays in regions of zero intensity. This option is *only* applicable on surfaces with intensity filters. If this flag is set (`IBZ sk YES`), rays hitting a region where the intensity approaches zero (< 0.001) are blocked. Specify `IBZ sk NO` to let rays pass irrespective of the intensity imposed by the filter. The `IBZ` option is particularly useful to model very complex aperture shapes. Any arbitrary shape provided in an INT-file or a bitmap file (BMP,PCX,PNG) may be attached as an intensity filter. `IBZ YES` on that surface will then define the complex aperture as all rays at zero intensity will be blocked.

8.27.8 Display Interferometric Deformation

Interferometric deformations attached to a surface can be viewed by the `PLO INT` command:

`PLO INT sk [?]`	Plot interferometric deformation attached to a surface. The question mark (optional) invokes a dialog box for editing plot parameters.

A sample plot of an interferometric deformation and the associated surface aperture is shown in Fig. 8.42 (page 143). This plot allows mapping of the interferogram file data to the surface aperture. Notice that the interferogram dimensions are queried at the time of loading/assigning deformation data. However, interferogram dimensions can be changed by the IRX, IRY commands.

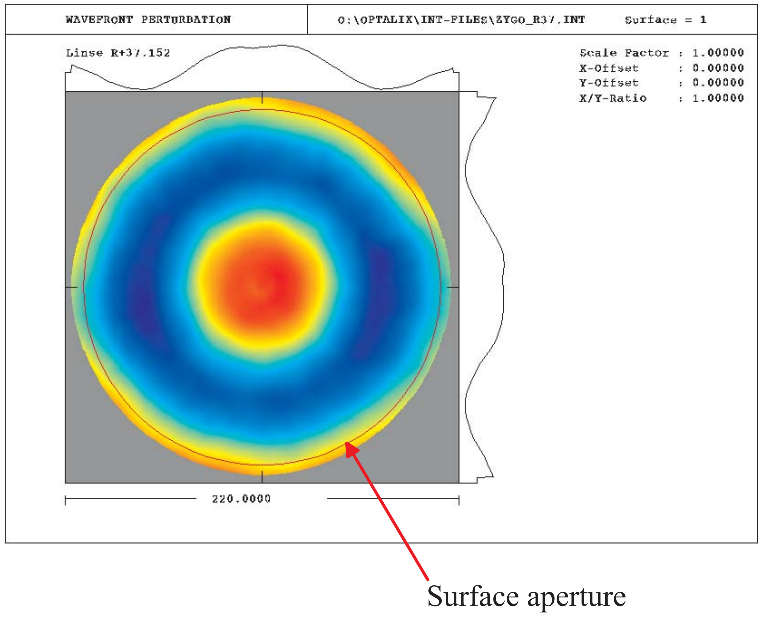


Figure 8.42: Display interferogram deformation on a surface. The surface aperture is shown in red colour which allows a direct comparison with the measured interferogram dimensions.

Interferometric deformations can be plotted in four styles, wire-grid plot, gray-scale plot, false-colour plot and as X/Y-sections. Currently, the plot style can only be defined within the option dialog box (i.e. `PLO INT ?`).

8.28 Zernike Surface

The Zernike surface is defined by the surface type "Z" which may be added to any other base surface (e.g. spherical, aspherical, toroidal, etc). Zernike surfaces are always defined in terms of "Finge Zernike polynomials". Zernike surfaces may be defined as surface or phase deformation:

- Zernike **surface** deformation: Defines a deformation of the surface, i.e. direction and optical path along a ray are altered by the law of refraction. The Zernike surface deformation is preferably applied to surfaces with an air/glass or glass/air interface.
- Zernike **phase** deformation: Introduces an additional phase component to the optical path (wavefront). The direction of rays is modified such that rays are always perpendicular to the phase additive. Zernike phase surfaces must be defined on surfaces with the same medium on both sides of the surface (preferably AIR/AIR interfaces).

Command Overview:

`ZRN [si..j|sk] SUR|PHA`	Define Zernike deformation on surface (SUR), or as phase/wavefront perturbation (PHA) at surface(s) `si..j`. The Zernike surface deformation is preferably applied to surfaces with air/glass, respectively glass/air interfaces, the Zernike phase surface should only be applied to air/air surfaces (i.e. dummy surfaces).
	continued on next page

continued from previous page

`ZRN [si..j|sk] ci..j`	Set Zernike coefficient `ci..j` at surface(s) `si..j`
`ZRN si..j|sk FIL f_name`	Load Zernike deformation coefficients from file `f_name` and attach it to a specific surface `sk` or a range of surfaces `si..j`. A description of the Zernike coefficients file format is given in section 32.4.
`ZRN WAV [fi]`	Fit Zernike polynomials to wavefront aberration at field `fi` at the reference wavelength. Make sure to have appropriate Zernike coefficients on wavefront activated (see `ZWACT` command below). See also the WZRN command to retrieve Zernike coefficients fitted to the wavefront.
`PLO ZRN si`	Plot Zernike-wave based on Zernike coefficients associated to surface `si`
`EDI ZRN si`	Opens a dialog box to edit Zernike coefficients associated to surface `si`.
`INR [si..j|sk] radius`	Connects the unit circle of Zernike data to a physical aperture on the surface(s) `si..j|sk`. The entered value is the radius on that surface(s). The default value for `INR` is the semi-diameter of the surface clear aperture. **Note:** If the given value of radius scales the Zernike deformation to a smaller value than the actual semi-aperture, the data outside the INR radius will be extrapolated, leading to false results! This case must be avoided.
`ZACT si..j|sk ci..j act1 act2 ...`	Activate/deactivate Zernike coefficients on a particular surface (or range of surfaces). Activating a coefficient means that it will be used in the performance analysis. ”`act`” is an integer number of 0 or 1, where 0 deactivates a coefficient and 1 activates it. In absence of a coefficients specifier ”c”, a sequence of integer values is expected (see third example below). Examples: `zact s2 c1 1` ! activates Zernike coefficient 1 at surface 2 `zact s2..3 c1..5 1` ! activates Zernike coefficients no. 1 to 5 at surface 2 to 3. `zact s2 1 0 1 0 1` ! activates coefficients no. 1,3 and 5, deactivates coefficients no. 2 and 4. Alternatively, coefficients may be activated/deactivated in the Zernike spreadsheet editor, which is invoked by the command `EDI ZRN` (see above). For the definition of Zernike coefficients see sect. 8.28.2).

continued on next page

continued from previous page	
`ZWACT ci..j act1 [act2 ...]`	Activate/deactivate Zernike coefficients used for wavefront fitting. Activating coefficients means that they will be used for fitting the wavefront. ”`act`” is an integer number of 0 or 1, where 0 deactivates a coefficient and 1 activates it. In absence of a coefficients specifier ”c”, a sequence of integer values is expected (see third example below. A surface qualifier is not required, since the ZWACT switches always apply to the wavefront Zernike coefficients. Examples: `zwact c1 1` ! activates Zernike coefficient 1 to be used for wavefront fitting, `zwact c1..5 1` ! activates Zernike coefficients no. 1 to 5 for wavefront fitting, `zwact 1 0 1 0 1` ! activates coefficients no. 1,3 and 5, deactivates coefficients no. 2 and 4. Alternatively, wavefront coefficients may be activated/deactivated in the Zernike spreadsheet editor, which is invoked by the command `EDI ZRN` (see above). Use the command WAV ZRN to actually fit the coefficients to the wavefront aberration at a particular field. For the definition of Zernike coefficients see sect. 8.28.2).
`WZRN Ci..j`	Set Zernike coefficients `ci..j` of wavefront. Fit Zernike coefficients to the actual wavefront at a specific field using the `ZRN WAV` command. (see above) and subsequently edit them by the `EDI ZRN` command.
`WZRN Ck fk`	In macros or from the commandline, retrieve a specific wavefront Zernike coefficient, where `ck` is the kth coefficient, and `fk` is field k. Example: `eva [wzrn c3 f1]`

Example 1:

Typical surface irregularities caused by fabrication errors can be simulated by adding Zernike deformations to particular surfaces. A likely effect in ”synchro-speed” generation of spherical surfaces can be modelled with good approximation using only one Zernike term, Z9, as shown in Fig. 8.43. We assume a measured irregularity $\tau = 0.5 wavesPV$ at 633nm on a surface exhibiting only this defect. Since in the unit circle $-0.5 < Z_9 < 1.0$, the PV value of Z_9 in the unit circle is 1.5, the coefficient Z_9 calculates to

$$Z_9 = \frac{\tau \cdot \lambda_{633}}{PV_{unit-circ}} = \frac{0.5 \cdot 0.000633}{1.5} = 2.11 \cdot E^{-4} \tag{8.86}$$

λ_{633} is the interferometer wavelength (633nm). This deformation is entered by the following commands (without typing the exclamation mark and the text right to it):

```
SUT s2 SZ            ! surface type is spherical + Zernike
ZRN s2 c9 2.11e-4    ! enters Zernike coefficient Z9 at surface 2
```

Alternatively, we may enter the coefficients in the Zernike spreadsheet editor, which is invoked by the EDI ZRN command. Find a more detailed explanation of the Zernike spreadsheet editor in section 8.28.1, page 146. The surface type can be changed in the surface spreadsheet editor, (use command EDI SUR, if not already open).

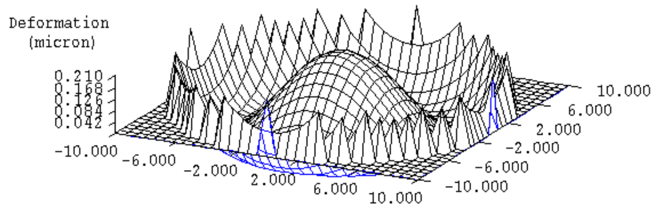


Figure 8.43: Zernike deformation, using only coefficient 9.

Example 2:

Fitting Zernike polynomials to the actual wavefront aberration at a particular field is accomplished with the ZRN WAV command. Suppose, we want to see the Zernike terms at field 2, we must first specify, which coefficients are to be included (activated) in the fitting process. Subsequently, fitting can be performed. Both operations are done, for example, by the commands

ZWACT 0 1 1 1 1 1 1 1	! activate Zernike coefficients 2-8 for wavefront fitting. Coefficients 1 and 9-36 are excluded from fitting.
ZRN WAV f2	! Perform wavefront fitting at field 2.

and obtain the following output for field 2 (the reference wavelength number is 2):

```
Zernike polynomial fit of wavefront at field   2   colour  2

   #       coefficient      coefficient    Remark
       (unit = micron)    (unit = wave)
   2      -0.817072827       -1.39053      Y-Tilt
   3       1.184744104        2.01624      Defocus
   4      -1.401898817       -2.38580      Astigmatism 3rd Order, 0 and 90 deg.
   5       0.000000000        0.00000      Astigmatism 3rd Order, +/- 45 deg.
   6       0.000000001        0.00000      X-Coma and Tilt, 3rd Order
   7      -2.191878576       -3.73022      Y-Coma and Tilt, 3rd Order
   8       1.450299352        2.46817      Spherical and Focus, 3rd Order
```

8.28.1 Zernike Spreadsheet Editor

Editing of Zernike coefficients can be performed in a more convenient manner via the Zernike spreadsheet editor (see Fig. 8.28.1). It is started from the command line by EDI ZRN and allows input of Zernike deformation coefficients at surfaces as well as fitting of the wavefront aberration. Any surface in the optical system (except the object and image surface) may be selected. If ”wavefront” is selected, the Zernike coefficients relate to the wavefront aberration in the exit pupil. For this case, it does not make much sense to enter coefficients (although it is possible), but this option is merely used to fit a Zernike polynomial to the existing wavefront. Select (activate) in the second column, which coefficients shall be included in the fit.

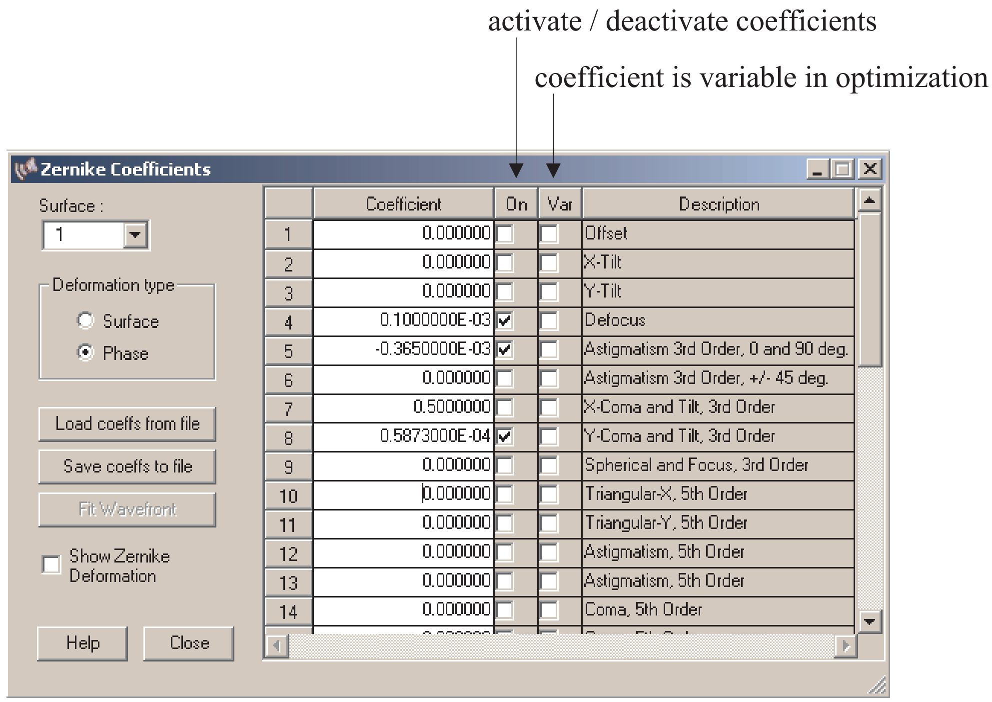


Figure 8.44: Editing of Zernike coefficients at surfaces, respectively fitting of wavefront aberration.

Zernike coefficients may be loaded from a file or stored into a file. The latter is particularly useful for fitted wavefront aberrations.

8.28.2 Definition of Fringe Zernike Polynomials

Zernike polynomials are circle polynomials in radius and azimuth.They are favoured in representing wavefront because they are orthogonal and normable within the unit circle. This implies that each term is independent from all others. Therefore, neither the inclusion or exclusion of a given term will affect the values of the other terms. This is strictly true only for continuous data, but it is approximately true for data that is uniformly spaced over a circular aperture. The Zernike polynomials have the general form

$$Z_n^m\left(r,\phi\right) = R_n^m\left(r\right)\left[\cos m\phi + sin m\phi\right] \tag{8.87}$$

where r and ϕ are polar coordinates within the unit circle. Typically, wavefront data are represented in the pupil of an optical system in cartesian pupil coordinates x_p, y_p. The relationship between $[r, \phi]$ and x_p, y_p is

$$x_p = r\cos\phi \tag{8.88}$$

$$y_p = r\sin\phi \tag{8.89}$$

We shall be concerned in the following treatment with the Fringe ZERNIKE polynomials, which are a subset of the standard Zernike polynomials but arranged in a different order. The first 37 coefficients can be written explicitly as:

1	1.0	Offset
2	$R\cos\phi$	X-tilt
3	$R\sin\phi$	Y-tilt
4	$2R^2-1$	Defocus
5	$R^2\cos 2\phi$	Astigmatism 3^{rd} order at $\phi =$ 0° or 90°
6	$R^2\sin 2\phi$	Astigmatism 3^{rd} order at $\phi =$ ±45°
7	$(3R^3-2R)\cos\phi$	X-coma and tilt, $3^{rd} order$
8	$(3R^3-2R)\sin\phi$	Y-coma and tilt, $3^{rd} order$
9	$6R^4-6R^2+1$	Spherical and focus, $3^{rd} order$
10	$R^3 cos(3\phi)$	Triangular-X, 5^{th} order
11	$R^3 sin(3\phi)$	Triangular-Y, 5^{th} order
12	$(4R^4-3R^2)cos(2\phi)$	Astigmatism, 5^{th} order
13	$(4R^4-3R^2)sin(2\phi)$	Astigmatism, 5^{th} order
14	$(10R^5-12R^3+3R)cos(\phi)$	Coma, 5^{th} order
15	$(10R^5-12R^3+3R)sin(\phi)$	Coma, 5^{th} order
16	$20R^6-30R^4+12R^2-1$	Spherical, 5^{th} order
17	$R^4 cos(4\phi)$	Quadratic-X, 7^{th} order
18	$R^4 sin(4\phi)$	Quadratic-Y, 7^{th} order
19	$(5R^5-4R^3)cos(3\phi)$	Triangular, 7^{th} order
20	$(5R^5-4R^3)sin(3\phi)$	Triangular, 7^{th} order
21	$(15R^6-20R^4+6R^2)cos(2\phi)$	Astigmatism, 7^{th} order
22	$(15R^6-20R^4+6R^2)sin(2\phi)$	Astigmatism, 7^{th} order
23	$(35R^7-60R^5+30R^3-4R)cos(\phi)$	Coma, 7^{th} order
24	$(35R^7-60R^5+30R^3-4R)sin(\phi)$	Coma, 7^{th} order
25	$70R^8-140R^6+90R^4-20R^2+1$	Spherical, 7^{th} order
26	$R^5 cos\,(5\phi)$	5-fold, 9^{th} order
27	$R^5 sin\,(5\phi)$	5-fold, 9^{th} order
28	$\left(6R^6-5R^4\right) cos\,(4\phi)$	Quadratic, 9^{th}order
29	$\left(6R^6-5R^4\right) sin\,(4\phi)$	Quadratic, 9^{th} order
30	$\left(21R^7-30R^5+10R^3\right) cos\,(3\phi)$	Triangular, 9^{th}order
31	$\left(21R^7-30R^5+10R^3\right) sin\,(3\phi)$	Triangular, 9^{th} order
32	$\left(56R^8-105R^6+60R^4-10R^2\right) cos\,(2\phi)$	Astigmatism,9^{th} order
33	$\left(56R^8-105R^6+60R^4-10R^2\right) sin\,(2\phi)$	Astigmatism, 9^{th} order
34	$\left(126R^9-280R^7+210R^5-60R^3+5R\right) cos\,(\phi)$	Coma, 9^{th} order
35	$\left(126R^9-280R^7+210R^5-60R^3+5R\right) sin\,(\phi)$	Coma, 9^{th} order
36	$252R^{10}-630R^8+560R^6-210R^4+30R^2-1$	Spherical, 9^{th} order
37	$924R^{12}-2772R^{10}+3150R^8-1680R^6+$ $420R^4-42R^2+1$	spherical, 11^{th} order

8.29 Zernike Phase Surface

The Zernike phase surface adds terms to the nominal wave front aberration of an optical system. It is most useful for the inclusion of measured interferometer data. Zernike phase surfaces must be defined on surfaces with the same medium on both sides of the surface (preferably AIR/AIR interfaces).

The following examples show definition of the Zernike phase surface, assuming surface 4.

In the command line :

```
zrn pha s4          ! define Zernike phase surface
inr s4 10           ! Connects Zernike unit circle to physical aperture
zrn s4 c5 0.00123   ! Zernike coefficient c5 at surface s4 is 0.00123
zact s4 c5 1        ! activate/enable coefficient c5 at surface s4
```

In the user dialog:

Invoke the Zernike editor from the menu *Edit / Zernike Coefficients* or from the command line by entering ”`EDI ZRN`” (without the quotes). A dialog box will pop up. The dialog is partially shown in Fig. 8.29.

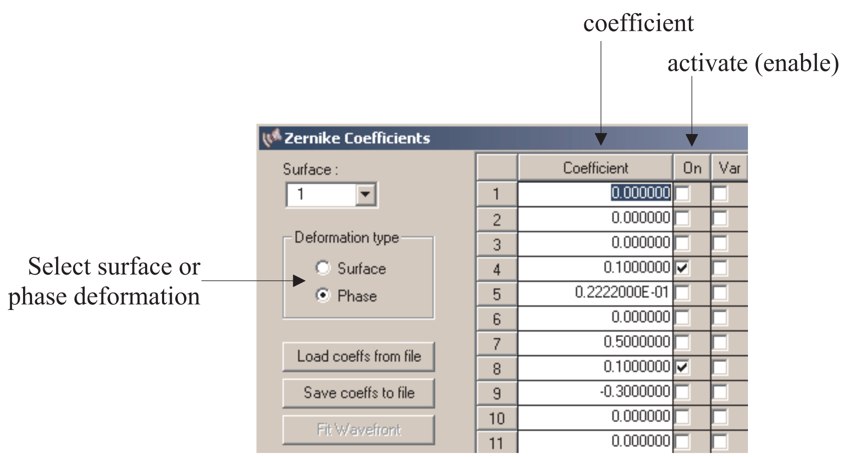


Figure 8.45: Editing of Zernike **phase** coefficients.

Check the radio button ”**phase**” and enter the appropriate coefficients. Do not forget to activate (enable) the coefficients by checking the ”On” field for each coefficient.

8.30 User-Defined Surface (UDS)

The user-defined surface allows interrupting the internal ray trace algorithms in $OpTaliX$ and take control of the ray trace. Internally, the ray trajectory is computed up to the surface immediately preceding the user surface, calls a user-written subroutine specified for the surface and then completes the ray trace through the remaining surfaces.

The designation of a surface as user-defined is done by entering the `UDS` command on that surface or setting the surface type (`SUT sk U`) directly. Coefficients for the user-defined surface, if any, are defined by the `UCO` command.

indexUser-defined!surface type

`UDS si..j|sk`	Change surface type to user-defined surface on surface(s) `si..j`, respectively surface `sk`. Alternatively, the surface type can be set to "U" (see SUT command on page 65). The UDS surface shape is entirely defined by the UCO coefficients (see below) and the user-written subroutine "`usersur.f90`" contained in a DLL.
`UCO si..j|sk ci..j coefficient`	Coefficient for describing user-defined surface (UDS) type on surface(s) `si..j|sk` using the user-written subroutine `usersur.f90`. The maximum number of coefficients is 91.

8.30.1 Creating a User-Defined Subroutine

The user need only program the (continuous) surface function and the surface derivatives in a FORTRAN or C subroutine called "`usersur.f90`" respectively "`usersur.c`". Note: The subroutine name must be exactly "`usersur`", no other name is permitted.

OpTaliX provides a sample subroutine in both FORTRAN and C programming languages, which is kept simple in order to demonstrate the programming interfaces. The sample subroutine defines a parabolic surface. It is found in the directories

`\optalix\usersur\Fortran`	for FORTRAN
`\optalix\usersur\C`	for C/C++

with appropriate subdirectories for Lahey/Fujitsu FORTRAN, Compaq Visual FORTRAN, Intel Fortran Parallel Studio, and Microsoft Visual C compilers. The source code of the `usersur` subroutine is given for each language and compiler in sections 8.30.3 to 8.30.6.

The `usersur` subroutine can also, if needed, call other subroutines or read data files. The subroutine `usersur` is successively called to iteratively compute the intersection point of a ray with a UDS type surface. After computing the intersection point of the ray with the surface, the surface slope at that point is determined. A special variable `icalc` must be queried in the `usersur` subroutine depending on whether the intersection point or the surface slope is to be calculated.

The `usersur` subroutine that you write in FORTRAN or C must have the following parameters:

`usersur(icalc,isur,curv,sdata,x,y,z,xn,yn,zn,i_err)`

where

`icalc`	Calculation mode (input). Indicates whether to calculate the surface function or the surface slope. 1 = calculate surface z coordinate at coordinates x,y 2 = calculate xn,yn,zn direction cosines at x,y,z
`isur`	Current surface number for which the function is to be evaluated. This is an input parameter which may be used to distinguish between various algorithms on different surfaces. If only one UDS type surface is used, this parameter is normally not needed. See also the note below.
`curv`	Surface vertex curvature (input). This parameter does not have to be used in the `usersur` subroutine, however, its value is also used to calculate first and third order properties of the optical system.

continued on next page

continued from previous page	
sdata	Data array with 91 elements for passing data between *OpTaliX* and the usersur subroutine. The elements of data correspond to the UCO coefficients C1 to C91.
x,y,z	Coordinates at a point along the ray.
xn,yn,zn	Direction cosines of the surface normal at the point (x,y,z).
i_err	Error flag. It should be set to 0 if there is no error generated and set to 1 otherwise.

Note: Only one usersur subroutine can be linked to *OpTaliX* at one time. Therefore all UDS type surfaces in the optical system must use the same usersur subroutine. However, it is possible to program more than one UDS surface description with different coefficients in the same usersur subroutine. The parameter isur designates the surface number currently in use for finding the surface intersection or surface slope. The following FORTRAN sample code illustrates this:

```
if(isur .eq. 3) then
    ! add code for surface 3 here
elseif(isur .eq. 7) then
    ! add other code for surface 7
endif
```

With this technique, there is virtually no limit on the number of different user-defined surface types in an optical system.

8.30.2 Languages and Compilers Supported

Both FORTRAN and C programming languages are supported. The following sections describe the specifics for various compilers. Sample subroutines are supplied with *OpTaliX* in both languages Fortran and C. These sample subroutines are located in the \optalix\usersur directory with subdirectories according to the programming language and compiler used.

Creating user defined surfaces is described for the following compilers:

- Lahey/Fujitsu FORTRAN,
- Compaq Visual FORTRAN,
- Intel FORTRAN Parallel Studio
- Microsoft Visual Studio

All supported compilers are 32 bit and 64 bit versions. 16 bit versions are no longer supported. All compilers must have version numbers equal or higher as listed below:

Lahey Fujitsu	FORTRAN-95, version 5.7 or later
Compaq/Intel	Visual FORTRAN, version 6.6 or later
Intel Parallel Studio	version 13 or higher
Microsoft Visual Studio	2012 or later

8.30.3 Compiling with Lahey/Fujitsu Fortran 90

Source code example of a user defined surface (UDS) in FORTRAN with specific instructions for the Lahey/Fujitsu compiler:

```
      subroutine usersur(icalc,isur,curv,sdata,x,y,z,xn,yn,zn,i_err)
!
!     Evaluate the function and its derivatives of a user defined surface
!
!     Parameters:
!     ----------
!     icalc  = 1 : calculate surface z coordinate at coordinates x,y (input)
!            = 2 : calculate xn,yn,zn direction cosines at x,y,z
!     isur       : surface number                                   (input)
!     curv       : curvature                                        (input)
!     sdata(91)  : Array containing the user-defined parameters     (input)
!                  For example, sdata(1) is the value entered with the
!                  command UCO C1.
!     x,y,z      : Coordinates of the current position of the ray with
!                  respect to the origin of the surface             (input)
!     xn,yn,zn   : Derivatives of the surface at coordinates (x,y,z) (output)
!     i_err      : Error flag (0 = no error, 1 = error)             (output)
!
!     Notes:
!     -----
!     The example code given below calculates coordinates and derivatives
!     of a parabolic surface based on the curvature "curv".
!     The user will typically substitute his own FORTRAN code for a
!     particular surface.
!
!     More than one surface description can be programmed in this subroutine.
!     Use the "isur" parameter to distinguish between surfaces and
!     determine the interpretation of the coefficients stored in "sdata"
!
      dll_export usersur
      integer          :: icalc,i_err,isur
      double precision :: x,y,z,xn,yn,zn,curv,sdata(91)
      double precision :: fnorm
!
      i_err = 0
!
      z = 0.5d0*curv*(x*x + y*y)    ! surface z-value, paraboloid
!
      if(icalc.ge.2) then           ! calculate surface derivatives at x,y,z
         xn = x*curv
         yn = y*curv
         fnorm = dsqrt(xn*xn + yn*yn + 1.0d0)
         xn = xn/fnorm
         yn = yn/fnorm
         zn = -1.0d0/fnorm
      endif
!
      return
      end
```

The parameter list in `usersur.f90` is fixed and must not be changed by the user. Compilation and creating a *dynamic link library* (DLL) with Lahey/Fujitsu FORTRAN-95 requires version 5.7 onwards. Note that earlier versions of Lahey/Fujitsu FORTRAN do not create compatible DLL's and libraries.

To create a 32-bit Windows DLL using Lahey/Fujitsu LF95, the `-dll` switch must be used. Example:

`LF95 usersur.f90 -dll -win -ml LF95`

In order to reference a procedure across a DLL interface, the compiler must be informed of the procedure name and told how to 'decorate' the external names in your DLL. The procedure name is defined by the '`dll_export`' statement in '`usersur.f90`'. Note that the procedure name '`usersur`' in the '`dll_export`' statement is case-sensitive. It must be written in small letters to be recognized by the $OpTaliX$ main program.

8.30.4 Compiling with Intel Fortran 90 and Compaq Visual Fortran

The Intel Fortran compiler (versions $\leq$ 8.xx) and the Compaq Visual Fortran compiler do seamlessly coexist. Current versions tested are Compaq 6.6 and Intel 7.1. here is the source code example of a user defined surface (UDS) in FORTRAN with specific directives for the Intel/Compaq Fortran compilers:

```
      subroutine usersur_(icalc,isur,curv,sdata,x,y,z,xn,yn,zn,i_err)
!
!------- for Intel Fortran V7.xx ---------------------------------------------
!
!     Evaluate the function and its derivatives of a user defined surface
!
!     Parameters:
!     ----------
!     icalc  = 1 : calculate surface z coordinate at coordinates x,y (input)
!            = 2 : calculate xn,yn,zn direction cosines at x,y,z
!     isur       : surface number                                    (input)
!     curv       : curvature                                         (input)
!     sdata(91)  : Array containing the special user-defined parameters (input)
!                  For example, sdata(1) is the value entered with the
!                  command UCO C1.
!     x,y,z      : Coordinates of the current position of the ray with
!                  respect to the origin of the surface              (input)
!     xn,yn,zn   : Derivatives of the surface at coordinates (x,y,z) (output)
!     i_err      : Error flag (0 = no error, 1 = error)              (output)
!
!     Notes:
!     -----
!     The example code given below calculates coordinates and derivatives
!     on a parabolic surface based on the curvature "curv".
!     The user will typically substitute his own FORTRAN code for a
!     particular surface.
!
!     More than one surface description can be programmed in this subroutine.
!     Use the "isur" parameter to distinguish between surfaces and
!     determine the interpretation of the coefficients stored in "sdata"
!
      !DEC$ ATTRIBUTES DLLEXPORT:: usersur_
      !DEC$ ATTRIBUTES ALIAS: 'usersur_':: usersur_   ! forces lower case
!
      integer          :: icalc,i_err,isur
      double precision :: x,y,z,xn,yn,zn,curv,sdata(81)
      double precision :: fnorm
!
      i_err = 0
!
      z = 0.5d0*curv*(x*x + y*y)    ! surface z-value (paraboloid)
!
      if(icalc.ge.2) then           ! calculate surface derivatives at x,y,z
         xn = x*curv
         yn = y*curv
         fnorm = dsqrt(xn*xn + yn*yn + 1.0d0)
         xn = xn/fnorm
         yn = yn/fnorm
         zn = -1.0d0/fnorm
      endif
!
      return
      end
```

The parameter list in `usersur.f90` is fixed and must not be changed by the user.

Intel compiler: Compilation and creating a *dynamic link library* (DLL) with Intel FORTRAN requires version 7.1 onwards. The DLL is created on the command line:

`ifl usersur.f90 /LD`

Compaq compiler: Compilation and creating a *dynamic link library* (DLL) with Compaq Visual FORTRAN from the OS-command line is accomplished by:

`DF /dll usersur.f90`

Both compilers Intel and Compaq FORTRAN require the following meta instructions:

The procedure name is defined by the '`!DEC$ ATTRIBUTES DLLEXPORT:: usersur_`' directive. Lower case is forced by the alias instruction '`!DEC$ ATTRIBUTES ALIAS: 'usersur_':: usersur_`'.

8.30.5 Compiling with Intel FORTRAN Parallel Studio

This section describes coding of user-defined surfaces for the "Intel Fortran Parallel Studio", versions 11.xx onwards. The former name was "Intel Visual Fortran Compiler". Here is the source code example of a user defined surface (UDS) in Intel Fortran Parallel Studio:

```
      subroutine usersur(icalc,isur,curv,sdata,x,y,z,xn,yn,zn,i_err)
!
!------- for Intel Visual Fortran Composer, > V9.xx ---------------------------------------------
!
!     Evaluate the function and its derivatives of a user defined surface
!
!     Parameters:
!     ----------
!     icalc  = 1 : calculate surface z coordinate at coordinates x,y (input)
!            = 2 : calculate xn,yn,zn direction cosines at x,y,z
!     isur       : surface number                                    (input)
!     curv       : curvature                                         (input)
!     sdata(91)  : Array containing the special user-defined parameters (input)
!                  For example, sdata(1) is the value entered with the
!                  command UCO C1.
!     x,y,z      : Coordinates of the current position of the ray with
!                  respect to the origin of the surface              (input)
!     xn,yn,zn   : Derivatives of the surface at coordinates (x,y,z) (output)
!     i_err      : Error flag (0 = no error, 1 = error)              (output)
!
!     Notes:
!     -----
!     The example code given below calculates coordinates and derivatives
!     on a parabolic surface based on the curvature "curv".
!     The user will typically substitute his own FORTRAN code for a
!     particular surface.
!
!     More than one surface description can be programmed in this subroutine.
!     Use the "isur" parameter to distinguish between surfaces and
!     determine the interpretation of the coefficients stored in "sdata"
!
      !DEC$ ATTRIBUTES DLLEXPORT:: USERSUR
!
      integer          :: icalc,i_err,isur
      double precision :: x,y,z,xn,yn,zn,curv,sdata(81)
      double precision :: fnorm
!
      i_err = 0
!
      z = 0.5d0*curv*(x*x + y*y)     ! surface z-value (paraboloid)
!
      if(icalc.ge.2) then            ! calculate surface derivatives at x,y,z
         xn = x*curv
```

```
          yn = y*curv
          fnorm = dsqrt(xn*xn + yn*yn + 1.0d0)
          xn = xn/fnorm
          yn = yn/fnorm
          zn = -1.0d0/fnorm
        endif
!
        return
        end
```

The parameter list in `usersur.f90` is fixed and must not be changed by the user.

Compilation and creating a *dynamic link library* (DLL) with Intel Fortran Parallel Studio requires version 13.xx onwards. The DLL is created on the command line:

```
ifort /dll usersur.f90
```

The procedure name is defined by the '`!DEC$ ATTRIBUTES DLLEXPORT:: USERSUR`' directive.

8.30.6 Compiling with Microsoft Visual Studio 2012 and higher

A program written in C must bridge the conventions on naming of functions, subroutines and arguments between FORTRAN and C. Since $OpTaliX$ is a FORTRAN package, in the example that follows we will modify the C side accordingly.

The FORTRAN call to the subroutine `USERSUR` will generate a requirement for an external symbol called `_USERSUR_`. For a subroutine written in C the entry point name must be `USERSUR_` (note the absence of the leading underscore, which will be added by the C compiler).

Typically, arguments in FORTRAN are passed by reference. C compilers, on the other hand, pass scalar variables by value, rather than its address. This essentially means that C functions should be set up so as to expect that all visible arguments are being passed by reference, or as "pointers" in the C lingo (hence the "*" in front of the variable names).

Also note that all C arrays start at 0 whereas FORTRAN arrays typically start at 1. The parameter adjustment `--sdata` accounts for this fact.

Notes for C++ : C++ allows function overloading. Therefore functions are stored differently in the *.lib files compared to the classical C. Because we are not overloading any functions here, we instruct the C++ compiler that we want to use traditional C. Note the following code excerpts,

```
#ifdef __cplusplus
extern "C" {
#endif
```

before the usersur declaration, and at the end of the source code

```
#ifdef __cplusplus
}
#endif
```

This makes the linker to store functions correctly regardless of the C compiler used. Here is the sample code of `usersur.c`, respectively `usersur.cpp` :

```
#include <math.h>
#include <string.h>
#include <windows.h>
#define PI 3.14159265359

/* Subroutine */
#define usersur_ USERSUR

#ifdef __cplusplus
extern "C" {
#endif

int __declspec(dllexport) usersur_(int *icalc, int *isur, double *curv, double *sdata, double *x, double *y, doub
/* Builtin functions */
/* uncomment the following line only if not declared in the math.h file */
/* double sqrt(); */

/* Local variables */
double fnorm;


/*     Evaluate the function and its derivatives of a user defined surface */

/*     Parameters: */
/*     ----------  */
/*     icalc  = 1 : calculate surface z coordinate at coordinates x,y (input) */
/*            = 2 : calculate xn,yn,zn direction cosines at x,y,z */
/*     isur       : surface number                                  (input) */
/*     curv       : curvature                                       (input) */
/*     sdata(81)  : Array containing the special user-defined parameters (input) */
/*                  For example, sdata(1) is the value entered with the */
/*                  command UCO C1. */
/*     x,y,z__    : Coordinates of the current position of the ray with */
/*                  respect to the origin of the surface            (input) */
/*     xn,yn,zn   : Derivatives of the surface at coordinates (x,y,z) (output) */
/*     i_err__    : Error flag (0 = no error, 1 = error)            (output) */

/*     Notes: */
/*     -----  */
/*     The example code given below calculates coordinates and derivatives */
/*     of a parabolic surface based on the curvature "curv". */
/*     The user will typically substitute his own C code for a */
/*     particular surface. */

/*     More than one surface description can be programmed in this subroutine. */
/*     Use the "isur" parameter to distinguish between surfaces and */
/*     determine the interpretation of the coefficients stored in "sdata" */


/* Parameter adjustments */
--sdata;

/* Function Body */
*i_err__ = 0;

*z__ = *curv * .5 * (*x * *x + *y * *y);

/* surface z-value (paraboloid) */
if (*icalc >= 2) {
/* calculate surface derivatives at x,y,z */
*xn = *x * *curv;
*yn = *y * *curv;
fnorm = sqrt(*xn * *xn + *yn * *yn + 1.);
*xn /= fnorm;
*yn /= fnorm;
*zn = -1. / fnorm;
}

return 0;
}
```

```
#ifdef __cplusplus
}
#endif
```

The parameter list in `usersur.c` or `usersur.cpp` is fixed and must not be changed. All entries after the comment line `/* Function Body */` may be freely modified by the user.

Microsoft Visual Studio 2012 or later is recommended. The newer versions allow improved processor-specific optimizations. Creating a DLL using Microsoft Visual Studio is accomplished in several steps:

1. From Microsoft Visual Studio select File ⟶ New ⟶ Project.
2. Select `Win32Project`
3. Give the project name: `usersur`
4. In the Windows Desktop Project window select application type: Dynamic Link Library (.dll)
5. Uncheck the ”Precompiled Header”
6. Uncheck the ”Security Development Lifestyle (SDL)” checks
7. In the *Solution Explorer* ⟶ *usersur* ⟶ *Source Files*, find the code `usersur.cpp` and delete it.
8. Right click the *Source Files* folder, select *Add* ⟶ *Existing item*, and add the template `usersur.cpp` source code copied from the `OpTaliX-PRO\usersur\C\MS-Visual-Studio_2015` directory.
9. Select *Configuration Manager* and make sure that Platform setting corresponds to the $OpTaliX$ edition used, i.e., x64 for 64 bit version and Win32 for 32 bit version of $OpTaliX$.
10. Compile your code by selecting *Build* ⟶ *Build usersur* (or *Build* ⟶ *Rebuild usersur*). When compilaton is successfull, the Output window reports locaton of created libraries.
11. Find the files `usersur.lib` and `usersur.dll` in the location above and copy them into the $OpTaliX$ installation directory `C:\Program Files\OpTaliX-PRO`

It is advised to make backup copies of original `usersur.lib` and `usersur.dll` files. $OpTaliX$ will not start if the libraries are not valid.

8.31 Lens Modules

A lens module is a black box with defined optical parameter on input and output, but hiding all internal properties and structure. Lens modules are usually selected when the detailed optical prescription is not known or only a conceptual layout of an optical system is required. Only first order properties of a lens can be modelled by a lens module. As a minimum parameter, the module focal length (`MFL`) must be provided.

`MOD sk|si..j`	Converts the surface type of two surfaces into a lens module. The surfaces must exist. If only one surface is specified, the surfaces sk and sk+1 will be converted.
`MFL sk mod_focal_length`	Module focal length. sk is the first surface of the module range.
`MRD sk red_ratio`	Module reduction ratio. Note that MRD is the negative magnification of the module. By default `MRD` = 0.
`MCO sk ci..j`	Module coefficients (reserved for future editions)

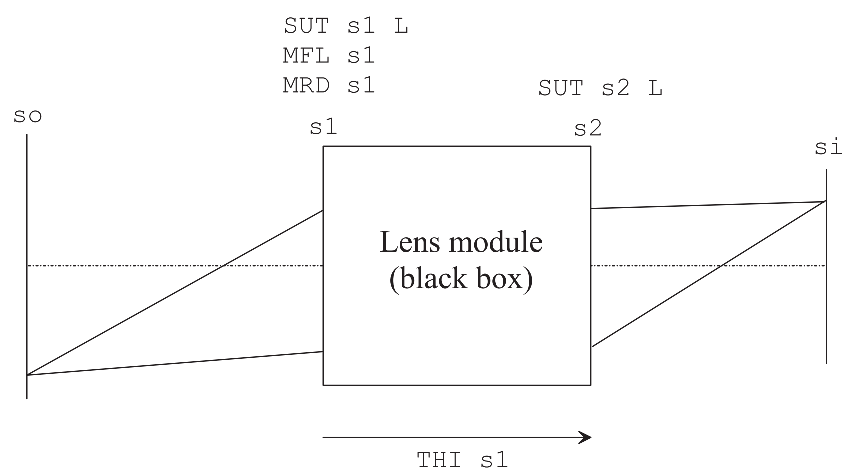


Figure 8.46: Lens module (perfect lens).

A lens module behaves as a *perfect lens* only at a single magnification which is defined by `MRD`. A lens module must always be defined by two consecutive surfaces of surface type ”L”. These surfaces define the entrance surface and exit surface of the lens module. Entrance and exit surface represent the principal planes of the module. For thick lenses or lens systems, the separation of the principal planes is defined by the thickness assigned to the entrance surface. All module parameters (`MFL`, `MRD`) must be specified at the entrance surface.

Lens modules can be applied only to finite conjugates. Infinite conjugates (object or image space) are approximated. For example, a reduction ratio of zero is modelled internally by 10^{-16}. Similarly, infinite magnifications are treated as 10^{+16}.

Example setting up a lens module:

```
ins s3..4     ! insert two surfaces which shall define the module
sut s3..4 L   ! make surfaces 3 to 4 module surfaces by setting surface type to ”L”,
              ! alternatively use the MOD s3 command
mfl s3 100    ! module focal length is 100 mm
mrd s3 1      ! module reduction ratio = 1 (module magnification = -1)
```

8.32 Surface Apertures

Apertures on surfaces are used to define and limit the light beam passing through a lens system. Up to 10 basic aperture shapes (rectangular, elliptical, circular and polygon) can be assigned to a surface.

Note that surface apertures must not be confused with the system aperture. For a detailed explanation of defining system aperture see sect. 7.3.5 (page 48).

Each basic aperture on an individual surface may be transmitting or obstructing, it can be decentered in X- and Y-direction from the local surface vertex and it can be rotated. Basic apertures may be logically combined by `.and.` respectively `.or.` operators. The operator `p` is used to address the different basic apertures on a given surface.

The following commands define apertures at surfaces:

`REX si..j pi..j [OBS|HOL|EDG|.or.|.and.] x_height`	Rectangular aperture. `x_height` is the semi-aperture in X-direction. See also notes below.
`REY si..j pi..j [OBS|HOL|EDG|.or.|.and.] y_height`	Rectangular aperture. `y_height` is the semi-aperture in Y-direction. See also notes below.
`ELX si..j pi..j [OBS|HOL|EDG|.or.|.and.] x_half_width`	Elliptical aperture. `x_half_width` is the semi-aperture (half width) in X-direction. See also notes below.
`ELY si..j pi..j [OBS|HOL|EDG|.or.|.and.] y_half_width`	Elliptical aperture. `y_half_width` is the semi-aperture (half width) in Y-direction. See also notes below.
`CIR si..j|sk pi..j [OBS|HOL|EDG|.or.|.and.] radius`	Defines circular aperture. `radius` is the semi-aperture of the circle. See also notes below.
`REC si..j|sk pi..j [OBS|HOL|EDG|.or.|.and.] x_height y_height`	Defines rectangular aperture. `x_height` and `y_height` describe the semi-apertures in X-direction and Y-direction respectively. If only `x_height` is specified, a square aperture is assumed.
`APT si..j cir|rec|ell|pol`	Set aperture type, i.e. the form of a surface aperture. It can be **cir**cular, **rec**tangular, **ell**iptical, or a **pol**ygon. This command is synonymous to the "CIR", "ELX", "ELY", "REX", "REY" commands. It was introduced to facilitate aperture shape definitions in a zoom/multiconfiguration environment.
`ADX si..j pi..j x_offset`	X-offset of aperture center
`ADY si..j pi..j y_offset`	Y-offset of aperture center
`ARO si..j pi..j rot_angle`	Rotate designated aperture on surface(s) `si..j`. Rotation is performed after `ADX`,`ADY`.

continued on next page

continued from previous page	
PLG si..j pi..j ck xk_vertex yk_vertex PLG si..j pi..j file data.plg	Polygon aperture. Two forms of defining polygon vertices are possible: The first form defines a single polygon vertex on surface(s) `si..j`, aperture element(s) `pi..j` and vertex (coefficient) `ck`. `xk_vertex, yk_vertex` are the polygon vertex coordinates. Example: `plg s3 p2 c4 12.0 3.0` The second form reads all polygon vertices from a file `data.plg`. Note that the "file" qualifier in the command is obligatory to interpret the subsequent string as a file name. The file format follows the conventions of INT files (see page 504). See also the detailed description for dialog-based entering of polygon data (section 8.32.1) and for reading polygon data from a file (section 8.32.1.2)
DEL APE sk|si..j pi..j|EDG	Delete aperture definition `pi..j` on surface(s) `si..j`. The alternate form `DEL APE sk|si..j EDG` deletes edges on the designated surfaces.

Notes:

- The parameter `p` may be omitted for the first sub-aperture, i.e. the commands

  ```
  cir s1 p1 30
  cir s1 30
  ```

  are identical.

- `OBS` means this is an obstructing aperture. Rays which hit the surface inside the border of an obstructing aperture element are blocked.

- `HOL` denotes a hole at the designated aperture, that is, rays inside a hole aperture are not affected by refraction or reflection on that surface, they "pass through" without any interaction. HOL aperture elements are used with sequential *and* non-sequential surfaces (see also sect.8.32.2).

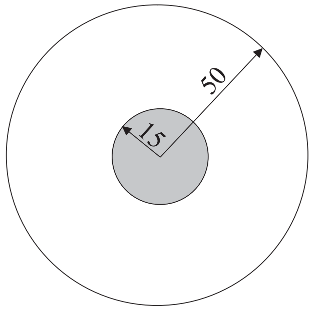
- `EDG` means this is the edge of the element following the designated surface. That is, it is only necessary to specify the `EDG` for the first surface of an element. `EDG` values specified on the rear surface of an element are ignored. Element edges are shown in the lens layout plots, are used in weight calculation and in lens element drawings. Edges, however, do NOT generate clear apertures. Use the FHY command instead for defining hard limiting (fixed) apertures.

- EDG apertures are deleted by defining a zero value, for example `CIR EDG s4 0`, or by the command `DEL APE sk|si..j EDG`.

- The `EDG` **option** used in `REX`, `REY`, `ELX`, `ELY`, `CIR` or `REC` commands must not be confused with the EDG **command**, which only defines how edges are drawn in the lens layout plot (VIE).

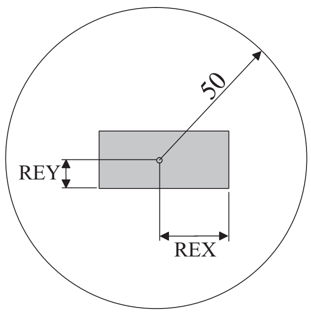
- By default, apertures do not limit or truncate ray beams, except where an obstructing (`OBS`) property is specified. However, apertures may limit or truncate beams by defining it "fixed" using the `FHY` command (see section 8.32.3, page 163 below). Then rays hitting a surface outside the aperture bounds will be blocked.

Examples of aperture shapes are shown below to illustrate usage of the commands:



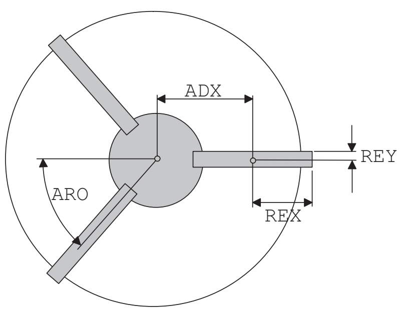
Circular aperture with central obscuration:

```
cir s1 50
cir s1 p2 obs 15
```



Circular aperture with rectangular obscuration:

```
cir s1 50
rex s1 p2 obs 20
rey s1 p2 obs 5
```



Circular aperture with circular central obstruction and spider with 3 vanes:

```
cir s1 50
cir s1 p2 20 obs
rex s1 p3 30 obs
rey s1 p3 5 obs
adx s1 p3 25
aro s1 p3 0
⋮
```

8.32.1 Polygon Apertures

Polygon aperture elements are constructed from up to 50 vertices and allow almost arbitrary aperture shapes. Polygon vertices are given as (X,Y) data pairs and are referred to the vertex of the optical surface. The entire polygon can be shifted and rotated by theADX, ADY and ARO commands.

Polygon apertures must be closed, i.e. the last vertex must have the same coordinates as the first vertex. Polygon apertures need NOT to be convex and any shape is allowed as indicated in Fig. 8.47. Up to ten polygon apertures are allowed on each surface, however, the total number of polygon apertures in an optical system is limited to 50.

8.32.1.1 Dialog-based editing of polygon apertures

Polygon apertures are edited in the surface spreadsheet editor (invoked by EDI SUR command) in the ”special apertures” tab. Set the aperture type in the first column of this tab to ”polygon”. The appropriate check box in the last column will be activated. Click on this check box and a dialog box as shown in Fig. 8.47 will be displayed.

The shape of the polygon (but not its absolute size) will always be updated as new vertices are entered. The polygon data can be uniformly scaled respectively a new set of polygon data can be imported from a file.

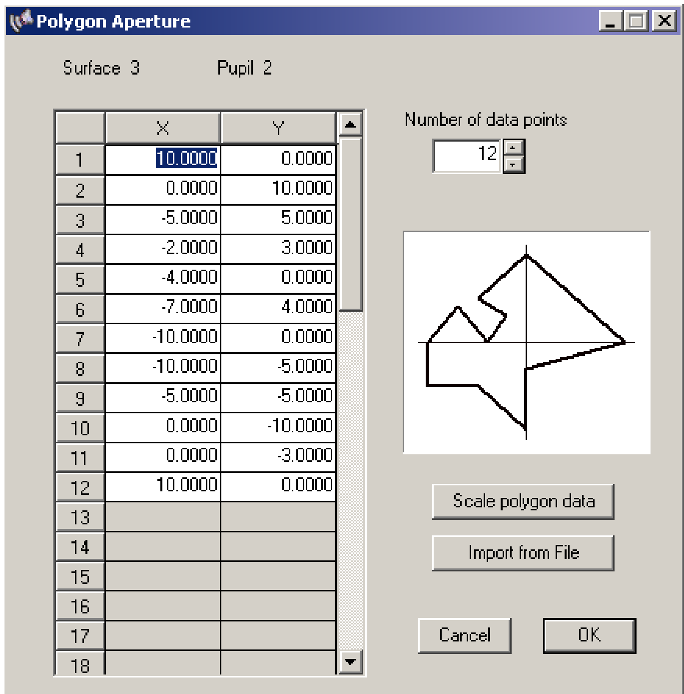


Figure 8.47: Dialog-based editing of polygon apertures.

8.32.1.2 Reading polygon apertures from a file

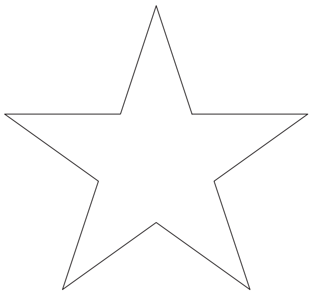
Complex polygon shapes can also be read in from an ASCII file. The data must be stored as (X,Y) data pairs, the file format must conform to the definition of INT-files as given in section 32.11, page 504. The file extension is preferably `.plg`, however, any other extension is also accepted. Fig. 8.48 shows an example polygon file of a five-pointed star (note that the first two lines in the file are mandatory):

8.32.2 Hole Aperture

On a "hole" aperture element, rays inside the specified hole aperture are passing through unaffected, i.e. they do NOT undergo refraction, reflection or diffraction on that surface. Hole apertures can be applied to both sequential and non-sequential surfaces. Hole apertures cannot be applied to the base aperture on a surface (i.e. aperture pointer p1), use `p2` or higher. Here is a concise command sequence for entering hole apertures:

```
cir s3 p2 5.0 hol   ! Defines a circular hole on surface 3, aperture element 2, with 5mm radius,
rex s4 p2 4.0 hol   ! Rectangular hole on surface 4, aperture element 2, X-height is 4mm,
rey s4 p2 2.0 hol   ! Rectangular hole on surface 4, aperture element 2, Y-height is 2mm,
```

Command:
`plg s3 p2 file c:\star.plg`



Contents of file star.plg:

```
Five pointed star
SSZ 1.0 !  scaling factor
11 !  number of polygon vertices
0 1
-0.2245 0.309
-0.9511 0.309
-0.3633 -0.118
-0.5878 -0.809
0 -0.382
0.5878 -0.809
0.3633 -0.118
0.9511 0.309
0.2245 0.309
0 1
```

Figure 8.48: Defining and assigning a five-pointed star polygon aperture from file `star.plg` to surface 3, pupil number 2.

Note that special apertures (such as obscurations, holes, polygons, etc.) are **only** active if the the fixed height (FHY) attribute has been assigned to the designated surface. A detailed description on ”fixed heights” is given in section 8.32.3.

In sequential systems only, hole apertures are ignored for calculation of the principal properties of an optical system, such as focal length, focus position, aperture ratio, etc., and for all ray aiming purposes. This behaviour assumes that sequential models are primarily based on traditional systems where the imaging function is determined by unobscured lenses/mirrors, and hole apertures were added for modeling additional features. Thus, for determination of system parameters (EFL, BFL, etc.) holes are ignored, whereas in all analysis options holes are correctly taken into account.

In order to study the effects of hole apertures, a simple example has been prepared. Load (restore) the file `$i\examples\Complex_Aperture\hole.otx` from the examples directory. A single lens is shown (see Fig. 8.49) bearing two hole apertures on surfaces 2 and 3.

8.32.3 Fixed Apertures (Heights)

It is sometimes necessary to set the aperture radius on a surface to a fixed value which must not change. In a pictorial way, one may say the aperture is ”frozen” to a certain dimension. This can be accomplished by the `FHY` command. Surfaces with fixed apertures are marked by a * (asterix) right to the APE-Y column in the prescription listing (LIS command) and in the surface editor. Rays outside the surface aperture marked by `FHY` are blocked.

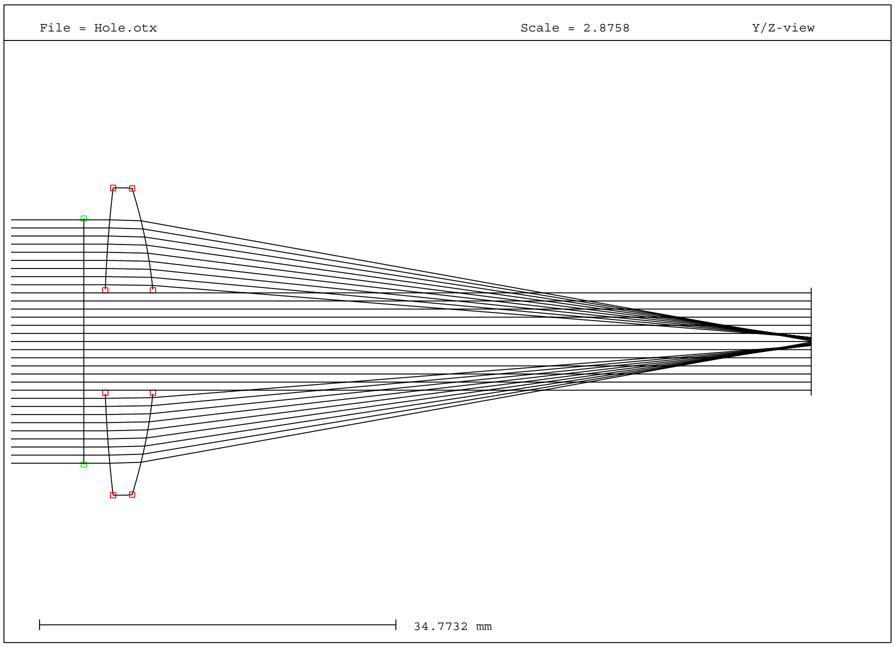


Figure 8.49: Hole apertures. Rays inside the hole aperture pass through unaffected. Here shown on a sequential model.

`FHY [si..j] 0/1`	Sets the apertures of surfaces `si..j` to fixed or floating. Surfaces marked by FHY = 1 block all rays which exceed the aperture radius. Also, aperture values of these surfaces will not be altered by the program, e.g. in modules which automatically set apertures (see `SET MHT` command).
`SET MHT [si..j, fi..j, zi..j, over_x, over_y]`	Automatically determines the maximum required surface apertures within the surface range `si..j`. The program takes the apertures of the stop surface and all surfaces marked `FHY` and computes the light beams going through the system. All apertures not marked `FHY` will be changed in according to the light beam. Note: Ray failures may be reported during maximum aperture determination, for example if total internal reflection occurs during ray iteration. This, however, will be resolved if there is a feasible solution. `over_x` and `over_y` are the oversizing factors for surface apertures (only for lens layout plot).

Example:

Light beams entering the system in Fig. 8.50 are defined by the stop surface (no. 5) and the surface apertures (heights) of surfaces 2 and 7. This way all off-axis beams get vignetted.

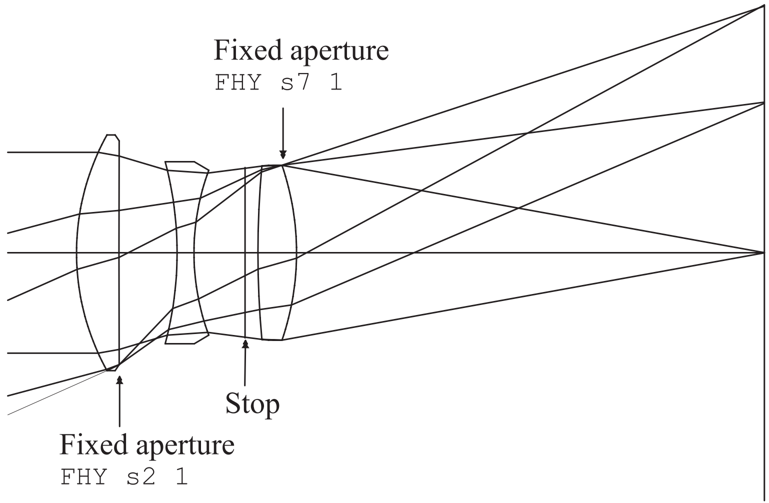


Figure 8.50: Defining vignetting characteristics with fixed apertures.

8.32.4 Editing Fixed Apertures in the Surface Editor

The fixed height (FHY) property may be edited in the surface editor in the column right to the `APE-Y` (aperture height) column:

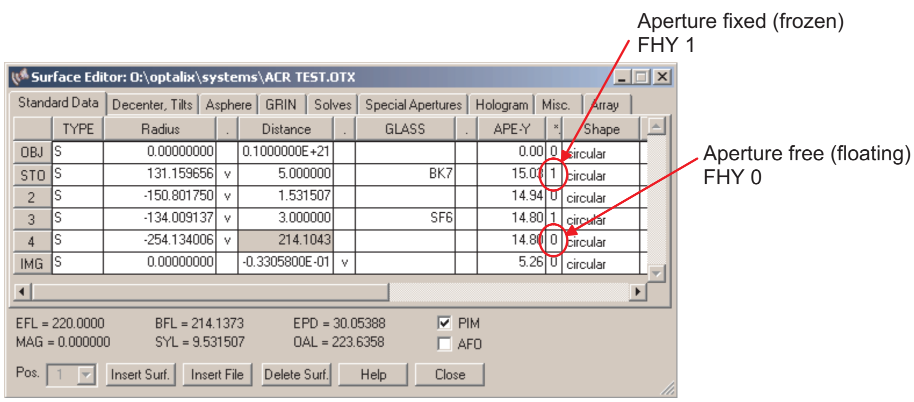


Figure 8.51: Defining fixed (frozen) apertures in the surface editor.

A fixed (frozen) aperture height is defined by 1 in the *-column right to APE-Y (corresponds to command `FHY sk 1`).
Floating apertures are defined by 0 in the *-column right to APE-Y (corresponds to command `FHY sk 0`).

8.33 Surface Comments

A comment field is provided for each surface, which accepts up to 80 characters of user text. This field is used for improving the readability of the lens data and has no impact on the lens analysis. Surface comments are entered using the command token ”`COM`”. For example:

```
COM s3..4 this is my comment
COM s1..2 "this is my comment"
```

Surface comments are listed by the command LIS COM or together with LIS ALL.

8.34 Insert, Invert, Copy, Move and Delete Surfaces

INS si..j target_surf [file file_spec]	Insert surfaces si..j before target surface. The optional parameter [file file_spec] inserts surfaces from a file. Examples: ins s3..4 ins s3..4 1 file c:/temp/mylens.otx The second example inserts surfaces 3 to 4 from the file c:/temp/mylens.otx before surface 1 of the current system.
INS MIR sk	Insert mirror surface before surface sk. By convention, the sign of radii, thicknesses and aspheric coefficients are reversed on surfaces following a mirror surface, which can be tedious if done manually. This command automatically inserts a surface, converts it to a mirror and reverts all necessary signs on subsequent surfaces. Example: ins mir s3
COP si..j target_surf [file file_spec]	Copies surfaces si..j to target surface. The target surfaces must exist. The optional parameter [file file_spec] copies the surfaces from a file. By default, the current directory is searched. Specify the full path if the file resides in a different directory. Examples: copy s3..4 8 ! copy surfaces 3-4 to surface 8 copy s3..4 8 file mylens.otx ! copy surfaces 3..4 from file mylens.otx to surface 8 and the following. copy s3..4 8 file c:\temp\mylens.otx ! As above but surfaces are copied from a file in a directory other than the current directory. The full path must be specified.
MOV si..j target_surf	Move surfaces si..j to the position of surface target_surf.
DEL si..j	Deletes surfaces si..j
DEL MIR sk	Delete mirror surface sk. This command combines two operations: It deletes the designated surface sk and reverts all necessary signs on subsequent surfaces. Surface sk must be a mirror surface, otherwise the command is ignored. Example: del mir s3
INV si..j	Invert surfaces si..j

8.35 Coatings / Multilayer Stacks

A complete package for design, analysis and optimization of thin film coatings is implemented in *OpTaliX* . This section describes how predefined coatings may be assigned (i.e. attached) to optical surfaces.

8.35.1 Attach Coatings to Surfaces

In the **command line**, attaching coating definitions to optical interfaces (surfaces) is accomplished by the following commands:

`ATT si..j [ FILE coating_name ]`	Attach a multilayer coating to surfaces si..j The `coating_name` refers to a file containing the coating prescription. It must exist. If the option `[FILE coating_name]` is absent, the default coating (if loaded) will be attached.
`COO si..j aut|nor|inv`	Orientation of coating when attached to an optical surface. `aut` = the orientation of the coating stack is automatically determined. `nor` = normal orientation, i.e. as defined in the coating file, `inv` = the coating is inverted (for example on a glass-air interface). Example: `coo s1..3 aut`
`DEL COA si..j`	Delete multilayer coating from surfaces si..j

In the **surface editor**, coatings (or multilayer stacks) may easily attached to surfaces by entering the coating file name into the ”coating” column of the surface editor, as shown in Fig. 8.52. The corresponding coating file must exist, either in the current directory (i.e. where the current optical design is stored) or in the general coating directory as defined in theprogram preferences (page 21).

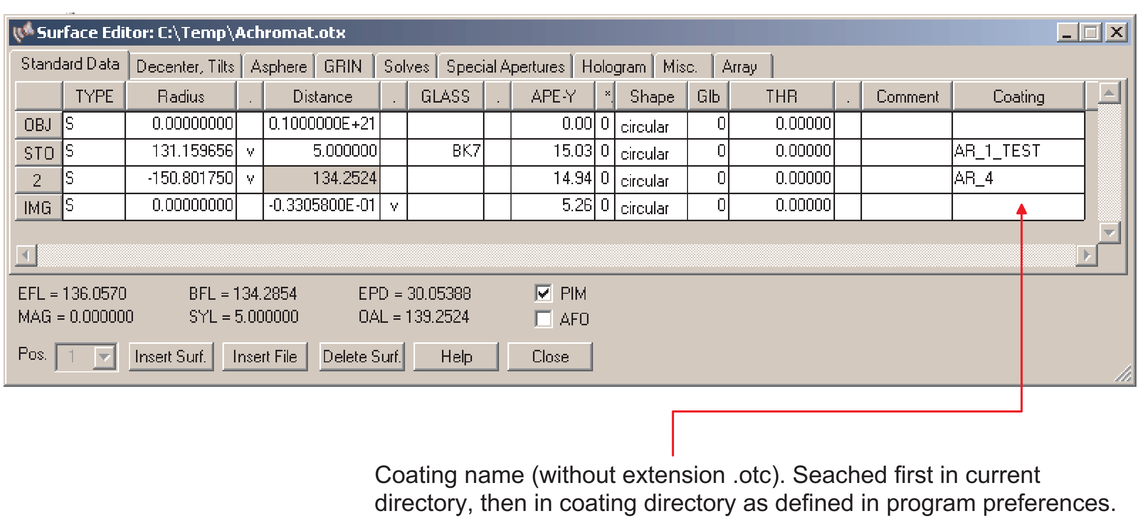


Figure 8.52: Enter coating name on optical surfaces. The coating name corresponds to the coating file name (without the extension `.otc`). The coating (file) is first searched in the current directory (i.e. where the current optical system resides) and, if not found, in the coating definition directory as defined in the program preferences (page 21).

8.35.2 Coating Orientation

Coatings are attached to surfaces as defined in the corresponding coating file. The regular orientation of coatings in OpTaliX is **air - layers - substrate**, respectively for cemented surfaces, **cement - layers - substrate**.

When attaching coatings to specific surfaces, $OpTaliX$ automatically detects the correct orientation of coatings. For example, on an air-glass interface, the coating is attached in normal orientation, i.e. as stored in the coating file, on glass-air interfaces, the coating is automatically inverted. This does not require any user interaction.

In special cases, however, it is advisable to explicitly specify the coating orientation to avoid any ambiguities. For example, cemented surfaces are a good example of overriding the automatic determination. Use the ”`COO NOR`” or ”`COO INV`” commands (without the double quotes), depending on how the layer sequence is defined in the coating file.

A detailed description on creating, changing and optimizing coatings is given in chapter20 on page 369.

8.36 Image Surface Definition

The image surface is typically the last surface in an optical system, however, it can be freely defined by use of the `IMG` command:

`IMG sk`	Defines the image surface number. `sk` must be less or equal the total number of surfaces in the optical system. The `IMG` command does not change the total number of surfaces in a system. Surfaces greater than `IMG` are '*inactive*' surfaces (i.e. not included in the ray trace) but are always stored/restored, irrespectively of the `IMG` setting.

Defining the image surface number is particularly useful in systems with intermediate images. The IMG command allows the re-definition of the image surface and the subsequent analysis and optimization at the new surface with a single command.

Note that the `IMG` command does not alter the total number of surfaces. That is, moving the image surface to a lower surface number still keeps surface data of **all** surfaces higher than `IMG` in memory. Also on storing/restoring optical systems, the total number of surfaces in a system is retained, irrespectively of the `IMG` setting.

For example, the system shown in Fig. 8.53 exhibits an intermediate image. Both the intermediate image and the final image can be simultaneously analyzed/optimized by defining the image surface number separately for each zoom position.

In the surface editor, surfaces greater than `IMG` are marked by blue colour to indicate that these surfaces are currently not active. Fig. 8.54 gives an example. Notice that parameters of inactive surfaces can always be edited.

There are a few restrictions connected with the `IMG` command:

- The IMG surface number must be less or equal the total number of surfaces in the optical system.
- The IMG surface number must not be the first surface or the object surface.
- The IMG surface number must always be greater than the stop surface number. If required, move the stop surface to a surface number lower than IMG (for example by zooming the stop surface).

Obligatory Surface Types		Optional Surface Types, add in arbitrary order	
S	Spherical surface	D	Decentered and/or tilted surfaces
A	Aspheric surface, see sections 8.6.1 to 8.6.5.	M	Mirror
L	Lens module (ideal lens)	G	Grating surface
X	"No-raytrace" surface. Only transforms surface coordinates without actually tracing rays to this surface. See sect. 8.22	H	Holographic surface
U	User-defined surface	F	Fresnel Surface
		I	Gradient index (GRIN) surface
		N	Non-sequential surface (NSS), must be used in combination with surface type "D"
		P	Pipe, (Light Pipe, step index fiber). The cone angle of tapered pipes/fibers is defined by the semi-apertures of the end surfaces
		R	Array (Lens Array)
		T	Total internal reflection (TIR) surface (see sect. 8.12, page 93)
		Z	Zernike surface
		C	Radial Spline deformation
		W	2-dimensional surface deformation, given as gridded data
		E	pure 2-dimensional spline (non-symmetric), no base surface. In preparation.

Table 8.2: Surface types

		K	$<$	-1	Hyperbola
		K	=	-1	Parabola
-1	$<$	K	$<$	0	Ellipse at major axis (prolate ellipse)
		K	$>$	0	Ellipse at minor axis (oblate ellipse)
		K	=	0	Sphere

Table 8.4: Geometric interpretation of conic constant K

Order	Tilt/decenter	Qualifier	Symbol
first	XDE (decenter X)	X	Δx
second	YDE (decenter Y)	Y	Δy
third	ZDE (decenter Z)	Z	Δz
fourth	ADE (tilt about X-axis)	A	α
fifth	BDE (tilt about Y-axis)	B	β
sixth	CDE (tilt about Z-axis)	C	γ

Table 8.21: Default tilt sequence and qualifying characters.

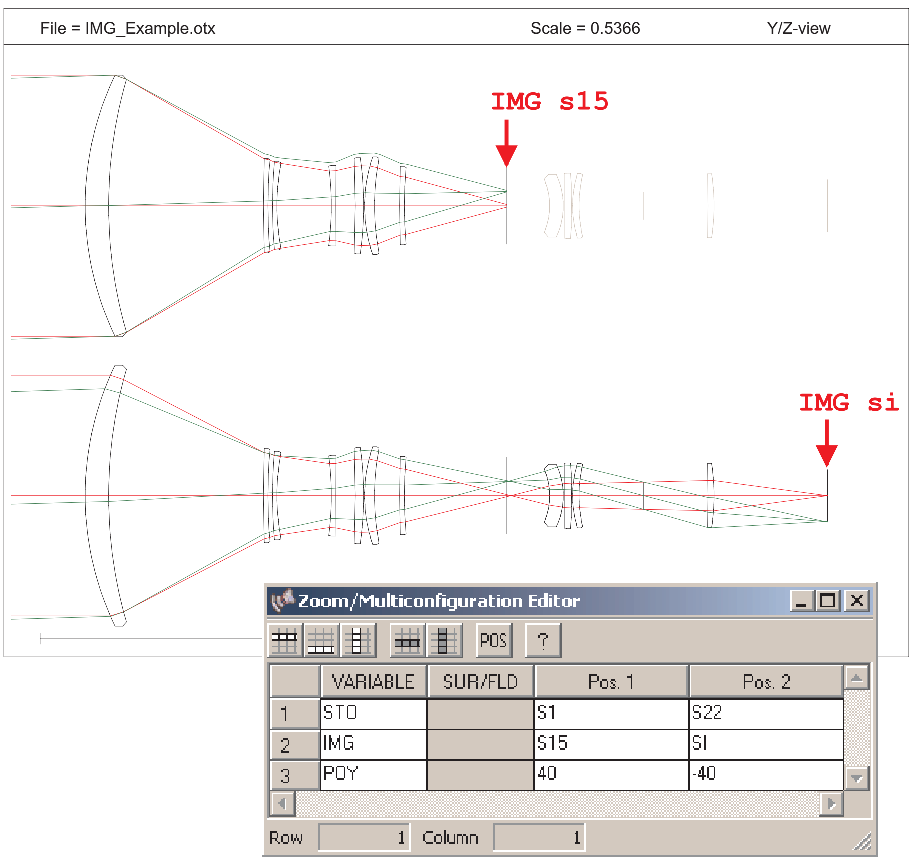


Figure 8.53: Re-defining the image surface number (IMG command) in a zoom/multi-configuration system. Top: Image surface is defined at an intermediate surface (`IMG s15`). Bottom: Image surface is the last surface in the system (`IMG si`). The corresponding zoom definitions are shown in the lower right corner (dialog box). The example file is found at: `$i\examples\zoom\img_example.otx`

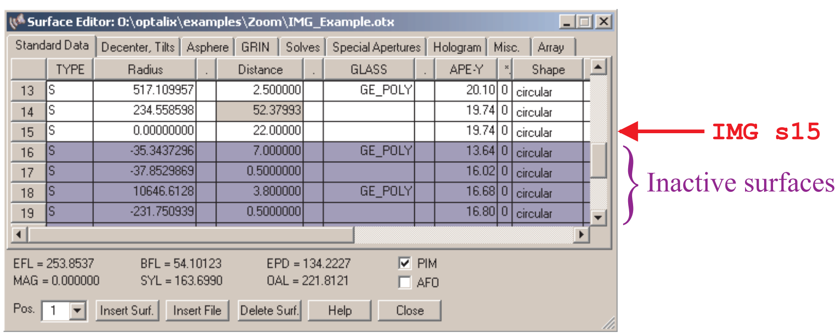


Figure 8.54: Inactive surfaces (i.e. surface numbers greater than `IMG sk`) are marked by blue colour in the surface editor, but still can be edited.

9

Listings, Reports

The `LIS` command gives an output of a complete lens description of the optical system. The listing also includes the first order properties as obtained from the FIR command.

9.1 List Prescription Data

Listings of prescription data and reports are obtained by the command:

`LIS [si..j] [ri..j] [options]`

or:

`LIS [options] > prn|filespec`

where `options` can be one of the following parameter

`RAY|GLA|ALG|IND|PIK|CNF|TXT|MUL|OPT|APE|TOL|TPL|COM|`
`CAM|OSP|PAR|DNDT|EXC|ALL`

Description of list options:

`ALL`	all options, list everything
`ALG`	alternative glasses with respect to a base glass. See also sect. 12.5 below.
`APE`	surface apertures (heights)
`CAM`	cam parameter.
`COM`	surface comment
`CNF`	configuration data
`DNDT`	absolute dn/dT of selected glasses. See also the notes below.
`EXC`	Linear expansion coefficient of selected glasses.

continued on next page

continued from previous page	
GLA	Lists all glasses in glass catalogue, which match a specified string. For example, `gla bk*` = all glasses beginning with ”bk” `GLA sch:bk*` = all glasses from SCHott beginning with ”bk” `GLA sch:*` = all glasses from Schott Note the use of the asterisk symbol ”*”, which does the wildcard matching. For example, the pattern `sf*` lists all glasses beginning with ”sf”, hence it will list SF1, SF2, SF11, SF6 and so on. The pattern ”sf” without asterisk will search for the glass ”sf”, which does not exist.
IND	refractive indices used in current system
MUL	multilayer definition
OPT	optimization data
OSP	optical spectrum
PAR	paraxial system data. See also FIR (page 242).
PIK	surface pickups (see also PKL surface pickups)
RAY	all rays
REM	remarks
TOL	tolerances
TPL	test plate list

Notes:

1. The redirection symbol ”>” allows immediate text-output to the printer (`prn`) or graphics output to the printer/plotter (`plt`) or to a file (`filespec`).
   Note: The output unit redirection is active only for one single command. Subsequent outputs will then appear on the default output device (screen) again.

2. The LIS DNDT command accepts an additional parameter, the temperature (in °C) at which dn/dT shall be calculated. For example, dn/dT data of Schott BK7 glass at 50°C are listed by:

   ```
   lis dndt bk7 50
   ```

   Omission of the temperature parameter resorts to the default temperature 20°C.
   dn/dT data is always listed for wavelengths defined in the system configuration.
   Glasses or wavelengths where dn/dT data is unavailable return -999.

Command Examples:

```
lis all            ! List all relevant surface data
lis > prn          ! Surface listing is redirected to printer (prn)
lis s1..5          ! List surfaces 1 to 5
lis ra             ! List all rays
lis r1..5          ! List rays 1 to 5
lis gla sf*        ! List all glasses beginning with ”sf”
lis dndt bk* 50    ! List absolute dn/dT for all glasses beginning with ”bk” at 50°C
```

9.2 List Alternative Glasses

Lists alternative (replacement) glasses with respect to a base glass. Alternative glasses are glasses having similar properties on refractive index and dispersion compared to the base glass and therefore may be used to replace the base glass in an optical system. The choice of alternative glasses is based on the given index difference (Δn_d) and the dispersion difference ($\Delta \nu_d$) at the d-line.

The syntax for listing alternative (replacement) glasses is:

`LIS ALG base_glass [delta_n delta_V]`[1]	List alternative (replacement) glasses with respect to a `base_glass`. By default, the tolerances on selecting an alternative glass are $\Delta n_d = 0.001$ on refractive index and $\Delta \nu_d = 0.8\%$ on dispersion, however, they may be overwritten by specifying `delta_n` and `delta_V`. See also the direct command 'ALG' (page 195).

Notice that the choice of alternative glasses is solely based on the Δn_d and $\Delta \nu_d$ differences. It is the designers responsibility to take other glass properties into account, such as partial dispersion, TCE, dn/dT, etc, depending on a particular application. This list is only intended to support you in selecting glasses from alternate vendors.

Example:

`LIS ALG N-BK7`

produces the following output:

```
ALTERNATIVE GLASS LIST :

Base glass              n_d       V_d    P(g,F)    P(C,s)      TCE     dndT   Melt  Price
   SCH:N-BK7       1.516798    64.141    0.5350    0.5612     7.10    1.160      1   1.00

Alternative glasses:
   SCO:BK7         1.516798    64.141    0.5350    0.5612     7.10    1.160      0   0.00
   SCO:UBK7        1.516800    64.264    0.5349    0.5603     7.00    1.102      0   0.00
   OHA:S-BSL7      1.516328    64.116    0.5353    0.5601     7.20    0.000      1   1.00
   OHA:L-BSL7      1.516328    64.039    0.5334    0.5646     5.80    0.000      0   0.00
   OHA:BSL7Y       1.516329    64.218    0.5343    0.5636     6.80    0.000      1   0.00
   COR:B1664       1.516802    64.198    0.5352    0.5609     6.80    0.000      0   0.00
   SUM:SU-BK7      1.516328    64.022    0.5346    0.5594     0.90    0.000      0   0.00
   HIK:H-E-BK7     1.516798    64.083    0.5358    0.5594     9.20    0.000      0   0.00
   HOY:BSC7        1.516797    64.172    0.5343    0.5615     7.50    0.541      1   0.00
   CDG:H-K9L       1.516797    64.172    0.5343    0.5615     7.50    0.000      0   0.00

Tolerance on nd     :  0.001
Tolerance on Vd     :  0.8 %
```

Notes:

- `dndT` values are always given as 10^{-6} units
- `Melt` indicates the glass manufacturers melt frequency. 1 corresponds to very high melt frequency, 5 corresponds to very low melt frequency. 0 means that there is no information available or that the glass is discontinued.
- `Price` is given relative to SCHOTT BK7. In absence of information, the relative price is `0.00`.

[1] Note that the previous command 'LIS ALT' is obsolete, however, still supported for backwards compatibility.

9.3 Description of Standard Listing Output

The data output with the `LIS` command are formatted to a fixed number of significant digits. If this is insufficient for a given item of data, full precision can be obtained with the EVA command (see also page 446). There are many options to the `LIS` command as described in section 9.1, however, the simplest form is just `LIS`. There are no qualifiers or data associated with the command (except for `LIS DNDT`, see page 174). You may also wish to direct output to a file with the OUT command (see page 431) prior to applying the `LIS` command.

The individual data listed with the `LIS` command, can be listed separately, as described in section 9.1. A standard listing is invoked by the command `LIS`, which is divided into three parts,

1. System data,
2. Surface data (standard),
3. Paraxial (first order) data.

An example listing (Double-Gauss lens from the examples library) indicates the three-parts logic as shown below:

Part 1, System Data:

```
FILE = DOUBLE_GAUSS.OTX                                          11.Jul.2004    15:49

Remarks:
  DOUBLE GAUSS - U.S. PATENT 2,532,751

Wavelength :       0.65630      0.58760      0.48610
Weight     :             1            1            1
REF =   2

XAN         0.00000        0.00000        0.00000
YAN         0.00000       10.00000       14.00000
FWGT            100            100            100
FACT              1              1              1

PIM = yes
SYM = yes
EPD =   25.0000
```

Part 2, Standard Surface Data:

```
  # TYPE         RADIUS     DISTANCE  GLASS          INDEX    APE-Y AP CP DP TP MP GLB
OBJ S          Infinity  0.10000E+21            1.000000     0.00  C  0  0  0  0   0
  1>S           28.7249      4.37333  BSM24      1.617644    15.00* C  0  0  0  0   0
  2 S           94.2300      0.14909            1.000000    14.60  C  0  0  0  0   0
  3 S           17.4436      6.21211  SK1        1.610248    12.71  C  0  0  0  0   0
  4 S          Infinity      1.88848  F15        1.605648    12.26  C  0  0  0  0   0
  5 S           10.7346      7.55393            1.000000     8.48  C  0  0  0  0   0
STO S          Infinity      6.46060            1.000000     7.74  C  0  0  0  0   0
  7 S          -13.5175      1.88848  F15        1.605648     8.44  C  0  0  0  0   0
  8 S          Infinity      5.41696  SK16       1.620408    10.45  C  0  0  0  0   0
  9 S          -17.4934      0.14909            1.000000    11.06  C  0  0  0  0   0
 10 S          293.3702      3.42909  SK16       1.620408    11.94  C  0  0  0  0   0
 11 S          -31.5576     31.52335            1.000000    12.00* C  0  0  0  0   0
IMG S          Infinity                         1.000000    12.62  C  0  0  0  0   0
```

Part 3, Paraxial Data:

```
PARAXIAL DATA AT INFINITE CONJUGATES:
   EFL                         50.00024       SH1 (Princ.Plane 1)       34.36081
   FNO                          2.00001       SH2 (Princ.Plane 2)      -18.43131

PARAXIAL DATA AT USED CONJUGATE:
   MAG (Magnification)          0.00000       SEP (Entr.Pup.Loc.)       27.93312
   NAO (Num.ape.object)         0.00000       EPD (Entr.Pup.Dia.)       25.00000
   NA  (Num.ape.image)          0.25000       APD (Exit Pup.Dia.)       28.68792
   BFL                         31.56893       SAP (Exit Pup.Loc.)      -25.80720
   DEF (Defocus)               -0.04558       PRD pupil relay dist     -16.21914
   IMD (Image distance)        31.52335       OAL (S1->Image)           69.04452
   OID (Object->Image)     0.10000E+21       SYL (System Length)       37.52117
```

9.4 List Global Coordinates and Global Matrices

Normally an optical system is described with respect to a chain of local coordinate systems for each surface (sequential model). However, it may be desirable to obtain the coordinates of each surface vertex in a global coordinate system. The following commands output the coordinates of surface vertices and the corresponding transformation matrices referred to a given surface.

For reference, see also the related commands for entering surface data referred to another surface (GLO command, page 114).

GSC [si..j]	Reports global surface coordinates referred to a reference surface which is defined by the GLO command (see below).
GSM [si..j]	Reports global surface matrix, referred to a reference surface which is defined by the GLO command (see below). The global surface matrix is a 3 by 4 matrix describing the global tilts and offsets of the surface vertices.
GLO sk [yes|no]	Set global coordinates analysis on/off. X/Y/Z surface coordinates for SIN, RSI and GSC (see above) are expressed relative to the single global origin defined by GLO. If GLO is not defined, sk defaults to s1. If sk is specified, the global surface coordinate output is referred to surface sk, otherwise s1 is used. Examples: glo s3 ! global surface coordinates are referred to surface 3 glo y ! Sets global surface output on. Reference surface is 1. glo yes ! As above, sets global surface output on. Reference surface is 1. glo ! Restore previous sk. If no previous GLO, uses s1. glo no ! Turn off global coordinate output.

Global coordinates of surface vertices may also retrieved from the lens database in EVA commands (page 26.9), in macros (page 439) and in optimization constraints (page 19.6):

XSC, YSC, ZSC	global vertex coordinates, referred to surface defined by GLO sk.
XSG, YSG, ZSG	global vertex coordinates, always referred to global system (no surface reference
CXG, CYG, CZG	global direction cosines of surface normal

Example Output: Global Surface Coordinates (GSC)

Command: `gsc`

```
*********** ABSOLUTE VERTEX COORDINATES REFERRED TO SURFACE   1 ************

        Surface vertex coordinates      :  Direction cosine of surface normal
------------------------------------------+-----------------------------------
  #          X           Y           Z :          NX          NY          NZ
                                                Alpha        Beta       Gamma
  1     0.00000     0.00000     0.00000 :   0.0000000   0.0392598   0.9992290
                                              2.25000     0.00000     0.00000
  2     0.00000  -116.19792 -1476.43457 :   0.0000000  -0.0155134   0.9998797
                                             -0.88889     0.00000     0.00000
  3     0.00000  -308.74461   273.85521 :  -0.0000020  -0.1651447   0.9862693
                                             -9.50564     0.00012     0.00000
```

The `GSC` command outputs X/Y/Z coordinates of each surface vertex referred to an arbitrary surface (see GLO command), the direction cosine of the surface normals and the global α, β, γ Euler tilt angles (in the sequence α, β, γ.

Example Output: Global Surface Matrices (GSM)

Command: `gsm`

```
GLOBAL SURFACE VERTEX COORDINATES AND TRANSFORMATION MATRICES:

Reference surface =  1

  #          M11          M12          M13             X     Alpha        Beta       Gamma
             M21          M22          M23             Y
             M31          M32          M33             Z

  1    1.0000000    0.0000000    0.0000000      0.000000     2.25000     0.00000     0.00000
       0.0000000    0.9992290    0.0392598      0.000000
       0.0000000   -0.0392598    0.9992290      0.000000

  2    1.0000000    0.0000000    0.0000000      0.000000    -0.88889     0.00000     0.00000
       0.0000000    0.9998797   -0.0155134    116.197921
       0.0000000    0.0155134    0.9998797   1476.434571

  3    1.0000000    0.0000000   -0.0000020      0.000000    -9.50564     0.00012     0.00000
      -0.0000003    0.9862693   -0.1651447    308.744609
       0.0000020    0.1651447    0.9862693   -273.855207
```

Surface tilts and decentrations can be conveniently described by a 3 x 4 matrix of the form:

$$\begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} & -X \\ m_{2,1} & m_{2,2} & m_{2,3} & -Y \\ m_{3,1} & m_{3,2} & m_{3,3} & -Z \end{bmatrix} \tag{9.1}$$

The $m_{i,k}$ coefficients hold the tilts whereas the fourth column contains the $X/Y/Z$ decentrations of the surface vertex with respect to the chosen reference. For a more detailed explanation of tilts defined by matrix notation see also section 8.18, page 111. In addition the α, β, γ Euler tilt angles (in the sequence α, β, γ are listed in the rightmost three columns.

9.5 List User-Defined Variables

`LVR`	Allows output of information about user-defined variables. Lists the current variable names and the associated arguments (numeric values or string). See also definition of variables in section 26.11, page 447.

9.6 List User-Defined Functions

LFC	Allows output of information about user-defined functions. Lists the current function names and the associated function definitions. See also definition of functions in section 26.16, page 452.

10

Lens Layout Plot

Plots the optical system as a cross-section or 3-D perspective drawing. The command accepts optional parameters to control the type of representation. See also the `GRA` command (section 25.1, page 432) for output to the printer and for export to other graphics formats.

`VIE [sec|si..j|zk|scale|?]`	Plots cross-section or perspective view of lens layout. `sec` is a single character describing the type of layout plot (optional): `X` : cross section in X/Z plane `Y` : cross section in Y/Z plane `P` : perspective view (wire frame) `si..j` = surface range, e.g. s3..7, (optional) `zk` = zoom position (optional) `scale` = plot scale (optional) `?` invokes a dialog box to edit lens plotting parameters. Example command: `vie Y s3..7 z4 0.5`
`VPT azimuth elevation`	Defines the azimuth and elevation angles (in degree) for three-dimensional perspective plot. The azimuth angle is measured in the X/Z-plane from -180° to + 180°, with 0° directing to the -X axis. The elevation angle is measured in the X/Y-plane, ranging from -180° to + 180°. The perspective distance is always at infinity (parallel projection). The graphics window containing the perspective plot will be automatically updated if it is opened.
`LDS`	Same as `VIE`, however, the layout plot is always drawn in a screen window, irrespective of other settings of graphics output units. See also setting of other graphic output units by the `GRA` command, page 432.

continued on next page

continued from previous page	
REN	Create an almost photo-realistic rendered image of the lens system. The rendering information is written to the file "optix.pov" in the $OpTaliX$ temporary directory (usually /optix/temp) and the POV rendering engine is subsequently called. See also section 10.1 on how to interface $OpTaliX$ to POV. The rendering information (POV-file) may also be separately written (exported) to a specific file using the EXP POV command (see page 481).
RSP	Traces a single ray in the Y/Z lens layout plot. The start coordinates of the ray can be interactively adjusted in field and aperture using slider bars in a dialog box. This command does not output any ray trace data. Use the command RSI on page 232 to obtain precise ray coordinates.
AAP yes|no	Plots asymmetric apertures. In lens plot, draws only the used aperture of a surface. AAP no plots the full surface aperture, irrespective of the actual area used by the light beams (surfaces are drawn symmetrical to their local axis). AAP no is the default.
POX,POY,POZ [zi..j|zk]	Plot offsets (in paper coordinates). Shifts the lens layout plot in x- and y-direction on the paper. For "zoomed" systems, individual values for POX,POY,POZ may be specified. In this case, the plot offsets must be preceded by a ZOO qualifier and specified as described in the zoom section (see page 188).
EDI LDR	Edit lens draw parameter for lens layout plot. A dialog box is invoked.
PPOS plot_pos	Plot zoom position. This is an extended variant of the POS command for setting a specific zoom position. If plot_pos, an integer number between 1 and the maximum defined zoom positions, is specified, only the layout of position plot_pos will be drawn. If plot_pos is 0, *all* positions will be plotted.
	continued on next page

continued from previous page

<table>
<tr>
<td>EDG [si..j|yes|no] edge_type_no

or

EDG [si..j|yes|no] edge_string</td>
<td>Edge drawing. Specifies how edges of lens elements are drawn. Edges may be specified by number (edge_type_no) or by a descriptive string (edge_string). See also Fig. 10.1 for an explanation of the various edge types.
<table>
<tr><th>edge type</th><th>edge string</th><th></th></tr>
<tr><td></td><td>yes</td><td>Edges are drawn on all elements for which edge_type_no is non-zero,</td></tr>
<tr><td></td><td>no</td><td>Without surface specifier, omits drawing of all edges, only surfaces are drawn on elements,</td></tr>
<tr><td>0</td><td>no</td><td>with surface specifier, omits drawing of edges only for the specified surfaces,</td></tr>
<tr><td>1</td><td>lin</td><td>connects edges linearly,</td></tr>
<tr><td>2</td><td>ang</td><td>connects edges by angled facets (default),</td></tr>
<tr><td>3</td><td>rec</td><td>connects edges by rectangular facets.</td></tr>
</table>
Example:
edg s5 3 or edg s5 rec ! Draws edges on lens element as rectangular facet.</td>
</tr>
</table>

Examples:

`vie y s1..5 2.5`	Lens draw, Y/Z-section, surfaces 1-5, scale = 2.5
`vie 1.5`	Lens draw , scale = 1.5, the other parameters are taken from the previous settings.
`vie 0`	Plot scaling is automatic. The program internally adjusts the plot scale to fit the layout plot onto the paper.
`vie ?`	Invokes a dialog box for adjustment of plot parameters prior to layout plotting.
`edg s5 3`	Draws edges on lens element as rectangular facet.
`edg s5 rec`	As above, draws edges on lens element as rectangular facet.

10.1 Using POV Rendering Engine

Creating photo-realistic pictures is accomplished by invocation of the Persistence of Vision (POV) renderer. POV is free and may be downloaded from http://www.povray.org. It must be installed separately and *OpTaliX* provides an interface to POV via the export module. In order to tell *OpTaliX* the location of POV, the path to the rendering engine must be modified in the *OpTaliX* configuration file `optix.cfg`. This may be accomplished in two ways:

1. Modify the file `optix.cfg`, which resides in the *OpTaliX* installation directory. Search for the key-word `RENDER` and change the path accordingly. Path names containing blanks must be enclosed in apostrophes. A typical example is

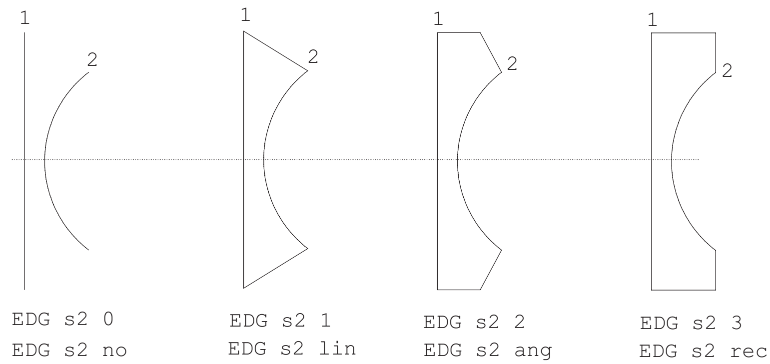


Figure 10.1: Various types of edge drawing.

```
RENDER = "c:/pov31a/bin/pvengine.exe"
```

2. From the main menu, select *FILE* $-->$ *PREFERENCES*. A dialog box appears to modify default search paths. The path to POV may be entered directly into the appropriate field or searched by clicking on the button right to the path-field.

Information: In order to use the POV interface, $OpTaliX$ must be installed on a writeable medium. If $OpTaliX$ is executed from a non-writeable medium (a CD_ROM for example), the whole $OpTaliX$ tree must be copied to a medium, which has write access.

10.2 Plot Rays

Only for purposes of plotting the lens layout, a set of special rays (hereafter denoted as *plot rays*) may be generated and stored with the optical system. These rays, however, are completely independent from rays generated internally by the program for image analysis.

Plot rays are generated by the following commands:

`SET RAY`	Generates a set of standard plot rays. These are typically 5 rays per field point: - a chief ray going through the stop center (or the entrance pupil center depending on the ray aiming method `RAIM`), - a meridional (tangential) upper limit ray - a meridional (tangential) lower limit ray - a sagittal upper limit ray - a sagittal lower limit ray.
`SET FAN [Y] [num_fan_rays]`	Sets a fan of rays in Y-direction. The number of rays (num_fan_rays) is uniformly distributed across the entrance pupil. Vignetted rays are not shown. Omission of the optional parameter `Y` or `num_fan_rays` uses the previous setting or the default setting (11 rays across aperture in Y-direction).

continued on next page

continued from previous page	
`SET FAN [X] [num_fan_rays]`	Sets a fan of rays in X-direction. The number of rays (num_fan_rays) is uniformly distributed across the entrance pupil. Vignetted rays are not shown. Omission of the optional parameter `X` or `num_fan_rays` uses the previous setting or the default setting (11 rays across aperture in Y-direction).
`SET FAN [XY] [num_fan_rays]`	Sets a fan of rays in both X-direction and Y-direction. The number of rays (`num_fan_rays`) is uniformly distributed across the entrance pupil. Vignetted rays are not shown. Omission of the optional parameter `XY` or `num_fan_rays` uses the previous setting or the default setting (11 rays across aperture in Y-direction).
`SET FAN [C] [num_circ_rays]`	Sets a fan of rays uniformly distributed around the used aperture circumference. Vignetting of the entrance beam is considered, thus, the plot rays may become elliptical in shape. Omission of the optional parameter `C` or `num_fan_rays` uses the previous setting or the default setting (11 rays across aperture in Y-direction).
`RAYX ri..j abs_X_value`	Absolute start coordinate X in entrance pupil for plot ray(s) `i..j`.
`RAYY ri..j abs_Y_value`	Absolute start coordinate Y in entrance pupil for plot ray(s) `i..j`.
`RAYCX ri..j cosine_x`	Direction cosine in X-direction in the entrance pupil for plot ray(s) `i..j`.
`RAYCY ri..j cosine_x`	Direction cosine in Y-direction in the entrance pupil for plot ray(s) `i..j`.
`DEL ri..j`	Deletes plot rays `i..j`.
`DEL ra`	Deletes all plot rays.

Note: Ray definitions may be overwritten, if automatic ray generation is checked in the lens layout plot (see command `EDI LDR`).

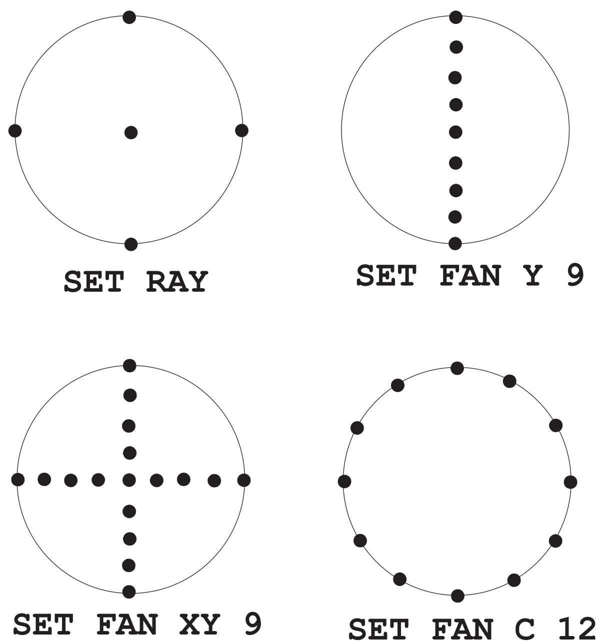


Figure 10.2: Examples of generating plot rays.

11

Zoom and Multi-Configuration

The term ”zoom” is used throughout the manual as a generally accepted synonym for ”multi-configuration” systems (bearing in mind that classical zoomed systems mainly alter the air-space between lenses while true multi-configuration systems allow the modification of *any* parameter). Thus, in ”true” multi-configuration systems, the lens can be used at different wavelengths, different tilt/scan angles, different object fields, to name a few.
The zoom features are:

- Almost any lens data parameter which can be edited may be zoomed
- all zoom data are saved as part of the lens,
- ”dezoom” lens data to any selected zoom position

A zoom or multi-configuration system is set up by the following steps:

1. Define the number of configurations
2. Define the parameter for each zoom configuration
3. define the optimization parameter for each configuration (if any)

Each step is described in detail in the following sections.

11.1 Number of Zoom Positions

The number of zoom positions in $OpTaliX$ is theoretically unlimited, however, there may be practical limitations imposed by your hardware configuration. The number of zoom positions is set by the command

`ZOO n_pos`

with `n_pos` = number of zoom positions.

11.2 Define Zoom Parameter

A ”zoomed” parameter always requires a preceding `ZOO` qualifier, if entered from the command line. For example, to make the thickness at surface 3 variable in a zoom/multiconfigurationn systems, the command would be:

ZOO THI S3 1.0 12.0 16.0

The number of parameter must match the number of zoom positions entered by the ZOO n_pos command. If the number of variables entered is less than the number of zoom positions, then the remaining variables are assumed zero (0).

Also note the command EDI ZOO which invokes a spreadsheet-like editor to define zoom/multi-configuration parameters (sect. 11.3).

The command syntax is:

ZOO n_pos	Define the number of zoom positions.
EDI ZOO	Edit zoom parameter. Invokes a text editor.
ZOO operand parameter_1 ... parameter_n	Converts a non-zoomed parameter into a zoomed parameter. "operand" can be any *OpTaliX* -command, "parameter" any value appropriate for the operand. Examples are given below this table.
ZED	Text based editor for editing zoom parameters. This option is *only* recommended if more than 50 zoom positions shall be handled. Otherwise use the "EDI ZOO" command explained above. The ZED command invokes an ASCII editor for modifying zoom position parameters in a command-like fashion.
POS zoom_pos	Sets a zoomed system to the zoom position "zoom_pos", which is then the current zoom position. All subsequent performance analysis (e.g. MTF, PSF, etc) are performed at the currently selected position. It is important to note, that the overall zoom parameter are not destroyed (as in DEZ command, see below). Example: POS 3 selects the current zoom position 3. A subsequent system listing (LIS-command) or a MTF-analysis will then be performed at zoom position 3. See also the PPOS command on page 182 for plotting only one specific zoom position.
DEZ zoom_pos	Dezoom: Freezes a zoomed system to a non-zoomed (single position) system at the position "zoom_pos". All zoom parameter are lost.
ZOO POX value(z1) ... value(zn) ZOO POY value(z1) ... value(zn) ZOO POZ value(z1) ... value(zn)	Set the plot offset for each zoom-position referred to the center of the paper plotting area. The offset values are given in mm. These commands were introduced to place the lens layout plots (lens drawings) on the paper for each zoom position individually. Example: zoo poy 80 40 0 -40 -80 ! Plots the lens layout plots for the zoom positions 1-5 vertically in Y-direction on the paper, that is position 1 is plotted 80mm above the paper center, position 2 is plotted 40mm above the paper center, and so on.

Examples:

ZOO 3	Select 3 zoom positions

continued on next page

continued from previous page

`ZOO THI s2 2 4 6`	Zoom thickness s2 is 2mm, 4mm, 6mm at position 1 to 3
`ZOO ADE s3..6 10 20 30`	Zoom X-tilt of surfaces 3-6 to values 10, 20 and 30 degree at positions 1 to 3
`DEZ 2`	"Dezooms" a system to a non-zoomed system at position 2. For the example given above, the following fixed settings are selected: `THI s2 4`, and `ADE s3..6 20`
`ZOO STO s1 s4 s6`	Zoom stop surface.
`ZOO STO 1 4 6`	as above, but without explicit surface qualifier.
`ZOO GLA s1 bk7 sf6 f2`	zoom glasses

11.3 Spreadsheet Zoom Editor

Zoomed parameter may also be conveniently entered in a spreadsheet like editor. The zoom spreadsheet editor window is capable of displaying and editing up to 50 zoom/multiconfiguration positions. If more than 50 positions are needed, enter zoom parameters in the command line or use the text base zoom editor (ZED command). The zoom editor spreadsheet is invoked by the command

```
EDI ZOO
```

Each parameter in the editor is displayed in a separate cell. For example, three fields (YAN) and three axial separations (THI) are zoomed in the examples file `$i\examples\zoom\laikin-35-1.otx`. In the command line, the zoom parameters would be entered as

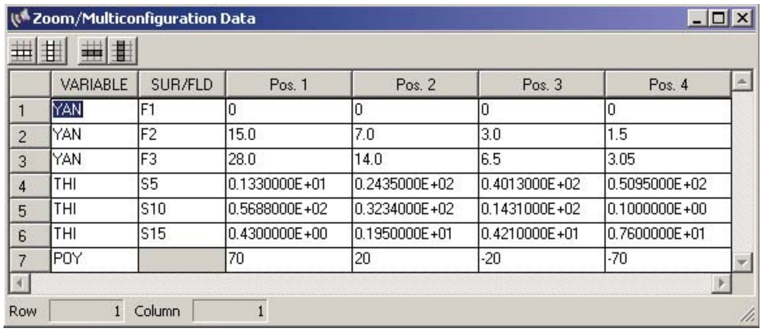
```
zoo 4
zoo yan f1 0 0 0 0
zoo yan f2 15.0 7.0 3.0 1.5
zoo yan f3 28.0 14.0 6.5 3.05
zoo thi s5  0.1330000E+01  0.2435000E+02  0.4013000E+02  0.5095000E+02
zoo thi s10  0.5688000E+02  0.3234000E+02  0.1431000E+02  0.1000000E+00
zoo thi s15  0.4300000E+00  0.1950000E+01  0.4210000E+01  0.7600000E+01
zoo poy 70 20 -20 -70
```

and in the zoom spreadsheet editor as shown in Fig. 11.1.

Notice that there is a limit on the maximum number of zoom parameter entries (rows/lines) in the spreadsheet zoom editor. Currently only 120 zoom parameter lines are accepted. This limit is only defined to limit system resources and allow *OpTaliX* to be run also on computers with limited memory.

The first column, labelled "VARIABLE", always holds the parameter to be zoomed. This can be any parameter describing the optical system such as curvatures (CUY), radii (RDY), distances (THI), tilt/decenter (XDE, ADE, ...), wavelength (WL), aperture (EPD,NA,NAO) and so on. Any parameter which can be changed in the command line will also be accepted in the zoom editor.

The second column, labelled "SUR/FLD" specifies surface number or field number or wavelength number. Since the cells in the zoom editor are a direct representation of the (string) parameters entered in the command line, a corresponding surface or field or wavelength letter symbol must preceding. Thus, like in the command line, surface 3 is specified as "`s3`" (without the quotation marks) in the corresponding cell. Field number 2 would be specified as "`f2`" and wavelength number 4 as "`w4`".



	VARIABLE	SUR/FLD	Pos. 1	Pos. 2	Pos. 3	Pos. 4
1	YAN	F1	0	0	0	0
2	YAN	F2	15.0	7.0	3.0	1.5
3	YAN	F3	28.0	14.0	6.5	3.05
4	THI	S5	0.1330000E+01	0.2435000E+02	0.4013000E+02	0.5095000E+02
5	THI	S10	0.5688000E+02	0.3234000E+02	0.1431000E+02	0.1000000E+00
6	THI	S15	0.4300000E+00	0.1950000E+01	0.4210000E+01	0.7600000E+01
7	POY		70	20	-20	-70

Figure 11.1: Zoom Editor window, showing the zoom parameters on the example of `$i\examples\zoom\laikin-35-1.otx`

All subsequent columns hold the parameter data for each zoom position.

Notes:
There are a few parameters which are not dependent on either field, surface or wavelength. These are 'PIM', 'POX', 'POY', 'POZ', 'DEF', 'EPD', 'FNO', 'NA ', 'NAO', 'MAG', 'RED', 'STO', 'WRX', 'WRY', 'ZWX', 'ZWY', 'RCX', 'RCY', 'M2 ', 'MFR'. For these cases the corresponding cell in the second column is greyed, indicating that no entry is required in this cell.

Analysis options such as MTF, PSF, etc) are always calculated at the currently selected zoom/multiconfiguration position. Thus, to do performance analyses for various zoom positions, the corresponding zoom position must be selected prior to the dedicated analysis. The zoom position is set by the command "`POS i`" where "`i`" is the zoom position. A few options such as spot diagram (SPO), rim ray fan (FAN) and lens layout (VIE) are designed to plot *all* positions in one graph.

11.4 Insert, Copy, Delete Zoom Positions

`INS zi..j`	Insert zoom positions `zi..j`. Zoom data at higher position numbers will be shifted accordingly.
`DEL zi..j`	Delete zoom positions `zi..j`. Zoom data at higher position numbers will be shifted accordingly.
`COP zk target_pos`	Copy zoom position `zk` to `target_pos`. This command **overwrites** data at the new position (`target_pos`). If required, insert a new zoom position (`INS zi..j`) prior to copying zoom position data. Only one position can be copied at a time.

Zoom positions may also be inserted or deleted from the zoom editor window by clicking on the appropriate icons in the zoom editor toolbar as shown in Fig. 11.1. An explanation of the icons is given below.

	Insert a new zoom parameter row.
	Insert a zoom position before the selected position (=column). To select a zoom position, put the cursor into any cell of the desired column (=position).
	Delete a zoom parameter row.
	Delete a zoom position (column in the surface editor).

11.5 Text based Zoom Editor

In addition to the spreadsheet zoom editor, a text based editor for zoom/multiconfiguration data is available. This option is offered because the spreadsheet zoom editor is currently limited to 120 parameter definitions (rows/lines). The reason is caused by the fact that the number of cells in a grid editor corresponds to the system resources. The larger the grid, the more system resources are required. In order to allow $OpTaliX$ to be run on computers with limited memory, this limitation has been deliberately defined.

However, the number of zoom/multiconfiguration parameters that can be edited in the *text based* zoom editor is unlimited. The text based zoom editor is invoked by the command

`ZED`	Invoke text based zoom/multiconfiguration editor. A spreadsheet zoom editor, if opened, is automatically closed.

This command opens an editor window similar to figure 11.2.

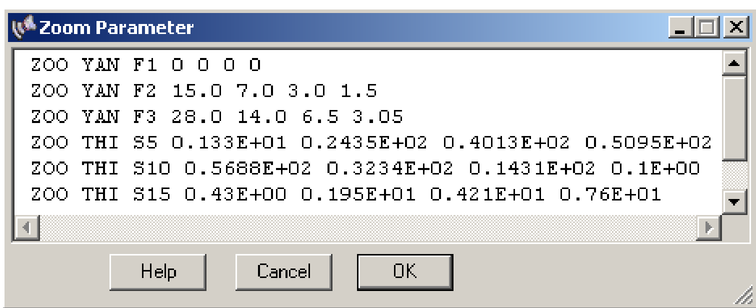


Figure 11.2: Text based Zoom Editor, showing the zoom parameters on the example of `$i\examples\zoom\laikin-35-1.otx`. See also the command "`EDI ZOO`" which displays the spreadsheet zoom editor (default).

11.6 Solves in Zoom Systems

Solves are active **only** in the first zoom position. The solved parameter is then unchanged for remaining positions.

12

Tools and Utilities

12.1 Autofocus

Finds the best focus of an optical system by adjusting the back focal distance or any other selectable axial separation. It provides an easy and quick means to put the image plane in focus. There are several function types according to which the focus is determined: minimum rms-spot size (also in X- or Y-direction), minimum wavefront error, maximum MTF or maximum coupling efficiency. The best focus location depends upon the criterion selected. Focusing can be accomplished at selected fields and wavelengths or as an average over the full field. For zoom systems, focusing is always performed at the currently selected position (see POS command).

Since only axial separations are altered, autofocus does not account for a tilted image plane. Adjusting the image plane tilt as well (for instance in non-symmetric systems) requires optimization by proper setting of surface tilts ADE, BDE, CDE as variables.

`AF function_type [ fi..j | wi..j | si | ? ]`

Searches best focus (autofocus) at selected fields and wavelengths by adjusting the axial separations (thicknesses). By default, the back focus will be adjusted. In case of "PIM yes", the defocus (DEF) is changed. In case of "PIM no", autofocus uses the axial separation of the last surface. In case a dedicated surface is specified (eq. sk), the axisl distance (thickness) at this surface is used to adjust the best focus.
`function_type` is one of the 3-character strings:

SPD	spot diameter, rms
SPX	spot diameter, rms, in X-direction only
SPY	spot diameter, rms, in Y-direction only
WAV	wavefront error, rms
MTF	modulation transfer function (MTF). The spatial frequency, at which MTF-autofocus is performed, is set by AFR (see page 263), or below
CEF	Coupling efficiency.

`AFR autofocus_frequency`	Spatial frequency used in AF command (see above). It is given in Lp/mm for focal systems, in Lp/mrad for afocal systems

Examples :

`af`	Autofocus without any parameter adjusts the back focus (default) for all wavelengths and fields at the currently selected zoom position.
`af ?`	invokes a dialog box to select from various autofocus options.
`af spd f1..3 w3`	determines the best focus for minimum rms-spot diameter at fields 1-3 and wavelength number 3.
`af mtf s4 f1`	searches best focus on the basis of maximum MTF at field point 1 and uses thickness 4 as variable.
`afr freq`	Sets spatial frequency for autofocus optimization to `freq`, in Lp/mm for focal systems, respectively Lp/mrad for afocal systems. This setting does not affect analysis frequencies, such as `MFR`, `MFRD`, `MFRF`.

12.2 Scaling

Scales the optical system (or part of it) by a defined factor. The command syntax is

`SCA si..j scale_factor`	Scale range of surfaces `si..j` by `scale_factor`.
`SCA sa scale_factor`	Scale entire system (`sa` = all surfaces) by `scale_factor`.
`SCA [EFL|OID|SYL|EPD|OAL] target_value`	Scale entire system by specifying a target value for either EFL, OID, SYL, EPD or OAL. Example: `sca efl 100` ! Scales entire system such that a focal length (EFL) of 100mm is obtained.

12.3 Invert System

Inverts the optical system (or part of it). Parameters, which describe the usage of the system (aperture, field, etc.), however, are not altered.

`INV si..j`	Invert (reverse) a range of surfaces `si..j`.

12.4 Convert fictitious Glasses to real Catalogue Glasses

Converts a fictitious glass to a catalogue glass (a ”regular” glass). Fictitious glasses are characterized either by a 6-digit MIL-number as described on page 220 or by DNO or DVO offsets (see page 228). The conversion searches for a nearest glass in the glass catalogues, based on n_d and ν_d. Partial dispersions are not taken into account.

There exist special glasses (like gradient index glasses, ”infrared” glasses) for which no valid MIL representation exist. In this case the program will not return meaningful results.

`REG [si..j | cat_code1 ... cat_code10 | ? ]`	Convert a fictitious glass to a regular catalogue glass by searching the nearest glass in the $n - \nu$ domain (glass map). The `cat_code` is a three character short code identifying the manufacturer. The allowed short codes are found in table 13.2 (page 217). Up to 10 catalogue codes may be specified simultaneously. Examples: `REG sa SCH` : replace all fictitious glasses by nearest Schott glasses. `REG s2..5 HOY HIK` : replace fictitious glasses on surfaces 2 to 5 by nearest catalogue glasses from Hoya or Hikari.

The catalogues to be searched for a nearest glass may also be conveniently selected in a dialog, accessible from the main menu ”*Tools*” −− > ”*Fictitious glass to catalogue glass*” as shown in Fig. 12.1. Select all glass catalogues that apply.

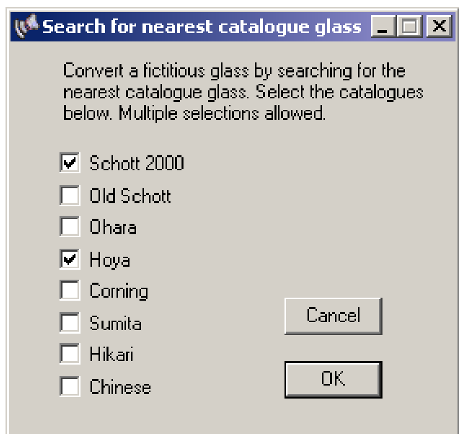


Figure 12.1: Select glass catalogues for converting fictitious glasses to regular catalogue glasses.

12.5 Find Alternative Glasses

Find alternative (replacement) glasses from a different vendor with respect to a base glass. Alternative glasses are glasses having similar properties on refractive index and dispersion compared to the base glass and therefore may be used to replace the base glass in an optical system. The choice of alternative glasses is based on the given index difference (Δn_d) and the dispersion difference ($\Delta \nu_d$) at the d-line. Hence, alternative glass finding is valid only for the visible (400-700nm) spectral range.

`ALG base_glass [delta_n delta_V]`	Find alternative (replacement) glasses with respect to a `base_glass`. By default, the tolerances on selecting an alternative glass are $\Delta n_d = 0.001$ on refractive index and $\Delta \nu_d = 0.8\%$ on dispersion, however, they may be overwritten by specifying `delta_n` and `delta_V`.

Notice that the choice of alternative glasses is solely based on the Δn_d and $\Delta \nu_d$ differences. It is the designers responsibility to take other glass properties into account, such as partial dispersion, TCE, dn/dT, etc, depending on a particular application. This list is only intended to support you in selecting glasses from alternate vendors.

12.6 Weight and Volume

This option calculates the weights, volumes and center of gravity of lenses in the optical system. Only the glass weight of the system is included, mechanical spacers and housing are ignored. The volume of spherical lenses with circular base aperture is calculated analytically. Aspheric surfaces and lenses with rectangular or elliptical base aperture are integrated numerically. The weight is computed from the specific gravity of the material as stored in the glass catalogues.

The diameter of the lens is taken from the maximum surface aperture (see sect. 8.32), independent of whether they are checked (fixed aperture) or not. The edge of the surface with the smaller aperture is squared up to the larger aperture.

If edges are specified (see EDG option in section 8.32), they define the enclosed volume. Use of `EDG` apertures allow the definition of 'edge allowances', or to match values assigned from the housing design.

The weight of front surface mirrors can be calculated provided thickness and specific gravity of the mirror are supplied using the `THM` and `SPG` commands (see table below). The back surface of front surface mirrors is always assumed plano.

`WEI [sk|si..j]`	Compute weight and volume of lenses. Includes aperture obscurations and holes. Tilted surfaces are not supported. For mirror surfaces, check also the commands `THM` and `SPG` for setting mirror thickness and specific gravity of mirror material.
`SPG [sk|si..j] gravity`	Specific gravity in g/cm^3. This command overwrites any pre-defined value stored in the glass catalogues. Enter `SPG sk|si..j 0` to delete any user-defined specific gravity data.
`THM [sk|si..j] mirror_thickness`	Center thickness of mirror. This command has no influence on the construction parameter, it is only required for weight calculation and for ISO element drawing of mirrors.

Example 1:

The following example is a standard double Gauss lens, taken from the examples library `\optix\examples\misc\double_gauss.otx` as shown in Fig. 12.2. It also indicates how edges are assumed in the `WEI` option.

The output table contains surface and element number, volume, specific gravity, weight and center of gravity. The centers of gravity given for the individual lenses refer to the vertex of the front surface, whereas the center of gravity for the entire system is referred to the first surface of the system.

```
WEIGHT CALCULATION:

         Element        Volume     Gravity      Weight   ---- Center of Gravity ----
```

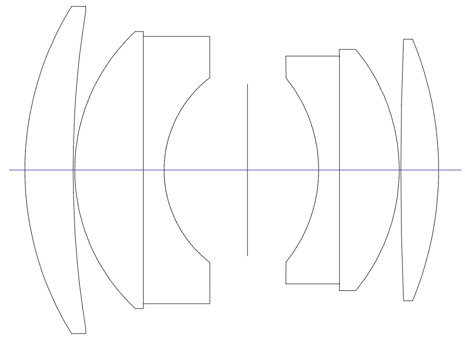


Figure 12.2: Double-Gauss example, showing edges used for weight calculation.

Surf.	Number	(cm^3)	(g/cm^3)	(g)	X	Y	Z
1-2	1	2.05929	6.300	12.974	0.000	0.000	3.280
3-4	2	1.84480	3.560	6.567	0.000	0.000	4.052
4-5	3	2.34401	3.480	8.157	0.000	0.000	2.676
7-8	4	1.31742	3.480	4.585	0.000	0.000	-0.163
8-9	5	1.35685	3.580	4.858	0.000	0.000	1.947
10-11	6	0.96652	3.580	3.460	0.000	0.000	1.299
	Total :	9.88889		40.600	0.000	0.000	16.629

```
Notes: Center of gravity of lenses are referred to the front surface of each element.
       Center of gravity of total system is referred to first surface.
```

We will now make all surfaces aspheric (use command `sut sa a`), which forces 2-D numerical integration. Volume and weights of the elements are slightly different due to the numerical integration.

WEIGHT CALCULATION:

	Element	Volume	Gravity	Weight	---- Center of Gravity ----		
Surf.	Number	(cm^3)	(g/cm^3)	(g)	X	Y	Z
1-2	1	2.05960	6.300	12.975	0.000	0.000	3.280
3-4	2	1.84512	3.560	6.569	0.000	0.000	4.052
4-5	3	2.34416	3.480	8.158	0.000	0.000	2.676
7-8	4	1.31743	3.480	4.585	0.000	0.000	-0.163
8-9	5	1.35702	3.580	4.858	0.000	0.000	1.947
10-11	6	0.96664	3.580	3.461	0.000	0.000	1.299
	Total :	9.88996		40.605	0.000	0.000	16.628

```
Notes: Center of gravity of lenses are referred to the front surface of each element.
       Center of gravity of total system is referred to first surface.
```

Example 2:
This example shows how to calculate the weight for systems containing (front-surface) mirrors. In order to obtain reasonable weight figures, a center thickness and a specific gravity of the mirror material must be assigned to mirror surfaces. This is accomplished by the commands THM and SPG

We restore (load) the Cassegrain telescope from the examples library `\optix\examples\mirror\cassegrain` and assign the following thicknesses to primary and secondary mirror:

```
thm s1 10.0
thm s2 5.0
```

Note that mirror thicknesses are always given as positive values. Next, specific gravities ρ must be specified for the mirrors. For example,

```
spg s1 3.1
spg s2 2.5
```

which specifies ρ in g/cm^3 units. Now that all relevant data are entered, the `WEI` command outputs weight and center of gravity.

```
WEIGHT CALCULATION:

         Element         Volume     Gravity      Weight    ---- Center of Gravity ----
Surf.    Number           (cm^3)     (g/cm^3)      (g)           X          Y          Z

1-2         1        349.23284      3.100     1082.622     0.000      0.000      3.812
                (     379.59602 circular, transmit)
                (     -30.36318 circular, obstruct)
2-3         2         13.46352      2.500       33.659     0.000      0.000     -2.731

        Total :      362.69636                1116.281     0.000      0.000    -12.279

Notes: Center of gravity of lenses are referred to the front surface of each element.
       Center of gravity of total system is referred to first surface.
```

Since a central obstruction has been assigned to the primary mirror (surface 1), weight calculation also reports the weight of the solid (unobstructed) mirror and the fictitious weight corresponding to the central obstruction, which is subtracted from the weight of the solid mirror.

12.7 Maximum Incidence Angles

This option traces ray bundles through the optical system for a given range of fields and zoom positions. The output reports the maximum ray incidence/refraction angles and the mean (average) incidence/refraction angles for each surface. Because the analysis is based on a full aperture ray trace, accuracy of the results may be increased by increasing NRD (number of rays across diameter).

Knowing the range of ray incidence angles is often helpful for designing multilayer coatings appropriately matched to the optical use of surfaces.

`MAXAOI [fk|fi..j zk|zi..j]`	Calculates the maximum angle of incidence on all optical surfaces for a given range of field numbers `fi..j` and zoom positions `zi..j`. A description of the output is given below. Related commands: AOI, AOR, NRD.

Description of output:

```
RAY INCIDENCE ANGLES:

Analysis is based on a full-aperture ray trace with  32  x  32  rays
in the entrance pupil for each field and zoom position.

Average values are given with consideration of uniform and apodized
intensity in the entrance pupil.

All incidence angles are given in degrees.

Zoom Positions :  1  -  1
```

```
Fields         :  1  -  1

        Average()      Average()      Maximum()    Surface comment
Sur      (uniform)      (apodized)
  1         0.00000        0.00000        0.00000
  2         2.87489        1.64474        4.37264    Lens 1
  3         9.62708        5.50348       14.70824
  4         1.77709        1.01125        2.78751    Lens 2
  5         8.90122        5.09164       13.55067
  6         6.74561        3.85643       10.30278
```

The average incidence angles are calculated in two variants. The column labeled 'uniform' assumes that all rays within the pupil have identical intensity (i.e. uniform intensity distribution), whereas the values in the column labeled 'apodized' take pupil apodization into account. The latter is often specified in systems using laser beams with a Gaussian intensity profile across the aperture.

12.8 Optimal Coating Indices for Gradient Index Surfaces

This option determines the optimal index of refraction to use when AR coating a gradient index lens (front and back surfaces). Particularly for steeper curvatures the refractive index may vary considerably (as this is the intention in the design process), however, some unique index must be determined for the coating substrate. A commonly accepted estimate is the index at 70% of the clear aperture. Another, probably better, approach is the area-weighted index value, which is calculated by

$$n = \frac{\sum_{i=1}^{k} n_i (r_i^2 - r_{i-1}^2)}{r_{max}^2} \tag{12.1}$$

Both cases are calculated and the indices at the surface vertex and the clear aperture are given in addition. The command syntax is

`CIND sk [ape1 ape2]`	Output refractive indices to be used for coating a gradient index surface `sk` within clear apertures `ape1` and `ape2` of front and rear surface respectively. If `ape1` and `ape2` are omitted, the currently set apertures are used. Example: `cind s2 10 9` ! Calculate optimal refractive indices at 10mm clear aperture (front surface) and 9mm clear aperture (rear surface).

A typical output in the text window would be

```
Refractive index values for AR-coating of gradient index lenses:

Wavelength :      0.58760

                Area            70%                          full        Clear
 Surf        weighted       aperture        on-axis      aperture     aperture
    1       1.7062033      1.7071323      1.7173626     1.6962368       12.500
    2       1.6816796      1.6816665      1.6815213     1.6818191       11.255
```

12.9 Surface Sag

Surface sag computes the sag at any point on any surface in the optical system. The command syntax is:

SAG sk x_height y_height [?]	Surface sag (z-component) at surface sk and surface coordinates x_height, y_height, measured from the surface vertex without regard to tilt and decentration.

12.10 User Defined Graphics (UGR)

In addition to graphics predefined by the program, graphics defined by the user can be created. These are two-dimensional plots of any variable parameter against any performance measure known to $OpTaliX$. Parameters and functions may be composed from any command, arithmetic expression, function or macro as it would be entered in the command line. For example, changing the lateral displacement of a fiber in a fiber coupling optics is accomplished by the command

```
FRY .001
```

which offsets the receiving fiber 1 μm from the nominal chief ray intercept in the image plane. In an user defined graphics (UGR), this misalignment may become a variable parameter by simply writing 'fry'. The function depending on this parameter, can also be any part of a command sequence, for instance 'SPD f3', which is the rms spot diameter at field number 3.

Let us assume, we want a plot of the coupling efficiency vs. the fiber misalignment. The commands required to achieve this are:

```
UGR X 'fry' LIM -0.005 0.005 0.001
UGR Y 'cef' LIM 0 1.0
```

The first line defines the variable parameter 'fry' to be plotted at the X-axis, the second line defines the dependent function 'cef', which is plotted at the Y-axis. The values following the token LIM define the lower and upper plot limits for X- and Y-axis and the variable step respectively. That is essentially all what is needed to define a user defined graphics (UGR). We may also want to add axis labels and a title to the plot:

```
UGR TIT 'Coupling efficiency vs. fiber misalignment'
UGR XLAB 'fiber decenter'
UGR YLAB 'CEF'
```

The plot is created with the command

```
UGR go
```

Here is a summary of all commands related to UGR:

`UGR X var_string [LIM xlow xhigh xstep]`	Define a variable used in UGR. `var_string` is a string (enclosed in apostrophes) containing the variable definition. `LIM` is optional. If given the plot limits are explicitly specified. Omitting `LIM` scales the X-axis automatically. For example, `ugr x 'thi s4'` ! Thickness at surface 4 is variable in UGR. `ugr x '$myvar'` ! Creates a user-defined variable to be varied in UGR. `ugr x 'yde s3 lim 0 30 .5'` ! UGR-variable definition with explicit limits.
`UGR Y func_string [LIM ylow yhigh]`	Define a function used in UGR. `var_string` is a string (enclosed in apostrophes) containing the variable definition. `LIM` is optional. If given the plot limits are explicitly specified. Omitting `LIM` scales the Y-axis automatically. For example, `ugr y 'efl'` ! Calculate EFL and plot it as function value. `ugr y 'efl lim 100 200'` ! EFL is plotted within fixed limits (100 - 200 mm).
`UGR TIT title_string`	Title string displayed in user-defined graphics. `title_string` should be enclosed in apostrophes if the string contains blank characters, otherwise apostrophes can be omitted. For example, `ugr tit 'My Title'` ! Plots 'My Title' (without apostrophes) as title. `ugr tit stuff` ! Plots 'stuff' (without apostrophes) as title.
`UGR XLAB x_label_string`	X-label displayed in user-defined graphics. `x_label_string` should be enclosed in apostrophes if the string contains blank characters, otherwise apostrophes can be omitted. For example, `ugr xlab 'X variable'` ! Plots 'X variable' (without apostrophes) as X-label. `ugr xlab x-value` ! Plots 'x-value' (without apostrophes) as X-label.
`UGR YLAB y_label_string`	Y-label displayed in user-defined graphics. `y_label_string` should be enclosed in apostrophes if the string contains blank characters, otherwise apostrophes can be omitted. For example, `ugr ylab 'spd f1'` ! Plots 'Y variable' (without apostrophes) as Y-label. `ugr xlab spd` ! Plots 'y-value' (without apostrophes) as Y-label.
`UGR LOG floor`	Select logarithmic display.

A more user-friendly way is from the menu *TOOLS –> User Defined Graphics*, which invokes a dialog box to enter all required parameters. Our example discussed above as well as the resulting plot would look like (Fig. 12.3 and Fig. 12.4),

The string fields for the variable parameter and the function can be edited and expanded within the

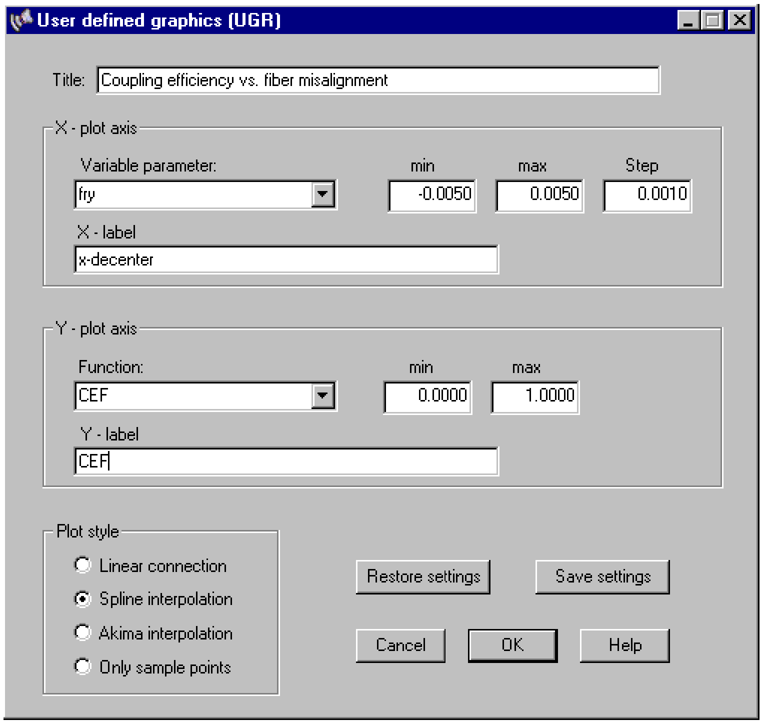


Figure 12.3: Dialog box to create an user defined graphics.

syntax rules given for each command. There are a limited number of predefined variables and functions, which may be accessed by clicking on the associated down arrows. A concise descriptive text is given to each variable/function string, separated by an exclamation mark "!". Text after the exclamation mark is considered as a comment and will thus be ignored. It is not part of the variable/function definition.

UGR definitions may be saved or restored (loaded) to/from a macro file with extension `*.ugr`.

Due to the numerous number of plots which can be created with user defined graphics, there are no intelligent defaults for the independent variables or the dependent functions. In case of uncertainty, it is advisable to test the commands and the resulting function values in the command line prior to using them in the UGR option.

Also note, that some variables only work if the corresponding system parameter are properly defined. For example, a variable decenter (XDE or YDE) requires that the surface can be decentered (add "D" to surface type if needed).

12.10.1 Variable Parameters in User-defined Graphics

Variable parameters in user-defined graphics (UGR) can be specified as follows:

- Any construction parameter that can be entered/edited on the command line can be made variable in UGR. For example, THI s4 (thickness at surface 4). Enter the parameter plainly, without quotes or apostrophes.

Specify any construction parameter as variable in UGR, just as you would enter it in the command line or in a macro.

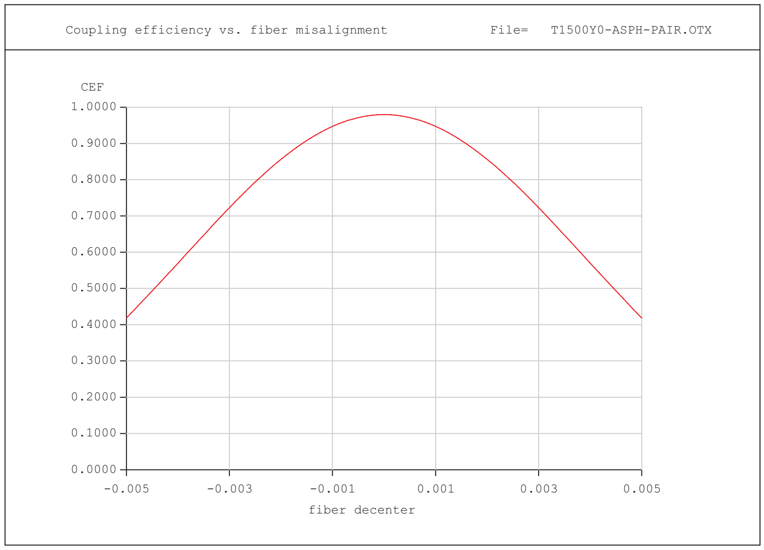
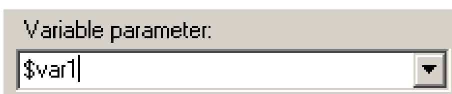


Figure 12.4: Example output of user defined graphics: CEF vs. fiber misalignment.

- Specify any valid user-defined variable. Note for brevity: User-defined variables *must* begin with a "$" character followed by at least one alpha-numerical character.

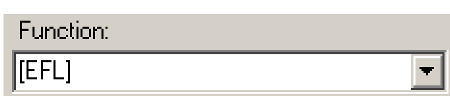


Enter a user-defined variable directly. The variable need not exist before, it is created during UGR execution.

12.10.2 Functions and Macros in User-defined Graphics

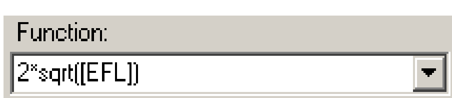
In user-defined graphics (UGR), the function values to be plotted on the Y-axis of a graph can be defined by various methods:

- A lens database item (LDI) provides the easiest access to a lens construction parameter. See for example Fig. 12.3 which asks for coupling efficiency (CEF). Enter the name of this parameter enclosed in square brackets in the function field. For example,



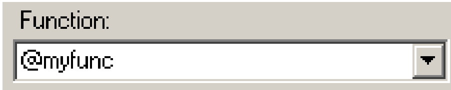
Specify a lens database item (LDI) directly. In this example, the function value is the "equivalent focal length" (EFL).

- Specify an arithmetic expression which may include variables and lens database items (LDI).



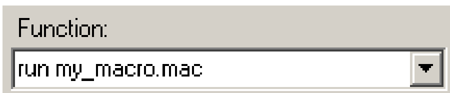
Define an arithmetic expression, including a LDI.

- Specify a function which must have been previously defined in a separate command or amacro. For example, if we have defined the function "`myfunc == $x^2`" (without the quotes), the square of variable `$x` would be returned.



Use a function previously defined for calculating the function value.

- Specify a macro which returns a value. In macros, (function-) values can be passed to the calling module using the RETURN statement (see page 456).



Run a macro which evaluates and returns the function value. See also RETURN (page 456), and RUN (page 440). The macro file is assumed in the macro directory as defined in the preferences settings (page 21) which is typically `c:\programs\optalix\macro`. For any different location you must explicitly specify the path.

12.10.3 UGR Command Example

In addition to the menu-based entry of user-defined parameters, as described in the previous sections, this section gives a concise overview on defining user-defined graphics from the command line respectively from macros.

`ugr X 'thi s2' LIM 0.5 1.0 0.05`

Define the independent parameter (variable) range for UGR-plot. The variable parameter in this case is 'THI s2', thickness at surface 2. The variable parameter (thi s2) is varied within the limits 0.5 to 0.1 at steps of 0.05.

`ugr Y 'spd f1 w1' LIM 0 0.1`

Specify the dependent parameter (i.e. function value). In this case the spot diameter at field 1, wavelength 1, (spd f1 w1) shall be calculated. The plot limits (i.e. along the Y-axis) are between 0.0 and 0.1. Note that these limits may change according to the parameter and functions defined.

`ugr tit 'My UGR Graphics'`

12.11 Analytical Setup

A few optical systems may be created from scratch by entering a few basic system parameters like focal length, aperture, field of view, etc. They are then automatically generated on the basis of third-order theory. This means, that the aberrations of the resulting systems are corrected to third order, neglecting any higher order aberrations. However, these systems provide a good starting point for further refinement or as building blocks to construct more complex systems.

12.11.1 Lens of best Form

Constructs a lens of best form, for which the third-order spherical aberration reaches a minimum for a given object distance s and power φ. Without reiterating third-order theory, we first define auxiliary variables

$$A = \frac{2n+1}{n-1}, \qquad B = \frac{n+1}{n}, \qquad C = \frac{n+2}{n} \tag{12.2}$$

The curvatures of the lens are then obtained by

$$c_1 = \frac{A\varphi + 4B \cdot \frac{1}{s}}{2C}\varphi \tag{12.3}$$

$$c_2 = \left(c_1 - \frac{1}{n-1}\right)\varphi \tag{12.4}$$

Command Syntax:

SETUP SLE	Single lens setup. The lens bending is chosen to minimizing third-order spherical aberration. This command invokes a dialog box.

12.11.2 Achromatic Doublet

Constructs a thin-lens achromatic doublet from selected materials and a given focal length. The algorithm is found in Laikin [29].

Command Syntax:

SETUP ACR	Thin-lens achromatic doublet setup. This command invokes a dialog box.

12.11.3 Lurie-Houghton Telescope

Constructs a catadioptric telescope of Luri-Houghton form. The ”Lurie-Houghton” telescope combines design elements from Lurie’s original proposal [31] (two-lens full-aperture corrector) with elements of the Houghton telescope [21] (spherical corrector). Both modifications greatly simplify manufacturing, however, at the expense of astigmatism. A distinct advantage of this design form is the improved correction of coma compared to other catadioptric telescopes (Schmidt-Newton, Wright). A design example of the Lurie-Houghton design form can be found in the `/examples/catadiop` directory.

Analytical setup of the Lurie-Houghton design form is accomplished by a few simple equations. From the auxiliary variables

$$A = \frac{n+2}{n(n-1)^2}, \qquad B = \frac{2(2n+1)}{(n-1)^2}, \qquad C = \frac{2(n+1)}{n(n-1)} \tag{12.5}$$

$$D = d \cdot \varphi, \qquad L = \frac{(D-2)(2A-B)}{C}, \qquad Q = \frac{(2-D)L^2}{2C} \tag{12.6}$$

we obtain the radii of the corrector

$$r_1 = -r_3 = \frac{2L(n-1)}{(Q+1)\varphi} \tag{12.7}$$

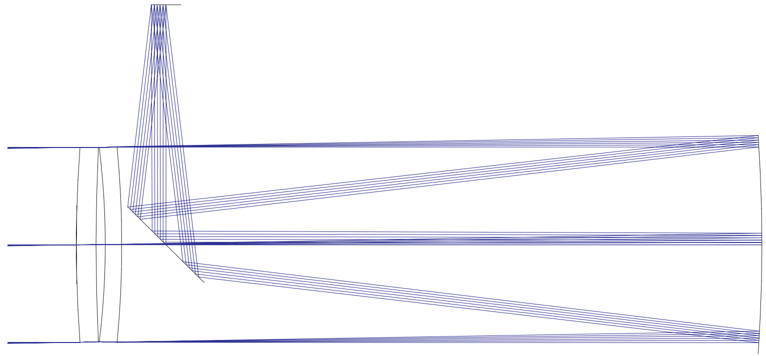


Figure 12.5: Lurie-Houghton design form.

$$r_2 = -r_4 = \frac{2L(n-1)}{(Q-1)\varphi} \tag{12.8}$$

with

φ optical power of the primary mirror = $2/r_m$

d distance of last corrector surface to primary mirror

Command Syntax:

SETUP LURIE	Setup of a Lurie-Houghton Telescope. A dialog box is invoked.

12.11.4 Reflecting Telescopes

This section describes the theory for the setup of basic reflective telescopes (e.g. Parabola, Cassegrain, Gregory, Ritchey-Chretien, etc.).

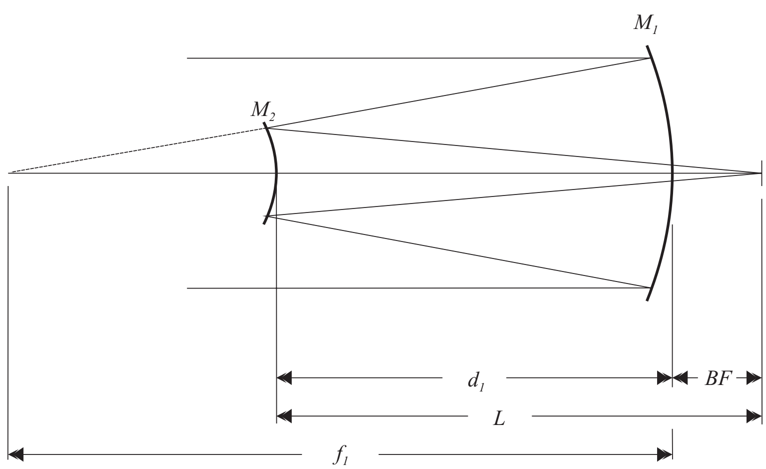


Figure 12.6: Paraxial quantities at a compound telescope

Command Syntax:

SETUP TEL	Setup of compound reflecting telescopes such as Cassegrain, Richey-Chretien, Gregory or Parabola. A dialog box is invoked, which allows selection of the various design forms.

The equations and formulae presented hereafter are deduced from R.N.Wilson [61]. The variables as shown in Fig. 12.6 are defined as

d_1	Separation of primary mirror and secondary mirror
L	Distance of focus from secondary mirror
BF	Back focus (distance of focus from primary mirror)
f_1	Primary mirror focal length
f_2	Secondary mirror focal length
m_2	Secondary mirror magnification

Note, that the sign convention is in accordance with the definitions given in chapter2.

12.11.4.1 Classical Cassegrain and Gregory Form

These forms are defined by a primary mirror of parabolic form ($K_1 = -1$). The position of the secondary mirror is defined by:

$$d_1 = \frac{m_2 f_1 + BF}{1 - m_2} \tag{12.9}$$

The power Φ_2 of the secondary mirror is:

$$\Phi_2 = \frac{1}{f_2} = \frac{1}{BF - d_1} - \frac{1}{f_1 - d_1} \tag{12.10}$$

The conic constant of the secondary mirror is then a function of the secondary mirror magnification m_2:

$$K_2 = -\left(\frac{m_2 - 1}{m_2 + 1}\right)^2 \tag{12.11}$$

12.11.4.2 The Aplanatic Telescope and its Ritchey-Chretien Form

The Ritchey-Chretien (RC) form is an important modification of the Cassegrain telescope. The RC-solution solves for the field coma of a 2-mirror telescope, which is zero for an aplanatic condition. The solution of the aspheric conic constants is achieved by:

$$K_1 = -1 + \frac{2L}{d_1 m_2^3} \tag{12.12}$$

$$K_2 = -\left[\left(\frac{m_2 - 1}{m_2 + 1}\right)^2 + \frac{2f'}{d_1\left(m_2 + 1\right)^3}\right] \tag{12.13}$$

The power of the secondary mirror M_2 is obtained from Eq. 12.10.

12.12 Slider Control

Sliders are used to interactively change any system or surface parameter. The result on system layout or performance can be immediately viewed in any analysis window. That is, the effect of changing values in the prescription of an optical system is immediately displayed in open analysis windows.

Sliders are invoked by the command SLID or from the main menu *Tools* - *Sliders*. A a dialog showing up to five slider controls is displayed (see Fig. 12.7).

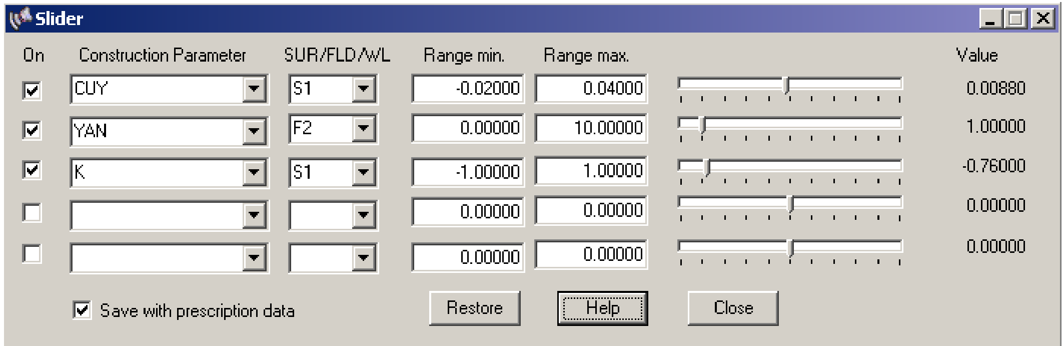


Figure 12.7: Slider Dialog. Allows definition of arbitrary construction parameters to be adjusted interactively while immediately viewing the analysis result in open windows.

Description of slider controls:

On Turns on/off a specific control.

Construction Parameter This is any construction or system parameter which can be entered in the command line. The pull-down menu offers a selection of pre-defined (mostly used) parameters, however, individual parameters can be entered in the first menu item (initially blank).

SUR/FLD/WL This field expects a surface, field or wavelength qualifier such as S3, F2, or W4. The allowable range of surface/field/wavelength qualifiers in the current optical system may be selected from the pull down menu.

Range min. The minimum allowable value of a construction or system parameter.

Range max. The maximum allowable value of a construction or system parameter.

Notes:

- Changes made to slider controls are immediately reflected in the surface editor. However, changes made in the surface editor directly (for example inserting or deleting surfaces) will not be updated in the slider dialog. If the optical system is changed, you are requested to close and reopen the slider dialog to update for the new parameters.
- Analysis windows that require long computing times (such as MTF, PSF, etc) may slow down window update significantly. If necessary, close computing intensive analysis windows.
- A copy is made of the data to be modified prior to displaying the slider dialog. The "Restore" button then restores the state of the optical system before the slider dialog was invoked.
- Slider settings can be saved with the current system by checking the "*save with prescription data*" check box found in the lower left corner of the dialog. This also implies that slider settings are specific to the current system.
- On closing the slider control dialog, the current slider settings are used for all subsequent analyses. Click on the "Restore" button before leaving the slider dialog if you want to return to the previous system (i.e. before the slider dialog has been invoked).

12.13 ECHO Command Line

`ECHO Y|N`	Echoes commands (entered in the command line) in the text output window. Enabled by ”`Y`” and disabled by ”`N`”. The default setting is ”`ECHO N`”. The ECHO command is only active for a particular session of $OpTaliX$. `ECHO` does not apply to commands executed within a macro. If you want to disable *all* text output, use the ”OUT SILENT” option (page 434).

12.14 CLS (Clear Screen)

`CLS`	Clears the contents of the text window (”clear screen”). For Code V compatibility, the CLS command can also be used for defining plot colours. See sections 7.2 (page 45) for defining field colours, 20.1 (page 372) for defining coating colours.

12.15 Time

`TIM`	Outputs an character string with the current time in 24 hour format HH:MM:SS.

12.16 Date

`DAT`	Outputs an character string with the current date in the format DD MMMM YYYY.

12.17 File Name

`FILENAME`	Outputs an character string containing the file name (without path).

12.18 File Path

`FILEPATH`	Outputs an character string containing the file path.

12.19 Operating System Command

SYS ['cmd_string' | ?]	Opens a command window (DOS-box) to execute operating system (OS) commands. Control is then transferred to the operating system and *OpTaliX* waits until the OS command window is closed (terminated). Under Windows 95/98/Me operating systems `command.com` is invoked. Windows NT/2000/XP operating systems call `cmd.exe` by default. The optional parameter `cmd_string` is the operating system command. It must be enclosed in apostrophes. The question mark "?" keeps the OS command window open, while omission of the question mark executes `cmd_string` in silent mode, except where SYS is given without any parameters.

Examples:

`SYS` invokes an OS command window. The window remains open. Type '`exit`' (without the apostrophes) to close the OS command window and give back control to OpTaliX.

`SYS 'dir *.*' ?` invokes an OS command window, executes the system command 'dir *.*' and waits for additional OS commands. Type '`exit`' (without the apostrophes) to close the OS command window and give back control to OpTaliX.

`SYS 'copy a.txt b.txt'` executes the OS command and gives back control to *OpTaliX* immediately.

Note that operating system commands may also be used in macros where the form without the question mark "?" is preferable to ensure uninterrupted execution.

12.20 Logging Ray Data

It is sometimes desirable to have access to ray data, in particular if a large number of rays is concerned (such as in spot diagrams or in illumination calculations). Ray data can then be logged (written) to a file for later reuse.

RAYLOG sk|off FIL log_file	Enables logging (i.e. writing) ray data at a specific surface `sk` to a file `log_file`. Specification of surface `sk` at which ray data are to be logged is mandatory. If omitted, the command is ignored. The "`off`" option or `s0` disables ray logging. Ray data are written to plain ASCII files without header. See sect. 32.13 for a description of the ray file format. Examples: `raylog s4 fil rays.txt !` logs all rays calculated in subsequent commands. `raylog off` ! disables ray logging. `raylog s0` ! same as above, disables ray logging

Use this command with great care! There are many analysis options (such as PSF, MTF, spot and illumination calculations) which generate a massive amount of ray data and therefore log-files may become huge. Also do not forget to disable ray logging by the ”RAYLOG off” command after you have acquired ray data. Otherwise rays may be inadvertently written to the file, thus using excess hard disc space and slowing down calculations due to hard disc writing.

The RAYLOG command is favorably used in a macro environment. For example, consider the following situation where ray data resulting from an illumination calculation at the image surface (the target surface) are stored in a file:

```
raylog si fil my_rays.txt    ! turn on ray logging
ill ?                        ! invokes illumination dialog for editing
                               illumination parameters
raylog off                   ! turn off ray logging
```

With the example above, the ray data are then found in the file my_rays.txt. See also sect. 32.13 (page 507) for a description of the ray file format.

13

Materials, Glasses

A large number of optical materials is available in $OpTaliX$. The optical and physical constants of refractive materials are stored in several catalogue files. The currently available catalogues are:

Identifier	**Manufacturer**
SCH	Schott, 2000 catalogue
SCO	Schott, old catalogue
OHA	Ohara
COR	Corning
SUM	Sumita
HIK	Hikari
HOY	Hoya
CAR	Cargille liquids
CHI	Chinese glasses
LPT	LightPath, axial gradients
SEL	NSG, Selfoc^{TM} radial gradients
GLC	Gradient Lens Corp.
GRT	Grintech, Jena
ARC	Archer OpTx
RPO	Rochester Precision Optics
SPE	Special materials (infrared, UV, plastic materials, liquids)

The optical materials can be homogeneous or inhomogeneous in their refractive index. Standard materials from different suppliers are available in the spectral range from 200nm to 30μm. Besides the refractive index information, a large number of additional optical and physical properties are provided:

- Partial dispersion
- Linear expansion coefficient
- Transformation and melting temperature
- Thermal conductivity
- Specific weight
- Hardness

- E-Module
- Chemical properties
- Temperature coefficient of refractive index
- Internal transmission

Most of these data can be viewed and partly edited in the glass manager (see section24, page 419).

Command Summary:

`GLA [si..j] [zi..j|zk] [man:]glass_name`	Glass name of manufacturer (e.g. BK7). `man` is optional and designates the manufacturer. The glass vendor may also be specified by preceding the glass name with the manufacturers short code followed by a colon, such as `SCH:BK7`. The length of the glass name, including the manufacturers short code is limited to 64 characters. See also section 13.3 for a list of manufacturers short codes.
`GL1 [si..j] gl1_name`	Glass in front of surface (`gl1_name` is identical to `GLA` in classical (i.e. sequential) systems.
`GL2 [si..j] gl2_name`	Glass at rear of surface (required for non-sequential surfaces only)
`AIR [si..j]`	Medium is air
`REFL [si..j]`	Medium is reflecting (mirror)
`REFR [si..j]`	Medium is refracting (lens)
`RMD [si..j] REFR|REFL|TIR`	Refractive/reflective mode. Available modes are `REFR` = refract all rays at surface(s) `si..j` = default mode. `REFL` = reflect all rays at surface(s) `si..j` `TIR` = only reflect rays that fulfil TIR condition This command complements the `REFR`, `REFL` and `TIR` commands.
`IND [si..j | wi..j] val_1 val_2 .... val_n`	Refractive index (ordinary) corresponding to defined wavelengths. See also wavelength definition on page 46. Only takes effect for private glasses (see section 13.4). Examples: `ind s3 1.541 1.540 1.490` ! defines indices for the first three wavelengths `ind s3 w2 1.540` ! defines index at wavelength number 2.
`INE [si..j] val_1 val_2 .... val_n`	Refractive index (extraordinary) for defined wavelengths
`DVO [si..j] delta_nue`	Dispersion shift $\Delta\nu$ (in absolute ν-values). Example: `DVO s3..5 4.2`. See also section 13.1.6 for definition of the primary dispersion.
`DNO [si..j] delta_n`	Index shift Δn at reference wavelength. Note: Reference wavelength is defined by `REF` command.
`PGO [si..j] delta_P(g,F)`	Offset of partial dispersion $P_{g,F}$ from catalogue value (see section 13.1.7 for definition of $P_{g,F}$).
`PCO [si..j] delta_P(C,s)`	Offset of partial dispersion $P_{C,s}$ from catalogue value (see section 13.1.7 for definition of $P_{C,s}$).

13.1 Dispersion

Dispersion describes the variation of the index of refraction as a function of wavelength. It is one of the most important factors in selecting optical materials. The "old Schott" formula and the Sellmeier formula are consistently used. The coefficients are stored in glass catalogue files, which requires only specification of the glass name. The correct indices of refraction are calculated from the coefficients for all specified wavelengths.

13.1.1 Old Schott Formula

Formerly, Schott described the index of refraction in the visible portion of the spectrum by a Laurent series, sometimes called the "Schott formula"

$$n^2(\lambda) = A_0 + A_1 \cdot \lambda^2 + A_2 \cdot \lambda^{-2} + A_3 \cdot \lambda^{-4} + A_4 \cdot \lambda^{-6} + A_5 \cdot \lambda^{-8} \tag{13.1}$$

where λ = wavelength in μm and n = refractive index.

13.1.2 Sellmeier Formula

The Sellmeier formula has recently been adopted by Schott and other glass manufacturers.

$$n^2\left(\lambda\right) - 1 = \frac{B_1\lambda^2}{\lambda^2 - C_1} + \frac{B_2\lambda^2}{\lambda^2 - C_2} + \frac{B_3\lambda^2}{\lambda^2 - C_3} \tag{13.2}$$

where λ = wavelength in μm.

13.1.3 Extended Sellmeier Formula

The extended Sellmeier formula adds more coefficients to the standard Sellmaier equation.

$$n^2\left(\lambda\right) - 1 = \frac{B_1\lambda^2}{\lambda^2 - C_1} + \frac{B_2\lambda^2}{\lambda^2 - C_2} + \frac{B_3\lambda^2}{\lambda^2 - C_3} + \frac{B_4\lambda^2}{\lambda^2 - C_4} + \frac{B_5\lambda^2}{\lambda^2 - C_5} \tag{13.3}$$

where λ = wavelength in μm.

13.1.4 Nikon Dispersion Formula

This form is used by Nikon:

$$n^2(\lambda) = A_0 + A_1\cdot\lambda^2 + A_2\cdot\lambda^4 + A_3\cdot\lambda^{-2} + A_4\cdot\lambda^{-4} + A_5\cdot\lambda^{-6} + A_6\cdot\lambda^{-8} + A_7\cdot\lambda^{-10} + A_8\cdot\lambda^{-12} \tag{13.4}$$

where λ = wavelength in μm.

13.1.5 Herzberger Formula

The Herzberger equation combines Sellmeier and power series terms. It was first developed for glasses and later applied to infrared crystalline materials.

$$n = A + \frac{B}{(\lambda^2 - \lambda_0^2)} + \frac{C}{(\lambda^2 - \lambda_0^2)^2} + D\lambda^2 + E\lambda^4 + F\lambda^6 \tag{13.5}$$

where the choice of the constant $\lambda_0^2 = 0.028$ is arbitrary in that it is applied to all materials. The wavelength λ is given in μm.

13.1.6 Primary Dispersion

The difference in the refractive indices at the wavelengths corresponding to the F and C lines referred to the wavelength at the d-line is called the *primary dispersion*. It is expressed by the Abbe number

$$\nu = \frac{n_d - 1}{n_F - n_C} \tag{13.6}$$

where n_d is the index of refraction at 0.5876μm, n_F is the index of refraction at 0.4861μm and n_C is the index of refraction at 0.6563μm.

13.1.7 Partial Dispersion

The partial dispersion is expressed as the ratio

$$P_{x,y} = \frac{n_x - n_y}{n_F - n_C} \tag{13.7}$$

for two selected wavelengths x and y. In $OpTaliX$, the two commonly used partial dispersions in the visible and near-infrared portion of the spectrum are

$$P_{g,F} = \frac{n_g - n_F}{n_F - n_C}\,, \qquad P_{C,s} = \frac{n_C - n_s}{n_F - n_C} \tag{13.8}$$

13.2 dn/dT

The basic Schott model is used for the absolute index change from the index at standard temperature and pressure. It is given by

$$\frac{dn_{abs}(\lambda,T)}{dT} = \frac{n^2(\lambda,T_0)-1}{2\cdot n(\lambda,T_0)} \cdot \left(D_0 + 2\cdot D_1 \cdot \triangle T + 3\cdot D_2 \cdot \triangle T^2 + \frac{E_0 + 2\cdot E_1 \cdot \triangle T}{\lambda^2 - \lambda_{TK}^2} \right) \tag{13.9}$$

with:
T_0 = Reference temperature (20°C)
T = Temperature (in °C)
$\triangle T$ = Temperature difference versus T_0
λ = Wavelength (in μm) in vacuum
λ_{TK} = average resonance wavelength (in μm)

Note that some glass manufacturers only provide dn/dT-data at discrete points (wavelengths and/or temperatures). In such cases, the data is fitted according to Eq. 13.2 in order to give a continuous representation of dn/dT. This may result in small (practically negligible) deviations from catalogue data in temperature calculations, when listing dn/dT (DNDT) data (see LIS DNDT command, page 173) or querying DNDT as lens database item (page 459).

13.3 Catalogue Glasses

Glasses from glass manufacturers are designated on surfaces by an alphanumeric code. This code (a character string) may contain the glass name as well as the manufacturer short code (a 3 character string). If both, manufacturer short code and glass name are provided, they are separated by a colon. The general syntax is:

`gla si..j [manuf:]name`	An alphanumeric code, limited to 64 characters, from a manufacturers catalogue is entered.

Examples:

```
gla s1..3 BK7
gla s4 lak9
gla s2 sch:bk7
```

The manufacturers short codes are derived from the first 3 characters of the manufacturers name, which are given in table 13.2.

Short Code	Glass Manufacturer
SCH	Schott 2000
SCO	Schott (old catalogue)
OHA	Ohara
HOY	Hoya
COR	Corning
SUM	Sumita
CAR	Cargille (liquids)
LPT	LightPath (Gradium glass)
GRT	GrinTech, Jena, gradient index glass
NSG	Nippon Sheet Glass Company
GLC	Gradient Lens Corp.
CHI	Chinese catalogue
ARC	Archer OpTx
SPE	Special Materials (Infrared, plastics, etc.)

Table 13.2: Short codes of glass manufacturers.

Glass name and manufacturer short code are case insensitive, e.g. BK7 and bk7 are treated as identical glasses.

13.4 Private Glasses

In most cases, the refractive index is implicitly defined by specification of a glass name. The refractive index is then calculated from coefficients stored in the glass catalogues. Other than the glass name, there is no further user interaction required to obtain the correct index. In some cases, however, it is necessary to explicitly enter the refractive index for given wavelengths, for example when exact coefficients are not available or to enter data for materials that are not included with $OpTaliX$.

With private glasses you enter your own glass names and associated index data. Private glasses are part of the lens in memory and only apply to that lens. Private glass data will be stored with the prescription data.

Private glasses must not be confused with melt glasses as described in the glass manager section 24.9, page 427. Melt glasses are also defined by wavelength/index pairs, however, they are stored in a separate glass catalogue file and are globally available within the $OpTaliX$ environment.

Private glasses only retain to the current lens. To make private glasses available for use with several lenses, create a sequence (.SEQ file) with the desired private glass commands for all the glasses to be included and execute this sequence with each lens. A private glass must be defined before it can be specified on a surface.

Definition of private glasses can be accomplished in three ways,

- by entering pairs of wavelength and index of refraction, or
- by Laurent dispersion coefficients, or
- by glass manufacturers Sellmeier dispersion coefficients.

13.4.1 Private Glass defined by Wavelength-Data Pairs

All private glass data are enclosed by the `PRV`, `END` commands. The example below shows definition of a private glass (`mybk7`) using wavelength-index data pairs:

```
PRV
PWL         0.435    0.479    0.547    0.587    0.656
'myBK7'     1.527    1.523    1.519   1.5168    1.514
END
```

13.4.2 Private Glass defined by Laurent Dispersion Coefficients

Private glasses using Laurent coefficients are defined by entering the glass name, dispersion formula type and dispersion coefficients. The Laurent dispersion formula uses the `LAU` designator right to the glass name:

```
PRV
'myBK7'   LAU   coeff1  coeff2  coeff3  ...  coeff n
END
```

13.4.3 Private Glass defined by Sellmeier Dispersion Coefficients

The Sellmeier dispersion formula uses the `GMS` (glass manufacturers Sellmeier) designator right to the glass name:

```
PRV
'myBK7'   GMS   coeff1  coeff2  coeff3  ...  coeff n
END
```

These command sequences may also be conveniently stored in a macro file and then executed by the RUN command. The wavelength/index pairs need not to be sorted for (ascending or descending) wavelength. Wavelength values should be specified in micrometers (the default in $OpTaliX$), however, wavelengths in nanometer are also recognized to support compatibility with Code V syntax. Wavelength data > 100 are interpreted as nanometers (nm), otherwise micrometer (μm) are assumed.

Private glasses may be specified on surfaces like any other catalogue glass, except that the glass name must be enclosed in apostrophes. Example:

`gla s2 'MYBK7'`

Also note that names given to private glasses are case sensitive, i.e. `'MYBK7'` and `'mybk7'` are treated as two separate glasses.

`PRV`	Start private glass entries. It accepts then `PWL` commands and 'glass_name' entries until terminated with and `END` command. Any other *OpTaliX* command can be used within the `PRV` ... `END` environment. See also the `END` command below.
`PWL wavel_1 ... wavel_20`	Enter wavelength (in μm) for next refractive indices. This command is only required for wavelength-index data pairs. Up to 20 wavelengths are accepted. Wavelength data may also be entered in nanometers (nm) for Code V compatibility. Values > 100 are interpreted as nanometers, otherwise in micrometers (μm). Private wavelength data should at least span the wavelengths to be used in calculations, as defined in the system data, or by the WL command. Interpolation will be done as necessary; extrapolation outside this range will be done, but accuracy is not assured.
`'glass_name' index_1... index_20`	**For wavelength-index data pairs**, enter up to 20 indices for the user-defined `'glass_name'` with index values corresponding in order and number to the prior `PWL` command. If `'glass_name'` matches a catalogue glass, the catalogue glass always takes precedence, i.e. the private glass data will be ignored.
`'glass_name' LAU|GMS coeff_1... coeff_6`	**For dispersion coefficients**, enter up to 6 coefficients for the user-defined `'glass_name'`. If `'glass_name'` matches a catalogue glass, the catalogue glass always takes precedence, i.e. the private glass data will be ignored.
`END`	Terminates entry of private glass data, started by `PRV`.
`IND sk [wk]`	Returns index of refraction at surface `sk` and wavelength number `wk` in macros and lens database queries. Omission of `wk` returns the index at the reference wavelength. Note that `IND` may also be used for direct index specification (see obsolete commands below).
Obsolete commands:	
	continued on next page

continued from previous page	
`IND sk|si..j index_1... index_11`	Directly specify indices for the wavelengths currently in use (see `WL` command) without an underlying dispersion model. That is, the indices entered on surface(s) `sk|si..j` must correspond to the system wavelengths. The obligatory glass name must be '`PRI`' (without the apostrophes), see also next row. Although still available, use of this command is discouraged. Use the PRV - END construct as described in the commands above. The problem with direct index specification arises if wavelengths are changed (for example using the `WL` command [page 46] or the `EDI CNF` command [page 42] via the configuration dialog). In such cases the refractive index data assigned to the surfaces cannot be updated for glasses with direct index specification. It is therefore the users responsibility, to take care of this index to wavelength relation.
`GLA sk|si..j PRI`	Defines a private glass with direct index specification. The refractive indices must correspond to the system wavelengths and must be entered using the `IND` command.

General Notes on Private Glasses:

Private glasses defined with the same name as an already existing private glass will change the data for the designated glass. Private glasses for which the glass name matches a catalogue glass, the catalogue glass always takes precedence, i.e the private glass data will be ignored.

Refracting indices for each system wavelength are fitted according to the old Schott formula (see Eq. 13.1).

13.5 Fictitious Glasses

In contrast to the finite number of real glasses, fictitious glasses are defined in a continuous glass model, and in theory allow an infinite number of available glasses. The dispersion of fictitious glasses is defined internally, and is derived from the Abbe-number ν and the partial dispersions $P_{g,f}$ and $P_{C,s}$. Fictitious glasses are defined by two parameter:

- the refractive index n_d at the wavelength $\lambda = 587.56nm$,
- the Abbe-number, which is a measure of the refractive index change with wavelength ($\lambda = 486.13nm$ and $\lambda = 656.27nm$) (see also section 13.1.6).

Fictitious glasses are denoted by a string of numeric digits of the following forms:

	`xxx.yyy`	where `xxx` = $n_d - 1$ and `yyy` = $10\nu_d$
or:	`xxxyyy`	where `xxx` = $n_d - 1$ and `yyy` = $10\nu_d$

The six-digit representation is also known as MIL-number. The length of the string is limited to 10 characters. Fictitious glasses are identified by the decimal point (anywhere within the string) or by the first character, which is a numeric digit. Consequently, a decimal point or a numeric digit as the first character is not allowed in any other glass codes. Since fictitious glasses are generic, properties other than refractive index and dispersion are not available. The fictitious glass model is restricted to the ”visible” wavelength region, i.e. between 400nm and 700nm. Extension to shorter and larger wavelengths is only possible with reduced accuracy.

Examples:

`GLA s3 514.642`	Define fictitious glass at surface 3 with $n_d = 1.514$ and $\nu_d = 64.2$
`GLA s3 514642`	Define fictitious glass by entering the SCHOTT code number (MIL-number)

Notes:

- Fictitious (or MIL-number) glasses are an approximation to real glasses. According to its definition, fictitious glasses should only be used in the visible range. Outside the visible wavelength range (ultraviolet or infrared) the fictitious glass model is not accurate and should be avoided.
- Fictitious glasses may be automatically converted to the nearest (regular) catalogue glasses as described in section 12.4 on page 194.

13.6 Special Materials

”Special” materials are all materials like plastic, crystals, liquids, semi-conductors etc. Also the Schott Glass filters are found in the special catalogue. The data used in the SPECIAL catalogue are from various literature sources and data sheets of material manufacturers. Many of the data provided are relatively inaccurate or were not measured at sufficiently small spectral intervals, respectively there are systematic differences among the literature sources. Apart from the measurement uncertainties, many of the data were taken at temperatures other than 20°C. This may cause incorrect results if a system is analyzed at 20°C while the refractive index base is at another temperature. The user should be aware of it.

13.6.1 Infra-red Materials, Plastics

Material name	Spectral range (μm)	Description	Reference
AIR	0.2 - 15	Air	Kohlrauch [28], see also section 13.7 on page 228.
AGCL	0.5 -14	Silver Chloride	JOSA Vol.40, No.8, p.540
AGCL_IR	6.0 - 20.0	Silver Chloride, infrared band	JOSA Vol.40, No.8, p.540
ALON	0.4 - 2.3	Aluminum Oxynitride (ALON) Spinel	Handbook of Optics, Second Edition,Vol2, 1995
AMTIR1	7.0 - 12.0	$Ge_{33}As_{12}Se_{55}$	P.Klocek, Handbook of Infrared Optical Materials
AMTIR1A	1.5 - 12.0	$Ge_{33}As_{12}Se_{55}$	Amorphous Materials, (www.amorphousmaterials.com)
AQUEOUS	0.36 - 1.1	Ocular medium	Navarro et.al., JOSA A, Vol2., No.8, pp.1274

continued on next page

continued from previous page			
AS2S3	1.0 - 9.0	Arsenic Sulfide	Handbook of Optics, 1978
B270	0.36 - 1.06	Desag float glass, super-white	Desag data sheet
BAF2	0.4 - 10.0	Barium Fluoride	JOSA Vol.40, No.8, p.540
BATIO3	0.4 - 0.7	Barium Titanate ($BaTiO_3$)	Handbook of Optics, Second Edition, Vol2, 1995
BGG	0.4 - 5.5	Barium Gallogermanate Glass	Appl. Opt., Vol.41, No.7, March 2002, pp. 1366
CAF2	0.42 - 5.0	Calcium Fluoride	Appl.Optics, Vol.2, No.11, p.1103
CAF2_IR	3.0 - 9.0	Calcium Fluoride, infrared band	Appl.Optics, Vol.2, No.11, p.1103
CAF2_UV	0.15 - 2.0	Calcium Fluoride, ultraviolet band	Schott Lithotec datasheet
CAF2_VIS	0.365 - 1.06	Calcium Fluoride, visible band, enhanced interpolation accuracy	Appl.Optics, Vol.2, No.11, p.1103
CERAM-Z	0.4 - 1.6	Clearceram-Z	Zero-expansion glass-ceramics, Ohara data sheet
CERAM-ZHS	0.4 - 1.6	Clearceram-Z HS	Zero-expansion glass-ceramics, Ohara data sheet
CDTE	1.0 - 30.3	Cadmium Telluride	Palik, Handbook of Optical Constants of Solids, Academic Press 1985
CLEARTRAN	0.45 - 10.0	”Cleartran” (water clear ZnS)	Rohm & Haas Advanced Materials data sheet (www.cvdmaterials.com)
COR9754	0.42 - 5.2	Germanate glass	Corning, France, data sheet
CORNEA	0.36 - 1.1	Ocular medium	Navarro et.al., JOSA A, Vol2., No.8, pp.1274
CSBR	0.5 - 22.0	Cesium Bromide	Journal of Research of the National Bureau of Standards, Vol. 51, No.3, 1953, p.123
CSJ	0.3 - 26.0	Cesium Iodide	JOSA, Vol.45, No.11, p.987
CSJ_IR	9.0 - 40	Cesium Iodide	JOSA, Vol.45, No.11, p.987
DIAMOND	0.3 - 20	CVD-Diamond	Diamond Materials, www.diamond-materials.com
EYELENS	0.36 - 1.1	Ocular medium	Navarro et.al., JOSA A, Vol2., No.8, pp.1274
GASIR1	2.0 - 14.0	Ge22As20Se58	Umicore technical data sheet
GASIR2	2.0 - 14.0	Ge22Sb15Se65	Umicore technical data sheet
GERMANIUM	2.99 - 13.2	Germanium, poly-crystalline	JOSA, Vol.48, Aug.1958, p.579, Salzberg & Villa
GE_POLY	2.99 - 13.2	Germanium, poly-crystalline	JOSA, Vol.48, Aug.1958, p.579, Salzberg & Villa
GE_MONO	2.9 - 22.0	Germanium, mono crystalline	JOSA, Vol.48, Aug.1958, p.579, Salzberg & Villa
HERASIL	0.22 - 2.3	Fused quartz	Heraeus datasheet
HOMOSIL	0.22 - 2.3	Fused quartz	Heraeus datasheet
			continued on next page

continued from previous page

INFRASIL	0.22 - 2.3	Fused quartz	Heraeus datasheet
IRG2	0.405 - 4.59	Chalcogenide glass	Schott datasheet
IRG3	0.656 - 4.59	Chalcogenide glass	Schott datasheet
IRG7	0.486 - 3.3	Chalcogenide glass	Schott datasheet
IRG9	0.404 - 3.3	Chalcogenide glass	Schott datasheet
IRGN6	0.486 - 3.3	Chalcogenide glass	Schott datasheet
IRG100	1.0 - 14.0	Chalcogenide glass	Schott datasheet
IRG11	0.58 - 4.59	Chalcogenide glass	Schott datasheet
IRTRAN1	1.1 - 6.2	MgF_2	P.Klocek, Handbook of Infrared Optical Materials
KBR	0.5 - 12.0	Potassium Bromide	SPIE, Vol.400, p.141
KCL	0.5 - 12.0	Potassium Chloride	SPIE, Vol.400, p.141
KRS5	1.0- 22.0	Thallium Bromoiodide	JOSA, Vol.46, No.11, p.956
LIF	0.19 - 5.5	Lithium Fluoride	The Infrared Handbook, IRIA, William L. Wolfe
LIF_IR	5.0 - 11.0	Lithium Fluoride, IR-band	Handbuch der Physik
LIF_UV	0.19 - 1.2	Lithium Fluoride, UV-band	Handbuch der Physik
LUMICERA	0.40 - 0.7	Lumicera, transparent ceramics	Datasheet from Murata Manufacturing Co. Ltd., 4-4-1 Higashi-Okino, Yokaichi city, Shiga 527-8558, Japan.
MACROLON	0.36 - 1.06	”Bayer” trade name	
MGF2	0.2 - 7.0	Magnesium Fluorite, ordinary index, wide spectral range, Sellmeier equation	Appl.Optics, Vol.23, No.12, p.1980
MGF2_O	2.2 - 7.0	Magnesium Fluorite, ordinary index	Appl.Optics, Vol.23, No.12, p.1980
MGF2_E	2.2 - 7.0	Magnesium Fluorite, extraordinary index	Appl.Optics, Vol.23, No.12, p.1980
MGF2_VO	0.2 - 3.0	Magnesium Fluoride	Appl.Optics, Vol.23, No.12, p.1980
MGO	0.5 - 5.1	Magnesium Oxide	E.D. Palik, Handbook of Optical Constants of Solids II
MGO_IR	2.5 - 5.55	Magnesium Oxide	E.D. Palik, Handbook of Optical Constants of Solids II
NACL	0.5 - 12.0	Natrium Chloride	
NOA61	0.36 - 2.3	Norland adhesive cement	Norland data sheet
PBF2	0.4 - 10.0	Lead Fluoride	
PMMA	0.36 -1.06	Polymethyl Methacrylate (Lucite, Plexiglass)	Photonics design and applications handbook, 1996
POLYCARB	0.36 - 1.06	Polycarbonate (Lexan, Merlon)	Germanow Simon Corp. datasheet
QUARTZ	0.2 - 3.5	Fused quartz	equivalent to Suprasil, data from Heraeus datasheet
QUARTZ_IR	0.9 - 3.4	Fused quartz	Heraeus datasheet

continued on next page

continued from previous page			
SAPPHIRE	0.27 - 5.4	Sapphire	JOSA, Vol.52, No.12, p.1377
SILICA	0.2 - 3.5	Fused quartz (Suprasil)	Heraeus datasheet
SILICON	2.43 - 11.2	Silicon	Applied Optics, Vol.19, No.24, pp.4130, (1980), Salzberg & Villa data. It appears that these data are also used in Code V.
SILICON2	1.4 - 9.0	Silicon	Eagle Pitcher data sheet
SILICON3	1.5 - 12.0	Silicon	H.H.Li, Refractive Index of Silicon and Germanium and its Wavelength and Temperature Derivatives, J.Phys.Chem. Ref.Data, Vol.9, No.3, 1980
STYRENE	0.36 - 1.06	Polystyrene (Dylene, Styron, Lustrex)	Germanow Simon Corp. datasheet
SUPRASIL	0.27 - 3.5	fused quartz	Heraeus datasheet
TGG	0.38 - 1.6	Terbium Gallium Garnet	U.Schlarb, B. Sugg, ”Refractive Index of Terbium Gallium Garnet”, physica status solidi (b) 182, K91 (1994)
TOPAS5013	0.4 - 1.07	Cyclic olefin copolymer (COC)	Ticona datasheet
VACUUM	0.2 - 1.1	Vacuum	F.Kohlrauch, ”Praktische Physik”, 1968, Vol.1, p.408
VITREOUS	0.36 - 1.1	Ocular medium	Navarro et.al., JOSA A, Vol2., No.8, pp.1274
WATER	0.38 - 0.72	Water	
WATER2	0.40 - 0.80	Water with dn/dt data	R.C.Millard, G.Seaver [38]
SEAWATER	0.40 - 0.80	Seawater with dn/dt data	R.C.Millard, G.Seaver [38]
ZEONEX330R	0.36 - 0.80	Cyclo Olefin Polymer	Zeon-Europe
ZEONEXE48R	0.36 - 1.7	Cyclo Olefin Polymer	Zeon-Europe
ZEONEX480R	0.40 - 1.0	Cyclo Olefin Polymer	Zeon-Europe
ZERODUR	0.4 - 0.7	Zerodur	Schott datasheet, and Schott TIE-43 ”Optical properties of Zerodur”
ZNS	0.4 - 0.8	Zink Sulphide, visible and medium infrared (Trade name:Cleartran)	Morton datasheet
ZNS_IR	3.0 - 12.0	Zink Sulphide, infrared	Morton datasheet
ZNS_M	0.4 - 8.0	Zink Sulphide, multispectral	Morton datasheet
ZNS_M_IR	3.0 - 12.0	Zink Sulphide, multispectral	Morton datasheet
ZNSE	0.54 - 10.2	Zink Selenide	Morton datasheet
ZNSE_IR	7.8 - 18.2	Zink Selenide, infrared	Morton datasheet

13.6.2 Schott Filter Glasses

BG3	FG03	GG385	KG01	NG01	OG515	RG09	UG01	VG06	WG225
BG4	FG13	GG395	KG02	NG03	OG530	RG610	UG05	VG09	WG280
BG7		GG400	KG03	NG04	OG550	RG630	UG11	VG14	WG295
BG12		GG420	KG04	NG05	OG570	RG645			WG305
BG18		GG435	KG05	NG09	OG590	RG665			WG320
BG20		GG455		NG10		RG695			
BG23		GG475		NG11		RG715			
BG24		GG495		NG12		RG780			
BG25						RG830			
BG26						RG850			
BG28						RG1000			
BG34									
BG36									
BG38									
BG39									
BG40									
BG42									

13.6.3 Schott Radiation Resistant Glasses

The impact of high energy photon- and particle radiation reduces the spectral transmission of optical glasses. For example, this effect can already be observed at Gamma radiation of 10^3 rad (1.25 MeV) as a browning of the glass. The intensity of this change in colour is not only a function of the type of radiation and its dose, it also depends on the energy of the ionizing radiation.

Doping glasses with CeO_2 stabilizes them against colouring. Typically, the threshold at which colouring begins is raised to about 10^6 rad, at the expense of a reduced transmission in the blue.

The glass name of CeO_2 doped glasses is appended with the letter ”G” and a 2-digit number, indicating the amount of cerium oxide. For example, BaK1 G12 corresponds to 1.2% cerium oxide.

Available radiation resistant glasses from Schott:

BK7G18	SSK5G06	BK7G25
LAK9G15	K5G20	LF5G15
BAK1G12	F2G12	SK4G13
SF5G10	SK5G06	SF6G05
SK10G10	SF8G07	KZFS4G20
GG375G34		

13.6.4 Gradient Index (GRIN) Glasses

The glass catalogues store gradient index materials with radial and axial index profile from Nippon Sheet Glass (NSG), Gradient Lens Corporation (GLC) and LightPath (LPT). The following materials are available:

Manufacturer	Code	Name	z_{max}	n(587nm)	Profile	Remarks/Product Code
LightPath	LPT	G14SFN	5.800	1.8049	axial	
LightPath	LPT	G14SFP	5.800	1.6489	axial	

continued on next page

continued from previous page

LightPath	LPT	G22SFN	9.100	1.7860	axial	
LightPath	LPT	G22SFP	9.100	1.6569	axial	
LightPath	LPT	G23SFN	9.400	1.7758	axial	
LightPath	LPT	G23SFP	9.400	1.6561	axial	
LightPath	LPT	G32SFN	12.100	1.7666	axial	
LightPath	LPT	G32SFP	12.100	1.6731	axial	
LightPath	LPT	G41SFN	12.10	1.7443	axial	
LightPath	LPT	G41SFP	12.10	1.6961	axial	
LightPath	LPT	G51SFN	14.800	1.7446	axial	
LightPath	LPT	G51SFP	14.800	1.6982	axial	
LightPath	LPT	G4LAKN	13.931	1.7384	axial	
LightPath	LPT	G4LAKP	13.931	1.6726	axial	
NSG	SEL	SLN20	-	1.5845	radial	
NSG	SEL	SLS10	-	1.5477	radial	
NSG	SEL	SLS20	-	1.5477	radial	
NSG	SEL	SLW10	-	1.5868	radial	
NSG	SEL	SLW18	-	1.5868	radial	
NSG	SEL	SLW20	-	1.5868	radial	
NSG	SEL	SLW30	-	1.5868	radial	
NSG	SEL	SLH18	-	1.6294	radial	
NSG	SEL	SLA06	-	1.5238	radial	
NSG	SEL	SLA09	-	1.5845	radial	
NSG	SEL	SLA12	-	1.5930	radial	
NSG	SEL	SLA06A	-	1.5238	radial	
NSG	SEL	SLA09A	-	1.5845	radial	
NSG	SEL	SLA12A	-	1.5900	radial	
NSG	SEL	SLA20A	-	1.6098	radial	
Gradient Lens	GLC	EG10	-	1.5204	radial	
Gradient Lens	GLC	EG20	-	1.5204	radial	
Gradient Lens	GLC	EG27	-	1.5204	radial	
Gradient Lens	GLC	EG31	-	1.5204	radial	

GrinTech rods: The GrinTech product code is represented in a short form. The number in the 'GT050', 'GT100', or 'GT180' strings denotes the focal length (e.g. 050 = 0.5mm focal length), whereas the appendix denotes the intended wavelength: 06 = 670nm, 08 = 810nm, 13 = 1310nm, 15 = 1550nm.

Grintech	GRT	GT050-06	-	1.62885	radial	GT-LFRL-050-025-50-CC (670nm)
Grintech	GRT	GT100-06	-	1.62885	radial	GT-LFRL-100-025-50-CC (670nm)
Grintech	GRT	GT180-06	-	1.62885	radial	GT-LFRL-180-025-50-CC (670nm)
Grintech	GRT	GT050-08	-	1.623	radial	as above, at 810nm
Grintech	GRT	GT100-08	-	1.623	radial	as above, at 810nm
Grintech	GRT	GT180-08	-	1.623	radial	as above, at 810nm
Grintech	GRT	GT050-13	-	1.616	radial	as above, at 1310nm
Grintech	GRT	GT100-13	-	1.616	radial	as above, at 1310nm
Grintech	GRT	GT180-13	-	1.616	radial	as above, at 1310nm
Grintech	GRT	GT050-15	-	1.615	radial	as above, at 1550nm
Grintech	GRT	GT100-15	-	1.615	radial	as above, at 1550nm
Grintech	GRT	GT180-15	-	1.615	radial	as above, at 1550nm
Grintech	GRT	GT100	-	1.530	radial	
Grintech	GRT	GT180	-	1.530	radial	

continued on next page

continued from previous page

			-			
Grintech	GRC	GC050-06	-	1.524	cyl.	GT-LFCL-050-024-20 (670nm)
Grintech	GRC	GC100-06	-	1.524	cyl.	GT-LFCL-100-024-20 (670nm)
Grintech	GRC	GC130-06	-	1.524	cyl.	GT-LFCL-130-024-20 (670nm)
Grintech	GRC	GC050-08	-	1.624	cyl.	GT-LFCL-050-024-50-CC (810)
Grintech	GRC	GC100-08	-	1.624	cyl.	GT-LFCL-100-024-50-CC (810)
Grintech	GRC	GC130-08	-	1.624	cyl.	GT-LFCL-130-024-50-CC (810)
Grintech	GRC	GC050-09	-	1.621	cyl.	GT-LFCL-050-024-50-CC (940)
Grintech	GRC	GC100-09	-	1.621	cyl.	GT-LFCL-050-024-50-CC (940)
Grintech	GRC	GC130-09	-	1.621	cyl.	GT-LFCL-050-024-50-CC (940)

13.6.5 Liquids and Gels

A few specialty optical liquids from *Cargille Laboratories Inc.*[8] are stored in the glass database. They are grouped according to intended application as recommended by the manufacturer:

Immersion	:	Immersion liquids permit detection of imperfection in transparent and translucent materials and examination for stress and strain effects.
Laser	:	High transmission and highly stable liquids for laser wavelengths.
EC-Series	:	High refractive index, abnormal dispersion liquids. Low stability.
E, H, M-Series	:	Ultra-high refractive index, toxic and corrosive.
Matched	:	Matches precisely the refractive index of fused silica and closely approximates its dispersion.
Gel	:	Optical couplant gel for optical fibers to reduce or eliminate internal reflections or for mode stripping.

Name	Application	$n_D(589.3nm)$ at 20.0°C
CG1050_1	Immersion	1.400
CG1050_2	Immersion	1.425
CG1050_3	Immersion	1.458
CG5040_4	Immersion	1.475
CG5040_5	Immersion	1.500
CG5040_6	Immersion	1.535
CG5040_7	Immersion	1.570
CG4550	Immersion	1.452
CG433	Laser	1.295
CG3421_1	Laser	1.320
CG3421_2	Laser	1.400
CG1056_1	Laser	1.400
CG1056_2	Laser	1.455
CG5610	Laser	1.475
CG5763_1	Laser	1.600
CG5763_2	Laser	1.630
CGEC_164	EC-Series	1.640
CGM_178	M-Series	1.780
CGH_181	H-Series	1.810
CGE_155	E-Series	1.550

continued on next page

continued from previous page

CG50350	Matched	1.4587
CG06350	Matched	1.4587
CG0607	Gel	1.457
CG0608	Gel	1.457

It is important to note that the index of refraction of liquids is highly dependent on temperature. Typically, the dn/dT values of liquids are about a factor of 100 larger than those of optical glasses. The dispersion coefficients stored in the glass catalogue are always based on 25.0°C.

13.7 Air, Vacuum

There are two predefined optical ”materials”, air and vacuum. Physically, the refractive index of air is $n_{air} = 1.000273$ at normal temperature (20°C) and normal pressure ($0.101325 \cdot 10^6$ Pascal). According to standard practice, however, the index of air is regarded to be 1.0, rather than its true value. This approach is justified because the vast majority of optical systems are designed and used under normal atmospheric conditions (sea level). In addition, all standard glass catalogues have indices expressed relative to 1.0. Only very few (specialized) designs are used in vacuum. Thus, when entering the medium ”air”, the refractive index is uniformly set to 1.000 for all specified wavelength.

The index of air is altered by temperature and pressure in accordance with standard physical models. A good approximation, which also accounts for the wavelength dependence, is [28, 48]

$$n_{Air}\left(\lambda, T, p\right) = 1 + \frac{n_{Air}\left(\lambda, 15C, p_0\right) - 1}{1 + 3.4785 \cdot 10^{-3} \cdot (T - 15)} \cdot \frac{p}{p_0} \tag{13.10}$$

$$n_{Air}\left(\lambda, 15C, p_0\right) = 1 + \left\{6432.8 + \frac{2949810 \cdot \lambda^2}{146 \cdot \lambda^2 - 1} + \frac{25540 \cdot \lambda^2}{41 \cdot \lambda^2 - 1}\right\} \cdot 10^{-8} \tag{13.11}$$

with

p_0	=	$0.101325 \cdot 10^6$ Pa (= normal pressure in Pascal)
p	=	Pressure of air in Pascal
λ	=	Wavelength in μm in vacuum
T	=	Temperature in °C

The temperature dependance of the index of air is given by [48]

$$\frac{dn_{Air}(\lambda, T)}{dT} = -0.00367 \cdot \frac{n_{Air}\left(\lambda, T, p\right) - 1}{1 + 0.00367 \cdot T} \tag{13.12}$$

13.8 Index and Dispersion Offsets

Offsets on refractive index and dispersion may be applied to predefined catalogue materials and fictitious materials. They are entered by the `DNO` and `DVO` commands:

`DNO delta_ind`	Index of refraction offset.
`DVO delta_nue`	Dispersion offset. The value `delta_nue` refers to the Abbe-number ν_d (also called V-number) given in absolute values. Example: The ν_d value of Schott BK7 is 64.17. A dispersion offset `DVO 3.0` results in a new dispersion $\nu_d = 67.17$. For special materials (e.g. infrared materials), the actual synthetic ν-value should be considered when specifying `DVO`. See also the options on fictitious glass models below.
`DVOM 1|2`	Defines the model for calculating dispersion offsets used by the `DVO` command. Examples: `DVOM 1` : Dispersion offsets are exactly calculated according to the Abbe normal line as defined in the partial dispersion glass diagram (see sect. 24.3, or command `NFNC`). Anomalous dispersion of glasses, if present, are ignored. `DVOM 2` : Anomalous dispersion characteristics of special glasses is maintained when applying `DVO` dispersion offsets.

DNO and DVO commands should be applied with great care, since the n and ν-offsets are based on standard MIL-glasses (i.e. conform to the so-called ABBE line in the Schott glass diagram). They normally do not take the anomalous dispersion properties of many glasses into account. In addition, `DNO` and `DVO` may be used as variables during optimization, to let index and dispersion vary.

Named catalogue glasses that have `DNO` and `DVO` offsets assigned are indicated in the surface editor by red colour. In the surface listing (`LIS`) an asterisk is appended to the glass name. An example is given in Fig. 13.1 and in the listing below.

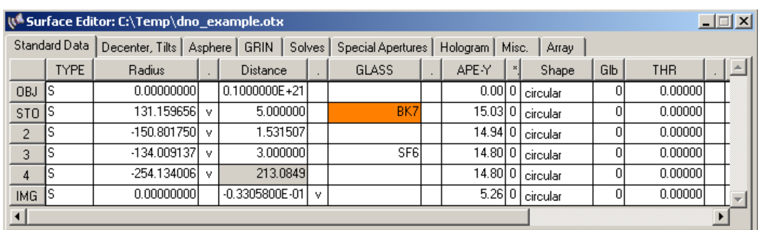


Figure 13.1: Glasses with `DNO`, `DVO` offsets are indicated by red colour.

```
  # TYPE         RADIUS     DISTANCE  GLASS          INDEX    APE-Y AP  CP  DP  TP  MP  GLB
OBJ  S         Infinity  0.10000E+21              1.000000     0.00  C   0   0   0   0    0
STO>S          131.1597      5.00000  BK7*        1.520304    15.03  C   0   0   0   0    0
  2  S        -150.8018      1.53151              1.000000    14.94  C   0   0   0   0    0
  3  S        -134.0091      3.00000  SF6         1.812665    14.80  C   0   0   0   0    0
  4  S        -254.1340    213.03629              1.000000    14.80  C   0   0   0   0    0
IMG  S         Infinity                           1.000000     5.26  C   0   0   0   0    0

GLASS OFFSETS:
                DNO            DVO             PGO            PCO         MIL-Code
  #     Index-offset      Nue-offset   P(g,F)-offset  P(C,s)-offset
  1      0.001500000    -2.347000000     0.000000000     0.000000000       5183.6181
```

13.9 Partial Dispersion Offsets

Partial dispersion offsets allow the simulation of anomalous dispersion properties of a real or fictitious glass. Since the values to be entered are offsets, `PGO` and `PCO` refer to

- the actual partial dispersions in case of a real glass (i.e. a glass from the catalogue)
- the Abbe normal line in case of a fictitious glass.

It should be noted that the partial dispersion offsets are not applicable to gradient index (GRIN) glasses.

Command syntax:

`PGO delta_P(g,F)`	Offset of partial dispersion $P_{(g,F)}$ from the nominal (catalogue) value, in case of fictitious glasses, from the Abbe normal line.
`PCO delta_P(C,s)`	Offset of partial dispersion $P_{(C,s)}$ from the nominal (catalogue) value, in case of fictitious glasses, from the Abbe normal line.

14

Image Evaluation

14.1 Geometrical Analysis

14.1.1 Paraxial Analysis

A standard collection of paraxial quantities is given in the prescription listing (see `LIS` command, page 173). These quantities refer to the entire system as indicated in Fig. 14.1. In addition, paraxial quantities may be obtained by specifying surface ranges (`si..j`) or zoom ranges (`zi..j`), as described in the table below.

`FIR`	Evaluate first order properties, such as focal length, magnification, etc.
`EFL [si..j | wi..j | zi..j]`	Retrieve the equivalent focal length for a range of surfaces or zoom positions. Without parameters, the `EFL` of the entire system is returned for all surfaces (`s1..i`), at the reference wavelength, for all zoom positions. Examples: `EFL` ! Focal length at reference wavelength, all zoom positions `EFL z1` ! Focal length at reference wavelength, zoom position 1 `EFL s1..4 z2 w3` ! Focal length of surfaces 1-4, zoom position 2, wavelength 3.
`BFL [wk|wi..j zi..j]`	Back focal length (distance from last surface to image plane) **at used conjugate**. Options are for wavelength numbers `i` to `j` and zoom positions `i` to `j`. If a wavelength qualifier (`wk`) is omitted, `BFL` is returned at the reference wavelength.
`SEP [zi..j]`	Evaluates the location of entrance pupil referred to first surface (not yet implemented)
`SAP [zi..j]`	Evaluates the location of exit pupil referred to last surface. Optional at zoom positions `zi..j`
	continued on next page

continued from previous page	
SAPI [zi..j]	Evaluates the reciprocal value of the location of exit pupil, referred to the last surface. That is, SAPI = 1/SAP. This function is particularly useful in optimization where the location of the exit pupil approaches infinity and the SAP function would be discontinuous. Zoom positions zi..j are optional.
PRD [zi..j]	Evaluates the pupil relay distance, that is the axial distance between the entrance and exit pupil. Optional at zoom positions zi..j
PRDI [zi..j]	Evaluates the reciprocal of the pupil relay distance, that is PRDI = 1/PRD. This function is particularly useful in optimization where the distance between entrance and exit pupil approaches infinity and the PRD function would be discontinuous.
OAL si..j zi..j	Overall length: Center thickness between surfaces si..j at zoom positions zi..j. If no parameters are given, the default setting for OAL is first surface to image for infinite objects, respectively object to image plane distance for finite objects.
OBD	Object distance. It is the separation from the object surface to the first surface in the system.
SYL si..j	Evaluate system length (= sum of thicknesses) for surface range si..j. If no surface range is specified, first surface to last surface (excluding object and image) will be assumed.
OID [si..j]	Axial distance from object surface to image surface. If a surface range (si..j) is specified, the axial distance between surfaces si..j is calculated. For objects at infinity, first surface to image surface is assumed. Note: The previously used command OOS is obsolete but retained for backwards compatibility.
SH1 [si..j] [zi..j]	Evaluates the location of the first (front) principal plane with respect to the first surface specified by si..j. If si..j is omitted, the first principal plane of the entire system is calculated.
SH2 [si..j] [zi..j]	Evaluates the location of the second (rear) principal plane with respect to the last surface specified by si..j. If si..j is omitted, the second (rear) principal plane of the entire system is calculated.
Related Commands	
UMY si..j zi..j	Paraxial direction angle of the marginal aperture ray (see page 105).
HMY si..j zi..j	Paraxial height of the marginal aperture ray (see page 105).
UCY si..j zi..j	Paraxial direction angle of the chief ray. See page 105.
HCY si..j zi..j	Paraxial height of the chief ray. See page 105.

14.1.2 Single Ray Tracing

Tracing a single ray through a system is accomplished by the following commands:

`sin [ si..j | gk | wi..j | zi..j | fi..j ] ape_absX ape_absY`

or

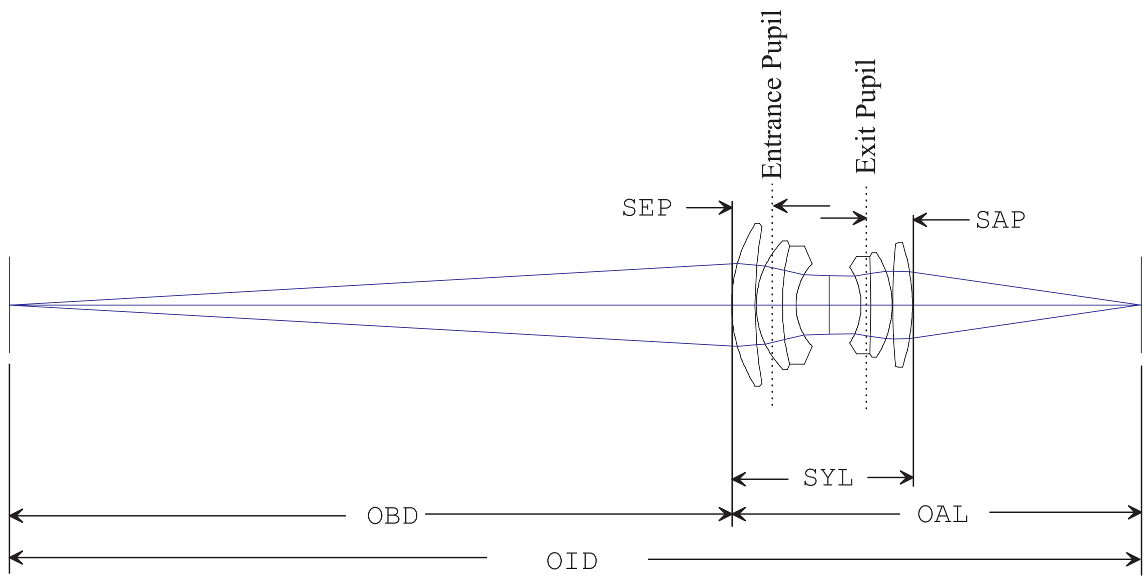


Figure 14.1: Definition of system data

```
rsi [ si..j | gk | wi..j | zi..j | fi..j ] ape_relX ape_relY
```

'`sin`' traces a single ray given absolute coordinates in the system entrance pupil, whereas '`rsi`' traces a single ray based on relative coordinates in the system entrance pupil.

The optional parameter are the designated zoom positions, wavelength, field, surface range and aperture. The ray coordinates at each surface are relative to the local coordinate system of each surface (i.e. the surface vertex).

Specifying a global reference surface `gk` outputs the ray coordinates with respect to the coordinate system at `gk`. If global coordinates (see `GLO` command on page 177) are activated, the ray coordinates are relative to the coordinate system of the surface specified by the `GLO`-command.

Notes on global coordinates output:
The `GLO sk` command is a permanent command. Once `GLO sk` is specified, ALL ray coordinates are referred to surface `sk` any time. Specify GLO N to disable global coordinates output. In contrast, in `rsi gk` commands (or `sin gk` commands), global output is active only for this particular command, irrespectively of `GLO Y|N|sk` settings.

Pupil coordinate definitions:

`ape_relX`	X-entrance pupil coordinate, a fraction of pupil X-radius. Values are between -1 and +1
`ape_relY`	Y-entrance pupil coordinate, a fraction of pupil Y-radius. Values are between -1 and +1
`ape_absX`	X-entrance pupil coordinate, absolute pupil coordinate. Values are absolute in mm.
`ape_absY`	Y-entrance pupil coordinate, absolute pupil coordinate. Values are absolute in mm.

Examples:

`rsi f1 w1 g3 0 1`	rim ray at field 1, wavelength 1, global ray coordinates referred to surface 3
`rsi f1 w1 0 1`	rim ray at field 1, wavelength 1, ray coordinates referred to local surface coordinates
`sin f1 w1 0 15`	rim ray at absolute entrance pupil coordinates (X/Y = 0/15) at field 1, wavelength 1, ray coordinates referred to local surface coordinates

14.1.3 Ray Aiming

`aim si [ wi..j | zi..j | fi..j ] ape_relX ape_relY`

Aims a ray to a specific (relative) aperture coordinate at a given surface `si` and at the designated zoom positions, wavelengths, and fields. The ray coordinates at each surface are relative to each surface's local coordinate system. If global coordinates (see `GLO` command on page 177) are activated, the ray coordinates are relative to the coordinate system of the surface specified by the `GLO`-command.

14.1.4 Single Ray Longitudinal Aberration

`LAX [ wi..j | zi..j | fi..j ] ape_relX ape_relY`	Computes the longitudinal aberration in the X-plane (sagittal) for a single ray. The aberration is always referred to the image surface.
`LAY [ wi..j | zi..j | fi..j ] ape_relX ape_relY`	Computes the longitudinal aberration in the Y-plane /tangential) for a single ray. The aberration is always referred to the image surface.

Note:
The longitudinal aberration is defined 'along' the optical axis. For ape_relX = 0 and ape_relY = 0, i.e. a ray going through the center of the aperture, LAX and LAY correspond to the sagittal and tangential astigmatism for the given fields and wavelengths.

14.1.5 Fan Aberration Curves (RIM Rays)

Fan rays are traced in either tangential or sagittal direction across the pupil. The aberrations may be plotted as transverse or longitudinal aberrations or as optical path difference.

`FAN [scale | ?]`	Transverse ray aberration fan. The optional parameter "`scale`" sets the aberration scaling for plotting. If not provided, the previous scaling value will be used. "`?`" invokes a dialog box to enter the plot scale.
`RIM [scale | ?]`	as above, only implemented as compatibility mode with CODE V.
`FANL [scale | ?]`	Longitudinal ray aberration fan. The optional parameter "`scale`" sets the aberration scaling for plotting. If not provided, the previous scaling value will be used. "`?`" invokes a dialog box to enter the plot scale.

continued on next page

continued from previous page	
OPDFAN [scale | ?]	Optical Path Difference (OPD). The aberrations are given in fractions of the reference wavelength (wave units). The optional parameter "scale" sets the aberration scaling for plotting. If not provided, the previous scaling value will be used. "?" invokes a dialog box to enter the plot scale.

The aperture axis in fan aberration plots, i.e. the axis representing the relative aperture coordinates, may be either plotted horizontal or vertical, depending on a users preference. This behaviour can be set in the program preferences (see page 3.2) by selecting from the main menu *File −− > Preferences* and then checking/unchecking 'Align ray fan curves horizontally' in the operations tab.

14.1.6 Spot Diagrams

A spot diagram collects the transverse aberrations in the image plane resulting from tracing a rectangular grid of rays (emerging from a single object point) through the system. Diffraction is ignored. The number of rays traced is approximately proportional to the square of the size of the rectangular grid in the entrance pupil as defined by the NRD command (see page 50). Increasing NRD will increase the accuracy of the spots but will also increase the computation time.

Spot diagrams may be displayed as a function of field, wavelength or zoom position. Note the optional parameter "?", which invokes a dialog box to modify the plot scale, i.e. the scale in which the aberrations are displayed. Alternatively, the plot scale may be specified explicitly as an additional parameter, which is useful in macro sequences.

SPO [plot_scale] SPO FLD [?] [plot_scale]	Spot diagram vs. field. This is the default.
SPO LAM [?] | [plot_scale]	Spot diagram vs. wavelength (colour)
SPO THF [?] | [plot_scale] [def_range]	Through Focus Spot diagram. plot_scale is the size of the aberration box in the plot and def_range is the ±defocus range along the optical axis.
SPO RIS [?] | [plot_scale]	Plots ray intersection points on a surface. See also section 14.1.8.
SPO ZOO [?] | [plot_scale]	Spot diagram vs. zoom position
SPO FF [?] | [plot_scale] [num_fields]	Array of spot diagrams extending over the full field, where plot_scale is the aberration scale of the spots, num_fields is the number of field points in X-and Y-direction (default = 3) Example: spo ff 0.02 5 ! Plots a 5x5 array of spots, scale is 0.02mm
SPR [fi..j, wi..j, zi..j] SPD [fi..j, wi..j, zi..j]	Evaluates rms-spot radius (SPR) respectively rms-spot diameter at fields fi..j, wavelengths wi..j and zoom positions zi..j. Results are given numerically.

continued on next page

continued from previous page	
SPR FLD [plot_scale] | [?]	Plots rms-spot diameter versus field. In case of zoomed systems, the currently selected zoom position (see POS command) is used. The maximum of the field definition is used. The question mark "?" invokes a dialog box for entering plot scale, settings of X-or Y-field and reference to chief ray or spot gravity center.
SPR LAM plot_scale [fi..j] | [?]	Plots rms-spot diameter versus wavelength (LAM holds for λ) at fields fi..j. In case of zoomed systems, the currently selected zoom position (see POS command) is used. The wavelength range is defined by the minimum and maximum wavelengths used (see WL command). The question mark "?" allows setting of X-or Y-field and reference to chief ray or spot gravity center. **Implemented in future release!**
SPO [fi..j | wi..j | zi..j] FILE file_name	Write spot aberrations to an ASCII file. No graphic output is generated. The qualifier 'FILE' is mandatory. If file_name is omitted, the user will be asked for a file name. Note that there is no default extension for the file name. The spot aberrations are written in a fixed format with the following columns: pos field colour X-abe Y-abe where pos = zoom position number (integer), field = field number (integer), colour = wavelength number (integer), X-abe = X-aberration relative to chief ray, Y-abe = Y-aberration relative to chief ray.
SPMS marker_size	Temporarily adjusts the size of markers used in spot diagrams. Marker size is defined in plot units (in mm) referred to the size of a standard A4 paper. The default spot marker size is 0.5mm. The spot marker size is predefined in the preferences section, miscellaneous tab.
IFO incr_in_focus	Increment in focus position

14.1.7 Spot Gravity Center

This option calculates the gravity center of the geometrical spot for all fields and wavelengths defined in the optical system.

XGR [fi..j wi..j] YGR [fi..j wi..j]	Calculates the X- and Y-coordinates of the spot gravity center on the image surface. Although XGR and YGR are functionally identical for reporting the image centroid, a distinction between X- and Y-coordinate is required when used in optimization, user defined graphics or tolerancing. This analysis includes the effects of wavelength weights (see WTW command, page 47).

Example command:

```
ygr f3
```

gives the following output in the ”Text Window”:

```
 Field        Wavel.  Rel.Wgt          X-Grav.       Y-Grav.     rel.Grav-X     rel.Grav-Y
     3      0.54600      1.00         0.000000     18.147916       0.000000      -0.002189
     3      0.45000      1.00         0.000000     18.141295       0.000000      -0.008810
     3      0.65000      1.00         0.000000     18.146546       0.000000      -0.003559
   Weighted gravity center:           0.000000     18.145252       0.000000      -0.004853
```

The ”X-Grav.” and ”Y-Grav” columns are the absolute gravity coordinates on the image surface referred to the vertex of the image surface. The ”rel.Grav-X” and ”rel.Grav-Y” columns are the gravity centers referred to the chief ray coordinate at the reference wavelength.

14.1.8 Surface Ray Intersection Plot

A square grid of rays, evenly spaced in the entrance pupil, is traced through the optical system and the intersection points of all rays on a designated surface are plotted. See Fig. 14.2. All fields, wavelengths and zoom positions are represented. Rays that are vignetted are not drawn, independently on which surface vignetting occurs. This way, usage of the light beam on a designated surface is shown. The number of rays in the grid are defined by the NRD command. The ray intersection plot is functionally equivalent with the footprint analysis (see page 403), both indicate the area on surfaces used by the beams. Ray intersection plots are more general, because they also take obscurations into account. Due to the finite sampling spacing of the rays, however, the exact boundary of the beam cannot be determined. If precise beam boundaries are required, the footprint option should be used.

`SPO RIS [ sk | plot_extent | ? ]`	Plots the intersection points of rays on surface `sk`. If `sk` is not specified, the default (surface 1) is used on the first plot, respectively for subsequent (repeated) plots the previously specified surface is used. The parameter `plot_extent` is optional and defines the maximum displayed area. Absence of `plot_extent` or a zero value invokes automatic determination of the plot extent on `sk`, except where the plot extent has already been determined by a previous plot. Rays are traced only in the reference wavelength.

14.1.9 Pupil Intensity Map

The pupil intensity map computes the intensity distribution in the system exit pupil for a given field, wavelength and zoom position. Typically, the intensity distribution across the exit pupil is uniform, however, effects like bulk material absorption or reflection losses at optical surfaces cause a spatial variation of the light intensity in the pupil. In this context, notice that any non-uniform illumination of the system pupil may be considered as *apodization*. Other influences leading to this effect are intensity filters (see INT command, page 136) on surfaces (loaded from an interferogram file, or non-uniform characteristics of the sources itself. For example, laser beams typically exhibit a Gaussian intensity profile which also modifies the effective intensity distribution in the pupil of a system.

Summarizing, the pupil intensity plot includes the effects of

- Pupil apodization (as defined in system configuration dialog or by PUI command, see page 51),

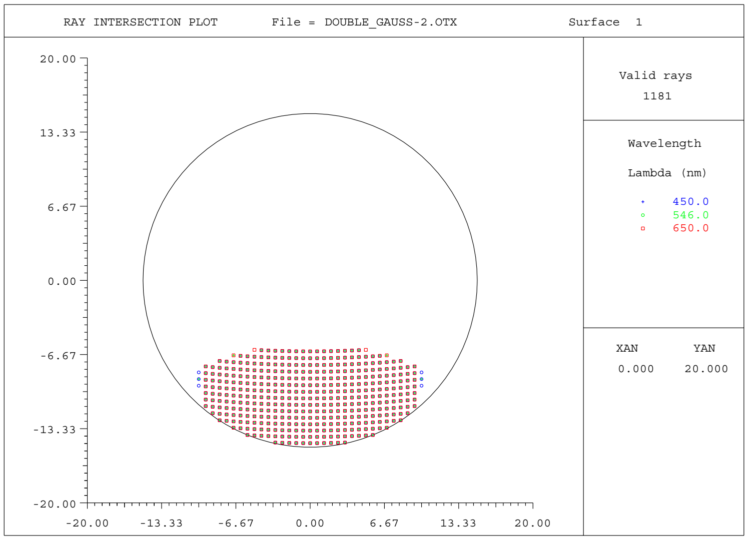


Figure 14.2: Ray intersection plot, indicating the area used on a surface. Here shown for a single field.

- Polarization or transmission (see POL and TRA commands, pages **??** and 325),
- Intensity filters, see INT format,
- Coatings and non-uniform coating thickness variations (see CTV).

Plots of the pupil intensity are used to control the intensity distribution in the exit pupil. This is an important feature, as any variation of the system transmission will result in a modification of the image performance. For example, the point spread function (PSF) of most optical systems can be computed by the Fourier Transform of phase and amplitude (the complex field) in the pupil. It is evident that any amplitude modulation will change the form of the PSF.

Pupil intensity maps are obtained by tracing bundles of rays through the entire system and monitor the reduction of the intensity of each ray caused by the above mentioned effects.

Pupil intensity plots are created by the command:

`PMA zk fk wk [WIR|GRY|FAL|CON|XY|?]`	Pupil map. Plots the intensity distribution across the system pupil at field number `fk`, wavelength number `wk` and zoom position `zk`. Plots can be displayed as wire grid (WIR) which is the default, gray level (GRY), false colour (FAL), contour plot (CON) or XY-slices (XY).

The command "`PMA ?`" (without the quotes) invokes a dialog box for editing plot parameters:

One single plot can be generated for a specific set of field, wavelength and zoom position. The check boxes "*include transmission*" and "*include polarization*" allow overriding of the configuration settings for a particular plot only. For example, unchecking the "*include transmission*" option ignores transmission effects in the pupil map plot, even though transmission analysis (seeTRA yes—no command) has been specified. In other words, the settings in this dialog box are temporarily and have no effect on the configuration settings (conditions of use).

The following figures (14.4 to 14.6) show various representations of pupil map intensity.

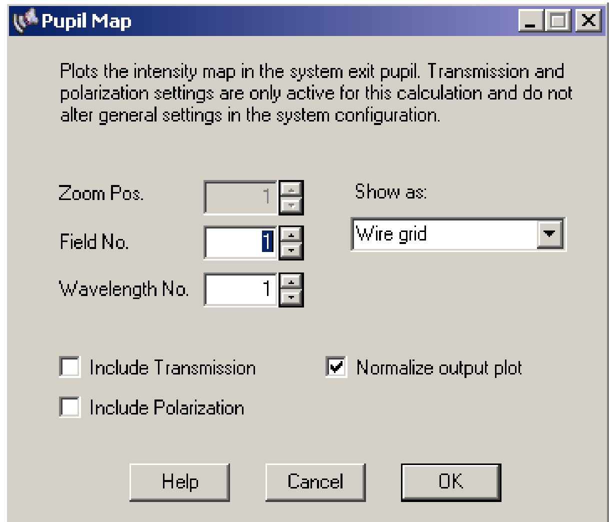


Figure 14.3: Dialog box for editing pupil intensity plot parameters.

14.1.10 Distortion

The distortion is expressed as the coordinate of the real image related to the paraxial image coordinate. It is given in % and may be analysed as chief ray distortion or spot gravity distortion.

$$D = \frac{y_{chief\_ray} - y_{paraxial}}{y_{paraxial}} \cdot 100 \tag{14.1}$$

$$D = \frac{y_{gravity} - y_{paraxial}}{y_{paraxial}} \cdot 100 \tag{14.2}$$

with

$y_{chief\_ray} =$ image height of the real chief ray
$y_{gravity} =$ image height of spot gravity center
$y_{paraxial} =$ paraxial image height (the expected distortion-free image height)

The distortion is always given in %. The paraxial image height $y_{paraxial}$ is calculated in two different ways:

$y_{paraxial} = \tan(w) \cdot EFL$ for conventional systems, i.e. the image coordinate is proportional to the tangent of the field angle

$y_{paraxial} = w \cdot EFL$ for F-Theta systems, i.e. the field coordinate is proportional to the field angle (in radians). This definition is widely used in scanning systems.

Afocal systems (i.e. object and image are at infinity) are not adequately described by the equations above. It is more appropriate to define an angular distortion which is the angular deviation of the outgoing beam from a nominal (distortion free) angle. Angular distortion is defined as

$$D_{\alpha} = \frac{\alpha_{real} - \alpha_{paraxial}}{\alpha_{paraxial}} \cdot 100 \tag{14.3}$$

with α = angle to the optical axis.

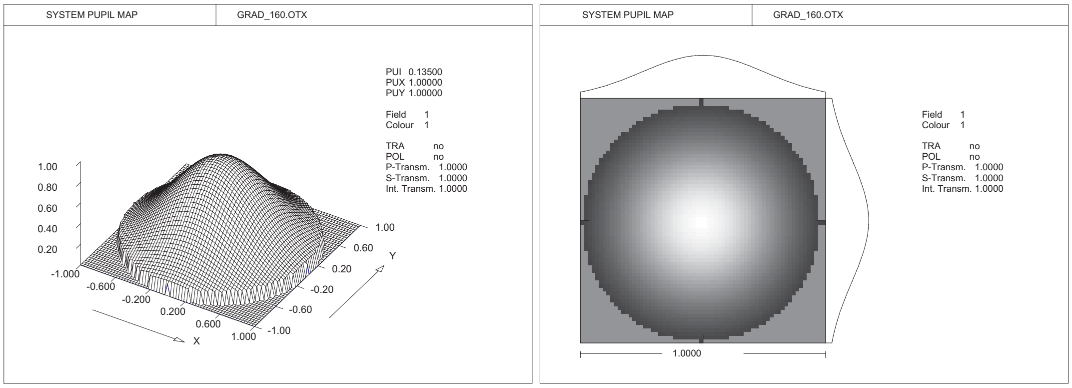


Figure 14.4: Pupil intensity map shown in wire-grid (`WIR`) and gray-scale (`GRY`) representations. Left: Wire grid plot, command: `PMA z1 f2 w3 WIR`, Right: Gray scale plot, command: `PMA z1 f2 w3 GRY`

The so-called F-Theta distortion is only meaningful in systems with an object at infinity. Here, the image height is proportional to the field angle which is mostly required in scanning systems. Strictly speaking, distortion is only valid for centered, rotationally symmetric systems with plane image surfaces, since the paraxial approximation does not account for such special systems.

Vignetting factors are ignored for chief ray distortion. However, for spot gravity distortion, vignetting is taken into account and may have impact on distortion.

Command syntax:

Numerical Distortion Analysis	
`DISX [fi..j, zi..j, GRAV]`	Distortion analysis for fields and zoom positions in X-direction. The optional parameter `GRAV` outputs distortion referred to the spot gravity center. Examples: `DISX f1..3` computes X-distortion at fields 1 to 3 `DISX GRAV f3 w2` computes spot gravity distortion in X-direction at field 3 and wavelength 2.
`DISY [fi..j, zi..j, GRAV]`	Distortion analysis in Y-direction.
`FDISX [fi..j, zi..j, GRAV]`	F-Theta distortion in X-direction.
`FDISY [fi..j, zi..j, GRAV]`	F-Theta distortion in Y-direction.
Distortion Plots	
`PLO DISY`	Plot distortion in Y-field direction. The entire field extension is plotted.
`PLO DISX`	Plot distortion in X-field direction. The entire field extension is plotted.
`PLO FDISY`	Plot F-theta distortion in Y-field direction. The entire field extension is plotted.

continued on next page

continued from previous page	
PLO FDISX	Plot F-theta distortion in Y-field direction. The entire field extension is plotted.
PLO DIG	Plot distortion grid. This is the deformation of a rectangular object grid caused by distortion. The full field extension is plotted. See description below.

14.1.11 Grid Distortion Plot :

The distortion grid plot also accounts for non-rotationally symmetric optical systems, which `DISX`, `DISY`, `FDISX`, `FDISY` do not because they are calculated in the Y/Z-plane only. Calculation of grid distortion assumes a perfectly rectangular grid at the object surface. The distortion of this grid when imaged through the system is then plotted at the image surface (see Fig. 14.7).

This analysis is performed for the full field extension in X- and Y-direction. If only the Y-field is specified (i.e. all X-field coordinates are zero), the full field is assumed circular with the maximum Y-field being the radius of the field circle. A square object field is then inscribed to this circle such that its diagonal (from lower left to upper right corner) is equal to the maximum field circle. The maximum extents of the image are derived from *paraxial* quantities. In extreme wide-angle systems (Fisheye) the paraxial image size may go to infinity if the full field angle approaches 180°, which may lead to problems in the plot diagram. To avoid this problem, a maximum image extension should be provided by the user. The command syntax is

PLO DIG [CHF|GRA|PSF] [enlargement_factor]	Plots the image of a rectangular object grid. `enlargement_factor` is the factor by which distortion (i.e. the deviation from the ideal grid) is enlarged in the plot. The distortion grid can be referred to chief rays (CHF), spot gravity centers (GRA) or PSF gravity centers (PSF). The default reference is chief ray (CHF). Examples: `plo dig 10.0` ! plot distortion grid at image surface, enlarged by factor 10. `plo dig gra 10` ! as above, however, grid is referred to gravity center of spot.

In particular when distortion is small, the distortion aberration may be enlarged in the plot by a user-defined factor. This may give a better impression of the shape of distortion. The distortion enlargement is defined by

$$x_{plot} = d_f\,(x_{ideal} - x_{real}) \tag{14.4}$$

where d_f is the `enlargement_factor`. That is, only distortion aberrations are plotted at an oversized (enlarged) scale, whereas the ideal grid is always plotted at the same size.

14.1.12 Field Aberrations - Astigmatism and Distortion Analysis

The field aberration option computes distortion, astigmatic field curves and optionally longitudinal spherical aberration. It provides a combined plot of all these three types of aberrations. Although

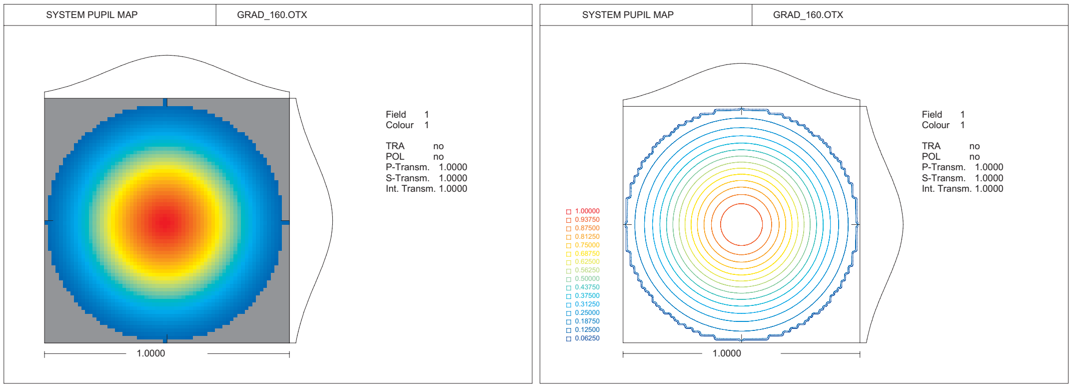


Figure 14.5: Pupil intensity maps shown in false-colour (FAL) and contour (CON) representations.
Left: False colour plot, command: `PMA z1 f2 w3 FAL`
Right: Contour plot, command: `PMA z1 f2 w3 CON`

longitudinal spherical aberration is not field dependent, it is sometimes desired for traditional reasons.

`FIE [LSA] [?]`	Plots field dependent aberrations: Astigmatism and distortion. The optional parameter `LSA` also plots longitudinal spherical aberration. The question mark invokes a dialog box for setting aberration scales (enter 0 for automatic scaling). For zoom systems the currently selected zoom position is used (see POS command). Figure 14.8 shows the plot layout.

Distortion is the change in magnification as a function of field. It is computed from tracing chief rays and is measured in percent relative to the paraxial field height. Astigmatism is represented in terms of longitudinal defocus for tangential (Y) and sagittal (X) planes at various field heights.

In addition to the combined plot, aberrations may also plotted separately. For distortion see sect. 14.1.10, page 239, for longitudinal spherical aberration see sect. 14.1.5, page 234.

14.1.13 First Order Analysis

`FIR`	Lists table of first-order (paraxial) system parameters (e.g. EFL, OAL, etc.) for all zoom positions. Note that paraxial system data are always output with the LIS command. See also the LIS PAR option (page 173).
`FIO [sk|si..j zk|zi..j]`	List paraxial data for marginal and chief rays for designated surface(s) `sk|si..j` and designated zoom position(s) `zk|zi..j`.

Although the ray-tracing equations used in $OpTaliX$ to evaluate an optical system are exact, they are complicated and provide little insight into the image-formation process. To reach simplified analytical results, a *first order* approximation is often a good starting point and in many applications precise enough. This is particularly valid when a common optical axis exists and when the light rays make small angles with the axis. Such rays are called *paraxial rays* and calculations in this domain are denoted as paraxial optics. Paraxial approximations were known already in the early 17^{th} century and Kepler used it when he first formulated the theory of the telescope. Paraxial calculations are derived from Snell's law $n \cdot sin\theta = n' \cdot sin\theta'$. If we recall that the sine may be expanded in a series

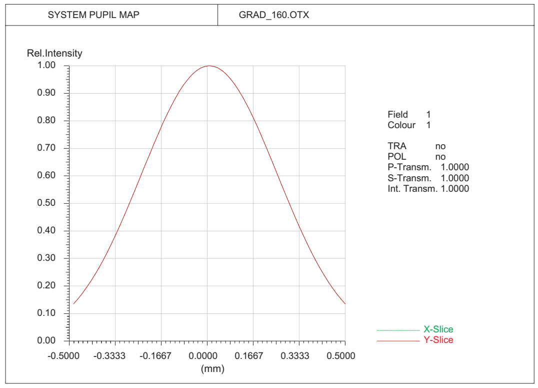


Figure 14.6: Pupil intensity maps shown by XY-slices (XY) representations.
Command: `PMA z1 f2 w3 XY`

$$sin\theta = \theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \frac{\theta^7}{7!} + \cdots \tag{14.5}$$

and assuming small values of θ, we may approximate $sin\theta \approx \theta$. This is the domain of what is called *first-order* or *paraxial* theory.

Paraxial quantities are displayed by the commands LIS, LIS PAR or FIR. For a detailed description of the output values see section 9.1 (page 173).

14.1.14 Third Order Analysis (Seidel Aberrations)

Third order aberrations are an approximation to the aberrations obtained by real (skew) ray trace. The advantage of third order[1] aberrations is that they can be calculated easily and quickly on the basis of paraxial quantities. In the contrary, exact ray trace equations are complicated as they involve the trigonometric functions of angles, instead of just the angles. When we speak of third order approximation, we truncate the series expansion given in Eq. 14.5 after the θ^3 term and only the first and third order terms in the expansion of the sine are retained. The resulting equations and corresponding aberrations are part of *third order optics*. In the same way that the sine was expanded in a series, the aberrations can be expanded. The first term in the expansion is known as the *third order aberration* (i.e. the first approximation to the total aberration).

To illustrate this point, Fig. 14.9 shows the spherical aberration of a lens based on real ray trace data. The aberration curve based on third order equations is shown as thick line.

Fig. 14.9 indicates that third order aberrations only give a more or less coarse approximation to the real aberration, in particular for larger apertures and/or fields. This behaviour depends on the system used. The beauty of third order aberrations, however, must be seen in the fact that they provide a deeper insight into the contributions of each surface onto the overall aberration of an optical system.

The astute reader may argue that an approximation involving fifth order aberrations may simulate the aberrations much better and give an even more deeper insight. However, fifth order (or even $7th$ order) equations are nearly as complex as real ray trace equations. Due to the advent of fast computers, exact ray trace aberrations, which include *all* orders, can be computed equally fast and there is no convincing reason any more to using 5^{th} order or higher order aberrations.

[1] sometimes also referred to as *tertiary* aberrations

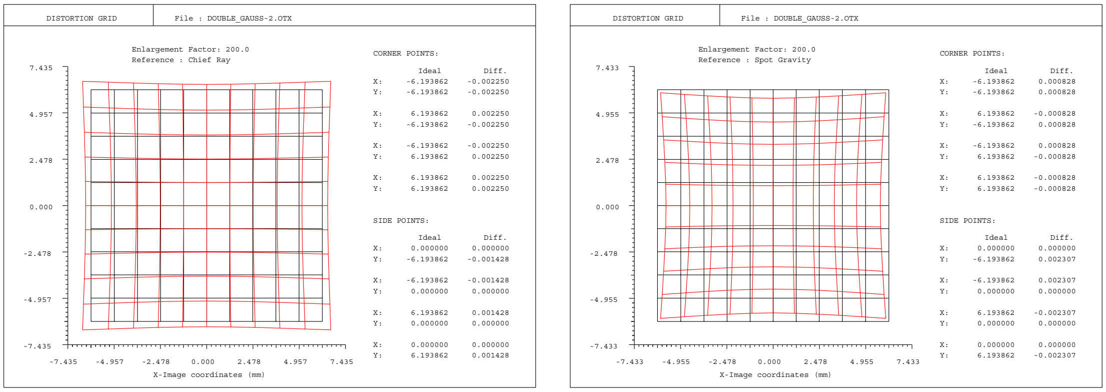


Figure 14.7: Grid distortion plots. Left: Distortion referred to chief rays. Right: Distortion referred to spot gravity center.

Command:

`THO`	Outputs the third order (Seidel) aberrations with surface contributions.

Third Order Formalism:

We refer to the paraxial quantities established in section 5.3 and define some system constants:

$$
\begin{aligned}
H &= n u_a h_b - n u_b h_a \quad \text{(Helmholz-Lagrange invariant)} && (14.6)\\
S &= \frac{Y'}{2H} && (14.7)\\
S_p &= \frac{Y' \cdot \Delta\omega}{H} && (14.8)\\
S_s &= \frac{Y'}{H} \cdot \left(\frac{\Delta\omega}{2}\right)^2 && (14.9)
\end{aligned}
$$

The paraxial image height is Y' and the Buchdahl chromatic variable ω is defined as (see [7],[46]),

$$\omega = \frac{\lambda - \lambda_0}{1 + 2.5\,(\lambda - \lambda_0)} \tag{14.10}$$

where λ_0 is the reference wavelength. For each surface, we define the following auxiliary variables:

$$
\begin{aligned}
i &= c \cdot h_a + u_a && (14.11)\\
j &= c \cdot h_b + u_b && (14.12)\\
b_a &= \frac{n}{n'}\left(n - n'\right) h_a \left(u_a + i\right) && (14.13)\\
b_b &= \frac{n}{n'}\left(n - n'\right) h_b \left(u_b + j\right) && (14.14)\\
a &= \left(n - n'\right)\left(k \cdot c^3 + 8A_4\right) && (14.15)\\
d_p &= \frac{\partial n}{\partial \omega} - \frac{n}{n'} \cdot \frac{\partial n'}{\partial \omega} && (14.16)\\
d_s &= \frac{\partial^2 n}{\partial^2 \omega} - \frac{n}{n'} \cdot \frac{\partial^2 n'}{\partial^2 \omega} && (14.17)
\end{aligned}
$$

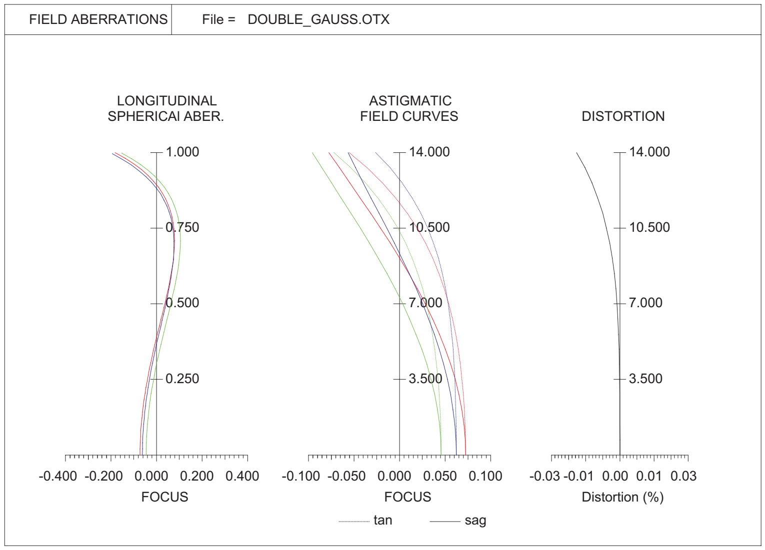


Figure 14.8: Field aberrations, astigmatism, distortion and longitudinal spherical aberration, combined in one plot.

From these constants, we obtain the surface contributions to the third order (Seidel) aberrations:

$$
\begin{array}{rlcl}
 & \text{spheric terms:} & & \text{aspheric terms:} \\
\text{Spherical: } A_i = & S \cdot b_a \cdot i^2 & + & S \cdot a \cdot {h_a}^4 \\
\text{Coma: } B_i = & S \cdot b_a \cdot i \cdot j & + & S \cdot a \cdot {h_a}^3 \cdot h_b \\
\text{Astigmatism: } C_i = & S \cdot b_a \cdot j^2 & + & S \cdot a \cdot {h_a}^2 \cdot {h_b}^2 \\
\text{Petzval: } P_i = & S \cdot H^2 \cdot \frac{n-n'}{n \cdot n'} \cdot c & + & 0 \\
\text{Distortion: } V_i = & S \cdot \left[b_b \cdot i \cdot j + H \left({u_b'}^2 - {u_b}^2 \right) \right] & + & S \cdot h_a \cdot {h_b}^3 \\
\text{Axial Color: } Fl_i = & S_p \cdot d_p \cdot h_a \cdot i & + & 0 \\
\text{Lateral Color: } Fq_i = & S_p \cdot d_p \cdot h_a \cdot j & + & 0
\end{array}
$$

The third order aberrations of the entire system are then the sum of the corresponding aberration contributions associated with the individual surfaces of the system, hence

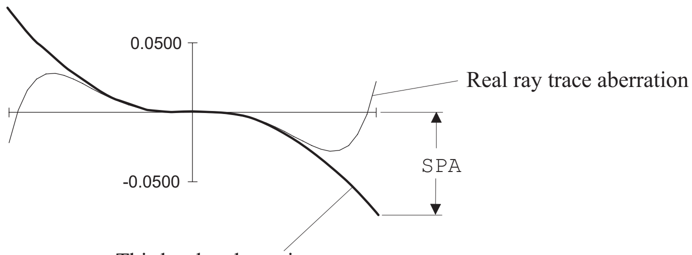


Figure 14.9: Third order aberration in comparison to real ray trace data, shown on the example of spherical aberration.

$$SPA = \sum_{i=1}^{n} A_i \quad (14.18)$$

$$COMA = \sum_{i=1}^{n} B_i \quad (14.19)$$

$$ASTI = \sum_{i=1}^{n} C_i \quad (14.20)$$

$$PETZ = \sum_{i=1}^{n} P_i \quad (14.21)$$

$$DIST = \sum_{i=1}^{n} V_i \quad (14.22)$$

$$LCA = \sum_{i=1}^{n} Fl_i \quad (14.23)$$

$$TCA = \sum_{i=1}^{n} Fq_i \quad (14.24)$$

(14.25)

14.1.15 Secondary Spectrum

The secondary spectrum (longitudinal colour) is the variation of the *paraxial* focus along the optical axis as a function of wavelength.

`SSP`	Secondary Spectrum, numerical output. Since this analysis is based on paraxial calculations, results may not be meaningful for non-paraxial (tilted, decentered or off-axis) systems.
`PLO SSP [ plot_scale | ? ]`	Plots the secondary spectrum. The optional question mark "?" invokes a dialog box for entering the plot scale.
`SSR [wi..j | zi..j]`	Secondary spectrum, weighted rms-value. It is computed as the rms-variation of the *paraxial* focus at wavelengths `wi..j` (including spectral weights) and at zoom positions `zi..j`. Since this analysis is based on paraxial calculations, results may not be meaningful for non-paraxial (tilted, decentered or off-axis) systems.

14.1.16 Lateral Colour

For a given wavelength, the lateral colour is the distance on the image surface with respect to the reference wavelength. A curve is plotted for each wavelength. Chief rays are used for this analysis.

Quite often the lateral colour is defined as the distance on the image surface from the shortest wavelength to the longest wavelength chief ray intercept. However, a lot of information is lossed by this approach, which may be misleading because the shortest/longest wavelength may not exhibit the worst aberration. This problem is avoided in $OpTaliX$.

`LAC wi..j [fi..j, zi..j]`	Lateral colour within wavelength range `wi..j`. A wavelength range is required, field and zoom specification are optional. It is the maximum lateral deviation for all wavelengths from the chief ray intercept of the ray at the reference wavelength. Wavelength weights are not in effect for this type of analysis.
`PLO LAC`	Plot lateral colour vs. field. For each wavelength, the lateral deviation from the chief ray intercept of the ray at the reference wavelength is plotted vs. field. A dialog box is opened to enter the plot scale.

14.1.17 Ghost Image Analysis

Optical systems can form unintended images due to reflections between pairs of surfaces. All lens surfaces reflect light to an extent depending on the refractive index of the glass itself respectively on the type of anti-reflection coating applied to these surfaces. Light reflected from the inner surfaces of a lens will be reflected again and may form reasonably well-defined images close to the image surface. Such spurious images are called *ghost images*.

The number of possible surface combinations (pairs) which may contribute to ghost images is $n(n-1)/2$, where n is the number of lens surfaces in the system. As the number of surfaces grows, the probability of ghost problems also increases. For example, a zoom lens with 10 lenses (20 surfaces) gives 190 possible ghost images.

As a guideline, the transmittance of a lens including all possible multiple reflections, but ignoring any loss of light by absorption in the glass, is given by [22]

$$t = \frac{1-r}{1+(N-1)r} \tag{14.26}$$

where r is the reflectance of each surface and N is the number of surfaces. Thus, the reflected portion $(1-t)$ does not contribute to the image formation, it is considered stray light. On the example of the above mentioned zoom lens with 20 air-glass interfaces, the amount of ghost radiation compared to the total radiation passing the lens is 45% for uncoated surfaces and about 17% if the surfaces are anti-reflection coated (1% reflection loss).

Most of this ghost radiation is harmless if it is diffuse enough, i.e. spread uniformly over the entire image area. However, if brought to focus near the image surface, ghost images can be quite intense even in case of anti-reflection (AR) coatings. It is therefore of utmost importance to control not only the amount of ghost (stray) radiation but also its intensity distribution.

$OpTaliX$ provides four types of analyses to study the effects of ghost images.

- Paraxial Analysis: Find the *paraxial* location and apparent diameter of the ghost image with respect to a target surface (typically the image surface, but can be any other surface as well).

- Calculate the spot diagrams based on exact ray trace along the ghost path (including the internal double-reflection).
- Plot a lens layout showing the ghost path.
- Create a photo-realistic image of ghost effects, including effects of anti-reflection coatings and ghost spot distribution.

`GHO SUR si..j`	Ghost surface range. The surfaces `si..j` denote the first and last surface to be included in ghost analysis.
`GHO TAR sk [x_ext, y_ext]`	Target surface at which ghost effects are to be analyzed. The optional parameter `x_ext`, `y_ext` define the extension of the analysis area at the target surface.
`GHO SRC`	Include the effects of the source. That is, the analysis includes the irradiation at the target surface caused by the source itself **plus** the effects caused by ghost radiation. Since the expected intensity differences between direct image and ghost image may be large, logarithmic display is recommended (see GHO LOG command below).
`GHO LOG [Y|N]`	Logarithmic display of ghost intensity. `Y` enables logarithmic display, `N` disables it (i.e. resorts to linear scale). Note the `GHO FLOOR` command below.
`GHO FLOOR i_min`	Defines the lowest intensity level I_{min} that can be displayed in logarithmic display (requires `GHO LOG Y`). I_{min} can be specified as linear or logarithmic value: Negative numbers are considered as $log(I_{min})$, positive numbers as linear value. Examples: `gho floor -3` ! Lowest relative intensity is $10^{-3} = 0.001$, `gho floor 0.001` ! Lowest relative intensity is 0.001
`GHP si..j target_sur [ALL]` `GHO si..j target_sur [ALL]`	Find the *paraxial* location and apparent diameter of the ghost image with respect to a target surface. `si..j` are the first and last surface where ghost reflections take place. The optional parameter ALL lists all possible surface pairs within the surface range `si..j`. The commands `GHP` and `GHO` are functionally equivalent. `GHO` was added for compatibility with Code V. See also the notes on paraxial ghosts below.

continued on next page

continued from previous page	
GHS si..j target_surf GHO SPO si..j target_surf	Calculate the spot diagrams based on exact ray trace along the ghost path. si..j are the first and last surface where ghost reflections take place. The target surface target_surf may be any surface including the image surface.
GHV si..j target_surf GHO VIE si..j target_surf	View lens layout plot including ghost ray trace. si..j are the first and last surface where ghost reflections take place.
GHR si..j target_surf x_rel_aperture y_rel_aperture GHO RAY si..j target_surf x_rel_aperture y_rel_aperture	Trace a single ghost ray. si..j are the first and last surface where ghost reflections take place.
GHO RGB si..j target_surf [ALL] [FILE file_spec]	Calculate an almost photo-realistic RGB-image. si..j are the first and last surface where ghost reflections take place. The optional parameter ALL includes the ghost contributions of all possible surface pairs within the surface range si..j, including coating effects (requires POL Y), transmission effects (requires TRA Y) and the spectral weighting of the system. The optional parameters FILE file_spec allow export of the ghost RGB data to a file specified in file_spec. Two formats currently supported are plain ASCII and Microsoft^{TM} Excel. The file format is derived from the file extension, i.e. a file name test.xls will create an Excel file, whereas any other extension defaults to ASCII. More information about "photo-realistic rendering of ghost effects" is also given on page 251. **Example:** gho rgb s3..7 12 all fil c:\temp\ghostrgb.xls
GHO SAV Y|N	Save ghost analysis parameters along with optical system prescription.

Limitations:
The current implementation of ghost analysis (respectively the underlying inverse ray trace) takes spherical surfaces, aspheric surfaces and decentered and/or tilted surfaces into account. Gradient Index (GRIN) media are also correctly simulated in the inverse ray trace, however, the end surfaces of GRIN elements must be centered.

14.1.17.1 Notes on paraxial ghost analysis:

Ghost analysis based on paraxial calculation provides a very fast means for identifying the most disturbing surface pairs. However, the results of paraxial ghost analysis should be observed with great care, because paraxial analysis does not account for geometrical aberrations along the ghost path. Ghost images, however, are not corrected to produce sharp images. Therefore, the more common case is that ghost images are blurred by large amounts of spherical aberration, coma and field curvature.

It is therefore likely that the effect of ghost images predicted by *paraxial* analysis does not match well with an exact ghost ray trace. Only for optical systems that exhibit small numerical apertures and small fields only, paraxial ghost quantities may reasonably represent real ghost effects. As an example, the paraxial ghost analysis shown below exhibits a relatively small ghost spot for the surface pair 5-7 (that is, first reflection is on surface 7, second reflection is on surface 5). However, the exact ghost ray trace, as shown in Fig. 14.10, reveals a large spread of the rays on the image surface caused by severe (uncorrected) spherical aberration along the ghost path.

Note that the often observed discrepancy between paraxial ghosts and *real* ray trace ghosts is not an implementation fault in $OpTaliX$ but is *only* due to the inherent limitations of paraxial theory (i.e. linear approximation of real world effects).

Thus, be warned NOT to trust paraxial ghost analysis as the sole means of performing ghost analysis, because it is fast, but always cross-check results of paraxial ghost analysis against other methods (for example ghost spot, ghost lens view or ghost RGB-analysis).

```
PARAXIAL GHOST ANALYSIS:

All ghost aberrations are referred to surface  12

  1st.Refl  2nd.Refl         GhostNA     GhostDiam    GhostFocus
       7         5          0.18700       0.89666      -2.39748
```

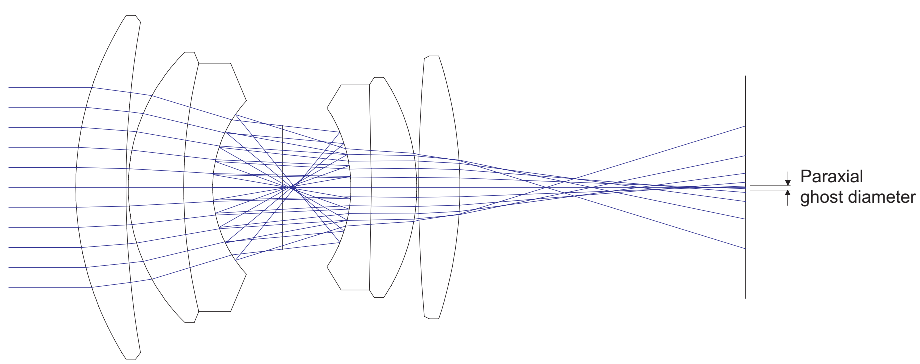


Figure 14.10: Ghost imaging. Note the spread of the rays on the image surface due to (uncorrected) spherical aberration along the ghost path as opposed to the size of the ghost image predicted by paraxial analysis.

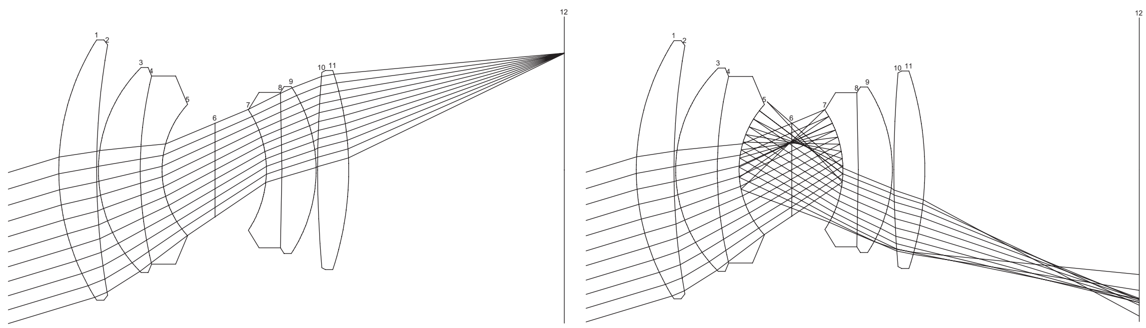
Thus, the user should be aware of the intrinsic limitations of paraxial ghost analysis, which may be appropriate in "slow" systems but may fail in systems with large numerical aperture or systems having a wide field.

Example:
The following example uses a Double-Gauss system (see `$i\examples\misc\double_gauss-2.otx`). First reflection takes place on surface 7, directing the rays backwards. The second reflection takes

place on surface 5, directing the ghost rays back to the image surface. The ghost ray trace is visualized by the command

```
ghv s5..7 12
```

where `s5..7` defines the surface range. The third parameter is the target surface (12). Fig. 14.11 shows the nominal imaging ray trace and the corresponding ghost ray trace for the surface pair 5 and 7. Also note the surface numbers, which are identical for both cases, indicating that extra surfaces (which describe the ghost path) are not required.



Direct imaging path Ghost imaging path

Figure 14.11: Ghost imaging. Left: conventional imaging path, right: ghost imaging path between surfaces 5 and 7.

14.1.17.2 Photo-realistic rendering of Ghost Effects:

The ”`GHO RGB`” option provides the most realistic and accurate ghost analysis. It offers a fully automatic search of ghost effects by evaluating *ALL* possible combinations of surface pairs in a lens which may contribute to ghosts. If enabled, the analysis also includes wavelength dependent effects of multilayer coatings on optical surfaces (”POL yes”), material absorption (”TRA yes”) and vignetting.

The colors in the RGB-plot are approximate to the ’real world’ colour rendition only for systems in the visible spectral range, that is approximately 400 - 700 nm. If other spectral ranges are used (for example ultra-violet or infrared spectral regions), then a ’blue’ colour in the plot only represents a shorter wavelength in that spectral range, respectively a ’red’ colour corresponds to a longer wavelength. In such cases, colors should be considered as ’pseudo’ colors only.
In order to create photo-realistic plots of ghosts, some preparatory work is recommended:

- We define a single object which is considered as the disturbing source, being either inside the specified field of view or outside.
- All surface apertures should be fixed (FHY sa 1) so that ghost rays hitting a surfaces outside its defined aperture are effectively blocked.
- Coatings should be appropriately attached to surfaces (see ATT command) in order to model ghost reflections realistically.
- Polarization and transmission analysis must be enabled (`POL Y`, respectively `TRA Y`) to include effects of coatings in the ghost analysis. `POL` and `TRA` may also be set separately in the

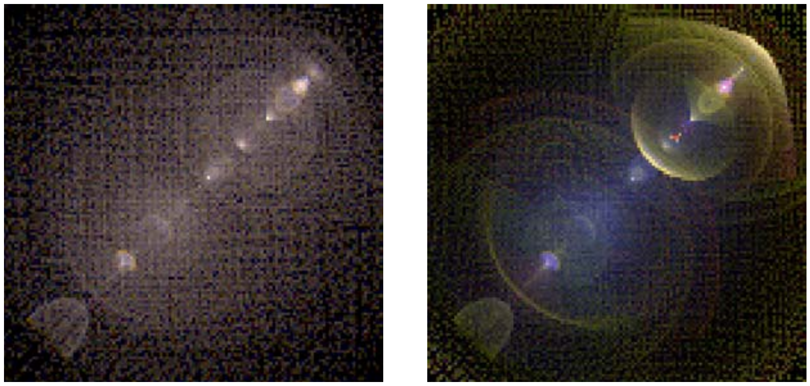


Figure 14.12: Almost photo-realistic rendering of ghost effects as a RGB-image on the example of `$i\examples\high_na\f15_33.otx`. The left image was obtained by ignoring coating or Fresnel reflection effects, whereas the right image is more realistic by including coating effects (`POL Y`, `TRA Y`)

ghost analysis dialog. Note that polarization calculation is computationally intensive, which may slow down the speed of the calculation by an order of magnitude. Therefore, it is sometimes helpful to do a first ghost analysis with `POL` and `TRA` disabled and study the geometrical effects of ghosts only. For a detailed and precise analysis, `POL` and `TRA` should be enabled to include the intensities of ghost images. For the differences of enabled/disabled coatings see Fig. 14.12.

For each pair of ghost surfaces the RGB-ghost analysis outputs the location and the relative intensity of the ghost image. This information helps to identify contributions to the ghost image from particular surface combinations. A typical output from a RGB-ghost analysis would be:

```
Surface sequence:  0 --> 4  --> 3  --> 21
     WL      Rays     X-grav.     Y-grav.         Rel.Int.
   0.55000    187     2.59402     2.19795      0.000000506
   0.43000    186     2.66335     2.25453      0.000003985
   0.62000    187     2.58922     2.19467      0.000000319
Surface sequence:  0 --> 5  --> 1  --> 21
     WL      Rays     X-grav.     Y-grav.         Rel.Int.
   0.55000     97     2.25870     2.26849      0.000000940
   0.43000     95     2.26453     2.25057      0.000001150
   0.62000     97     2.26605     2.27590      0.000000163
Surface sequence:  0 --> 5  --> 2  --> 21
     WL      Rays     X-grav.     Y-grav.         Rel.Int.
   0.55000    145    -2.21976    -2.28230      0.000001083
   0.43000    145    -2.10141    -2.16481      0.000000826
   0.62000    145    -2.25932    -2.32177      0.000000173
```

Output is given for each wavelength defined in the system. The "`X-grav`" and "`Y-grav`" coordinates are the intensity-weighted gravity centers of the ghost image at the target surface. It helps to easier identify the location of a particular ghost in the RGB-image. The relative intensity (`Rel.Int.`) column gives the average intensity of a particular ghost in relation tho the intensity of the light entering the optical system. The `Rel.Int.` column does not give a measure of the ghost irradiance on the target surface.

14.1.17.3 Writing Ghost Data to Files (ASCII or Excel

Irradiance distributions resulting from photo-realistic Ghost RGB (red-green-blue) analyses may also be written to a file. The supported file formats are Excel (.XLS), or ASCII (.TXT or .DAT), whereas the file format is derived from the extension itself.

On export, all channels are written successively into a single file, that is, red, green, blue channels and the composite ”white” channel.

ASCII-Format:
Each channel of the Ghost RGB image is preceded by two comment lines indicating the channel and the maximum ray intensity (max value) in that channel. Data of each channel are then written normalized with respect to the channel maximum intensity. The typical output format is shown below:

```
! red:
! max value =  581.4250488281250
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
! green:
! max value =  406.0000000000000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
! blue:
! max value =  635.9754028320312
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
! white:
! max value =  1623.400390625000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 .....
```

Excel-Format:

Images from ghost RGB analyses may be written to Excel. Each red-green-blue component of the ghost image is then written to a separate sheet in the Excel file. The fourth channel ”white” is a composite of the three RGB channels.

14.1.18 Vignetting Analysis

Vignetting is a reduction in the size of the entrance pupil, for off-axis fields, because several surfaces may limit the transverse extension of the beam. Using this definition there is no vignetting on-axis.

53	4.311024	11.49606	26.86614	31.17717	11.49606	4.311024	4.748032
54	5.748032	12.93307	57.48032	63.22835	12.93307	5.748032	4.311024
55	5.748032	15.80709	177.189	200.1811	15.80709	5.748032	5.748032
56	8.622047	11.93307	43.11024	48.85827	11.93307	8.622047	5.748032
57	7.18504	11.05906	20.11811	24.42913	11.05906	7.18504	5.748032
58	6.18504	10.05906	7.18504	10.05906	10.05906	6.18504	2.437008
59	5.748032	5.748032	5.748032	8.622047	5.748032	5.748032	1.437008
60	2.874016	5.748032	4.311024	8.622047	5.748032	2.874016	1.437008
61	0	0	2.874016	4.311024	0	0	0

Tabelle1 / Tabelle2 / Tabelle3 / **Red** / Green / Blue / White

Bereit

Figure 14.13: The RGB components from ghost images are written to separate tabs in an Excel file, including the composite ”white” channel. The preceding tabs labeled ”Tabelle1” to ”Tabelle3” are dummy sheets and should be ignored.

Vignetting leads to a decrease of the illuminance of the image towards the edge of the field. Also, vignetting is often used in the design stage to have a better control of aberrations.

In $OpTaliX$ vignetting properties of an optical system are *solely* defined by surface apertures which have the ”fixed height” property assigned (see FHY command, page 163). Vignetting analysis is always referred to the first field (F1) in the field list, which, for centered systems, is assumed the axial case. For non-centered systems, i.e. systems which contain decentered/tilted surfaces or have a non-symmetrical field, the reference field must be specified in the first position (F1) of the field list.

Commands:

`VIGP`	Plots vignetting as a function of field. In case of zoom systems, all vignetting is overlayed for all positions in a single plot.
`VIG [fi..j wi..j zi..j]`	Evaluate vignetting numerically at discrete fields `fi..j`, and zoom positions `zi..j`. Vignetting is always integrated and spectrally weighted over wavelengths `wi..j`. Values are returned between 0 (100% vignetting) and 1 (no vignetting). By that definition, it is a measure of relative illumination. If fields are not specified, the maximum field will be used. If zoom positions are not specified, zoom position 1 is used.

14.1.19 Geometric Modulation Transfer Function

Calculates the geometrical approximation of the modulation transfer function (MTF). This analysis is appropriate when the wavefront aberration is large compared with the wavelength. We may then approximate the optical transfer function (MTF) by [34]

$$\hat{H}(f_x, f_y) = \iint_{-\infty}^{+\infty} A(x,y)e^{i}(\Delta x\omega_x + \Delta_y\omega_y)dx, dy \tag{14.27}$$

where

$$\begin{aligned} \omega_x &= 2\pi f_x \\ \omega_y &= 2\pi f_y \end{aligned} \tag{14.28}$$

and Δ_x, Δ_y are the transverse aberrations, f_x, f_y are the spatial frequencies of interest and $A(x, y)$ is the relative amplitude associated to each ray. The geometric aberrations (Δ_x, Δ_y) are obtained from tracing a bundle of rays through the system, rectangularly gridded across the entrance pupil. With this assumption, by dividing the aperture in small squares, the geometrical transfer function may be written as

$$\hat{H}(f_x, f_y) = A(x, y)\left\{\sum_{i=1}^{N} cos(\Delta_x\omega_x + \Delta y\omega_y) + \sum_{i=1}^{N} sin(\Delta_x\omega_x + \Delta_y\omega_y)\right\} \quad (14.29)$$

where the sum is performed for all rays N on a spot diagram. This geometrical approximation is surprisingly accurate when the aberrations are larger than a few wavelength. In very well corrected systems, for example where geometric aberrations are in the order or smaller than the Airy-diameter, the geometric approximation of the MTF yields better results than are physically possible. The diffraction based MTF should be used instead (see section 14.2.1, page 262).

`MTF FRE|FLD|DEF [NUM] GEO`	Geometric MTF. The optional parameters can be specified in any order. Note that polarization effects are ignored for geometrical response calculations. Examples: `MTF FLD GEO` ! geometric MTF vs. field, `MTF GEO FLD NUM` ! only numeric output of geom. MTF vs. field.
`GMTFT [fk zk]`	Tangential geometric MTF at field fk, zoom position zk. For use in optimization, `UGR` and `EVA`luation commands only.
`GMTFS [fk zk]`	Sagittal geometric MTF at field fk, zoom position zk. For use in optimization, `UGR` and `EVA`luation commands only.
`GMTFA [fk zk]`	Average geometric MTF at field fk, zoom position zk. `GMTFA = 0.5(GMTFT + GMTFS)`. For use in optimization, `UGR` and `EVA`luation commands only.

14.1.20 Geometric Point Spread Function (GPSF)

The GPSF analysis is a purely geometric approximation to the image of a point source. Since only ray aberrations are included, diffraction effects are completely ignored. This analysis may be useful in systems where aberrations are large compared to the diffraction limited performance. Use thePSF option (page 265) if diffraction effects shall be taken into account.

This analysis includes spectral weighting (as defined in the system configuration), transmission effects (requires `POL yes` and `TRA yes`) and aperture apodization.

By default, the calculation is performed for all fields and wavelengths defined in the system configuration.

`GPSF zk fi..j wi..j img_size [VIE|CON|FAL|XY] [?]`	Geometric point spread function. This analysis is based on geometric effects only. It is most appropriate where aberrations are large. Use the PSF command (see page 265) to include diffraction effects. `img_size` is the patch size at the image surface. Plot options: `VIE`: perspective plot (wire grid), `FAL`: "false" colour geometric PSF. The intensity of the PSF is coded into a rgb-model. Blue colour represents low intensities, red colour represents high intensities. `CON`: contour plot of geometric PSF `XY`: cross sectional plots (in X- and Y-direction) GPSF traces grids of rays for all fields and wavelengths specified and plots the *relative* intensity in the image plane.
`GNRD num_rays_diam`	Number of rays across diameter for geometric PSF calculations only. Note that `GNRD` is equivalent to NRD, however, it is effective only during GPSF-calculations. Also, `GNRD` does not change `NRD`. Any positive number for `GNRD` is allowed.

Example commands:

`GPSF f2..3 0.05 FAL`	Calculates geometric PSF for fields 2-3. Intensity distribution is shown on a 0.05mm image patch as false-colour coded image.
`GNRD 30`	sets number of rays across diameter for GPSF calculation exclusively.
`GPSF ?`	invokes a dialog box for adjusting parameters prior to calculating GPSF.

14.1.21 Encircled Energy (Geometric)

Calculates the fraction of energy by counting all rays that pass the optical system (i.e. are not vignetted) and hit the image surface within a specified area (defined by its diameter). An evenly-spaced rectangular grid of rays in the entrance pupil (see NRD) is traced to the image surface for specified wavelengths, field and zoom positions. Each ray is assigned an energy proportional to its wavelength weight (WTW), aperture apodization and relative transmission.

`RAD fi..j [wi..j] diam_x [diam_y] [X posx Y posy]`	Fraction of energy contained in an image area defined by `diam_x`, `diam_y`. Solely based on geometrical analysis, diffraction is ignored. For diffraction encircled energy see ECE command (page 275). If `diam_y` is omitted (that is only `diam_x` is specified), the image area is assumed circular. Both values, `diam_x` and `diam_y` must be specified for a rectangular/square area. The center of the image area is assumed to lie at the location of the chief ray coordinates in the image plane, except when the optional parameter set `[X posx Y posy]` is specified (see below). Includes wavelength weight (WTW), transmission and apodization. The optional parameter set `[X posx Y posy]` clamps the specified area at a fixed position (`posx`, `posy`) on the image surface rather than defining the area with respect to the chief ray locations for each field. This way, rays are integrated on the same area for all fields and zoom positions.
`ECG fi..j|zk image_radius [NUM GRV]`	Plots geometric encircled energy. Entirely ray based analysis. Takes into account transmission (see TRA/POL) and apodization effects (see PUI/PUX/PUY), if enabled. Use the `NUM` option to list numerical values. The optional parameter GRV refers analysis to the spot gravity center. If omitted, the chief ray reference at the designated fields, respectively the last setting is used. Two curves are plotted, one for the geometric energy contained in a defined image circle (*encircled* energy) and one contained in a defined square (*ensquared* energy). See also Fig. 14.14 for the expected plot.

Examples:

`RAD f3 0.01 0.02`	! Output geometric encircled energy at field 3 contained in a rectangular area of X = 0.01mm, Y = 0.02mm.
`eva [RAD f3 0.01 0.02]`	! Evaluate geometric encircled energy at field 3 contained in a rectangular area of X = 0.01mm, Y = 0.02mm.
`RAD f1..4 .5 X 0.0 Y 0.0`	! Geometric encircled energy within a circular area of 0.5mm diameter with fixed location at X = 0, Y = 0.
`ECG f1..2 z3 0.1 NUM`	! Plot geometric encircled energy at fields 1-2, zoom position 3, image diameter 0.1mm and report numerical values.

14.1.22 Quadrant Detector Analysis

The quadrant detector analysis (`QUA`) option shows the scanned response of a quadrant detector to the image at each field. As in all geometric analyses, diffraction effects are ignored.

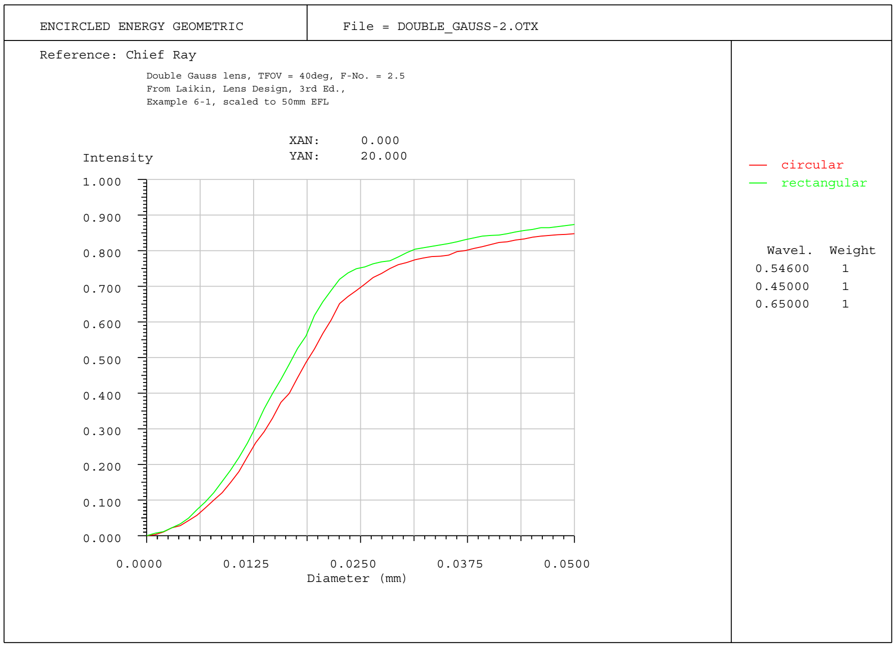


Figure 14.14: Encircled Energy geometric (ECG). Plots the fraction of energy associated to rays that hit a defined circle (or square) at the image surface. Includes transmission and apodization effects.

A quadrant detector is a semiconductor photodiode divided into four sensitive areas. Such devices are typically used to provide alignment information, as determined by comparisons of the illumination levels of opposing quadrants.

The computation lists the scanned response of a simulated quadrant detector to the image at each field point. Scanning is done for both X- and Y-directions. It assumes proper coupling of the quadrants in each half. See Fig. 14.15.

`QUA [STE scan_step_size] [fi..j] [zk]`	Quadrant detector analysis, showing the scanned response of a quadrant detector to the image at fields `fi..j` and zoom position `zk`. Diffraction is ignored.
`QST scan_step_size`	Quadrant step size, in lens units at the image plane.
`QSM smooth_diam`	Gaussian smoothing diameter, in lens units.

Notes:

Quadrant detector analysis is based on the number of rays across the pupil diameter (NRD) and it takes into account apodization and wavelength weights. If enabled (TRA Y and/or POL Y), transmission and polarization effects are also taken into account.

The scanned response may be smoothed by a small spot of Gaussian shape. The diameter of the smoothing Gaussian (`QSM`) is defined at an intensity 50% of the peak intensity.

Description of Output:

In addition to the plot output, a listing is generated for each field activated (see FACT command). The listed output shows the response of the two detector halves in X- and Y-direction as well as the ratio

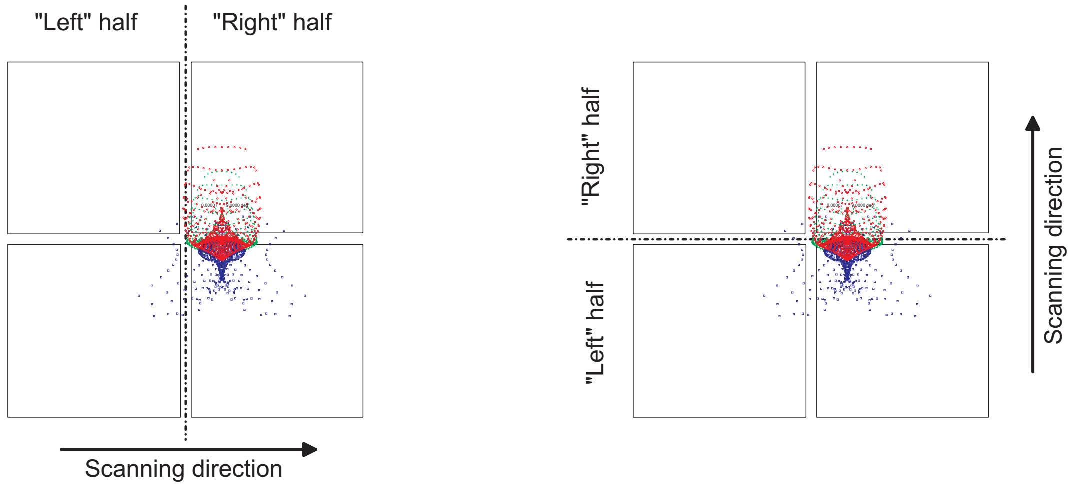


Figure 14.15: Movement of the halves of a quadrant detector across the spot at a given field. Shown are the two scan directions, in X (left) and in Y (right).

of responses from the two halves of the detector as a function of the scan position.

As an example, we will restore the "Double-Gauss" file from the examples library (`$i\optalix\examples\double_gauss.otx`). The settings are `QST 0.02` and `QSM 0.02`. Plot and numerical output are invoked by `QUA f1`.

```
QUADRANT DETECTOR ANALYSIS:

Field :  1      X =       0.00000      Y =       0.00000

     X-Shift      Left Half     Right Half            Ratio
    -0.06000        0.00000        1.00000         0.000000
    -0.04000        0.00250        0.99750         0.002507
    -0.02000        0.02293        0.97707         0.023464
     0.00000        0.47937        0.52063         0.920737
     0.02000        0.97707        0.02293        42.618182
     0.04000        0.99750        0.00250       398.833333
     0.06000        1.00000        0.00000   1000000.000000

     Y-Shift      Left Half     Right Half            Ratio
    -0.06000        0.00000        1.00000         0.000000
    -0.04000        0.00250        0.99750         0.002507
    -0.02000        0.02293        0.97707         0.023464
     0.00000        0.47937        0.52063         0.920737
     0.02000        0.97707        0.02293        42.618182
     0.04000        0.99750        0.00250       398.833333
     0.06000        1.00000        0.00000   1000000.000000
```

14.1.23 Biocular Analysis

The term "biocular" relates to viewing viewing with both eyes simultaneously. This term must not be confused with "bi**n**ocular" systems. In biocular systems, both eyes look through the same optical system, in bi**n**ocular systems, the human eyes look through two identical (mirror symmetrical) optical systems (telescopes), mounted side-by-side. The latter is often denoted as "Feldstecher" (German) or field glasses.

The biocular analysis (BIO) is useful in optical systems that provide an enlarged image of a display and which is observed from different (typically two) eye locations. The BIO option computes chief rays over a grid of viewing angles and displays the differences of the images.

Essentially, a biocular magnifies a small display and presents the enlarged image to the visual system. Biocular systems are viewed through with both eyes simultaneously (as opposed to binocular systems). Other typical applications of the BIO option are head-up-displays (HUD) and simulators.

The BIO option allows analysis of the following parameters:

Convergence: The human eyes are focusing to an object at a distance closer to infinity, that is the simultaneous inward movement of both eyes toward each other.

Divergence: The human eyes are focusing to an object in excess of infinity distance (i.e. a virtual image) and the eyes are forced to simultaneously move outward with respect to each other. This is a situation that the eyes cannot perform and that leads to eye strain and headache.

Dipvergence: The two images observed by the eyes are laterally displaced in vertical directions. Again, this may lead to eye strain and headache.

Biocular FOV: The angular range within observation of the display is possible with both eyes simultaneously. Typically, the FOV seen by the left or right eyes are different and do only partly overlap. The ”biocular FOV” is only the overlapping region.

The locations of the left and right eye are modelled via two zoom positions. The aperture stop is usually at the eye locations in front of the optics and is decentered to model the standard interpupillary eye distance of 64mm. The first zoom position decenters the stop -32mm in X-direction for the left eye while the second zoom position decenters it +32mm in X-direction for the right eye. The stop diameter is set in accordance to the diameter of the eye pupil (typically 5mm). Figure14.16 indicates the preferred condition.

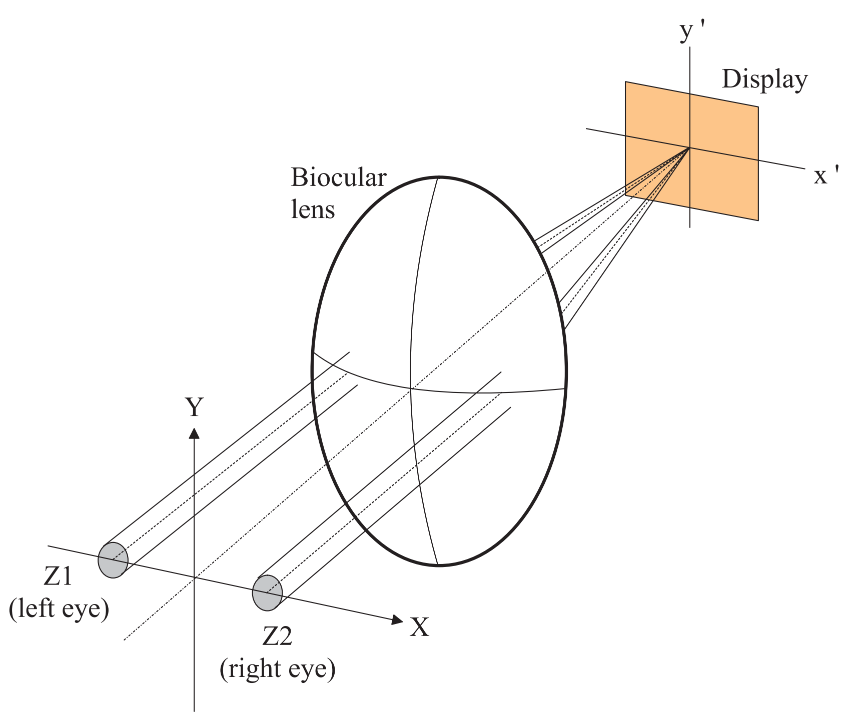


Figure 14.16: Optical setup for biocular analysis. Left and right eyes are modelled by small decentered apertures via two zoom positions.

`BIO FOV|CON|DIP [?|NUM]`	Biocular analysis. `FOV` : Plot biocular field-of-view `CON` : Plot convergence/divergence `DIP` : Plot dipvergence `NUM` : optional parameter, outputs numerical data. Example: `BIO FOV NUM ?` : Plots biocular FOV, outputs numerical data and invokes a dialog box for setting analysis parameters.
`BIO FOVX|FOVY fov_min fov_max`	Defines field-of-view (FOV) in horizontal (FOVX) or vertical (FOVY) direction. Values must be given in degrees. Examples: `BIO FOVX -15 +15` : Defines horizontal FOV from -15 to +15 degrees. `BIO FOVY -12 +12` : Defines vertical FOV from -12 to +12 degrees.
`BIO STPX|STPY step_x step_y`	Angular steps in horizontal (STPX) and vertical (STPY) directions. The total field-of-view is therefore scanned by a rectangular array with FOVX/STPX sampling points horizontally and FOVY/STPY sampling points vertically.
`BIO LEFT|RIGHT zk`	Specifies the zoom/multi-configuration position representing the left or right eye, respectively. Requires that the optical system is a zoom/multi-position system. Example: `BIO LEFT z1` : Position 1 is taken as the left eye. `BIO RIGHT z2` : Position 2 is taken as the right eye.
`BIO FACT scale_factor`	The scale factor converting chief ray differences at the display into angular aberrations. Since `scale_factor` is constant for all viewing angles, a linear (perfect) optical system is assumed as a reference.

Example:

An example system showing the use of the biocular option is found in the examples directory `$i\examples\biocular\biocular_1.otx`, as shown in Fig. 14.17. This is a zoom system with two positions, each position representing the left and right eye respectively. The left eye pupil is decentered at X = -32mm, the right eye pupil is decentered at X = +32mm, giving a total pupil distance of a typical human body of 64mm.

The apparent field of view (FOV) is limited for both eyes because the lens diameters are limited in size (see FHY command) and therefore truncate rays at extreme $\pm X$ viewing directions. This condition does allow a biocular view (i.e. with both eyes simultaneously) only in the central field of view, but not over the full field. The fields, seen by the left and right eye, respectively, are indicated in the field of view (FOV) plot (see `BIO FOV` command, and Fig. 14.18).

The convergence and divergence plots, as shown in Fig. 14.19, indicate the amount of accommodation required by the human eye to get sharp vision at various field points.

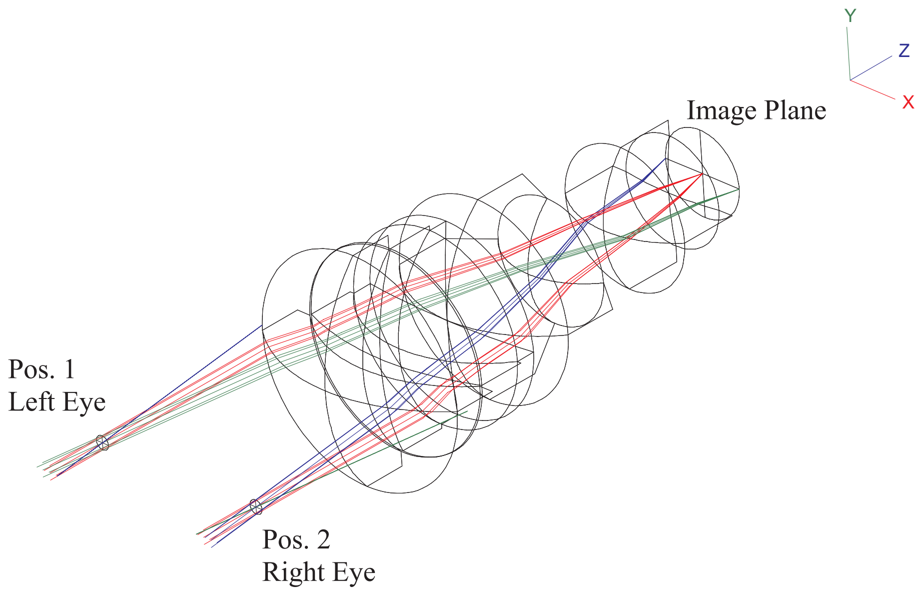


Figure 14.17: Biocular system with left and right eye modelled in a zoom configuration. Here shown as a 3D wire grid view.

14.2 Diffraction Analysis

14.2.1 Diffraction Modulation Transfer Function (MTF)

The diffraction Modulation Transfer Function (MTF) takes into account the extended nature of objects. It is a measure of the accuracy with which different frequency components are reproduced in the image. By default the sine wave MTF is calculated. Note, that the accuracy of the MTF calculation also depends on the density of the ray grid going through the system. Check NRD. The MTF is always calculated for the current zoom position. Use POS command to select a different position.

`MTF FRE|FLD|DEF [NUM]`	Plot Modulation Transfer Function versus: FRE = spatial frequency FLD = fields (default) DEF = defocus The optional parameter `NUM` gives a numerical table instead of a plot.
`MTFA [fi..j wi..j zi]`	Calculates mean value of sagittal and tangential MTF at the specified field points (`fi..j`), wavelengths (`wi..j`) and zoom position `zi`. Produces numerical output only. MTF is computed at spatial frequency defined by the MFR command (see below). The resulting MTF values are in the range between 0 and 1. When used as a function in UGR or optimization, only one field or zoom position can be specified.
	continued on next page

continued from previous page	
`MTFS [fi..j wi..j zi]`	Calculate MTF in sagittal direction at specified field points (`fi..j`), wavelengths (`wi..j`) and zoom position `zi`. Produces numerical output only. MTF is computed at spatial frequency defined by the MFR command (see below). The resulting MTF values are in the range between 0 and 1. When used as a function in UGR or optimization, only one field or zoom position can be specified.
`MTFT [fi..j wi..j zi]`	Calculate MTF in tangential (meridional) direction at specified field points (`fi..j`), wavelengths (`wi..j`) and zoom position `zi`. Produces numerical output only. MTF is computed at spatial frequency defined by the MFR command (see below). The resulting MTF values are in the range between 0 and 1. When used as a function in UGR or optimization, only one field or zoom position can be specified.
`MTF2D [fi|zi|max_freq]`	Plot 2-dimensional MTF at specified field point `fi` and zoom position `zi` for a maximum spatial frequency `max_freq`. MTF2D without any parameter uses field1, zoom position 1.
`MFR max_freq`	Maximum spatial frequency used in MTF analyses vs. spatial frequency. It is given in Lp/mm for focal systems, in Lp/mrad for afocal systems. See also `MFRF` which defines the maximum frequency for MTF vs. field analyses.
`MFRF max_freq_field`	Maximum spatial frequency used in MTF analyses vs. field. It is given in Lp/mm for focal systems, in Lp/mrad for afocal systems. See also `MFR` which defines the maximum frequency for MTF vs. frequency analyses.
`MFRD max_freq_defocus`	Maximum spatial frequency used in MTF analyses vs. defocus. It is given in Lp/mm for focal systems, in Lp/mrad for afocal systems.
`IFR frequency_increment`	Increment in frequency (in Lp/mm for focal systems, in Lp/mrad for afocal systems). The default id `MFR`/20.
`AFR autofocus_frequency` or `MFRA autofocus_frequency`	Spatial frequency used in autofocus option. It is given in Lp/mm for focal systems, in Lp/mrad for afocal systems

The calculation of the modulation transfer function follows the treatment of Malacara [33]

$$\hat{H}(f_x, f_y) = \iint_{-\infty}^{+\infty} \hat{P}(x,y)\hat{P}^*\left(x - \lambda R f_x,\ \ y - \lambda R f_y\right) dx dy \tag{14.30}$$

where R is the reference radius and (f_x, f_y) are the spatial frequencies in either x- or y-direction. Complex quantities are indicated by carets $\hat{}$ on the corresponding symbols. $\hat{P}(x,y)$ is the pupil function defined by

$$\hat{P}(x,y) = A(x,y)e^{ik \cdot W(x,y)} \tag{14.31}$$

where $W(x,y)$ is the wavefront deformation, $A(x,y)$ is the amplitude of the wave and (x,y) are the coordinates in the exit pupil. Thus, the pupil function gives the variation in amplitude and phase across the exit pupil of the system. The phase is deduced from the wavefront aberration and the

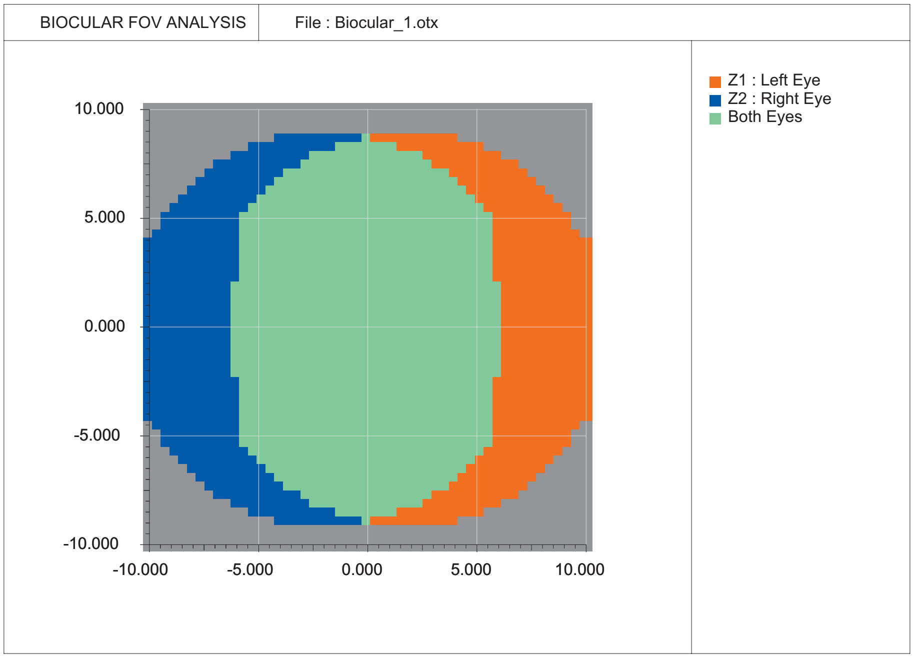


Figure 14.18: Biocular field of view example.

amplitude is derived from the intensity of each ray[2] across the exit pupil of the system. We also note the relation of amplitude and intensity response

$$I(x,y) = [A(x,y)]^2 \tag{14.32}$$

In almost all textbooks on optics, a uniformly illuminated pupil is assumed and since for this condition $A(x,y)$ and $I(x,y)$ are constant (unity) at every point within the aperture, it can be omitted. However, when the transmission property of the pupil is disturbed (e.g. by obstructions of the pupil or by apodization), the amplitude factor will accurately model these effects.

We can now write the integral explicitly

$$\hat{H}(f_x, f_y) = \iint_{-\infty}^{+\infty} \overline{A} \cdot e^{ik \cdot W(x,y)} e^{-ik \cdot W(x-\lambda R f_x,\, y-\lambda R f_y)} dxdy \tag{14.33}$$

with

$$\overline{A} = A(x,y) \cdot A(x - \lambda R f_x,\ y - \lambda R f_x) \tag{14.34}$$

$$k = 2\pi/\lambda \tag{14.35}$$

The integral of equation 14.33, when normalized with respect to its value at $f_x = f_y = 0$, is called the *optical transfer function* (OTF). It represents the convolution of the pupil and the laterally sheared image of it. Thus, the frequency response $\hat{H}(f_x, f_y)$ for incoherent illumination, apart from a constant factor, is the auto-correlation function of the pupil function. The optical transfer function is a complex quantity, its real part is called the modulation transfer function (MTF) , the imaginary part

[2] In this context we mean the apparent intensity of rays passing the system at different pupil coordinates (x, y). Intensity variation across the pupil occurs if the system exhibits varying transmission as a function of pupil coordinate (for example in systems with high numerical aperture) or if the source itself does not emit uniformly over spatial coordinates (e.g. apodization in laser applications).

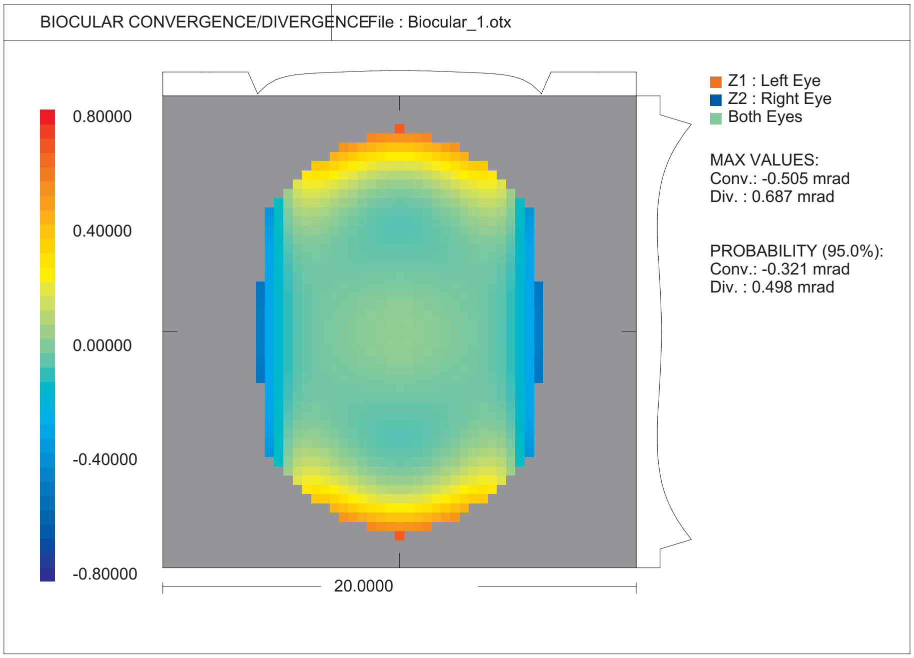


Figure 14.19: Biocular convergence/divergence example.

is the phase transfer function (PTF).

Square Wave MTF : (reserved for future releases)

The square wave response is calculated by resolving the square wave into its Fourier components and taking the sine wave response to each component:

$$S(v) = \frac{4}{\pi}\left[M(v) - \frac{M(3v)}{3} + \frac{M(5v)}{5} - \frac{M(7v)}{7} + ...\right] \tag{14.36}$$

with:

$S(v)$	= square wave MTF
$M(v)$	= sine wave MTF
v	= spatial frequency

14.2.2 Point Spread Function (PSF)

The diffraction point spread function (PSF) describes the intensity of the diffraction image formed by the optical system of a single point source in the object space. The point spread function is computed from the wavefront in the exit pupil of an optical system by a double Fourier integral as given in Eq. 14.39. Aperture obstructions and non-uniform illumination of the aperture (apodization) are correspondingly taken into account. In case of polychromatic analysis, the monochromatic PSF's are integrated over the wavelengths according to the assigned wavelength weights.

The amplitude distribution $A(x, y)$ in the exit pupil and the corresponding wavefront aberration $W(x, y)$ define the complex pupil function $P(x, y)$. The normalized coordinates in the exit pupil are (x, y).

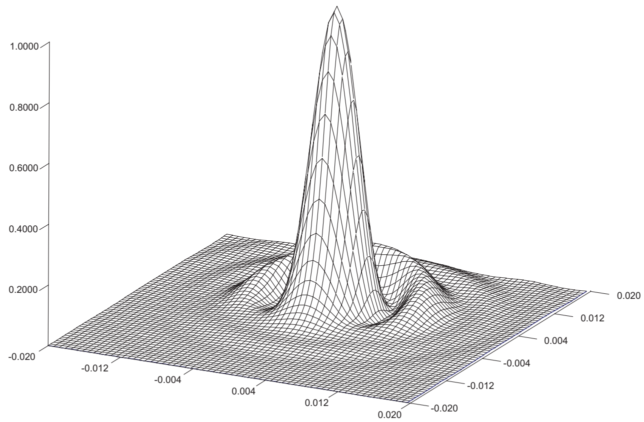


Figure 14.20: Diffraction PSF in perspective view.

$$P(x,y) = \begin{cases} A(x,y)e^{2\pi j \cdot W(x,y)/\lambda} \\ 0 \end{cases} \tag{14.37}$$

The pupil function P is zero outside the pupil. The intensity distribution $I(x,y)$ in the exit pupil is given by

$$I(x,y) = [A(x,y)]^2 \tag{14.38}$$

The diffracted irradiance $|h(u,v)|^2$ of a point-source object in the image plane with coordinates (u,v) is well approximated by

$$|h(u,v)|^2 = \frac{\left[\iint_{-\infty}^{+\infty} P(x,y)e^{-2\pi j(x \cdot u + y \cdot v)}\, dx\, dy\right]^2}{\left[\iint_{-\infty}^{+\infty} P(x,y)\, dx\, dy\right]^2} \tag{14.39}$$

14.2.2.1 Patch Size

A Fast Fourier Transform (FFT) is used to compute the integral in Eq. 14.39. Due to the unit transformation properties of the Fourier Transform, there is a relation between the sampling in the exit pupil of the optical system (defined by the ray grid, see `NRD` command) and the sampling period in the image plane. Thus, the computed area in the image plane is a function of three parameters, the sampling period in the exit pupil, the reference wavelength and the numerical aperture of the optical system. The default sampling in the exit pupil is a grid of 32 x 32 rays (`NRD` = 32). The maximum patch size in the image plane which can be calculated is then determined by

$$x_{image} = \frac{\lambda N_p}{2 \cdot sin(u')} \tag{14.40}$$

with :

λ = wavelength in μm
u' = numerical aperture in image space
N_p = number of sampling points across pupil (see NRD command)

If necessary, the maximum allowed patch size can be increased by increasing NRD (number of rays across diameter). Image patches smaller than the default value (i.e. calculated by Eq. 14.40) can be freely specified.

Another technique for the computation of the PSF is the direct integration of the complex pupil function (Huygens). This method allows direct specification of the image patch, however, it is computing intensive. It is therefore only available for the cross sectional PSF in two orthogonal sections (see `PSF XY` command).

`PSF fk [zk] [VIE|GRY|CON|XY|ZOO| norm|log] [img_size]`	Calculate and plot diffraction point spread function (PSF). The parameter are: `PSF VIE`: perspective plot of the PSF `PSF GRY`: gray level plot of PSF `PSF TRU`: pseudo true-colour plot of PSF. The colour components, contributing to the polychromatic PSF are coded into a rgb-model to give an impression of chromatic aberrations in the PSF. `PSF FAL`: "false" colour PSF. The intensity of the PSF is coded into a rgb-model. Blue colour represents low intensities, red colour represents high intensities. `PSF CON`: contour plot of PSF `PSF XY`: cross sectional plots (in X- and Y-direction) `PSF ZOO`: zoom (resample) the PSF to a desired image area. `norm`: can be used in conjunction with `PSF VIE` and normalizes the PSF to unity, independent of the actual value of the Strehl-ratio. `log` : plots the PSF on a logarithmic scale. `img_size`: Size of the image patch. See sect. 14.2.2.1 for restrictions on patch size.
`PSF FF patch [FIL file_name] [?]`	Full field PSF. The gray-scale PSF is computed at nine discrete field points within the maximum field. See also section 14.2.5 for a detailed description.
`PSF DF [img_size | fi..j]`	Diagonal field PSF. The PSF is computed at all specified field points and displayed in a single gray-coded bitmap. See also section 14.2.4 and Fig. 14.21.
`PSF fk [zk] [img_size] FIL file_name`	Write PSF intensity data to file `file_name`. The file written is a ASCII-file with 4*`NRD` columns and rows.

14.2.2.2 Exporting PSF-Data

Intensity distributions resulting from point spread function (PSF) calculations may also be written to a file. The file format is plain ASCII as described in sect. 32.12.

`PSF fk [zk] [img_size] FIL file_name`	Write PSF intensity data to file `file_name`. The file format is either ASCII or Excel, defined by the file extension (`*.txt` or `*.dat` for ASCII, `*.xls` for Excel). The number of rows and columns is 4*`NRD`, i.e. NRD 32 will write a 128 x 128 matrix. See sect. 32.12 for a description of the file format.

14.2.3 PSF Diameter in X and Y, Ellipticity

The diameter of a PSF can be calculated along two slices, in x- and y-direction. The intensity level at which the diameter is calculated can be freely defined. By default this level is at the $1/e^2$ intensity.

`PSDX fi zi [threshold]`	Calculates diameter of PSF in X-direction for a given field `fi` and zoom position `zi`. The diameter is determinated at a certain intensity level, defined by the `threshold`, a value between 0 and 1. `threshold` is optional with a default value = $1/e^2$ = 0.135. Example: `psdx f1 z3 0.135` Calculate the diameter of the PSD in X-direction at an intensity treshold of 0.135
`PSDY fi zi [threshold]`	Calculates diameter of PSF in Y-direction for a given field `fi` and zoom position `zi`. The diameter is determinated at a certain intensity level, defined by the `threshold`, a value between 0 and 1. `threshold` is optional with a default value = $1/e^2$ = 0.135.
`PSE fi zi [threshold]`	Calculates the ellipticity of the PSF for a given field `fi` and zoom position `zi`. The ellipticity of the PSF is defined as the ratio of x-diameter (PSDX) and y-diameter (PSDY) at the (optional) `threshold`-intensity, a value between 0 and 1. The PSE of a perfect (round) PSF is 1.

14.2.4 Diagonal Field PSF

It is sometimes desirable to simultaneously show the dependency of the PSF over the whole field of view instead for a single object point only, as (for example) provided by the `PSF GRY` command. To accomplish this, the PSF is computed at all field points specified in the field configuration (page 42) and displayed in a single bitmap image. Usually, for rotationally symmetric systems, fields are selected from the axis (center of field) to the maximum field, the diagonal of the x- and y-fields. Hence the name *diagonal-field* PSF. However, this option is also well suited for analysis of non-rotationally symmetric systems if the field points are appropriately specified in x- and y-directions.

`PSF DF [patch] [FIL file_name] [?]`	Calculates PSF at discrete field points arranged along the diagonal of the full-field circle. The patch size (`patch`) is the area at the image plane. If omitted or 0, `patch` is calculated automatically on the basis of NRD, wavelength and numerical aperture of the system. The resulting bitmap image may be saved to a bitmap file where the file extension defines the file format. For example, `*.bmp` = Windows bitmap, `*.pcx` = ZSoft PC Paintbrush, `*.png` = Portable Network Graphics. Example: `psf df 0.05 fil c:\psf.bmp`

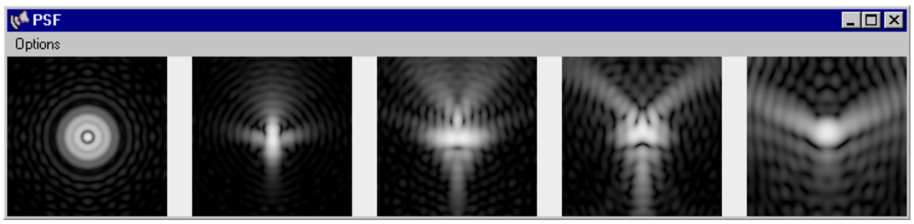


Figure 14.21: Diagonal field PSF as gray-coded bitmaps, using the `PSF DF` command.

14.2.5 Grid Field PSF

This option calculates the diffraction PSF at discrete field points arranged in a grid and displays the resulting PSF's as gray-scale images in a single bitmap image. Note, however, that the PSF's are always calculated including all wavelengths and corresponding weights whereas only the display is gray-scale.

This calculation takes the maximum (full) field circle as defined in the field configuration (page 42) and fits a square grid of field points into this circle. The field can be divided into grids defined by 3x3 and 5x5 field points. Currently the grid numbers can only be defined from the option dialog (use `PSF FF ?` command to invoke the dialog).

`PSF FF [patch] [FIL file_name] [?]`
or
`PSF GRD [patch] [FIL file_name] [?]`
	Calculates PSF at discrete field points arranged in a grid enclosed in the full-field circle. The patch size (`patch`) is the area at the image plane. If omitted or 0, `patch` is calculated automatically on the basis of NRD, wavelength and numerical aperture of the system.
	The resulting grid image may be saved to a bitmap file where the file extension defines the file format. For example, `*.bmp` = Windows bitmap, `*.pcx` = ZSoft PC Paintbrush, `*.png` = Portable Network Graphics.
	Example:
	`psf grd 0.05 fil c:\psf.bmp`

Example Commands:

`psf ff`	Calculates PSF's on a 3x3 field grid with automatic scaling of image area (patch).
`psf ff ?`	Invokes a dialog box for editing parameters
`psf ff 0.05`	Calculates PSF's on a 3x3 field grid with fixed scaling (0.05mm) of image area (patch)
`psf ff 0.05 fil 'c:\temp\psf.bmp'`	Calculates PSF's on a 3x3 field grid with fixed scaling (0.05mm) of image area and writes the grid image to file `c:\temp\psf.bmp`. Note that the apostrophes are only required in case of blanks in file name or folder name.

Two examples of a full-field (grid) PSF are given in Fig. 14.22

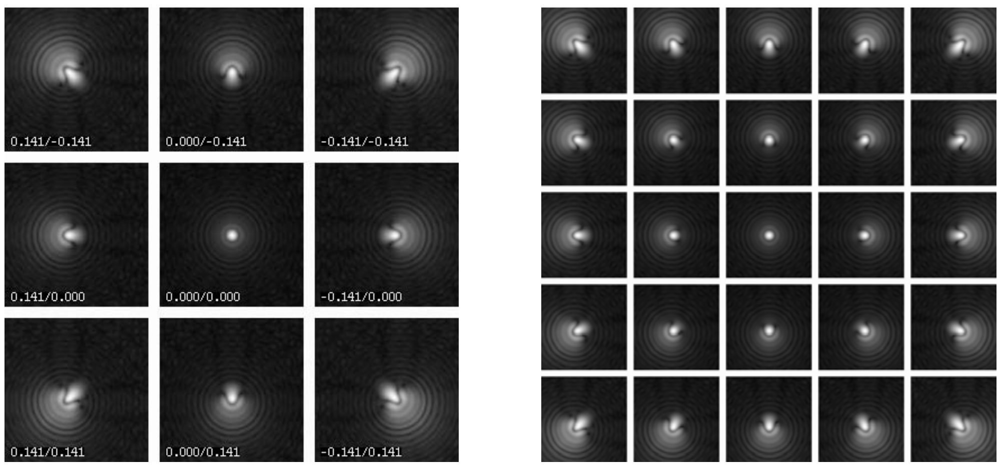


Figure 14.22: Full field (grid) diffraction PSF. Left: 3x3 field grid with X/Y field coordinates, right: 5x5 field grid.

Rendering of the PSF-images at each grid point may be reversed in case of systems with intermediate images. This option is currently only available via the option dialog box (i.e use command `PSF GRD ?`).

14.2.6 X/Y Cross Sections of PSF

Plots cross sections of the PSF in both X-section (sagittal) and Y-section (tangential) for each field specified. The PSF is referred to the coordinates of the chief ray at the reference wavelength. For afocal systems (see AFO YES), units are measured in milli-radians (mrad).

14.2.7 Extended Objects (Fourier Method)

This section deals with image analysis of spatially coherent and spatially incoherent objects of finite extension. It is based on Fourier theory and accounts for the limited frequency response, aberrations and diffraction effects of real optical systems on image formation. The user should be familiar with Fourier Optics (see for example the excellent book by J.Goodman, Ref. [17]) before meaningful conclusions can be drawn from this analysis.

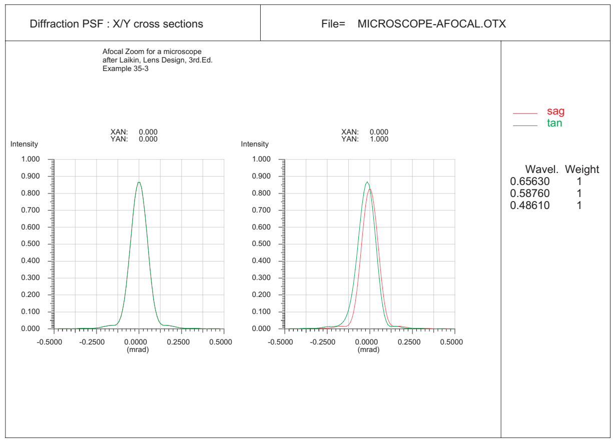


Figure 14.23: X/Y cross section of PSF

When we speak of extended objects, or alternatively and equivalently of extended images, the spatial extension of the object area must be small so that the optical transfer function (OTF) of the optical systems does not change noticeably. Thus, for a selected field point the object of interest must be confined to the region for which the OTF remains stable.

`EIMD fk wk obj_type ext_x ext_y | ? | fil bitmap_file`

Extended object/image, based on diffraction analysis. Uses Fourier techniques to calculate the image of an extended object at field number `fk` and wavelength number `wk`.

`obj_type` specifies the object type from a set of predefined objects, which can be

`CIR` = top hat, circular
`ELL` = top hat, elliptical
`REC` = top hat, rectangular, (`ext_x`, `ext_y` define the width in X/Y-direction)
`GAU` = Gaussian profile, (`ext_x`, `ext_y` define the $1/e^2$ diameters)
`GRA` = grating, (`ext_x` defines the grating period)
`PIN` = double pinhole, (`ext_x`, `ext_y` define the pinhole X/Y-separations)

`?` The question mark is optional and invokes a dialog box for editing parameters.

`fil bitmap_file` specifies a RGB-bitmap file as object. Supported file formats are BMP, PCX, PNG and INT. The physical extensions of the bitmap (`ext_x, ext_y`) must always be smaller than the maximum allowed object extension (see also Fig. 14.24 and the discussion below. Otherwise increase NRD).

Examples:
`eimd f3 w2 rec 0.1 0.05` Calculates imaging of a rectangular object (width = 0.1mm, height = 0.05mm) at field number 3 and wavelength number 2.
`eimd f3 w2 fil c:\mybitmap.bmp` A bitmap is used as object.
`eimd ?` Invokes a dialog box for editing all parameters.

The extended images calculated by this option may also be exported to files. Currently the INT-format (see section 32.11) and a "raw" format are available. The data in the "raw" file span the numerical range between 0 and 1. Export to INT or "raw" files, however, is only possible from the option dialog of an extended image window.

Since the algorithm used for calculating the extended image is based on Fast Fourier Transforms (FFT), the physical size of the object array respectively the maximum allowed size of the extended object x_{object} cannot be freely chosen. Due to the unit transformation of the Fourier Transform, the sampling in the exit pupil (see NRD command) and the sampling in the object/image plane are closely related. Thus, the maximum extension x of the object/image area is defined by the number of sampling points in the pupil (N_p = NRD), the wavelength used and the numerical aperture ($sin(u)$).

$$
\begin{aligned}
x_{max.object} &= \frac{\lambda N_p}{2 \cdot sin(u)} \\
x_{max.image} &= \frac{\lambda N_p}{2 \cdot sin(u')}
\end{aligned} \tag{14.41}
$$

Therefore, a denser aperture sampling (larger NRD) must be chosen to increase the maximum allowed object/image patch.

The object extensions must not be confused with the maximum array extensions, which are defined by Eqs. 14.41. Fig. 14.24 shows the definition of object extensions, which must always be smaller than the array dimensions, independently whether the structure is given in the object space or in the image space.

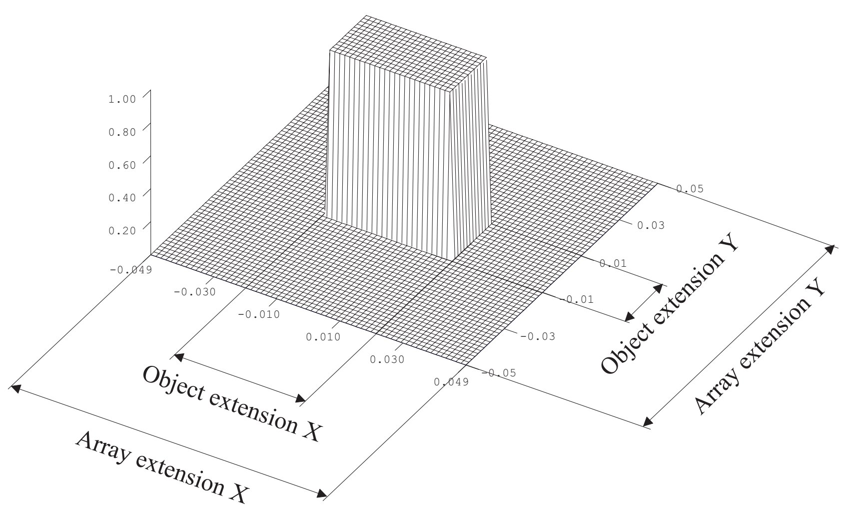


Figure 14.24: Extended object, definition of object extensions and array extensions.

Theory:

To analyse the imaging properties of extended objects (extended images) several assumptions are made. All imaging elements of an optical systems are combined in a single "black box" whose optical interfaces consist of the planes containing the entrance and exit pupils (see Fig. 14.25). It is furthermore assumed that the passage of light between the entrance and exit pupils is completely described by geometrical optics (i.e. using rays).

All diffraction effects are associated with either of these pupils and diffraction which might occur inside the optical system (the black box) is ignored. This point of view is the major difference to the physical optics beam propagation approach (see chapter 16, page 311), which does account for these effects, however, at the expense of increased computing overhead.

In describing the underlying theory of extended source imaging we shall follow the excellent description of Fourier optics by Goodman [17]. In this section only a condensed summary is given. The reader interested in a more complete treatment may wish to consult Goodman's book.

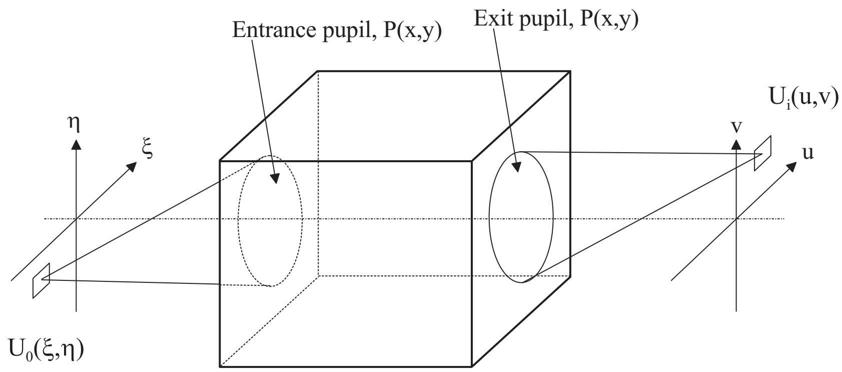


Figure 14.25: Generalized black-box model of an optical system.

The image amplitude $U_i(u, v)$ is represented by the superposition integral

$$U_i(u, v) = \iint_{-\infty}^{\infty} h(u, v) U_0(\xi, \eta) d\xi d\eta \tag{14.42}$$

where $h(u, v)$ is the (complex) amplitude in the image plane in response to a point-source object at coordinates (ξ, η) and $U_0(\xi, \eta)$ is the amplitude distribution of the object. For an ideal (diffraction limited) system, h is simply the Fraunhofer diffraction pattern of the exit pupil, centered at coordinates $u = m \cdot \xi$, $v = m \cdot \eta$ where m is the magnification. See also section 14.2.2, in particular Eq. 14.39, for computation of h.

In the general case, for an aberrated system, we can regard the image as being a convolution of the image predicted by geometrical optics with an impulse response that is the Fraunhofer diffraction pattern of an aperture with amplitude transmittance P, where P is defined as

$$P(x, y) = A(x, y) e^{jkW(x,y)} \tag{14.43}$$

$W(x, y)$ is the wavefront aberration as predicted by the optical path difference (OPD) with respect to a reference sphere and $A(x, y)$ is the relative amplitude in the exit pupil. Eq. 14.43 is equivalent to the optical transfer function (OTF) for the coherent case.

Using Fourier optics, we define the frequency spectra of the components

$$G_0(f_x, f_y) = \iint_{-\infty}^{\infty} U_0(u, v) e^{-2\pi j(f_x u + f_y v)} du dv \tag{14.44}$$

$$G_i(f_x, f_y) = \iint_{-\infty}^{\infty} U_i(u, v) e^{-2\pi j(f_x u + f_y v)} du dv \tag{14.45}$$

$$H(f_x, f_y) = \iint_{-\infty}^{\infty} h(u, v) e^{-2\pi j(f_x u + f_y v)} du dv \tag{14.46}$$

Applying the convolution theorem, it follows directly that

$$G_i(f_x, f_y) = H(f_x, f_y) G_0(f_x, f_y) \tag{14.47}$$

where we have expressed the effects of imaging in the frequency domain.

The coherent case:

For coherent imaging, the optical transfer function $H(f_x, f_y)$ can be directly related with the amplitude transmittance P

$$H(f_x, f_y) = P(\lambda z_i f_x, \lambda z_i f_y) \tag{14.48}$$

where z_i is the distance from the exit pupil to the image plane.

The incoherent case:

$$H(f_x, f_y) = \frac{\mathcal{F}|h(u,v)|^2}{\iint |h(u,v)|^2 du dv} \tag{14.49}$$

which is equivalent to Eq. 14.46, except that the phase information of the complex amplitude of the point-source image is rejected. H now specifies the complex weighting factor applied by the system to the frequency component at (f_x, f_y). Note that the modulus $|H|$ is known as the modulation transfer function (MTF). See also section 14.2.1, where the autocorrelation method is used to calculate $|H|$.

Operator Notation:

Both coherent and incoherent imaging can also be expressed in operator notation, where $\mathcal{F}$ denotes Fourier Transform and $\mathcal{F}^{-1}$ denotes the inverse Fourier Transform.

Coherent case:

$$U_i(u,v) = \mathcal{F}^{-1}\left[\mathcal{F}\left[U_0(\xi,\eta)\right] P(x,y)\right] \tag{14.50}$$

Incoherent case, without explicit notation of the normalization integral in Eq.14.49:

$$U_i(u,v) = \mathcal{F}^{-1}\left[\mathcal{F}\left[U_0(\xi,\eta)\right] \mathcal{F}^{-1}\left[|h(u,v)|^2\right]\right] \tag{14.51}$$

14.2.8 Knife Edge Function (KEF)

The knife edge function, also called ”edge spread function” or ”slant edge function”, calculates the response of a ”sharp edge” in the image plane.

`KEFS fk`	Compute the width of the knife edge function (KEF) in the sagittal direction at field `fk`. Typically, the width is defined by the 10% and 90% intensity points of the KEF. See the KEFL and KEFH commands to set the intensity levels of the KEF.
`KEFT fk`	Compute the width of the knife edge function (KEF) in the tangential direction at field `fk`.
`KEFL I_low I_high`	Set the low and high intensity levels for calculating the width of the knife edge function. The levels must be entered in percent (%). Default values are I_low = 10, I_high = 90.
`KEFH I_high`	Set the high intensity level for calculating the width of the knife edge function. The level must be entered in percent (%). Default value is I_high = 90.
`PLO KEF fk [?]`	Plot knife edge function (KEF) in sagittal and tangential directions at field fk. The optional question mark invokes a dialog box for editing plotting and calculation parameters. Specify a field number `fk`, otherwise the field from a previous calculation will be used (default `fk` = 1).

14.2.9 Encircled / Ensquared Energy (Diffraction based)

The encircled energy is the fraction of total energy in the point image enclosed within a circle or square of a given size. This type of analysis is particularly useful on a detector array with square pixels to determine which fraction of total energy is contained within the size of one pixel.

Encircled/ensquared energy calculations are based on integration of the diffraction point spread function (PSF) referred to the centroid of the diffraction PSF.

The accuracy of the calculation depends on the ray grid (see NRD, number of rays across diameter). The larger NRD (i.e. the denser the rays in the pupil are) the more accurate results can be obtained.

`ECE fk diam`	Compute encircled energy within a diameter (`diam`) at field `fk`. Calculation is referred to the center of gravity of the PSF function. See also the EQE command below.
`EQE fk diam`	Compute ensquared energy within a diameter (`diam`) at field fk. Calculation is referred to the center of gravity of the PSF function.
`PLO ECE|EQE fk diam [NUM] [?]`	Plot encircled or ensquared energy within diameter (diam) at field fk. Calculation is referred to the center of gravity of the PSF function. The optional question mark invokes a dialog box for editing plotting and calculation parameters. Specify a field number `fk`, otherwise the field from a previous `PSF` or `ECE` calculation will be used (default `fk` = 1). The parameter `NUM` outputs encircled/ensquared energy data numerically in the text window. Two curves will be plotted for *encircled* energy and *ensquared* energy separately. The ensquared energy curve is always higher than the encircled energy curve.

Notes:
The encircled/ensquared energy is computed from the diffraction point spread function (PSF). First, the center of gravity of the PSF function is searched and from that point integration over the diameter is started. In case of non-symmetric PSF-distributions, however, the center of gravity will not be in the center of the computational FFT-grid and the integration range may be smaller than computed in the FFT-grid. The corresponding encircled energy plot will then report a smaller integration range than requested.

14.2.10 Strehl Ratio

The Strehl ratio (also called Strehl definition) is the ratio of the peak value of the PSF to the peak of the PSF for an equivalent ideal (*unaberrated*) system. The Strehl ratio is a number between 0 and 1, where a Strehl ratio 1 corresponds to the ideal system.

`STREHL [zi..j fi..j wi..j]`	Numerical output of Strehl ratio for zoom positions `zi..j` fields `fi..j` and wavelengths `wi..j`
`PLO STREHL FLD`	Plot Strehl ratio vs. field
`PLO STREHL LAM [y-min y-max]`	Plot Strehl ratio vs. wavelength. The Y-plot range can be adjusted by the optional parameters `y-min`, `y-max` (range 0 - 1).

The Strehl ratio is computed from the complex pupil function $P(x, y)$ by

$$STREHL = \frac{\left[\iint P(x,y)\;dx\;dy\right]^2}{\left[\iint A(x,y)\;dx\;dy\right]^2} \tag{14.52}$$

where the integration takes place over the exit pupil with coordinates (x, y). $A(x, y)$ is the amplitude distribution in the exit pupil as defined in Eq. 14.37.

It is interesting to note that for systems with small aberrations the Strehl ratio is directly related to the variance of the wavefront $(\triangle W)^2$

$$STREHL \;\sim\; 1 - \left(\frac{2\pi}{\lambda}\right)^2 (\triangle W)^2 \tag{14.53}$$

14.2.11 Wavefront Aberration (Optical Path Difference)

The *wavefront aberration* (or *optical path difference*) is the departure of the actual wavefront from the reference sphere. The reference sphere has its center of curvature at the geometrically perfect point image. There is some freedom in choosing the radius of the reference sphere. By default, $OpTaliX$ locates the reference sphere in the exit pupil of the optical system. For the purpose of calculating the wavefront, the center of the reference sphere is always at the location of the chief ray in the image plane. Note, that in other diffraction calculations (e.g. MTF) the minimum variance of the wavefront for all wavelengths is chosen.

Wavefront calculations always include phase changes introduced by coatings on optical surfaces, if applied. This effect is normally small, however, may noticeable affect wavefront on systems with steep incidence angles (e.g. wide-field systems or high numerical aperture systems). See also section 20.6.

WAV [TLT] [fi..j | wi..j | zi..j]	Evaluate RMS wavefront aberration at fields fi..j, wavelengths wi..j or zoom positions zi..j. Output is given numerically. By default wavefront tilt is not subtracted. The TLT option, however, allows subtraction of wavefront tilt.
WAVZ [fi..j | wi..j | zi..j]	Evaluates RMS wavefront aberration as in the WAV command given above, however, allows subtraction of Zernike wavefront components like defocus, astigmatism, etc. Any order of Zernike terms is permitted. Use the ZWACT command (page 145 to define the Zernike terms to be subtracted prior to evaluating RMS wavefront aberration. Numerical output only.
PLO WAV [FLD | LAM] [TLT] [zk]	Plot wavefront aberration vs. field (FLD) or wavelength (LAM). The default is FLD. A plot scale (in microns) is queried in a dialog box. Choosing plot scale 0 will automatically adjust the scale to the maximum wavefront aberration at each field/wavelength/zoom position. By default wavefront tilt is not subtracted. The TLT option, however, allows subtraction of wavefront tilt.
OPD [fi..j | wi..j | zi..j] rel_apeX rel_apeY	Optical path difference (in mm) along a single ray, referred to the chief ray.
OPDW [fi..j | wi..j | zi..j] rel_apeX rel_apeY	Optical path difference along a single ray, expressed in wave units at the reference wavelength.

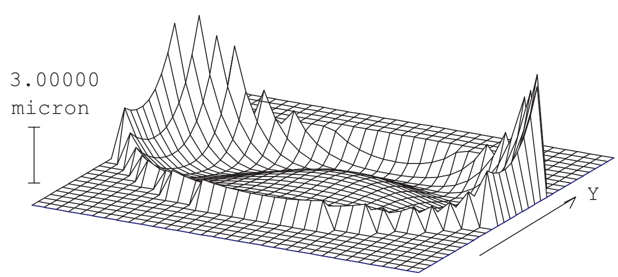


Figure 14.26: Wavefront aberration, shown for one discrete field point.

14.2.12 Conrady D-d Chromatic Aberration

DMD [fk|fi..j wk|wi..j] x_ape y_ape	Conrady D-d chromatic aberration expressed as wavelength difference at given wavelengths. Uses spectral weights as defined in the system configuration (page 46). A detailed description of Conrady's D-d chromatic aberration is given in section 14.2.12, page 277).

In achromats or apochromats, correcting the axial chromatic aberration for paraxial rays (for example see SSR command) does not mean that the longitudinal (axial) aberration vs. wavelength is also corrected for marginal rays. The variation of spherical aberration with wavelength is called chromatic spherical aberration, or spherochromatism.

The Conrady method [9] of controlling spherochromatism is defined as

$$DMD = \sum_{i=0}^{k} (D - d) \cdot (n_F - n_C) \tag{14.54}$$

where D is the optical path of a ray through the aperture center and d is the optical path for a marginal ray. Often the best choice is to correct the chromatic aberration at an aperture height `y_ape` = 0.7.

14.2.13 Single-Path Interferogram

Simulates an interferogram as it is expected from the wavefront deformation in a typical interferometer setup. Note that this analysis does not simulate a "true" two-path interferometric setup where two wavefronts physically interfere. It merely relates the optical path difference (wavefront) to the reference wavelength and displays the amount of constructive/destructive interference. Simulation of interferometric setups with two paths (arms) is discussed in the next section (dual-path interferogram).

The analysis accounts for vignetting and special apertures (central obstructions, spider, etc.). A tilt of the (interferometer) reference plane may be introduced to control the orientation of the fringes.

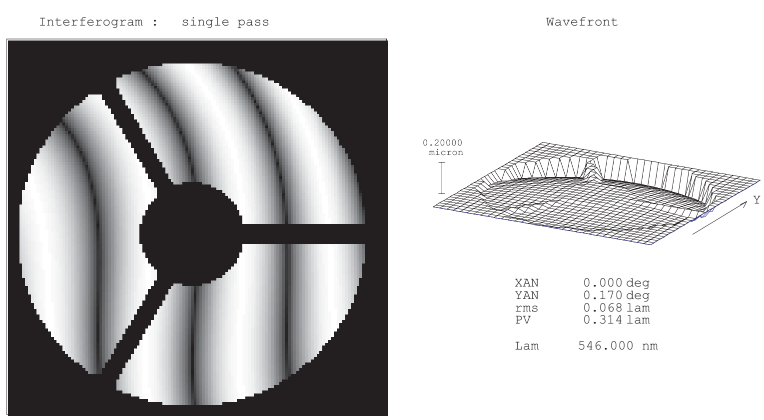


Figure 14.27: Interferogram, computed from wavefront aberration at one discrete field point.

Command syntax:

`IFG field_number`	Compute the interferogram from the wavefront deformation at the reference wavelength.

14.2.14 Dual-Path Interferogram

This option calculates the wavefront of two separate configurations and superimposes it according to the law of constructive/destructive interference. The output signal is therefore similar to that seen in a typical interferometer.

`INT2P sk [?]`	Two-path interferogram. Traces two paths in an interferometric setup and superimposes the resulting wavefronts. Based on constructive/destructive interference of the two wavefronts, the interferometer output is simulated and displayed. The two paths must be defined in a zoom/multi-configuration setup containing at least two positions. `sk` is the target surface at which the superposition of the two wavefront is analyzed. The optional question mark opens a dialog box for editing more parameters.

Simulation of two separate paths in an interferometer requires a zoom/multi-configuration setup with at least two positions. An example (Mach-Zehnder interferometer) is shown in Fig. 14.28. See also the examples library in the interferometer section.

The aperture of the target surface defines the area over which the interferogram is constructed. The aperture extension (e.g. `CIR`, or `REY`, or `ELY`) of the target surface should be at least the size of the expected beams to cover the full interferogram.

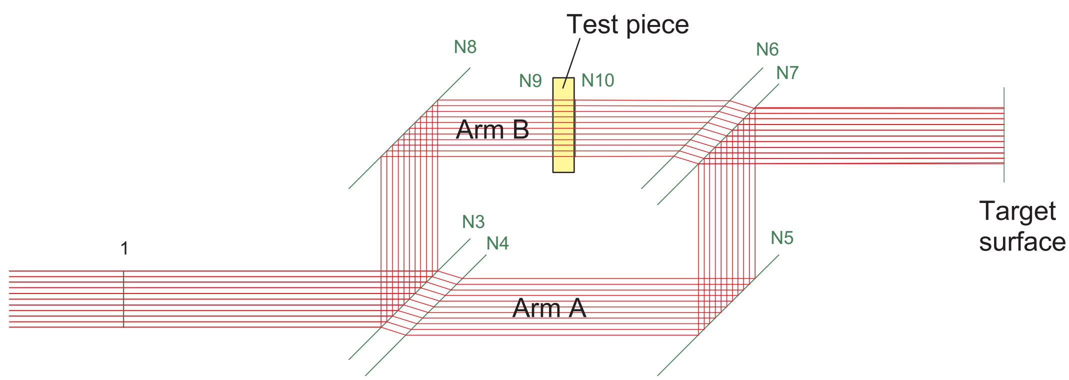


Figure 14.28: Example of a Mach-Zehnder interferometer with a test piece in arm B. Non-sequential surfaces and zoom configurations create the interferometer. The surface error of the test piece is described by a Zernike deformation.

14.3 Gaussian Beams

Gaussian beams, such as the laser beam, are highly directional and have a spatially non-uniform (radially symmetric) intensity distribution. Its Fourier transform is also a Gaussian and it remains Gaussian at every point along its path of propagation through the optical system. The Gaussian has no obvious boundaries, so the commonly agreed definition of the size of Gaussian is the radius at which the intensity has decreased to $1/e^2$ of its value on the axis.

`BEA [wi..j	zi..j |?]`	
	Gaussian beam analysis at wavelength numbers `i..j`, zoom positions `i..j`. The reference wavelength is used if no wavelength range (`wi..j`) is given. The input beam has a gaussian intensity profile and starts at the object surface, i.e. the waist of the beam is assumed at the object surface. Analysis requires proper setting of waist size (see `WRX, WRY` below). The optional question mark invokes a dialog box for editing of `WRX, WRY, ZWX, ZWY, RCX, RCY` and `M2`.
`WRX x_rad [sk |wi..j|zi..j]`	
	Waist radius (in mm) in X-direction at object surface, respectively relative to surface `sk` at zoom position `zi..j`|`zk` and wavelength(s) `wi..j`|`wk`. Only one parameter may be given in a command, either `x_rad` or `sk`|`zk`|`wk`. The optional surface parameters `si..j`|`sk`, `zi..j`|`zk` and `wi..j`|`wk` (without `x_rad`) are only applicable when `WRX` is used as a function. Examples: `wrx 0.005` ! waist X-radius at object plane is 0.005mm `wrx s6` ! returns waist X-radius at surface 6 in buffer for use in UDG or optimization. `wrx s6 z3 w2` ! same as above, but returns waist X-radius at surface 6 for zoom position 3 and wavelength 2 in buffer for use in UDG or optimization. Note, that the `zk` parameter is obligatory for zoomed systems.
`WRY y_rad [sk |wi..j|zi..j]`	
	Waist radius (in mm) in Y-direction at object surface, respectively relative to surface `sk` at zoom position `zk` and wavelength(s) `wi..j`|`wk`. Only one parameter may be given in a command, either `y_rad` or `sk`|`zk`|`wk`. The optional surface parameters `si..j`|`sk`, `zi..j`|`zk` and `wi..j`|`wk` (without `y_rad`) are only applicable when `WRY` is used as a function. Examples: `wry 0.005` ! waist Y-radius at object plane is 0.005mm `wry s6` ! returns waist Y-radius at surface 6 in buffer for use in UDG or optimization. `wry s6 z3` ! same as above, but returns waist Y-radius at surface 6 for zoom position 3 in buffer for use in UDG or optimization. Note, that the `zk` parameter is obligatory for zoomed systems.
`RCX wave_rad_x [sk |wi..j|zi..j]`	
	Radius of curvature of wavefront in x-direction at object plane, respectively relative to surface `sk` at zoom position `zi..j`|`zk` and wavelength(s) `wi..j`|`wk`. Only one parameter may be given in a command, either `wave_rad` or `sk`|`zk`|`wk`. The optional surface parameters `si..j`|`sk`, `zi..j`|`zk` and `wi..j`|`wk` (without `wave_rad`) are only applicable when `RCX` is used as a function.

continued on next page

continued from previous page	
	Examples: `rcx 0` ! wavefront X-radius of curvature at object plane is infinity. `rcx s6` ! returns wavefront X-radius of curvature at surface 6 in buffer for use in UDG or optimization. `rcx s6 z3` ! same as above, but returns wavefront X-radius of curvature at surface 6 for zoom position 3 in buffer for use in UDG or optimization. Note, that the `zk` parameter is obligatory for zoomed systems.
`RCY wave_rad_y [sk |wi..j|zi..j]`	
	Radius of curvature of wavefront in y-direction at object plane, respectively relative to surface `sk` at zoom position `zi..j|zk` and wavelength(s) `wi..j|wk`. Only one parameter may be given in a command, either `wave_rad` or `sk|zk|wk`. The optional surface parameters `si..j|sk`, `zi..j|zk` and `wi..j|wk` (without `wave_rad`) are only applicable when `RCY` is used as a function. Examples: `rcy 1000` ! wavefront Y-radius of curvature at object plane is 1000mm `rcy s6` ! returns wavefront Y-radius of curvature at surface 6 in buffer for use in UDG or optimization. `rcy s6 z3` ! same as above, but returns wavefront Y-radius of curvature at surface 6 for zoom position 3 in buffer for use in UDG or optimization. Note, that the `zk` parameter is obligatory for zoomed systems.
`ZWX z-waist-x [sk |wi..j|zi..j]`	
	Location of beam waist relative to object plane for x-direction, respectively relative to surface `sk` at zoom position `zi..j|zk` and wavelength(s) `wi..j|wk`. Only one parameter may be given in a command, either `z-waist-x` or `sk|zk|wk`. The optional surface parameters `si..j|sk`, `zi..j|zk` and `wi..j|wk` (without `z-waist-x`) are only applicable when `ZWX` is used as a function. Examples: `zwx 1.3` ! X-waist is 1.3mm from object plane `zwx s6` ! returns X-waist position relative to surface 6 into buffer for use in UDG or optimization. `zwx s6 z3` ! same as above, but returns X-waist position relative to surface 6 for zoom position 3 in buffer for use in UDG or optimization. Note, that the `zk` parameter is obligatory for zoomed systems.
`ZWY z-waist-y [sk |wi..j|zi..j]`	
	Location of beam waist relative to object plane for Y-direction, respectively relative to surface `sk` at zoom position `zi..j|zk` and wavelength(s) `wi..j|wk`. Only one parameter may be given in a command, either `z-waist-y` or `sk|zk|wk`. The optional surface parameters `si..j|sk`, `zi..j|zk` and `wi..j|wk` (without `z-waist-y`) are only applicable when `ZWY` is used as a function. Examples: `zwy 1.3` ! Y-waist is 1.3mm from object plane `zwy s6` ! returns Y-waist position relative to surface 6 into buffer for use in UDG or optimization. `zwy s6 z3` ! same as above, but returns Y-waist position relative to surface 6 for zoom position 3 in buffer for use in UDG or optimization. Note, that the `zk` parameter is obligatory for zoomed systems.
	continued on next page

continued from previous page	
`M2`	M^2 factor, describing the departure of real beams from the Gaussian ideal. See also Eq. 14.60. M^2 is the amount by which the beam waist product exceeds the diffraction limit of an ideal Gaussian beam of the same wavelength. $M^2 = 1$ for the ideal beam.
`SRX sk wi..j zi..j`	
	Returns the Gaussian spot size in the X/Z plane at surface `sk`. It takes the Gaussian source parameters (such as `WRX, WRY, RCX, RCY`, etc.), hence they must be properly set before this function may be used. This is a function, not a command, to be used in `UGR` or optimization.
`SRY sk wi..j|zi..j`	
	Returns the Gaussian spot size in the Y/Z plane at surface `sk`. It takes the Gaussian source parameters (such as `WRX, WRY, RCX, RCY`, etc.), hence they must be properly set before this function may be used. This is a function, not a command, to be used in `UGR` or optimization.
`GDX sk wi..j|zi..j`	
	Returns the divergence of a Gaussian beam in the X/Z plane at surface `sk`. It takes the Gaussian source parameters (such as `WRX, WRY, RCX, RCY`, etc.), hence they must be properly set before this function may be used. This is a function, not a command, to be used in `UGR` or optimization.
`GDY sk wi..j|zi..j`	
	Returns the divergence of a Gaussian beam in the Y/Z plane at surface `sk`. It takes the Gaussian source parameters (such as `WRX, WRY, RCX, RCY`, etc.), hence they must be properly set before this function may be used. This is a function, not a command, to be used in `UGR` or optimization.
`RRX sk wi..j|zi..j`	
	Returns the Rayleigh range of a Gaussian beam in X-direction at surface `sk`. It takes the Gaussian source parameters (such as `WRX, WRY, RCX, RCY`, etc.), hence they must be properly set before this command may be used. This is a function, not a command and may only be used in `UGR` or optimization.
`RRY sk wi..j|zi..j`	
	Returns the Rayleigh range of a Gaussian beam in Y-direction at surface `sk`. It takes the Gaussian source parameters (such as `WRX, WRY, RCX, RCY`, etc.), hence they must be properly set before this command may be used. This is a function, not a command and may only be used in `UGR` or optimization.

Mathematics:

Because of the self-Fourier Transform characteristics, complex integrals to describe the propagation of Gaussian beams are not required, since only the radius of the Gaussian (”spot size”) and the radius of curvature of the wavefront change.

The variation of spot size w and wavefront radius of curvature R with distance z can be described explicitly as

$$w^2(z) = w_0^2 \left[1 + \left(\frac{\lambda z}{\pi w_0^2} \right)^2 \right] \tag{14.55}$$

and

$$R(z) = z\left[1 + \left(\frac{\pi w_0^2}{\lambda z}\right)^2\right] \tag{14.56}$$

The spot size has its minimum value at $z = 0$, which is equal to the beam waist w_0. The wavefront radius of curvature becomes infinity at the beam waist as illustrated in Fig. 14.29. The far-field divergence angle θ is given by

$$\theta = tan^{-1}\left(\frac{\lambda}{\pi w_0}\right) \approx \frac{\lambda}{\pi w_0} \tag{14.57}$$

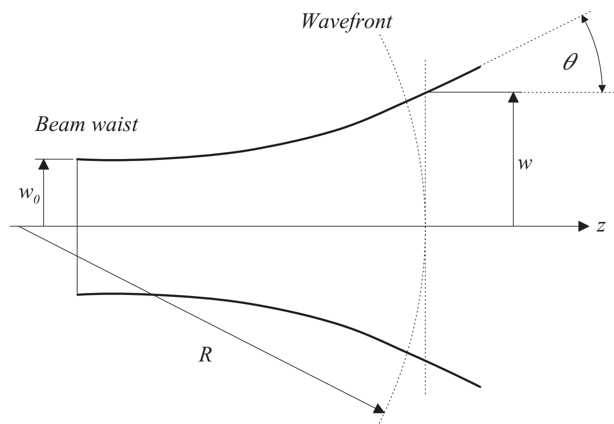


Figure 14.29: Propagation of a Gaussian beam.

The entire beam behaviour is completely specified by any two of the four parameters w, w_0, R and λ. The Rayleigh range is the distance from the waist to the axial point of minimum wavefront radius of curvature

$$z_r = \frac{\pi w_0^2}{\lambda} \tag{14.58}$$

R has its minimum value at $z = z_r$. In the free space between lenses, Eqs. 14.55 and 14.56 completely describe the beam. When the beam passes through an optical interface (lens, mirror), the wavefront curvature is changed, resulting in new values for size and position of the beam waist. At the optical interface, the beam diameter does not change.

A so-called M^2 factor has been introduced by Siegman[50] to describe the departure of a real beam from a Gaussian ideal beam. From Eq. 14.57 we see that the product of beam waist and far-field divergence angle is constant for a given wavelength

$$w_0\theta = \frac{\lambda}{\pi} \tag{14.59}$$

For a real beam the corresponding product can be written as

$$M^2 w_0\theta = M^2\frac{\lambda}{\pi} \tag{14.60}$$

Thus, the propagation of the spot size of real beams described by an M^2 factor is described by the same equation as for an ideal Gaussian.

It has been shown by Kogelnik and Li [27] and Herloski, Marshall and Antos [20], that the propagation and transformation of anastigmatic *Gaussian beams* can be modelled by an orthogonal characteristic ABCD matrix in the paraxial domain and, furthermore, can be represented by two paraxial rays. Following the model of Arnaud [2], we choose a waist ray (tangent to the input beam at the waist) and a divergence ray (tangent to the input beam at infinity), as shown in Fig. 14.30. Recalling the equations of Kogelnik and Li, we obtain

$$w' = \sqrt{y_d^2 + y_w^2} \quad (14.61)$$

$$z' = \frac{y_d v_d + y_w v_w}{v_d^2 + v_w^2} \quad (14.62)$$

$$w_0 = \frac{y_w v_d - v_w y_d}{\sqrt{v_d^2 + v_w^2}} \quad (14.63)$$

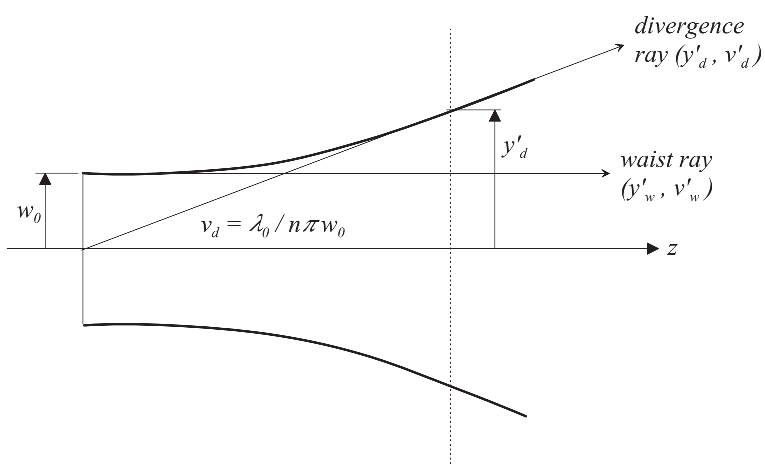


Figure 14.30: Equivalent paraxial rays for modelling of Gaussian beam propagation.

14.4 Fiber Coupling Efficiency

Calculation of coupling efficiency (`CEF`) includes apodization, clipping of the input beam, reflection losses by coated or uncoated surfaces and bulk absorption.

`CEF [ ? | fi | wi ]`	Calculate linear coupling efficiency (CEF). The question mark (optional) invokes a dialog box for editing properties of source fiber and receiving fiber.
`CEFDB [ ? | fi | wi ]`	Calculate coupling efficiency in decibel instead of returning the linear value. See also the CEF command above.
	continued on next page

continued from previous page	
MPR GAU | STE | FIL	Mode profile. Select GAU for Gaussian mode profile, STE for step-index, FIL for user defined profile loaded from file (in preparation).
FLO FIX|CMP	Fiber location in either a fixed (FIX) or compensated (CMP) position. FIX : The fiber is in a fixed position in the local coordinate system of the image surface (see also the second form of the FLO command below). The location of the fiber is independent of the beam location. CMP : The fiber position follows the chief ray. This is the default mode. The fiber is optimally shifted/tilted to give an optimized coupling efficiency.
FLO x_pos y_pos	Specify the coordinates of the (receiving) fiber position with respect to the local coordinate system of the image surface.
FSR rad_x rad_y	Fiber source radius in X- and Y-direction (in mm). Elliptical source profiles are specified by different values for the x- and y-extension. If only one value is given, the mode profile is assumed circular.
FSD div_x div_y	Far-field fiber source divergence. Elliptical far-fields are specified by different values for the x- and y-extension. If only one value is given, the far-field is assumed circular.
FSA alpha_tilt	Fiber source α-tilt in degree. Specify the rotation angle of the source fiber in the YZ plane. The rotation angle is defined in the local coordinate system.
FSB beta_tilt	Fiber source β-tilt in degree. Specify the rotation angle of the source fiber in the XZ plane. The rotation angle is defined in the local coordinate system.
FRR mode_radius	Receiving fiber mode-field radius (in mm).
FRD div	Far-field divergence of receiving fiber (in rad).
FRA alpha_tilt	Receiving fiber α-tilt in degree. Specify the rotation angle of the receiving fiber in the YZ plane. The rotation angle is defined in the local coordinate system. See also Fig. 14.31 for a definition of signs.
FRB beta_tilt	Receiving fiber β-tilt in degree. Specify the rotation angle of the receiving fiber in the XZ plane. The rotation angle is defined in the local coordinate system. See also Fig. 14.31 for a definition of signs.
FRX x-offset	Receiving fiber x-offset (in mm) with respect to the chief ray.
FRY y-offset	Receiving fiber y-offset (in mm) with respect to the chief ray.
	continued on next page

continued from previous page	
`WDX wedge_angle_x`	Wedge angle (cleavage angle) of the front face of the fiber in the X-direction, i.e. in the local XZ plane. The angle is measured in degree. See also Fig. 14.31 for a definition of signs.
`WDY wedge_angle_y`	Wedge angle (cleavage angle) of the front face of the fiber in the Y-direction, i.e. in the local YZ plane. The angle is measured in degree. See also Fig. 14.31 for a definition of signs.
`FSN1 source_core_index`	Source fiber, index of refraction n_1 of core material
`FSN2 source_cladding_index`	Source fiber, index of refraction n_2 of cladding material
`FSCR source_core_rad`	Source fiber, core radius in mm.
`FRN1 receiver_core_index`	Receiving fiber, index of refraction n_1 of core material
`FRN2 receiver_clad_index`	Receiving fiber, index of refraction n_2 of cladding material
`FRCR receiver_core_rad`	Receiving fiber, core radius in mm.
`FIBS prod-spec`	Specify source fiber by product (e.g. by manufacturers type number). A single command inserts all relevant optical data from a fiber catalogue. This option is currently only available from the menu.
`FIBR prod-spec`	Specify receiving fiber by product (e.g. by manufacturers type number). A single command inserts all relevant optical data from a fiber catalogue. This option is currently only available from the menu.
`TGR fft_grid`	Transformation grid. Because the coupling option uses a Fast Fourier Transform (FFT), a 2^n transform grid must be specified. The default value of `TGR` = 128, but it may be adjusted to 64, 128, 256, 512 or 1024. Smaller values of TGR are not recommended, as the accuracy of the computation will be reduced (sampling density is to coarse). Note, that a change of TGR also affects `NRD` (number of rays across pupil diameter). The relation is `TGR` = 4 * `NRD`.
`FSMM max_modes_source`	Fiber source maximum modes. Limits the number of modes calculated in the source fiber. `max_modes_source` must be less than less than the highest number of possible modes N in that fiber (see Eq. 14.76). Enter `FSMM -1` to always search for all modes possible (N).
`FRMM max_modes_receiver`	Fiber receiver maximum modes. Limits the number of modes calculated in the receiver fiber. `max_modes_receiver` must be less than the highest number of possible modes N in that fiber (see Eq. 14.76). Enter `FRMM -1` to always search for all modes possible (N).
`MMF`	Display field of a multi-mode fiber at selected modes. Opens a dialog box for editing fiber parameters. See a detailed description in sect. 14.4.3.

Notes:

- Coupling efficiency is normally computed for systems with finite object and image distances (fiber-fiber or diode-fiber applications). For systems, where the object is at infinity, the pupil will be assumed uniformly illuminated. All computations are then referred to the total energy incident upon the entrance pupil. Only for this special case, the Gaussian beam profile (e.g. from a collimated laser) must be properly set by the apodization factors `PUI`, `PUX` and `PUY` respectively. For finite object and image distances, apodization should be switched off (PUI=PUX=PUY=1), as the Fourier Transformation property based on the fiber mode profile already yields the correct far-field amplitude profile in the entrance pupil.

- **The only approximation made in the computation method as described below is that diffraction effects that occur between entrance and exit pupil are neglected.** In many cases this approximation is sufficiently accurate, but in special cases, for example when the beam is very small or when the free space in the optics is large, a diffraction beam propagation method (BPR) must be applied. The Fresnel number is a good indicator, whether CEF or BPR is appropriate. The Fresnel number N is a property of the beam semi diameter w, wavelength λ and propagation distance L. It is given by $N = \frac{w^2}{\lambda L}$. For small Fresnel numbers ($N < 1$), beam propagation should be used, otherwise CEF can be used with sufficient accuracy.

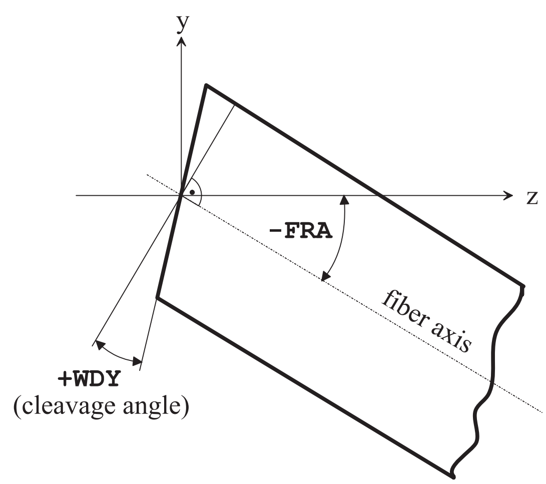


Figure 14.31: Definition of fiber tilts (FRA, FRB) and cleavage angles (WDX, WDY), here shown in the Y/Z plane only. The sign of the angles is in accordance to surface tilts. It follows mathematical convention, i.e. it is positive for counter-clockwise rotation and negative for clockwise rotation.

The calculation of coupling efficiency (also known as insertion loss) involves components and optical systems, which collect light from a source (a laser, a fiber, etc.) and couple it into a receiving fiber. The basic problem is to account for the effects of aberrations, fiber misalignments and fiber-mode mismatch.

The coupling efficiency T is defined as the normalized overlap integral of the image field distribution $U(x', y')$ and the mode pattern of the receiving fiber $\psi(x', y')$

$$T = \left| \frac{\iint U(x', y') \cdot \psi^*(x', y') dx' dy'}{\sqrt{\iint U(x', y') \cdot U^*(x', y') dx' dy' \iint \psi(x', y') \cdot \psi^*(x', y') dx' dy'}} \right|^2 \tag{14.64}$$

where $*$ denotes the complex conjugate. For computational purposes, the method described by Wagner and Tomlinson [57] is applied in $OpTaliX$ for which the overlap integral is transformed to the exit pupil of the coupling optics. The power-coupling efficiency T is then expressed as a single integral with an integrand that is the product of the complex far-field distributions of the source-fiber mode profile $\Psi_S(\zeta,\eta)$, the far-field distribution of the receiving-fiber mode profile $\Psi_R(\zeta,\eta)$ and the coherent transfer function of the optical system $L(\zeta,\eta)$

$$T = \left| \int \Psi_S(\zeta,\eta) \cdot L(\zeta,\eta) \cdot \Psi_R(\zeta,\eta) da \right|^2 \tag{14.65}$$

where (ζ,η) are the normalized coordinates in the exit pupil. Ψ_S and Ψ_R are the scaled Fourier transforms of the source and receiving fiber mode profiles ψ_s and ψ_r respectively. The coherent transfer function is expressed as $L = exp\left[-ikW(\zeta,\eta)\right]$ where W is the wavefront aberration and $k = 2\pi/\lambda$. Thus, all aberrations (optical system wavefront error, fiber misalignments and mode profile mismatch) are described in the exit pupil of the optical system, allowing coupling effects to be handled in a manner consistent with accepted conventions in classical optics.

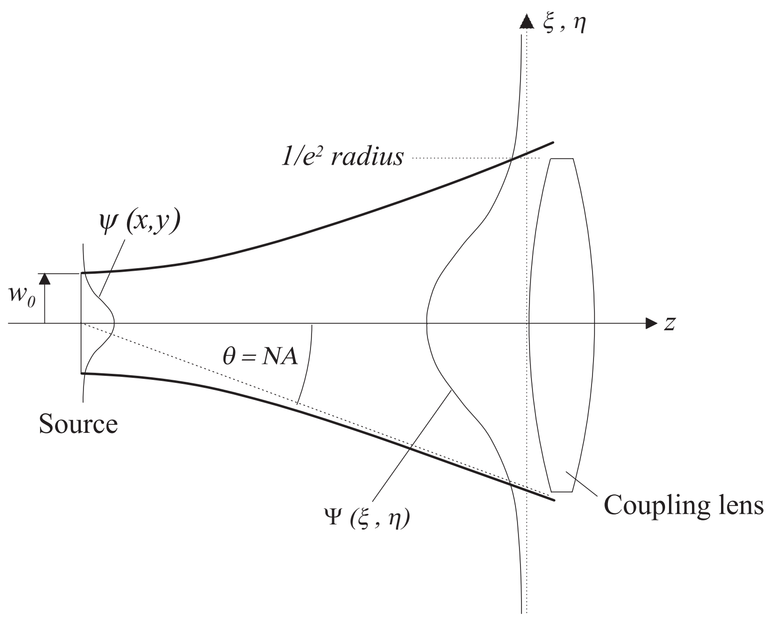


Figure 14.32: Transformation of the source profile (fiber or laser) to the entrance pupil of the optical system (not to scale). In the example shown, the numerical aperture (NA) of the coupling system matches the far-field divergence θ of the source (which is defined at the $1/e^2$ point). Hence, only a fraction of the emitted energy is transferred by the coupling optics, because the foot of the Gaussian field is truncated by the aperture stop of the optical system.

Using the quantities and relations given above, the far-field diffraction angle θ, which is usually defined at the $1/e^2$ intensity, must not be confused with the numerical aperture (NA) of the fiber and of the coupling optics. For multi-mode fibers the maximum angle of the beam radiated from (or accepted by) a fiber is determined by the refractive index difference between core and cladding and is defined by

$$NA = \sqrt{n_1^2 - n_2^2} = n_1\sqrt{2\Delta} \tag{14.66}$$

where

$$\Delta = \frac{n_1^2 - n_2^2}{2n_1^2} \approx \frac{n_1 - n_2}{n_1} \tag{14.67}$$

and n_1 is the index of refraction of the core, n_2 is the index of refraction of the cladding. NA is conventionally used as a measure of that index difference.

For a single-mode fiber, not only the core-cladding index difference but also the core size (precisely the mode-field diameter) and the wavelength of the light define the angular beam spread. With this definition, about 25% of the emitted power propagates at angles larger than θ (see also Fig. 14.32). In order to avoid substantial truncation of the beam, the lens NA must be extended beyond the emitted $1/e^2$ far-field divergence angle θ. The divergence angle, at which the far-field intensity has fallen to the 1% point is about 1.5 times larger than the $1/e^2$ angle and the lens NA must be oversized by this factor for efficient coupling.

Assuming identical source and fiber modes (i.e. the Gaussian beams perfectly match), the theoretical coupling efficiency can be expressed as a function of the numerical aperture of the optics (NA) and the far-field divergence θ of the fiber

$$T = \left(1 - \exp\left[-2\left(\frac{NA}{\theta}\right)^2\right]\right)^2 \tag{14.68}$$

For the above mentioned case, where $NA/\theta = 1.5$, the coupling efficiency is 0.978 (-0.097 dB).

14.4.1 Single-Mode Fibers

Single-mode fiber applications are different to classical optical imaging in that the source fiber, coupling optics and receiving fiber comprise a coherent system. In single-mode fibers, only one mode propagates because the core size (typically $5 - 10\mu m$) approaches the operational wavelength λ. The form of the mode pattern in single-mode fibers is well described by a Gaussian function of the form

$$\psi(x', y') = \exp\left[-\left(\frac{r'}{r_0}\right)^2\right] \tag{14.69}$$

The Gaussian mode is completely specified by the radius r_0 at which the amplitude drops to its $1/e^2$ value. Recalling Eq. 14.57, the mode profile at the fiber end also governs the $1/e^2$ far-field divergence angle

$$\theta = tan^{-1}\left(\frac{\lambda}{\pi w_0}\right) \approx \frac{\lambda}{\pi w_0}$$

if $w_0 = r_0$ is the waist radius of the mode profile at the $1/e^2$ intensity.

14.4.2 Multi-Mode Fibers

As their name implies, multi-mode fibers propagate more than one mode. The number of modes depends on the core radius a and numerical aperture (NA) and is given by $V^2/2$, with

$$V = \frac{2\pi}{\lambda_0} a \sqrt{n_1^2 - n_2^2} = \frac{2\pi}{\lambda_0} a n_1 \sqrt{2\Delta} \tag{14.70}$$

V is known as the *normalized frequency* or *waveguide parameter*. As the value of V increases, the number of modes supported by the fiber increases. A step-index fiber becomes single-mode for a given wavelength when $V < 2.405$.

Three parameters are required to specify a step-index or graded-index multi-mode fiber: the refractive index of the core material n_1, the refractive index of the cladding material n_2 and the radius of the cylindrical core a.

The mode pattern of the fundamental mode in a weakly guiding fiber is given by

$$\psi(r') = \begin{cases} \frac{A}{J_l(U)} J_l\left(\frac{Ur}{a}\right)\begin{bmatrix} cosl\phi \\ sinl\phi \end{bmatrix}, & r < a \\ \frac{A}{K_l(W)} K_l\left(\frac{Wr}{a}\right)\begin{bmatrix} cosl\phi \\ sinl\phi \end{bmatrix}, & r > a \end{cases} \tag{14.71}$$

where

$$U = a\left(k_0^2 n_1^2 - \beta^2\right)^{1/2} \tag{14.72}$$

$$W = a\left(\beta^2 - k_0^2 n_2^2\right)^{1/2} \tag{14.73}$$

$k_0 = 2\pi/\lambda$ and β is known as the propagation constant and $r = \sqrt{x^2 + y^2}$. For guided modes we must have $k_0^2 n_2^2 < \beta^2 < k_0^2 n_1^2$, or with the normalized propagation constant

$$b = \frac{\beta^2/k_0^2 - n_2^2}{n_1^2 - n_2^2} = \frac{W^2}{V^2} \tag{14.74}$$

we must have $0 < b < 1$. We can then write the eigenvalue equations for the mode structure

$$\begin{aligned} V(1-b)^{\frac{1}{2}} \frac{J_{l-1}\left(V(1-b)^{\frac{1}{2}}\right)}{J_l\left(V(1-b)^{\frac{1}{2}}\right)} &= -Vb^{\frac{1}{2}} \frac{K_{l-1}\left(V(b)^{\frac{1}{2}}\right)}{K_l\left(V(b)^{\frac{1}{2}}\right)}, \qquad l \geq 1 \\ V(1-b)^{\frac{1}{2}} \frac{J_1\left(V(1-b)^{\frac{1}{2}}\right)}{J_0\left(V(1-b)^{\frac{1}{2}}\right)} &= -Vb^{\frac{1}{2}} \frac{K_1\left(V(b)^{\frac{1}{2}}\right)}{K_0\left(V(b)^{\frac{1}{2}}\right)}, \qquad l = 0 \end{aligned} \tag{14.75}$$

where J, K are the J- and K-Bessel functions. For a given value of l, there will be a finite number of solutions of the eigenvalue equations (Eq. 14.75) and the m^{th} solution (m = 1,2,3,...) is referred to as the LP_{lm} mode.

A derivation of this mode structure can be found in Gloge [15] and Ghatak [13]. The maximum number of modes N is approximated by

$$N \approx \frac{V^2}{2} \tag{14.76}$$

for $V >> 1$.

$OpTaliX$ calculates the mode structure for all possible modes in a multi-mode fiber and performs a coupling efficiency calculation for each mode separately. The individual results are combined for a total coupling efficiency.

Note that computing time will increase significantly with increasing number of modes calculated on both source and receiver fiber, because CEF must be computed for each mode combination separately. For example, allowing only 10 modes in both source-fiber and receiver-fiber results in 100 separate calculations of coupling efficiency. It is therefore recommended to limit the maximum number of *calculated* modes by the `FSMM` and `FRMM` commands.

14.4.3 Display Fiber Modes

The individual modes of a multi-mode fiber can be displayed using the `MMF` command, which opens a dialog box for editing fiber parameters (see Fig. 14.33).

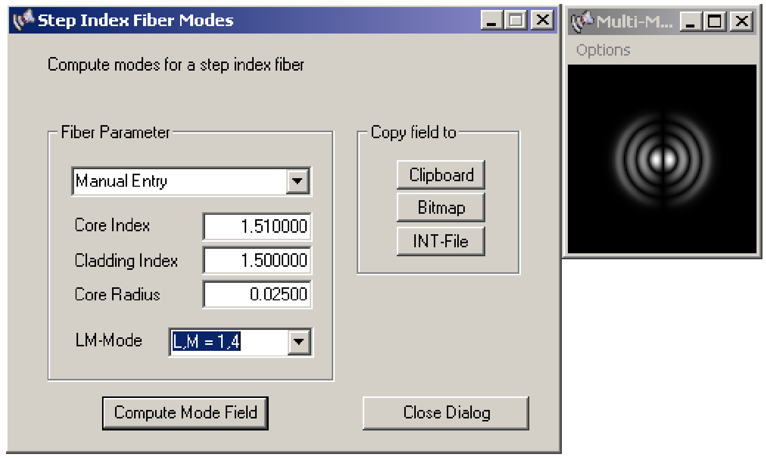


Figure 14.33: Calculation and display of fiber modes.

The maximum number of modes that can be calculated and displayed is 200. Fiber parameters such as core index, cladding index and core radius can be explicitly specified in the appropriate fields or obtained from predefined fibers from the pull-down menu. Note that on selecting new fiber parameters, the program automatically searches for all possible modes (¡ 200), which may take a while depending on the parameters selected and on computer speed. Clicking on the ”Compute Mode Field” button displays the selected mode profile. The intensity of the mode field can be saved as bitmap file (BMP, PNG or PCX) or INT-file (Code V compatible).

14.4.4 Fiber Coupling Example 1

As our first example, we choose a SELFOC^{TM} SLW10 gradient index rod-lens from NSG and for source and receiving fiber a single-mode fiber SMF28 from Corning is selected. This configuration, as shown in Fig. 14.34, can be found in the examples library (`selfoc-coupler.otx`). The pitch of the gradient index lens has been adjusted to 0.5, which gives unit magnification and therefore optimum coupling conditions for the selected fibers.

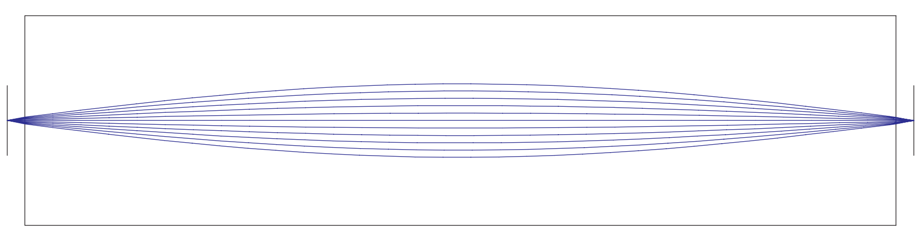


Figure 14.34: Coupling of two Corning SMF28 fibers with NSG-SELFOC^{TM} lens SLW10.

From the main menu, selecting *Diffraction Analysis*$--$ $>$*Fiber Coupling*, invokes a dialog box (Fig. 14.35 on the following page), which allows editing of all relevant coupling parameters. In this example, they are already preselected from the fiber catalogue. Mode-field radius and $1/e^2$ divergence are automatically updated, if a fiber is selected from the catalogue. The source fiber is assumed at the selected field position (as defined by the XOB and YOB commands) and the receiving fiber is assumed at the position of the chief ray coordinates in the image plane.

Important: The correct amplitude distribution in the pupil of the coupling optics is automatically calculated by the transformation process from the source fiber end to the entrance pupil. It is therefore not necessary to adjust the amplitude profile by the apodization parameter PUI, PUX and PUY. In order to obtain correct results in fiber-to-fiber coupling, PUI, PUX and PUY shall be 1. Check the corresponding settings.

Only in the special case of a parallel laser beam entering the coupling optics (object at infinity) should the apodization be properly adjusted, since transformation of the source will be skipped for this condition.

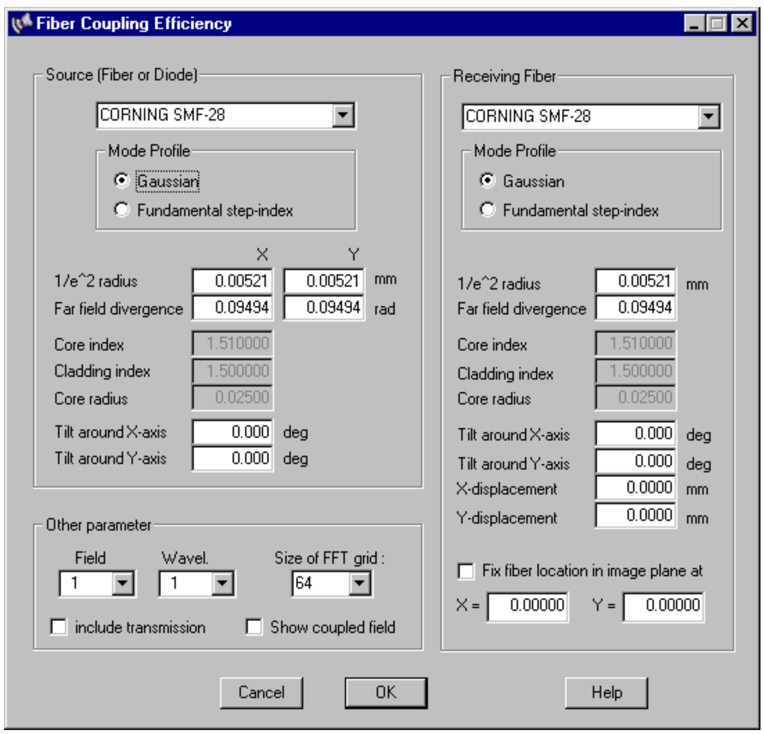


Figure 14.35: Dialog box showing coupling options for the setup shown in Fig. 14.34.

```
Fiber Coupling Efficiency:

Field number          :  1     (     0.0000/     0.0000 )
Image coordinates     :        (     0.0000/     0.0000 )
Wavelength number     :  1     (     1.5500 micron )
Transformation grid   :   64

                             SOURCE        RECEIVER   Unit
Fiber type            :      SMF-28         SMF-28
1/e^2 radius          :     0.00520        0.00520     mm
Far-field divergence  :     0.09488        0.09488    rad
Tilt around X-axis    :     0.00000        0.00000    deg
Tilt around Y-axis    :     0.00000        0.00000    deg
X-displacement        :                    0.00000     mm
Y-displacement        :                    0.00000     mm

Transmission          : not considered
```

```
Power coupling        :    0.99271 (  -0.032 dB)
Power coupling (ideal):    0.99953 (  -0.002 dB)
```

This example shows very little basic insertion loss (-0.032dB), since the NA of the coupling optics is about 2.1 times larger than the fiber divergence (0.09488). The ideal power coupling (-0.002dB) is the theoretical maximum efficiency if the optics introduced no aberrations and does not truncate the beam. It is a representation how good source fiber and receiving fiber match.

14.4.5 Fiber Coupling Example 2

The second example will be a demultiplexer, which we load from the examples library (`demux.otx`). Since the design employs a diffraction grating, it is basically a spectrometer, which separates the wavelengths (channels) into different fibers.

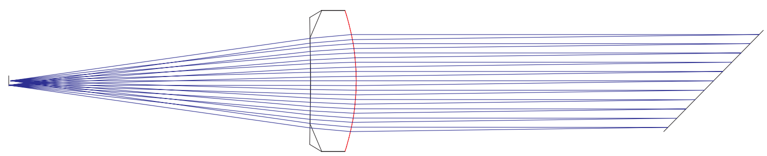


Figure 14.36: A simple demultiplexer, shown at only one wavelength.

The system is defined at three wavelengths, which describes the spectral range of interest. We will also switch to ”spectrometer” mode (this relates all aberrations to the current wavelength, rather than to the base wavelength), which is currently only possible from the configuration dialog (from the main menu, select *Edit−− >Configuration* and then the tab ”General”).

We will now define a user defined graphics UGR (see section 12.10, page 200) to plot coupling efficiency (CEF) versus wavelength. User defined graphics is found under the *tools menu*. In the dialog to appear, predefined settings may be restored. We will do so and restore (load) from the macro subdirectory `cef_vs_wl.ugr`. All settings should be right for our example and we immediately run the plot.

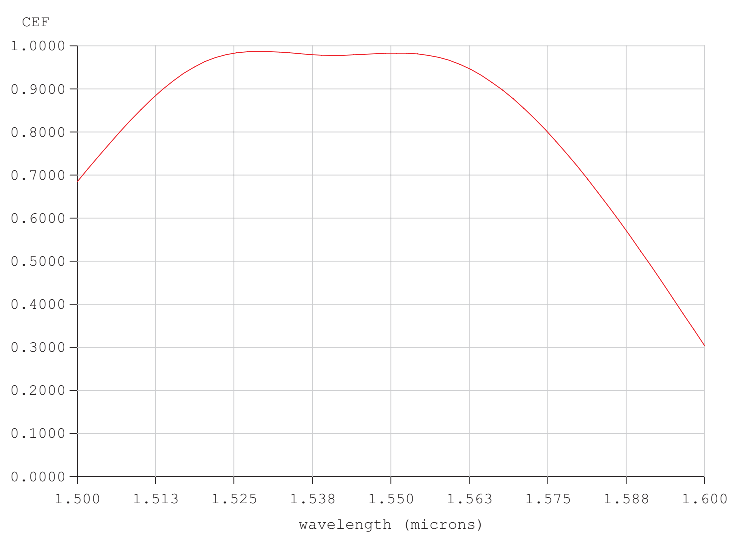


Figure 14.37: Coupling efficiency versus wavelength.

15

Illumination Analysis

The illumination option is used to compute the illuminance/radiance distribution at any surface of the system, including the image surface. As opposed to point-like objects (defined by 'fields", see sect. 7.3.1, page 42), illumination sources are extended in the spatial domain. $OpTaliX$ currently supports two types of illumination sources,

- **flat emitting sources**. There are predefined flat sources, such as circular, elliptical or rectangular flat shapes, Gaussian, double pinhole, or flat sources defined by bitmap images.
- **ray sources**, that is, sources defined by a collection of rays.

Point sources (fields) are defined in the *optical system configuration* (see sect. 7.3.1, page 42) and are *always* located on the object surface. Thus, object coordinates ("fields") are always referred to the vertex of the object surface. The location of the object surface itself is defined, for example, by the object distance (S0), x-decenter of the object surface (XDE s0), etc.

Sources used in illumination calculations *always* exhibit a finite spatial extension and their locations may be referred either to the global coordinate system or the object coordinate system. See page 27 for definition of coordinate systems.

15.1 Commands for Defining Illumination Sources

Command line entries for illumination source parameters allow two forms: a long form and a short form. Note that the short form is required in defining zoom/multi-configuration systems.

Also, do not confuse the qualifier "sk" used for sources and surfaces. In this section, and for illumination purposes only, "sk" is exclusively used for sources. For all other commands, not related to illumination or source properties, "sk" always refers to surfaces!

SRC n_sources	Without any other qualifier SRC n_sources specifies the number of sources that can be defined. Example: src 3 ! specifies 3 sources.
	continued on next page

continued from previous page

SRC sk TYPE [FIL file_name] Short form: Sxxx sk	Defines source type. The command exists in a long and a short form. sk is the source number. The short form is required in zoom definitions and LDM queries. In the long form, TYPE can be any one of CIR top hat circular ELL top hat elliptical REC top hat rectangular GAU Gaussian BMP Bitmap file (*.BMP, *.PCX, *.PNG) INT INT file GRA Grating PIN Double pinhole CHE Checker board RAY Rays defined in file_name In the short form, xxx is a place holder for the source type. It is defined as follows: SCIR top hat circular SELL top hat elliptical SREC top hat rectangular SGAU Gaussian SBMP Bitmap file (*.BMP, *.PCX, *.PNG) SINT INT file SGRA Grating SPIN Double pinhole SCHE Checker board Examples: src s1 ELL ! top hat elliptical source, srec s2 ! short form: top hat rectangular source no.2 , src s2 RAY FIL c:\rayset.dat ! ray source.
SRC USE sk Y|N Short form: SUSE sk Y|N	Use source sk. Once defined, sources can be included or excluded in illumination ray trace. The short form is required in zoom definitions and LDM queries. Examples: src s1 use y ! Source 1 is used (included) in illumination analysis, src s2 use n ! source 2 is ignored (excluded) in illumination analysis.
SRC REF sk O|G Short form: SGREF sk O|G	Source sk is referenced to object coordinate system (O) or global coordinate system (G). Examples: src s1 ref o ! Source 1 is referred to object coordinate system, sgref s2 g ! source 2 is referred to global coordinate system.

continued on next page

continued from previous page

`SRC PWR sk power` Short form: `SPWR sk power`	Source total emitted power. Examples: `src pwr s1 1.0!` Total emitted power for source 1 is 1.0 Watts. `spwr s2 3` ! Total emitted power for source 2 is 3 Watts.
`SRC sk XEXT x_ext` Short form: `SXEX sk x_ext`	Defines source X-extension (in mm). `sk` is the source number. If omitted, `sk` defaults to source 1. See also Fig. 15.5.
`SRC sk YEXT y_ext` Short form: `SYEX sk x_ext`	Defines source Y-extension (in mm). `sk` is the source number. If omitted, `sk` defaults to source 1. See also Fig. 15.5.
`SRC XDE sk x_dec` Short form: `SXDE sk x_ext`	Defines source X-decenter (in mm). Decenter is measured from the vertex of the source coordinate system (object or global). `sk` is the source number. If omitted, `sk` defaults to source 1. See also sect. 15.2 for definition of source coordinate system.
`SRC YDE sk y_dec` Short form: `SYDE sk y_dec`	Defines source Y-decenter (in mm). Decenter is measured from the vertex of the source coordinate system (object or global). `sk` is the source number. If omitted, `sk` defaults to source 1. See also sect. 15.2 for definition of source coordinate system.
`SRC ZDE sk z_dec` Short form: `SZDE sk z_dec`	Defines source Z-decenter (in mm). Decenter is measured from the vertex of the source coordinate system (object or global). `sk` is the source number. If omitted, `sk` defaults to source 1. See also sect. 15.2 for definition of source coordinate system.
`SRC ADE sk x_tlt` Short form: `SADE sk x_tlt`	Tilt of source normal about X-axis (in degrees). `sk` is the source number. If omitted, `sk` defaults to source 1. See also sect. 15.2 for definition of source coordinate system.
`SRC BDE sk y_tlt` Short form: `SBDE sk y_tlt`	Tilt of source normal about Y-axis (in degrees). `sk` is the source number. If omitted, `sk` defaults to source 1. See also sect. 15.2 for definition of source coordinate system.
`SRC CDE sk z_tlt` Short form: `SCDE sk z_tlt`	Tilt of source normal about Z-axis (in degrees). `sk` is the source number. If omitted, `sk` defaults to source 1. See also sect. 15.2 for definition of source coordinate system.
`SRC DIVX sk div_x` Short form: `SDIVX sk div_x`	X-divergence of source emission (in degrees), full with. `sk` is the source number. If omitted, `sk` defaults to source 1.
`SRC DIVY sk div_y` Short form: `SDIVY sk div_y`	Y-divergence of source emission (in degrees), full with. `sk` is the source number. If omitted, `sk` defaults to source 1.

continued on next page

continued from previous page	
SRC COS sk cos_power_factor Short form: SCOS sk cos_power_factor	Cosine power factor. Defines source emittance as a function of the emittance angle. sk is the source number. If omitted, sk defaults to source 1. The intensity of rays emitted from an extended source can be controlled by the cosine power factor (SCOS) in dependence of the angle at which the ray is launched from the source normal. The emitted ray intensity is described by the following function: $$I_{ray} = cos(\alpha)^{SCOS} \quad (15.1)$$ where α is the angle at which the ray is emitted and SCOS is the cosine power factor. See also section 15.3.1 for a more detailed description. Examples: SCOS 0.0 : All rays are emitted at the same intensity, irrespective of the emittance angle at which the ray is launched. SCOS 1.0 : Ray intensity follows the Lambertian Law, $I = cos(\alpha)^{1.0}$
SRC ARAY sk analysis_rays Short form: SARAY sk analysis_rays	Source analysis rays. Number of rays traced in illumination analysis for source sk. If sk is omitted, source 1 is assumed.
SRC PRAY sk plot_rays Short form: SPRAY sk plot_rays	Source plot rays. Number of rays displayed in layout plots for source sk.
LIS SRC [sk]	List illumination sources. sk is the source number. If sk is absent, all sources defined are listed.
ILL SAV Y|N	Save illumination data along with prescription data, Y=yes, N=no.

15.2 Illumination Sources Coordinate Definition

The position and orientation of flat and real sources may be freely chosen in 3D space. As described in 15.3 sources may be referred to the object coordinate system or the global reference coordinate system (see also the SGREF command). If the illumination source is referred to the object surface, its position and orientation also depend on the object surface location/orientation. Fig. 15.1 shows the dependencies of source position and orientation with respect to the object surface.

15.3 Defining Illumination Sources in the GUI

Source parameters can also be defined in dialogs from the graphical user interface (GUI). From the command line, invoke the illumination dialog by

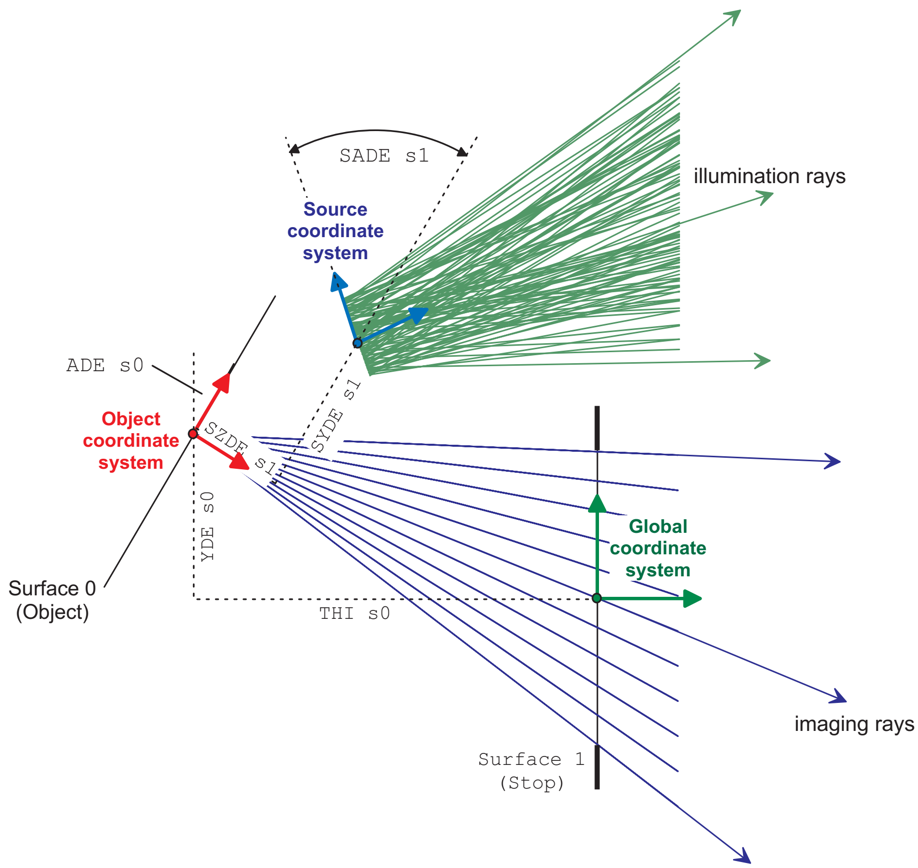


Figure 15.1: Definition of source coordinate system in relation to object coordinate system.

`ILL [?]`	Runs illumination analysis. The optional parameter "?" invokes a dialog box for editing illumination parameter prior to illumination analysis.

or from the main menu *Geom.Analysis -> Illumination*. Because illumination sources are mostly extended objects, in contrast to the point-like objects normally used in optical analysis (also called 'field objects'), illumination (extended) sources may also defined in the configuration dialog. It is invoked by

`EDI CNF`	Edit configuration parameters. Select the **Illum.Source** tab to define illumination source parameters.

The following graphic (Fig. 15.2) shows the dialog for defining various illumination sources.

15.3.1 Controlling Source Emittance Characteristics

The emittance characteristics of a source, i.e. its apparent intensity as a function of the viewing angle from the source normal, can be defined for flat sources (circle, rectangle, etc.) by theSCOS parameter. The emitted intensity as a function of the emittance angle α is described by

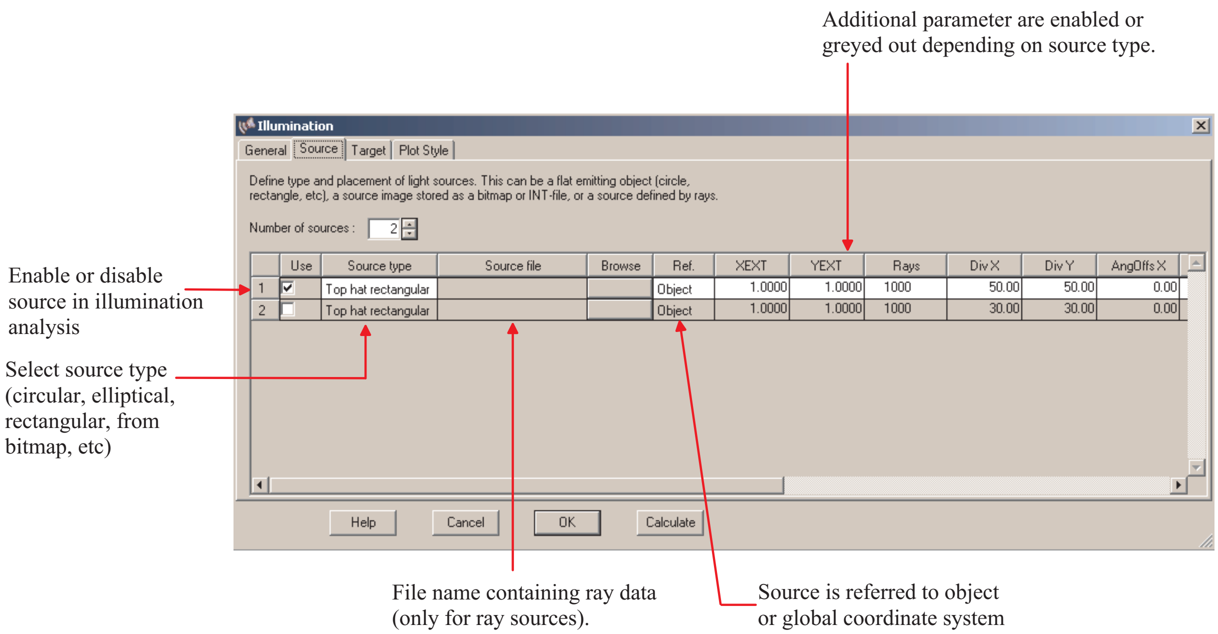


Figure 15.2: Dialog for defining illumination sources. Invoked by commands `ILL ?` or by `EDI CNF`.

$$I(\alpha) = I_0 \cdot cos(\alpha)^{SCOS} \tag{15.2}$$

Figure 15.3 indicates the effect of the SCOS parameter on the angular emittance function.

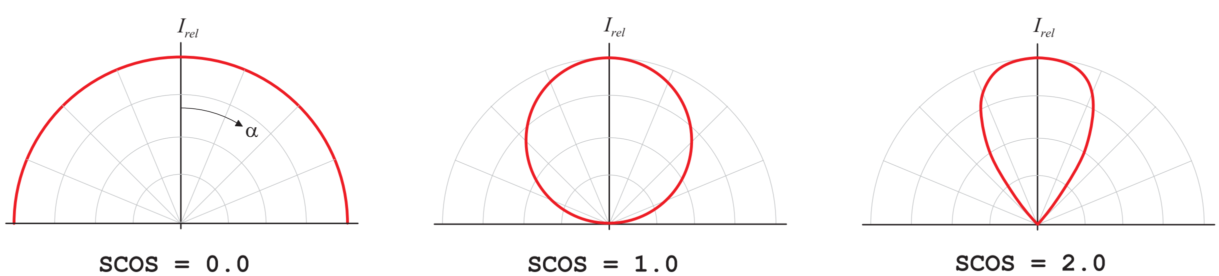


Figure 15.3: Effect of SCOS parameter on the angular emittance of a source, shown in a polar diagram.

Note that the SCOS parameter is ineffective for sources that are defined by a collection of rays ("ray sources").

15.3.2 Controlling Source Rays in the Lens Layout Plot

An important means to control the correct setting of source parameters is the visualization of rays emitted by the sources. By default, plotting of rays emanating from (extended) sources is disabled in the layout plot. Because analysis of illumination sources usually involves a massive amount of rays, this would significantly slow down rendering of sources (and the rays) in the lens layout plot. Source rays, however, can be enabled in lens layout plots by enabling the check box "***Show illumination source rays***" in the option dialog box (right click in the lens layout window), as shown in Fig.15.4

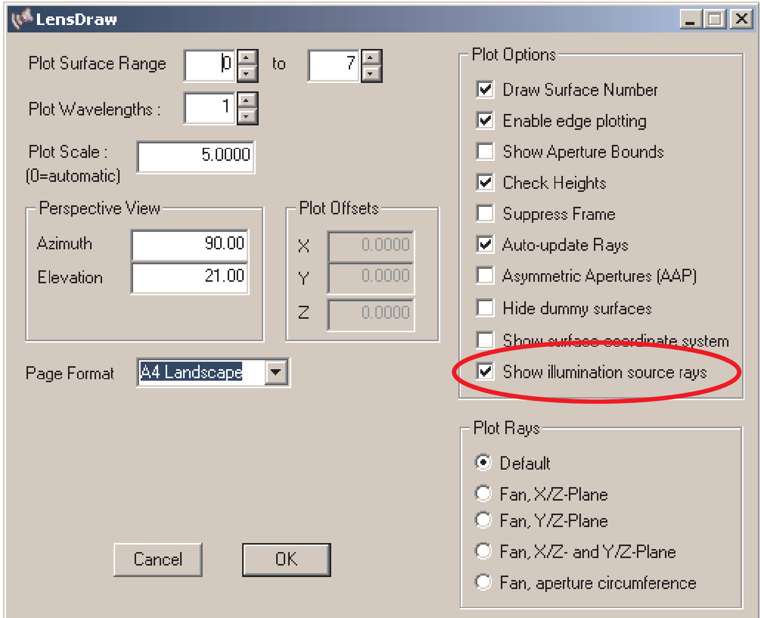


Figure 15.4: Enabling plot of source rays in lens layout plot. The setting is saved with the lens prescription.

15.3.3 Flat emitting Sources

The illumination option allows specification of flat sources, such as circular, elliptical or rectangular flat shapes, Gaussian, double pinhole, etc. Flat sources are defined on a plane surface only, as indicated in Figure 15.5. Flat sources emit at a constant intensity at every point of the source area confined by (SXEX, SYEX).

Note that standard field specifications, as defined for point sources in the ”fields” tab of the optical system configuration (EDI CNF), are ignored in illumination analysis.

Wavelength weights (WTW) are used to model the spectral transmission of the system, not the source. Initially, all sources are emitting spectrally uniformly at all specified wavelengths. Wavelength weights will then act as a spectral filter applied to the source.

A flat source (object) is defined by its full extension in X- and Y-direction (SXEX, SYEX). The source is located at (SXDE, SYDE, SZDE) with respect to the reference system which is either the global coordinate system or the object coordinate system. The flat surface may also be tilted by the angles (SADE, SBDE, SCDE) to indicate an emission direction different from the coordinate Z-axis.

The light emission is confined in a cone defined by the divergence parameters (SDIVX, SDIVY).

15.3.4 Flat Source with Gaussian Profile

Flat sources with a Gaussian profile are characterized by a non-uniform intensity across the source area. The profile is scaled to the source extensions SXEX, SYEX, such that the 50% of the peak intensity is obtained exactly at 1/3 of the source extension, the 1/e2 intensity is obtained at 0.567 of the source extension, and the source intensity at the rim of the source extension is 0.1954%. Fig.15.6 illustrates these relationships.

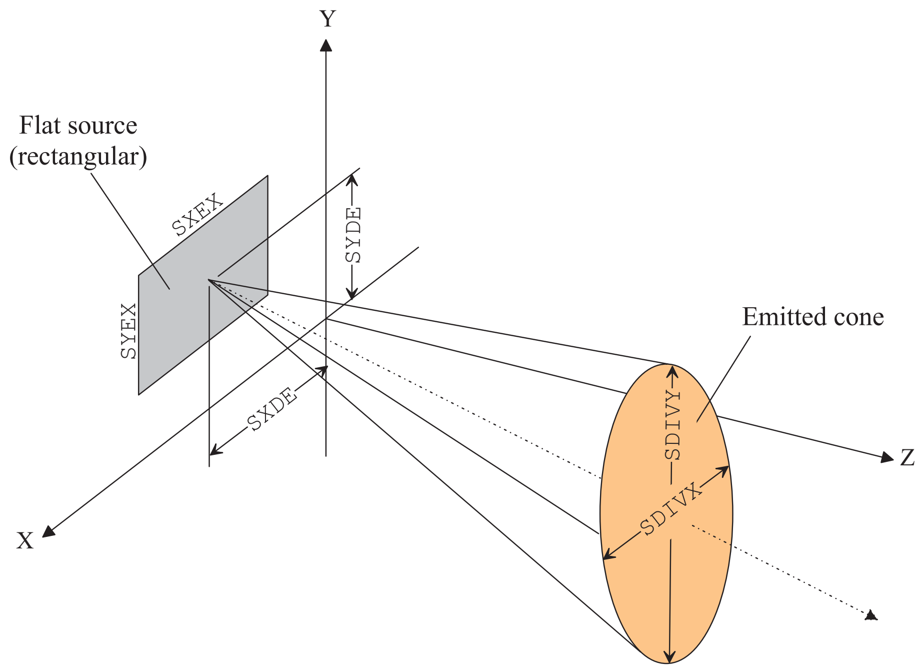


Figure 15.5: Definition of flat (surface-like) sources.

15.3.5 Sources defined by Rays

A volume source models any real-world source such as an incandescent lamp, LED, or laser diode. Instead of defining a precise geometrical model, the radiant source is modelled in $OpTaliX$ by a three-dimensional space-angular source characterization in terms of a collection of rays, in the following called *ray source*.

Individual rays in a ”ray source file” are defined by spatial ray coordinates (X, Y, Z), direction cosine (α, β, γ), intensity and wavelength, stored in a user supplied file. Rays provided in a ”ray source file” must obey to the file format as given in sect. 32.13.

Ray sets (i.e. a collection of rays) defining a source may also be generated from third party software provided by other vendors, such as

- Radiant Imaging^{TM} source files using the ProSource^{TM} software [44]. For more information about ProSource^{TM} ray files see section 15.3.9, page 305,
- ”Luca raymaker” software provided by Opsira, Germany [40]. For more information about ray files generated by ”Luca raymaker” see section 15.3.10, page 306,
- DIS-files. These are ray files in a binary format originally defined and mainly used in the optical analysis package ASAP. The typical extension of these files is ’`*.dis`’, hence the name DIS-files,
- or provide ray sets in a text file using any standard (ASCII) editor. The file format is explained in section 32.13, page 507.

Rays emanating from a source are assumed to be located either at the object coordinate system or the global coordinate system. Sect. 5.2 (page 27) describes these coordinate systems.

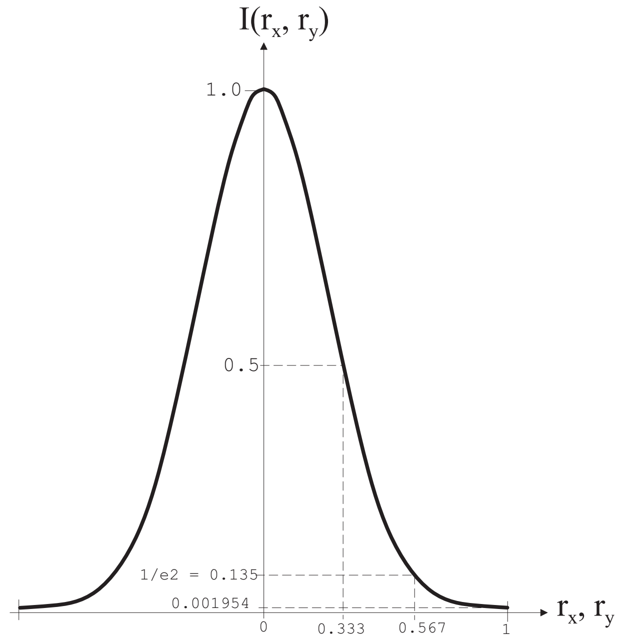


Figure 15.6: Definition of a Gaussian source. The profile is scaled to the source extensions `SXEX`, `SYEX`.

15.3.6 Source Rays aimed to System Entrance Pupil

In some cases it might be difficult to trace a sufficient number of rays emitted from an illumination source through the optical system. In the wide-angle system, as illustrated in Fig. 15.8, the majority of the emitted rays are wasted because they don't pass through the narrow entrance aperture. One would normally increase the number of source rays in order to obtain a decent number of rays at the target (image) surface.

In order to avoid this inefficient situation, a second option is offered by which rays from the extended source are directly aimed to the entrance pupil, instead of blindly launched from the source within the specified emittance cone. In the illumination ray aiming option, the source emittance characteristics is then completely ignored. Fig. 15.9 indicates the definition of illumination rays for this option.

Note that this option requires dedicated selection on how the source itself and the entrance pupil are sampled. For example (compare with Fig. 15.9),

Object sampling = 50: Divides the source area in 50 x 50 cells from which source rays are randomly generated.

Pupil sampling = 8: From each object cell, 8 x 8 rays are aimed to the entrance pupil.

In total, 50x50x8x8 = 160000 rays are then used for each particular source and wavelength.

15.3.7 Ray Source Viewer

"*Ray sources*" are sources defined by a collection of rays. The ray data is stored in plain ASCII files. Even though the data may be viewed in conventional ASCII editors, typically the sheer amount of data prevents a thorough understanding and interpretation of the source itself. The "*ray source viewer*" option provides a means for visualizing this data.

In addition to only viewing ray data, ray sets may also be transformed (shifted, rotated) and subsequently saved as a new ray file.

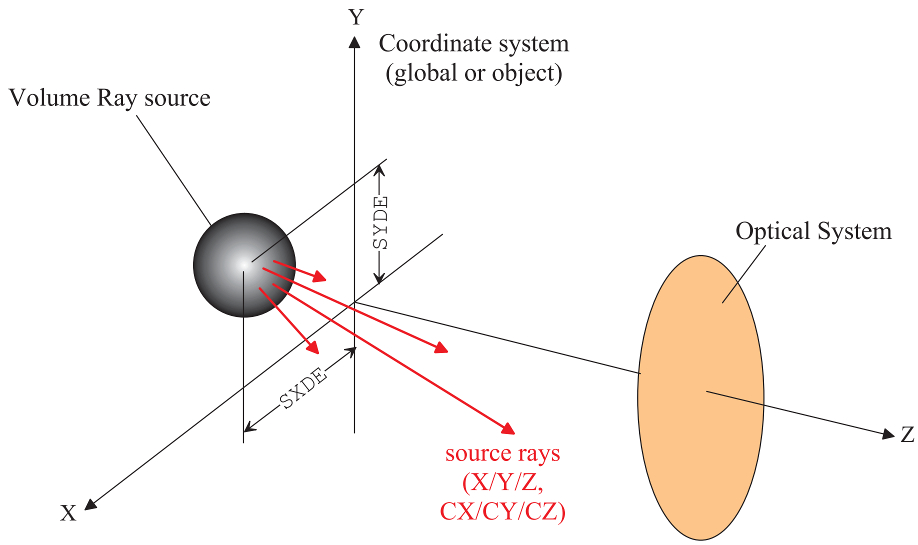


Figure 15.7: Coordinate system for defining rays in a "ray-source" model.

The ray source viewer is invoked from the command line by

`VIE SRC FIL source_file`	View ray source defined in `source_file`. The file name of the ray source may have extensions `*.txt`, `*.dat`, or `*.ray` for plain ASCII formats, respectively `*.dis` for the ASAP binary format. Other ray formats will be added later.

or from the main menu: *Display − > Ray Source Viewer*. A dialog box is invoked which allows viewing orientation (azimuth, elevation), zoom, and visualization of arrows indicating the ray direction.

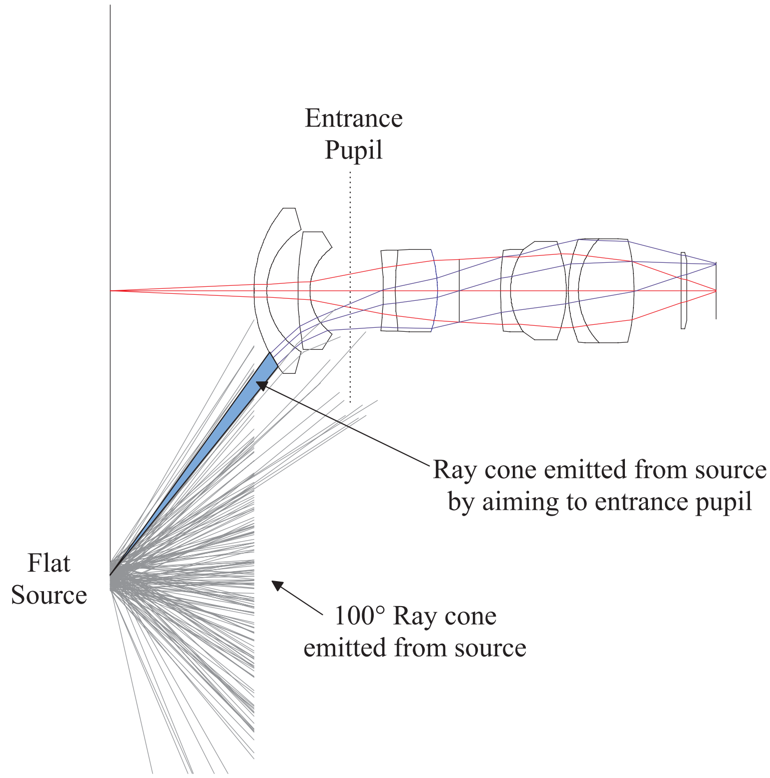


Figure 15.8: Aiming source rays to entrance pupil.

Figure 15.11 shows the source represented by ray coordinates and ray directions proportional to the ray intensity.

15.3.8 Transforming Ray Data

Source rays may be arbitrarily transformed in 3D-space. This is accomplished from within the ray source viewer dialog (see previous section).

Note that applying a transformation is cumulative. In order to 'undo' a transformation you must apply shift/rotate parameters with reverse sign. If more that one transformation (e.g. shift + rotation) is simultaneously applied and if you want to undo (reverse) this operation, you should keep in mind that coordinate transformations are not commutative (i.e. depend on order of operation). From this point of view it is advisable to apply only one parameter at a time. The result of ray transformation is then immediately visible in the ray source viewer.

Once transformed, ray data may also be stored in a separate file for later use. Select a file name and export the transformed ray data by pressing the Export button in the dialog shown above (Fig. 15.12). Two different output formats are provided, ASCII or binary. Note that the binary file format for $OpTaliX$ ray sources is compatible to the ASAP binary format.

15.3.9 Generating Source Rays from ProSource^{TM} Software

$OpTaliX$ allows import of (ray) source files generated by the ProSource^{TM} software from Radiant Imaging [44]. For a specific type of source (lamp, LED, laser diode, etc), the ProSource^{TM} software offers the option to export a collection of rays representing the source irradiation in a file (ASCII or binary).

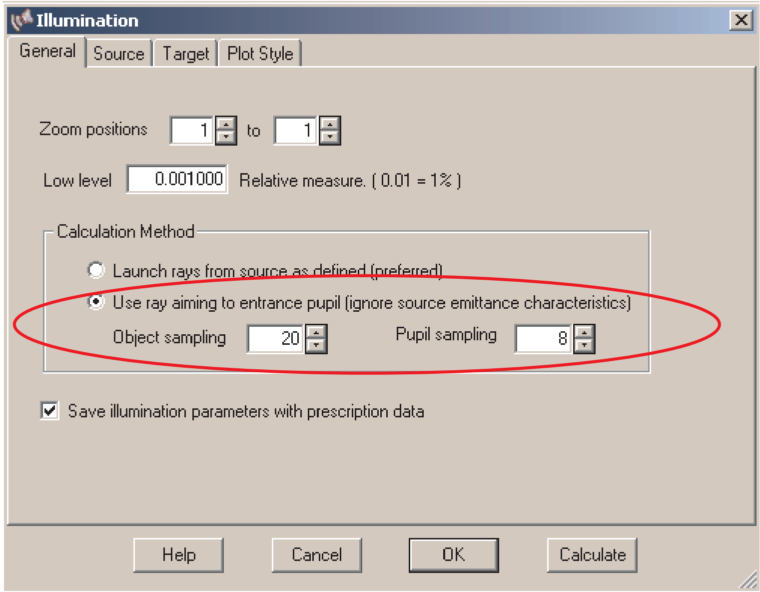


Figure 15.9: Selecting alternative illumination option: Aiming source rays to the entrance pupil directly. The number of rays traced per source is $(objectsampling)^2 \cdot (pupilsampling)^2 \cdot NumberOfColours$

Note that in the current implementation, $OpTaliX$ **only** accepts ray source files given in ASCII format(extensions `*.txt`, `*.dat`, `*.ray`) and in ASAP binary format (extension `.dis`)!

For optimal conversion of ProSource^{TM} sources to $OpTaliX$ the following settings in the ProSource^{TM} viewer software are recommended (see also Fig. 15.13):

- ASCII format: generic
- Coordinate system: left handed
- Ray origination: undefined

15.3.10 Generating Source Rays from ”Luca Raymaker” Software

$OpTaliX$ allows import of (ray) source files generated by the ”Luca raymaker” software from Opsira GmbH [40]. For a specific type of source (lamp, LED, laser diode, etc), the ”Luca raymaker” software offers the option to export a collection of rays representing the source irradiation in various file formats.

Note that in the current implementation, $OpTaliX$ **only** accepts ray source files given in ASCII format(extensions `*.txt`, `*.dat`, `*.ray`) and in the ASAP^{TM} binary format (extension `.dis`)!

15.4 Illumination Analysis Options

`ILL [?]`	Runs illumination analysis. The question mark invokes a dialog box for setting of parameters prior to analysis.
	continued on next page

continued from previous page	
`ILL SAV Y|N`	Store illumination data with prescription data, Y=yes, N=no.
`ILL EXP FIL out_file [RAW|INT|XLS]`	Save irradiance distribution at target surface to file. Requires preceding illumination analysis (use ILL command above). The full path specification including file extension must be given. The specific file format is recognized by one of the file extensions, RAW (raw file, ASCII-format), TXT (raw file, ASCII-format), INT (interferogram file) or XLS (Excel file). The default file format option is RAW.
`ILL TAR sk`	Target surface for illumination. This is the surface at which the irradiance distribution is computed. Examples: `ill tar s5` ! Illumination target surface is 5, `ill tar si` ! Illumination target surface is image surface.
`ILL IMX x_ext`	X-image extension (full width) of analysis region at target surface.
`ILL IMY y_ext`	Y-image extension (full width) of analysis region at target surface.
`ILL NXI X_Img_Cells`	Divides the image (target) extension `IMX` into `NXI` cells.
`ILL NYI Y_Img_Cells`	Divides the image (target) extension `IMY` into `NYI` cells.
`ILL FIL out_file [RAW|INT|XLS]`	Save irradiance distribution at target surface to file. The full path specification must be given. The specific file format is defined by one of the (optional) parameters, RAW (raw file), INT (interferogram file) or XLS (Excel file). The default file format option is RAW.
`RPWR`	Database item: Return total received power, including all activated sources, using illumination ray tracing. Example: `eva [rpwr]`
`EPWR`	Database item: Return emitted power from all activated sources. Example: `eva [epwr]`
`NILR`	Database item: Return number of successfully received illumination rays at target surface, including all active sources. Example: `eva [nilr]`

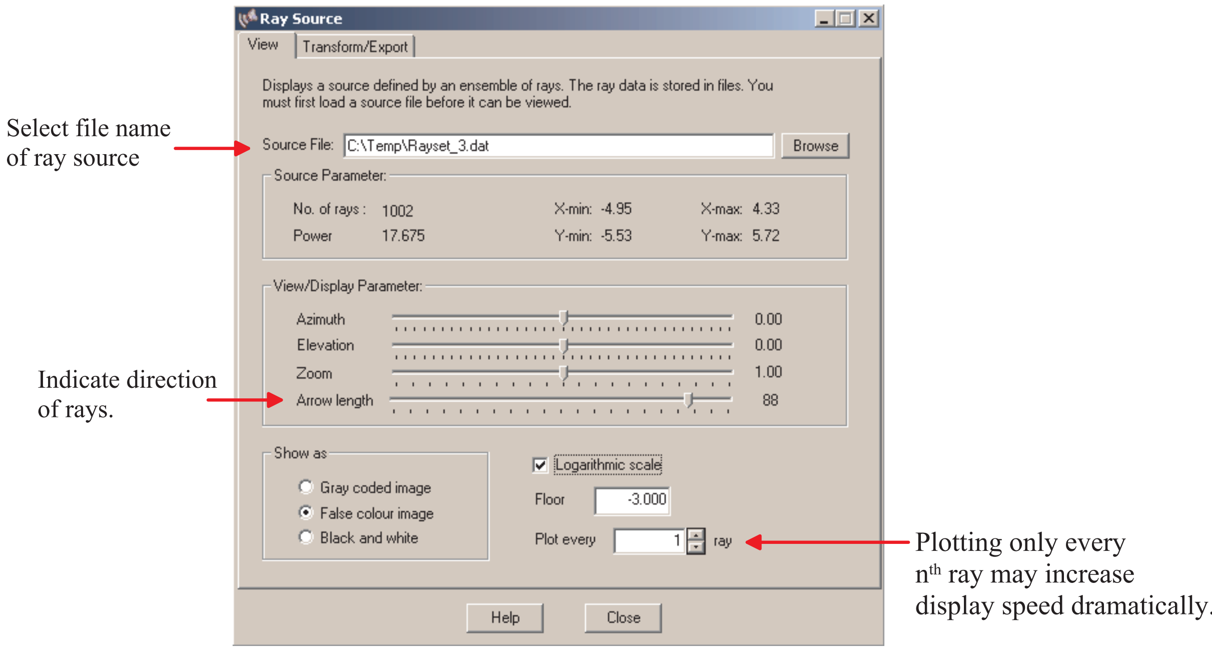


Figure 15.10: Dialog for visualizing ray source data.

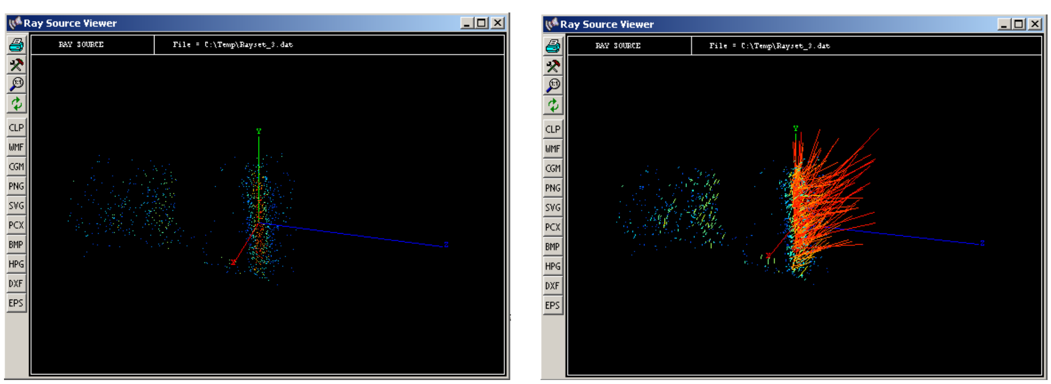


Figure 15.11: Visualization of ray data. Left: Shows ray coordinates only (arrow length = 0), right: Arrow length ≥ 0. The length of the arrows indicates relative intensity of the rays.

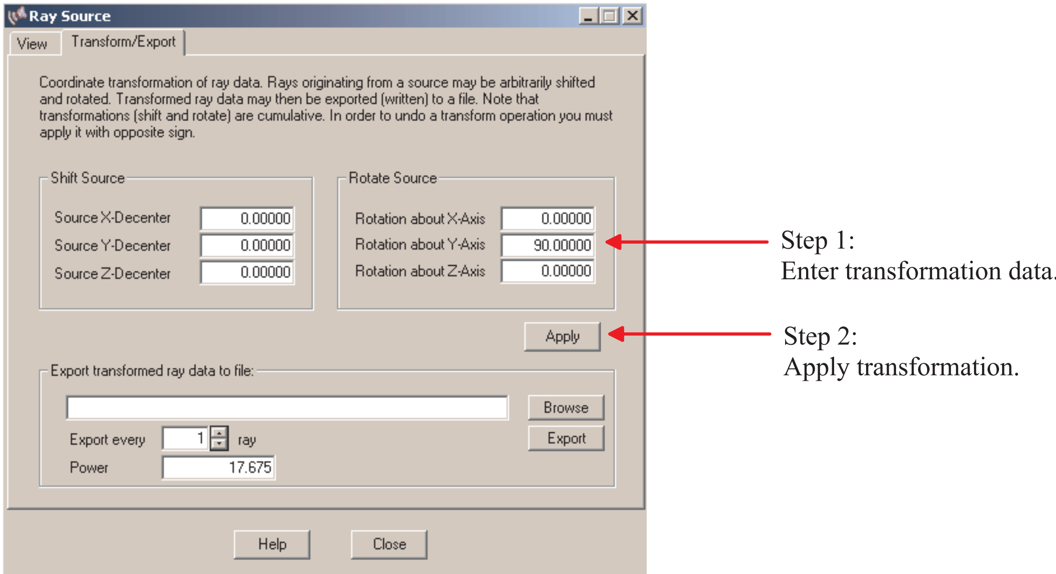


Figure 15.12: Transformation (shift, rotate) of ray data.

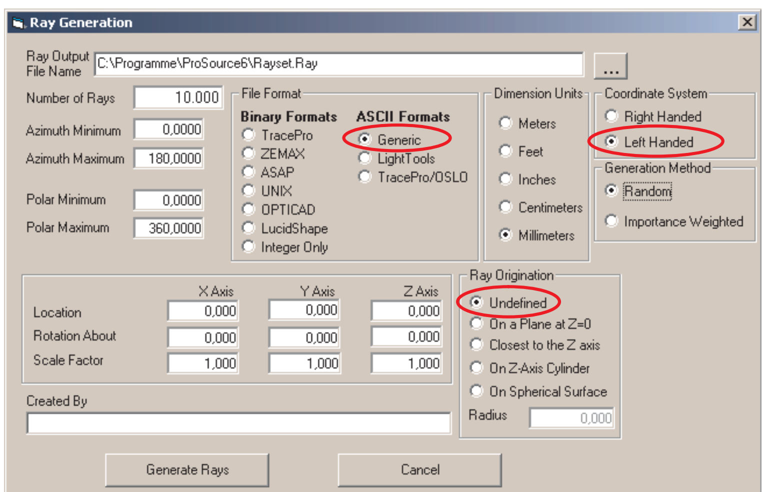


Figure 15.13: ProSource^{TM} dialog. The red ellipses indicate the preferred settings for exporting rays to $OpTaliX$

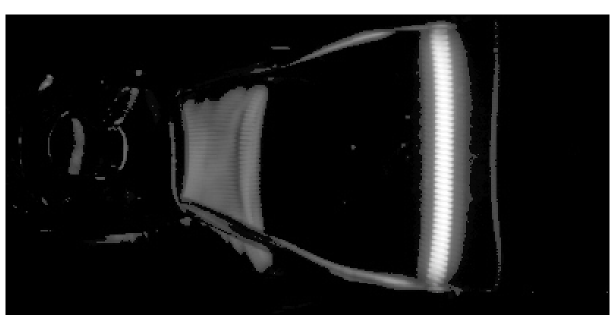


Figure 15.14: Image of incandescent lamp as displayed by the ProSource^{TM} software.

16

Physical Optics Propagation

(Diffraction Based Beam Propagation)

Optical modelling consists largely of geometrical ray tracing in which the light is represented by a set of rays which are normal to the wavefront. Diffraction effects in ”conventional” systems, such as a photographic objective, are small and localized to the edge of the beam. Rays are used to determine the pupil function and do a far-field diffraction analysis. This is a fast and well established method to calculate diffraction PSF and MTF, as described in sections 14.2.1 and 14.2.2.

This method, however, breaks down if noticeable diffraction occurs inside optical systems. A common example is a simple spatial filter (pinhole) located at the focal point of a laser system. Ray optics is unable to predict removal of the phase aberrations by the pinhole. Also, it cannot account for the beam spreading of Gaussian beams. In this context, note that the Gaussian beam analysis (BEA) as described in section 14.3 only models *paraxial* quantities of ideal Gaussian bemas and does not include wave aberrations.

For such cases, physical optics methods must be used. It models a *coherent* optical beam by a complex-valued function (amplitude and phase), describing the transverse beam distribution. In the computer, the beam is represented by a complex 2-dimensional array of discretely sampled points. The entire array (beam) is then propagated through the optical system. This approach is also commonly called *diffraction based beam propagation*.

Physical optics propagation is based on several algorithms, which are described in the following sections. For a detailed study of the underlying physical principles, see Goodman [17].

16.1 Propagation of the Angular Spectrum

If the complex field (amplitude and phase) is Fourier-transformed across any plane, the various spatial Fourier components can be considered as plane waves travelling in different directions. The field across any other plane can be calculated from the phase shifts these plane waves have undergone during propagation.

Let us assume a wave field $U(x, y, z_1)$ incident on a plane and we wish to obtain the resulting field $U(x, y, z_2)$ across a second, parallel plane at distance z to the right of the first plane. At the $z = 0$ plane the two-dimensional Fourier transform ($\mathcal{F}$) of the field U is given by

$$A(f_x, f_y, 0) = \iint_{-\infty}^{\infty} U(x, y, z_1) e^{-2\pi j(f_x x + f_y y)} dx dy \tag{16.1}$$

and correspondingly U can be obtained from the inverse Fourier transform ($\mathcal{F}^{-1}$) of its spectrum,

$$U(x,y,z_1) = \iint_{-\infty}^{\infty} A(f_x,f_y,z_1)e^{2\pi j(f_x x + f_y y)}df_x df_y \tag{16.2}$$

Physically the integrand of Eq. 16.2 can be interpreted as a plane wave propagating with wave vector $\overrightarrow{k}$ with magnitude $2\pi/\lambda$. It has direction cosines (α,β,γ) as shown in Fig. .... The complex phasor amplitude of the plane wave across a constant z-plane is given by

$$P(x,y,z) = e^{j\overrightarrow{k}\overrightarrow{r}} = e^{\dfrac{2\pi j}{\lambda}(\alpha x + \beta x)} \tag{16.3}$$

The complex exponential function $e^{2\pi j(f_x x + f_y y)}$ may be regarded as representing a plane wave propagating with direction cosines

$$\alpha = \lambda f_x \tag{16.4}$$
$$\beta = \lambda f_y \tag{16.5}$$
$$\gamma = \sqrt{1-(\lambda f_x)^2 - (\lambda f_y)^2} \tag{16.6}$$

The complex amplitude of the plane wave component is evaluated in the Fourier domain of U at the spatial frequencies $f_x = \alpha/\lambda$, $f_y = \beta/\lambda$. Hence, the function

$$A\left(f_x,f_y,z_1\right) = \iint_{-\infty}^{\infty} U(x,y,z_1)e^{-2\pi j(f_x x + f_y y)}dxdy \tag{16.7}$$

is called the angular spectrum of the field $U(x,y,z_1)$. The angular spectrum of U across a plane parallel to the z_1 plane but at a distance z from it is written in the form

$$A\left(f_x,f_y,z_2\right) = A(f_x,f_y,z_1)exp\left[\frac{2\pi j}{\lambda}\Delta z\sqrt{1-(\lambda f_x)^2-(\lambda f_y)^2}\right] \tag{16.8}$$

Thus, propagation of a complex field from one plane to another can be written in terms of operators for Fourier transform $\mathcal{F}\{U(z_1)\}$ and free space propagation $\mathcal{T}\{z_2 - z_1\}$

$$U(z_2) = \mathcal{F}^{-1}\left[\mathcal{T}\{z_2 - z_1\}\mathcal{F}\{U(z_1)\}\right] \tag{16.9}$$

This is a straightforward procedure in which the input field is Fourier transformed (i.e. decomposed into its frequency components), the plane wave propagator applied (i.e. adding the relative phases of the components of the angular spectrum) and then the resulting distribution inverse Fourier transformed. Since the angular spectrum method can only propagate a field between parallel planes, we will subsequently refer to it as the plane-to-plane (PTP) operator.

The direction cosines of the plane waves must satisfy the condition

$$\alpha^2 + \beta^2 < 1 \tag{16.10}$$

otherwise evanescent waves are obtained, which are not covered by the angular spectrum model.

16.2 Propagation using the Fresnel Approximation

In the Fresnel approximation the field $U(x, y, z_2)$ is calculated from the initial field $U(\xi, \eta, z_1)$ where the propagation distance is $\Delta z = z_2 - z_1$. The field is given by

$$U(x, y, z_2) = \frac{e^{jkz_2}}{j\lambda\Delta z} e^{\frac{jk}{2\Delta z}(x^2+y^2)} \iint_{-\infty}^{\infty} \left\{ U(\xi, \eta, z_1) e^{\frac{jk}{2\Delta z}(\xi^2+\eta^2)} \right\} e^{-j\frac{2\pi}{\lambda\Delta z}(\xi x+\eta y)} d\xi d\eta \tag{16.11}$$

This is the Fourier transform of the complex field at the initial plane multiplied by a quadratic phase exponential. It can also be written in operand notation

$$U(z_2) = \left[\frac{e^{jkz_2}}{j\lambda\Delta z}\right] \mathcal{Q}\{x, y, \Delta z\} \mathcal{F}\left[\mathcal{Q}\{\xi, \eta, \Delta z\} U(\xi, \eta, z_1)\right] \tag{16.12}$$

where $\mathcal{Q}\{\xi, \eta, \Delta z\} = e^{\frac{jkr^2}{2\Delta z}}$ is the quadratic phase exponential with $r^2 = \xi^2 + \eta^2$. The term $\mathcal{Q}\{\}$ outside the integral may be omitted if the resultant field is referred to a sphere of radius z instead a plane. At this point it is worthwhile to remember that the field is actually defined on a parabola (quadratic approximation), however, within the scope of the Fresnel approximation we have already assumed $(\xi, \eta) << z$. Referring the phase to a sphere is the preferred choice, since the phase variations are much smaller rather than referring the field to a plane. Eq. 16.12 can now be redefined as the waist-to-sphere (WTS) operator

$$U(z_2) = \left[\frac{e^{jkz_2}}{j\lambda\Delta z}\right] \mathcal{F}^s\left[\mathcal{Q}\{\xi, \eta, \Delta z\} U(\xi, \eta, z_1)\right] \tag{16.13}$$

and

$$s = \frac{\Delta z}{|\Delta z|} \tag{16.14}$$

The sphere-to-waist (STW) propagation is obtained by reversing the operations,

$$U(z_2) = \left[\frac{e^{jkz_2}}{j\lambda\Delta z}\right] \mathcal{Q}\{x, y, \Delta z\} \mathcal{F}^s\left[U(\xi, \eta, z_1)\right] \tag{16.15}$$

Note that the term e^{jkz_2} in equations 16.13 and 16.15 can normally be neglected, since it is a constant phase propagation term.

Using a Fast Fourier Transform (FFT) algorithm and representing the field in a two-dimensional complex-valued array, the sampling period at the z_2 plane or sphere is not constant but scales linearly by

$$\Delta x = \frac{\lambda|\Delta z|}{N\Delta\xi} \tag{16.16}$$

where N is the number of sampling points in the array.

16.3 Propagation through Optical Interfaces

The angular spectrum and Fresnel propagators are used for propagating through homogeneous space. At optical interfaces the complex transmittance function of optical elements (lenses, diffractive surfaces, aspheres, etc) are required to calculate the complex field after the element. Since these functions are not analytically known (except in the strict paraxial approximation), a combination of classical ray tracing and wave optics is used. This requires conversion of the field after free space propagation into rays, doing refraction/reflection at the optical interface and converting the resultant rays back into the complex field description.

16.3.1 Converting Field into Rays

The field is assumed at a sphere or plane, which is the result from a previous propagation operator (angular spectrum or Fresnel). The complex wave amplitude at the coordinates (x, y) in a two-dimensional array of data points is given by

$$U(x_m, y_n) = a(x_m, y_n)e^{j\Phi(x_m,y_n)} \tag{16.17}$$

where a is the amplitude and Φ is the phase in $2\pi/\lambda$ units. The coordinates (x_m, y_m) are assumed to form an equidistant mesh. Since the wave-optical propagation delivers the phase modulo 2π, a phase unwrapping algorithm must be used. This is, in the absence of noise, a straightforward operation. Following an arbitrary continuous path through the gridded data, the following decision rule is applied:

$$\Phi_{k+1} = \begin{cases} \Phi_k + \Delta_k - 2\pi & \text{if } \ \Delta_k > \pi \\ \Phi_k + \delta_k + 2\pi & \text{if } \ \Delta_k < \pi \\ \Phi_k + \delta_k & \text{else} \end{cases} \tag{16.18}$$

where k is the path index and Δ_k is is the adjacent-pixel phase difference. From the unwrapped phase the ray direction vector $\overrightarrow{v}$ is obtained by

$$\overrightarrow{v} = \frac{\lambda}{2\pi}\left[\frac{\partial\Phi}{\partial x}, \frac{\partial\Phi}{\partial y}, \sqrt{\left(\frac{\lambda}{2\pi}\right)^2 - \left(\frac{\partial\Phi}{\partial x}\right)^2 - \left(\frac{\partial\Phi}{\partial y}\right)^2}\right] \tag{16.19}$$

16.3.2 Transfer at Optical Interfaces

Starting from the input reference sphere, the ray is traced through the optical interface to the output reference sphere using geometric optics techniques. See also Fig. 16.1. Generally, input sphere and output sphere will be in the immediate vicinity of the optical interface.

The phase Φ is derived from the path length L of the ray between input reference and output reference and is added to the complex input field.

$$L = \frac{2\pi}{\lambda}\sum n_i \cdot L_i \tag{16.20}$$

where n_i is the index of refraction along the sub-path L_i. The total optical path may include a single optical interface or even a series of interfaces (surfaces).

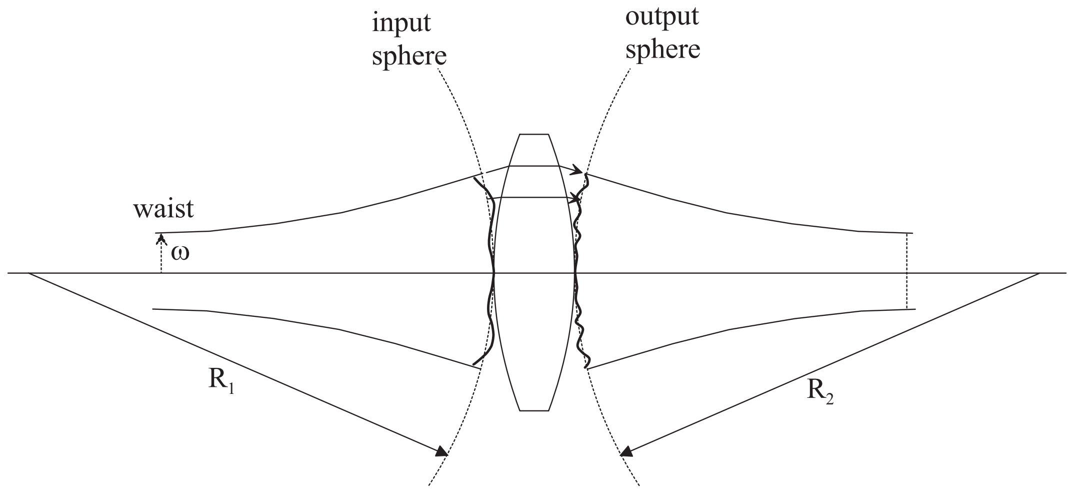


Figure 16.1: Relationship between diffraction-based beam propagation and geometrical ray tracing shown at the example of a Gaussian beam.

16.3.3 Converting Rays into Field

The phase $\Delta\Phi$ introduced in the geometric ray trace section of the path is derived from the optical path length between input sphere and output sphere and is added to the phase component of the complex field. Real and imaginary parts of the output field are then obtained by

$$R = a(x_m, y_n)\cos(\Phi + \Delta\Phi) \quad (16.21)$$

$$I = a(x_m, y_n)\sin(\Phi + \Delta\Phi) \quad (16.22)$$

If the output mesh is substantially distorted, resampling of the data points into a rectangular grid must be performed.

16.4 Propagation Control

Surrogate Gaussian beams are used to determine the algorithms to be used. These beams are considered to represent approximately the actual beam and since they have an easily calculated width at all points in space, they allow a convenient method of determining the size of the two-dimensional array holding the field data. Any complex input field may be approximately fit to a Gaussian beam of radius ω and phase radius R. From these values, the Gaussian waist size ω_0 and the distance to the waist z_w are calculated. The radius R_1 of the input sphere is then obtained by

$$R_1(z) = z\left[1 + \left(\frac{\pi\omega^2}{\lambda z}\right)\right] \quad (16.23)$$

where z is the distance from the waist. The radius R_2 is calculated by the lens law

$$\frac{1}{R_1} - \frac{1}{R_2} = \frac{1}{f} \quad (16.24)$$

where f is the focal length of the optical interface. Since the beam spreads due to diffraction it may overfill the array. Fortunately, near-field propagators (angular spectrum) and far-field propagators

(Fresnel) may be combined to control the size of the array so that aliasing due to the finite sampling is sufficiently suppressed. The sampling period of the near-field (angular spectrum) propagator is constant, while the sampling period of the far-field (fresnel) propagator scales linearly with propagation distance Δz according to Eq. 16.16. An appropriate transition point from a constant sampling period to a linearly scaling sampling period is chosen by the Rayleigh range $z_R = \omega_o^2\pi/\lambda$. This choice minimizes the phase error if a plane reference inside the Rayleigh distance and a spherical reference outside the Rayleigh distance is selected. Fig. 16.2 indicates the array sizes inside and outside the Rayleigh range.

The control of the propagation algorithm should allow movement from any point in space to any other. To do so the previously defined primitive operators, plane-to-plane (PTP), waist-to-sphere (WTS) and sphere-to-waist (STW) are appropriately combined. We define four new operators, which cover all possible cases (see also Fig. 16.2)

$$\begin{array}{lcll}
\mathbf{II}(z1,z2) & = & \mathbf{PTP}(z_2 - z_1) & \text{inside } z_R \text{ to inside } z_R \\
\mathbf{IO}(z1,z2) & = & \mathbf{WTS}(z_2 - z_\omega)\mathbf{PTP}(z_\omega - z_1) & \text{inside } z_R \text{ to outside } z_R \\
\mathbf{OI}(z1,z2) & = & \mathbf{PTP}(z_2 - z_\omega)\mathbf{STW}(z_\omega - z_1) & \text{outside } z_R \text{ to inside } z_R \\
\mathbf{OO}(z1,z2) & = & \mathbf{WTS}(z_2 - z_\omega)\mathbf{STW}(z_\omega - z_1) & \text{outside } z_R \text{ to outside } z_R
\end{array} \tag{16.25}$$

The primitive operators are defined in equations 16.9, 16.13 and 16.15 respectively.

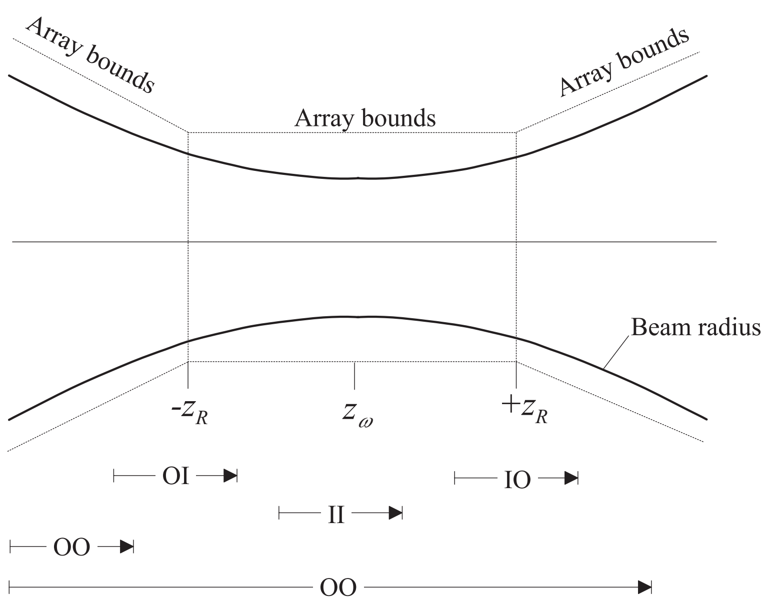


Figure 16.2: Variation of array size inside and outside the Rayleigh range. The four different possibilities in propagating inside/outside the Rayleigh range are indicated by the acronyms II, IO, OI, OO.

For practical usage of the algorithms described above, three major issues should be considered:

- The sampling interval,
- the oversizing of the array relative to the beam size,
- and the use of reference surfaces.

The sample spacing Δx and Δy determines the highest spatial frequency, which can be represented. The region of space which is covered by the whole array is $M\Delta x$ and $N\Delta y$, where M, N are the number of sample points in x- and y-direction. The sample spacing and the array size should be chosen as to overfill the beam by a factor 3-5. The choice of this factor depends largely on the profile of the input beam. For Gaussian profiles a factor 3 may be appropriate while for top hat functions factors of 5-10 are recommended. If the width of the array is too small, aliasing will occur. Aliasing is due to the discrete sampling and the finite extent of the computer arrays. Because of propagation a collimated beam expands and the field may grow beyond the array bounds. The portions of the beam which fall outside the array then ”fold back” and will cause aliasing.

16.5 Command Overview

`EDI BPR`	Invokes a dialog box for editing beam propagation parameter. Currently, parameters can only be defined in the dialog, there are no equivalent commands yet. See a detailed description of the relevant parameters in the following section 16.6 (Propagation Parameters).
`BPR`	Executes beam propagation and displays resulting field.

16.6 Propagation Parameters

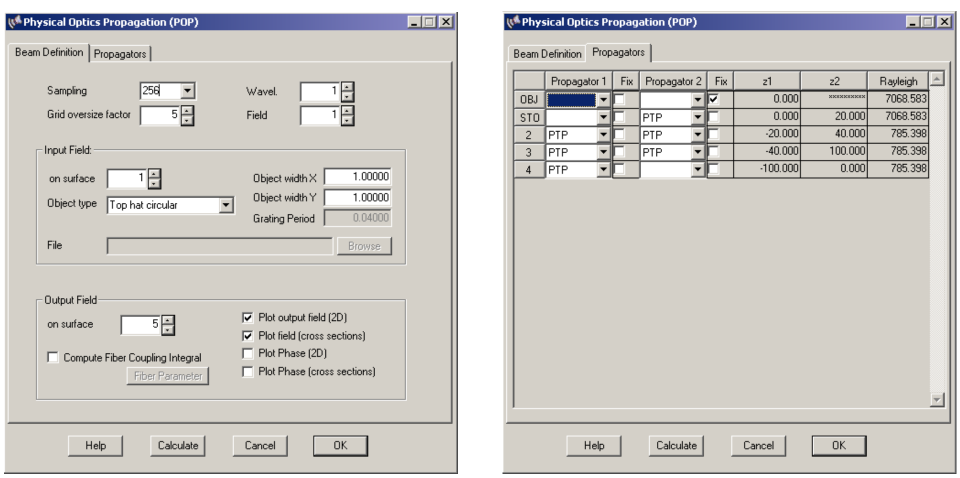


Figure 16.3: Parameter dialogs of the free-space propagation example.

The first tab of the dialog (see Fig. 16.3), labelled ’Beam Definition’, defines the parameter of the beam and other auxiliary propagation parameter.

Beam Definition Tab:

Sampling: Defines the number of sampling points across the data grid. This number is somewhat arbitrary, however, it should be noted that accuracy of simulation increases with higher numbers. Low numbers (≤ 128) should only be selected if there is little high spatial frequency content in the source profile (such as Gaussian) and if little spreading of the beam is expected. 'Top-hat' profiles contain relatively high spatial frequency components (due to the sharp edge) and therefore sampling numbers ≥ 256 should be selected. Also note that computing time goes with the square of the sampling number, that is computing time is 4 times higher on 256 sampling points as compared to a 128 sampling.

Grid oversize factor: Defines the physical size of the array in relation to the beam dimensions. The array must always be larger as to overfill the beam by the grid oversize factor and ensures that all frequency components of the beam profile are contained in the array. This factor also depends on the beam profile. Typical values are 3-5 for Gaussian beams, 8-10 for 'top-hat' profiles.

Object type: Select from several predefined profiles. (Import from a file not yet functional).

Object width: Specifies the maximum physical extension of the source beam in X-direction and Y-direction respectively. The physical extension of the array used in beam propagation is then 'grid oversize factor' * max(object_width_X, object_width_Y).

Input field surface: The surface number where the source beam (object) is placed and where from the propagation starts.

Grating period: This field is only accessible for amplitude grating sources and defines the grating period (one cycle) in X-direction.

Output field surface: The surface number at which the the propagation is terminated and the field components are displayed.

Fiber Coupling Integral: Takes the resultant field and convolves it with the profile of a receiving fiber in order to compute coupling efficiency.

Propagators Tab:

Propagator: There are 5 types of propagators:

PTP: Plane-to-Plane. Uses the angular spectrum method (sect. 16.1) to propagate a field from a plane surface over a distance z to another plane surface.

WTS: Waist-to-Sphere. Propagates a field defined on a plane surface (near the beam waist) over a distance z to a spherical (reference) surface, using the Fresnel approximation (sect. 16.2). The distance z must be 2 times larger than the Rayleigh range.

STW: Sphere-to-Waist. Propagates a field defined on a spherical surface (far from the beam waist) over a distance z to a plane (reference) surface, using the Fresnel approximation (sect. 16.2). The distance z must be 2 times larger than the Rayleigh range.

Ray: Does a conventional ray trace (ignores diffraction) over the distance z. This propagator is used in GRIN media (where FFT propagation fails) or where diffraction effects are expected to be neglected. This speeds up calculation.

Blank: A blank field means, no propagation is performed.

Fix: If checked, fixes (freezes) propagator selection and overrides automatic propagator selection. See also notes below.

Notes:

The program traces a pilot ray through the optical system. This is a paraxial Gaussian beam and allows very rapid finding of the location of waists with respect to surfaces, calculation of Rayleigh range and calculation of the reference spheres/planes at the optical surfaces. On this basis, the best propagator is selected and displayed in the dialog box (see Fig. 16.3, right). This selection can be overruled by the user by checking the appropriate check boxes in the columns 'Fix 1' and 'Fix 2' respectively.

Propagation between surfaces is typically performed in two steps, using two propagators successively. To illustrate the point, consider Fig. 16.4

Since there is no Sphere-to-Sphere propagator (yet), the field is first propagated from the reference sphere at surface 2 to the waist location over the distance z_1, using a STW (sphere-to-waist) propagator. From this location the field is propagated to the reference sphere at surface 3 over the distance z_2 (in negative direction).

This is why two propagators are offered for each surface in the BPR dialog (Fig. 16.3). The Rayleigh range z_R is a convenient measure for selecting the appropriate propagator.

$$z_R = \omega_o^2 \pi / \lambda \tag{16.26}$$

where ω_o is the beam radius (semi-diameter). The Rayleigh range indicates that axial range around the waist where the field (the wavefront) may still be considered with good accuracy as plano. Outside the Rayleigh range, beam spreading and wavefront curvature are noticeable. We also refer to the operators description in Eq. 16.25 and Fig. 16.2 to describe the four possible cases of propagation.

The simplest case is the 'inside-inside' (**II**) case. That is, propagation distance z is less than the

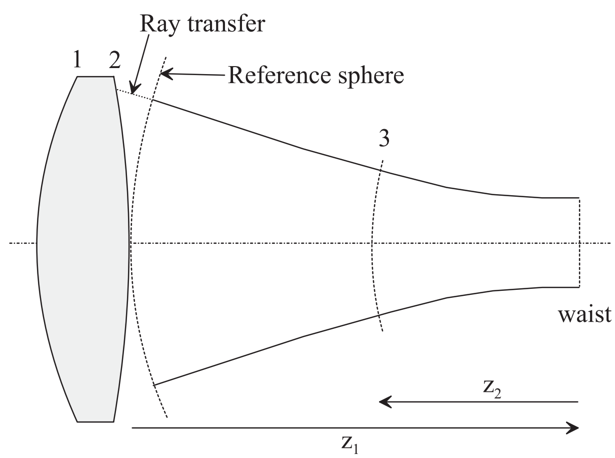


Figure 16.4: Propagation from surface 2 to 3.

Rayleigh range ($-z_R$ to $+z_R$). The radius of the wavefront is infinity or nearly infinity. Thus, a beam travelling inside this range may be well modelled by the angular spectrum method, which propagates between plano (infinity radius of curvature) surfaces. Therefore, this propagator is called **PTP** (plane-to-plane).

If the propagation distance is larger than the Rayleigh range z_R, the **IO** case ('inside-outside'), respectively the **OI** case ('outside-inside') apply. The radius of the wavefront at the start surface (OI case) respectively at the receiving surface (IO case) is no longer infinity. The Fresnel approximation is now used as propagator, which propagates a field from a sphere to a waist (STW) respectively from a waist to a sphere (WTS).

16.7 Examples

The examples to follow give a step-by-step introduction to propagating coherent (monochromatic) beams through optical systems. All the $OpTaliX$ files referred to in the subsequent sections are found in the examples directory `\optalix\examples\pop\`

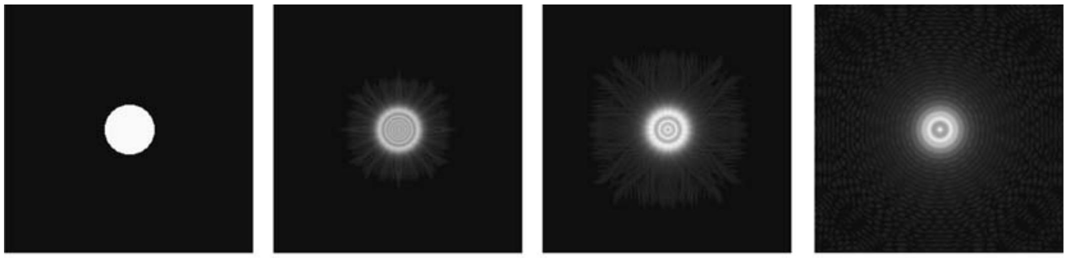
16.7.1 Free-Space Propagation

Fig. 16.5 shows the optical setup for propagating a plane wave over a certain distance in free space. The predefined $OpTaliX$ file is found under `\optalix\examples\pop\freespace.otx`. The input field is a 'top-hat' amplitude profile defined by a circular screen (aperture) of 1mm diameter on surface 1. We will calculate the field at the subsequent surfaces 2-5, which are placed at various distances to the screen (surface 1).

Figure 16.5: Optical setup for simple free-space propagation

The BPR dialog (click on the BPR icon underneath the main menu or enter `EDI BPR` in the command line) shows suitable predefined parameter for this example: The beam starts at surface 1 with a diameter of 1mm and a circular 'top-hat' amplitude profile. Since the we start with a plane wave the waist is also at surface 1. The size of the grid array is 256 x 256 and it overfills the beam by a factor 5.

The output surface, i.e. the surface on which the output field is displayed may be freely selected between 1 and 5. The resulting fields are shown in Fig. xxx.



Surface 1, z = 0mm   Surface 2, z = 20mm   Surface 3, z = 40mm   Surface 4, z = 100mm

Figure 16.6: Fields at various propagation distances.

16.7.2 Talbot Imaging

The Talbot imaging phenomenon is present for any periodic structure. At a specific distance, defined by the wavelength and the period of the periodic structure (typically an amplitude grating), a perfect image is obtained. A multiplicity of such images appear behind the grating, without the help of lenses. The z-locations at which the perfect image (also called a self-image) can be observed must satisfy the condition

$$z = \frac{2nL^2}{\lambda} \tag{16.27}$$

where L is the period of the periodic structure and n is an integer.

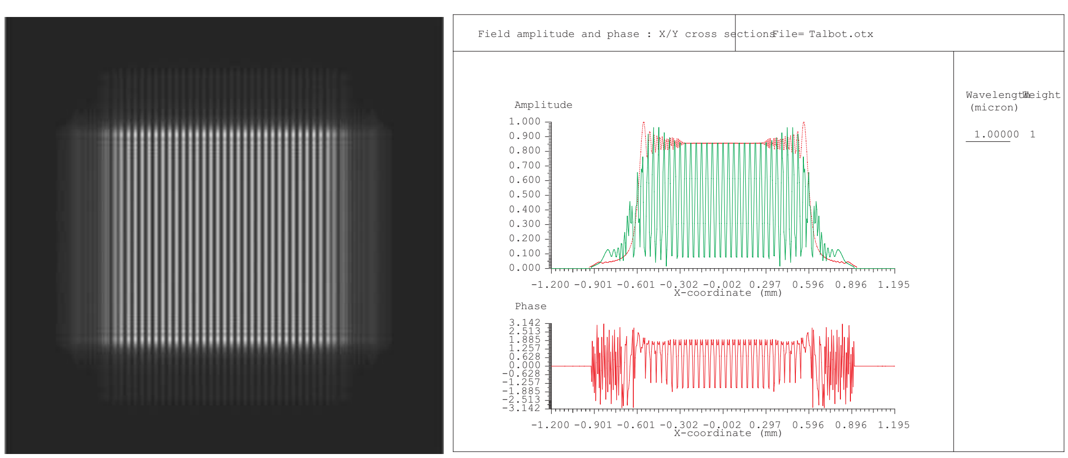


Figure 16.7: Talbot imaging

Note that the side lobes are due to the finite extent of the grating structure.

16.7.3 Coupling Efficiency Example

This example uses a symmetrical optical configuration to couple the output of a single mode fiber into another single mode fiber. The design file is found under `\optalix\examples\pop\coupling-efficiency.otx`. We have seen in section 14.4 (page 284) that fiber coupling efficiency (CEF) algorithms based on geometrical ray tracing predict coupling efficiency reasonably well if diffraction effects inside the optical system can be neglected.

We will now consider a case where diffraction effect play a significant role. The axial separation between the aspheric coupling lenses is 200mm. Due to the small diameter the beam will spread out (diverge) as it propagates in the free space. Due to diffraction, the beam diameter at the receiving lens will be larger than predicted by purely geometric ray tracing and the wavefront will no longer be plano. That gives rise to a different location of the focus position as compared to the geometric spot.

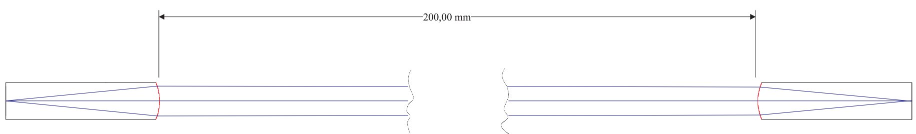


Figure 16.8: Fiber coupling 1:1 relay optics.

The source and receiving fibers are standard Corning SMF-28 types with $5.2\mu m$ mode field radius. Since the fibers are single mode, their emitted respectively exited field is close to a Gaussian and we may run a Gaussian beam analysis (see BEA option in sect. 14.3) in order to obtain a first quick overview about the expected the beam parameters:

```
Gaussian Beam Analysis:

 Wavelength =      1.55000 micron
 M-squared  =      1.00000

 Y/Z-Plane :
```

#	Spot Size SRY	Waist Size WRY	Waist Dist ZWY	Divergence GDY	RFR Radius RCY	Rayleigh R. RRY	Fresnel No.
0	0.005200	0.005200	0.000000	0.094598		Inf	
1	0.005200	0.005200	0.000000	0.065612	-0.14440E+21	0.054806	0.003
2	0.338163	0.294006	99.561456	0.001678	0.40786E+03	175.198763	0.738
3	0.294007	0.294006	-0.438544	0.001678	-0.69992E+05	175.198763	0.558
4	0.338893	0.005189	5.146211	0.065750	0.51474E+01	0.054574	14.399
5	0.005189	0.005189	0.000146	0.094797	0.20344E+02	0.054574	1000000.000
6	0.005189	0.005189	0.000146	0.094797	0.20344E+02	0.054574	

We see that the focus, i.e. the location of the waist, is practically identical to the position of surface 6. The geometric analysis (use spot or fan aberration plots), however, indicates a clear defocus.

This example is also a good exercise for selecting the correct propagators based on the Rayleigh range. For example, propagation from surface 2 to 3 over 100mm distance is completely within the Rayleigh range ($z_R = 175.199mm$), so the PTP operator will be initially proposed by the program. The waist, however, is not exactly at surface 3 but 0.439mm in front of surface 3. Since propagation is always performed from and to the waist, the program proposes propagation in two steps, first PTP over 99.561mm and secondly PTP over 0.439mm. Since surface 3 is so close to the waist, we override the program's choice by disabling the second propagator. Check the 'Fix' check box and select a blank field in the menu. That will also reduce computation time. In a future release, the program will automatically recognize such conditions.

In order to calculate coupled energy, the receiving fiber must be specified. Click on the 'Fiber Parameter' button in the 'Output Field' section of the dialog. A new dialog will be opened. In fact, this is the

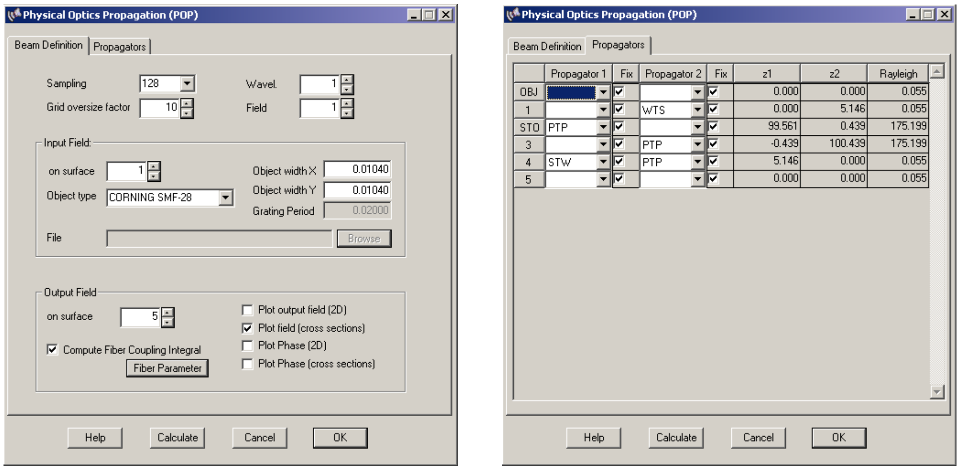


Figure 16.9: Dialogs for physical optics based calculation of coupling efficiency

dialog used in the CEF option (geometrical ray trace based) where only the receiving fiber parameters may be edited. The source fiber (source field) parameters are greyed out because the source field is already specified in the BPR dialog.

The output in the text window is:

```
BEAM PROPAGATION :

Source Parameter:
  Object width :  X =      0.01040   Y =      0.01040
  Object patch :  X =      0.10400   Y =      0.10400
  Sampling     :  128
  Source type  :  CORNING SMF-28

Linear coupling efficiency :    0.9935
Coupling loss              :   -0.0283 dB
```

As already expected from the Gaussian beam analysis (BEA) shown on page 322, coupling is nearly perfect. In contrast to this result, the geometric optics based CEF option calculates a relatively high loss, which corresponds to the defocus of the geometric spot.

```
Linear coupling efficiency :    0.619749
Coupling loss              :     -2.0778 dB
```

16.8 Restrictions

Diffraction beam propagation assumes *coherent* (monochromatic) radiation. Partial coherence or non-monochromatic light cannot be modelled by this option.

In the current implementation, only axial conditions can be modelled. Decentered and/or tilted configurations or skew beams should be avoided. This capability is subject to later releases.

17

Transmission Analysis

Computes the transmittance of a single ray or a bundle of rays through the optical system. The transmission is computed as a fraction of the incident intensity which is normalized to 1 (i.e. 100%). The transmission calculation accounts for vignetting due to clipping apertures or obscurations, ray losses (clipping due to ray trace errors), reflection losses at coated or uncoated surfaces, material bulk absorption, gaussian pupil apodization, surface intensity filters and the polarization state of the source radiation.

Calculation of the transmittance can be controlled in $OpTaliX$ by four options (see also Fig.17.1).

1. Absorption of radiation *within* optical materials is controlled by the `TRA` command. Use `TRA yes` or `TRA no` to activate/deactivate bulk material transmittance in calculations.

2. Reflection losses at optical interfaces (coated or uncoated) are controlled by the `POL` command, which activates/deactivates polarization ray tracing. See `POL yes|no` command to include/exclude effects from coated or uncoated surfaces.

3. Intensity filters (surface apodization) modify the intensity transmission along a ray path. These filters may be loaded from `INT`-files and associated to optical surfaces.

4. The system pupil may be apodized using the commands `PUI`, `PUX`, `PUY`. This feature is mainly used to model non-uniform source radiation such as lasers.

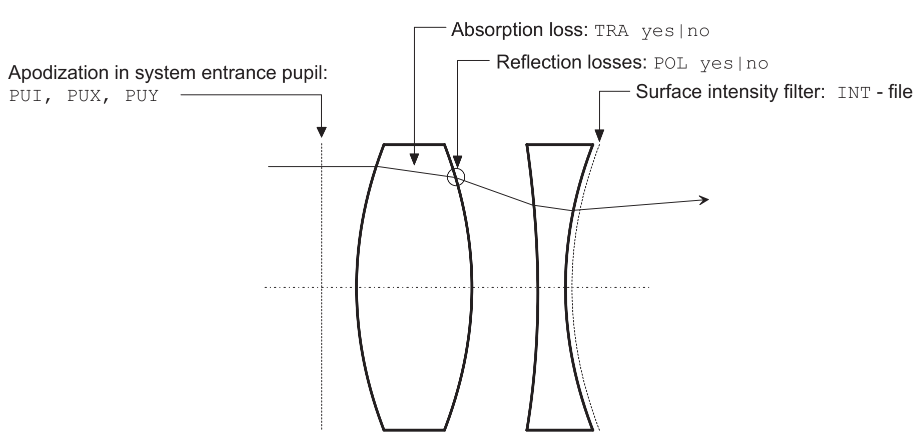


Figure 17.1: Effects on transmission.

Thus, in order to calculate transmission through an optical system including the effects of bulk material absorption and surface reflection losses, the following options must be activated:

```
TRA yes
POL yes
```

Likewise, the combination `TRA yes`, `POL no`, includes the effects of material absorption but ignores all surface reflection losses, whether they are coated or not.

If polarization ray trace is enabled (`POL yes`), output of transmission analysis depends on the polarization state of the source radiation. Use the POLSTATE command to select between polarized or unpolarized input radiation (see also section 18, page 333). By default, the source radiation is assumed unpolarized.

Bulk absorption losses of each material in the optical system are obtained from the glass types. Absorption losses are dependent on the integrated path-length, the material and the wavelength. If bulk absorbtion data is not available for a given glass (e.g. for fictitious glasses), the transmission along the ray path in this material will be assumed 100%.

17.1 Effect of Coatings/Cement on Transmission

By default, each air-glass surface is assumed to be uncoated, i.e. the Fresnel reflections at each air/glass interface are computed, if polarization ray trace is activated (`POL yes`). Mirrors without coating specification are assumed as "perfect" (100%) reflectors.

Attach real multilayer coatings to surfaces (see also `ATT` command on page 371) in order to get most accurate results. Multilayer coatings may be loaded, analyzed and optimized in the coatings menu and then assigned (attached) to any surface. The surface can be converted to an uncoated surface using the DEL MUL command.

A default coating can be applied on each surface for transmission analysis. It is assumed to be single layer MgF_2 with a quarter wave thickness normal to the surface at the reference wavelength. The default coating is defined and attached to a surface by the

```
ATT si..j|k DEF
```

command (see also `ATT` command on page 371), or by entering `DEFCOAT` in the coating column of the surface editor. An example is shown in Fig. 17.2

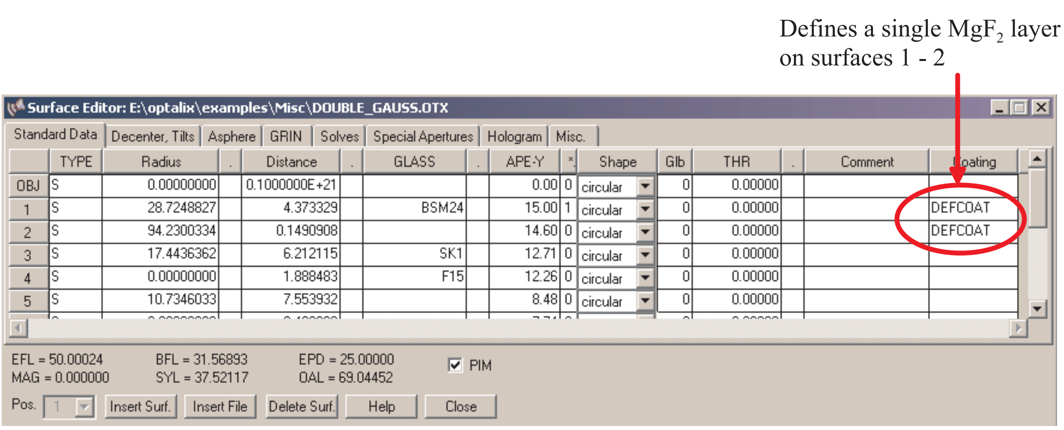


Figure 17.2: Defining a "default" coating (single MgF2 layer) on surfaces.

Cemented surfaces (glass-glass interfaces) are assumed uncoated; the transmission losses are derived from Fresnel reflection losses caused by the index difference of the two adjacent materials. In order to exactly model the effect of cement, split the cemented surface into two surfaces which enclose the cement material.

17.2 Transmission along Chief Ray

By default, transmission is based on the chief ray tracing only. Thus, only one ray (the chief ray) is used to calculate transmission. Using this option, all aperture related effects are ignored. In particular for systems with large numerical apertures, large field angles or large ray incidence angles at surfaces, transmission analysis which integrates over the aperture should be preferred (see section17.3).

Command syntax:

`TRA yes|no`	Includes bulk absorption in transmission analysis. "Yes", includes bulk absorption effects in all subsequent calculations (e.g. PSF, MTF). "No" ignores transmission effects and the aperture is assumed uniformly illuminated (except when apodization of the system has been explicitly specified, see commands PUI,PUX,PUY.
`TRA STEPS n_steps`	Number of wavelength intervals (steps) within the wavelength range as defined in the system configuration. Used in `TRA LAM` plots (see below).
`TRA LAM [FIL filename]`	Plot (chief ray) transmission vs. wavelength (`LAM`). Transmission data may be exported to a file (in ASCII or Excel format) if a file name following the `FIL` qualifier is specified. Note that the extension of the file specification determines the file format (`.txt` or `.dat` for ASCII format, `.xls` for Excel format).
`TRA FLD`	Plot (chief ray) transmission vs. field
`TRA SUR`	Plot chief ray transmission decomposed to surface contributions at all fields and wavelengths. For aperture averaged analysis add the optional parameter `AVG` to this command (section 17.3).
`TRA NUM`	Print chief ray transmission for all fields and wavelengths defined in the optical system. See also transmission integrated over the aperture in section 17.3).
`TRA fk wk sk zk`	Mean Transmission along a single ray. Only available as lens database item (LDI). Example: `eva [tra si f1 w3 z4 0 1]` Calculates transmission at surface si (image) along single ray defined at field 1, wavelength number 3, zoom position 4. The data pair (0 1) defines the relative coordinates in the entrance aperture. The example here describes the marginal Y-ray in the pupil.
`TRAS fk wk sk zk`	Transmission for S-polarized light along a single ray. Only available as lens database item (LDI). Example: `eva [tras si f1 w3 z4 0 1]` Calculates S-pol transmission at surface si (image) along single ray defined at field 1, wavelength number 3, zoom position 4. The data pair (0 1) defines the relative coordinates in the entrance aperture. The example here describes the marginal Y-ray in the pupil.
`TRAP fk wk sk zk`	Transmission for S-polarized light along a single ray. Only available as lens database item (LDI). Example: `eva [trap si f1 w3 z4 0 1]` Calculates P-pol transmission at surface si (image) along single ray defined at field 1, wavelength number 3, zoom position 4. The data pair (0 1) defines the relative coordinates in the entrance aperture. The example here describes the marginal Y-ray in the pupil.
`TRR`	Print transmission of user defined plot rays. See the commands SET RAY and SET FAN in section 184 for definition of various ray bundles.

Example:
We assume a simple achromatic doublet and attach the standard 3-layer coating "`ar_1`" (W-type antireflection coating) from the coating library to surfaces 1 and 2. We leave surfaces 3 and 4 uncoated.

This is accomplished by the commands, assuming the doublet is already in use:

`att s1..2 file ar_1`	! Attach coating "ar_1" to surfaces 1 - 2
`tra sur`	! compute transmission vs. surfaces

The incident intensity is always 1. The output gives the relative intensity along the chief ray. As shown below, transmission values are listed at each wavelength. The ratio of output to input intensity is given for each source of loss, where reflection losses are designated `REF` and absorption losses (occurring in the bulk material) are designated `ABS`.

```
   Wavel.:      0.400     0.450     0.500     0.550     0.600     0.650

---------- Field  1  ------------

REF:   0       1.0000    1.0000    1.0000    1.0000    1.0000    1.0000

REF:   1       0.9747    0.9990    0.9982    0.9968    0.9991    0.9996
ABS:           0.9980    0.9990    0.9990    0.9990    0.9990    0.9990
REF:   2       0.9747    0.9990    0.9982    0.9968    0.9991    0.9996

REF:   3       0.9085    0.9124    0.9149    0.9166    0.9179    0.9188
ABS:           0.9697    0.9960    0.9988    0.9994    0.9994    0.9994
REF:   4       0.9085    0.9124    0.9149    0.9166    0.9179    0.9188

Total          0.7588    0.8268    0.8322    0.8336    0.8397    0.8422
```

This example shows the effects of surface reflection losses and bulk absorption losses. Since no coating is specified at surfaces 3 and 4, Fresnel reflection losses are calculated for these surfaces. Fresnel reflection R on *uncoated* surfaces for normal incidence is given by

$$R = \left(\frac{n-1}{n+1}\right)^2 \tag{17.1}$$

Note also the steep falloff of transmission at shorter wavelengths (400-450nm), which is caused by bulk absorption in the second lens and the lower antireflection efficiency of this coating in the blue spectrum.

17.3 Transmission integrated over Aperture

A bundle of rays is traced through the optical system which fills the entire pupil. The output of this analysis is the mean transmission value of all rays. Note that this calculation is computing intensive and the result may be outputted delayed, depending on computer speed. The transmission calculation accounts for vignetting due to clipping apertures or obscurations, ray losses (clipping due to ray trace errors), losses at coated and uncoated surfaces and material bulk absorption.

Command syntax:

`TRA LAM AVG`	Plot transmission vs. wavelength (`LAM`), integrated over full aperture.
`TRA FLD AVG`	Plot transmission vs. field, integrated over full aperture.
`TRA SUR AVG`	Plot and list transmission integrated over full aperture and decomposed to surface contributions at all fields and wavelengths.
`TRA NUM AVG`	Print transmission integrated over aperture for all fields and wavelengths defined in the optical system.

A sample output for the '`TRA NUM AVG`' command is shown below:

```
TRANSMISSION ANALYSIS (full aperture):

TRA yes
POL no

Wavelength:     0.656     0.588     0.486

---------- Field  1  ------------

Transmittance :       0.9626    0.9787    0.9803
Proj. solid Angle     0.1937    0.1939    0.1938
Effective NA          0.2483    0.2484    0.2483
Relative Illum.       1.0000    1.0000    1.0000

---------- Field  2  ------------

Transmittance :       0.9627    0.9787    0.9805
Proj. solid Angle     0.1418    0.1419    0.1405
Effective NA          0.2124    0.2125    0.2115
Relative Illum.       0.7321    0.7319    0.7252

---------- Field  3  ------------

Transmittance :       0.9637    0.9793    0.9809
Proj. solid Angle     0.0966    0.0962    0.0956
Effective NA          0.1753    0.1750    0.1745
Relative Illum.       0.4991    0.4964    0.4939
```

For each field, wavelength and zoom position, output reports transmittance, projected solid angle, effective numerical aperture and relative irradiance.

Transmittance includes losses at air-glass interfaces (coated or uncoated surfaces) and material absorption losses. Set `POL yes` to enable air-glass losses and `TRA yes` to enable absorption losses.

Proj. solid Angle Defines the solid angle of the bundle of rays as seen from the image point. This is purely a geometric factor and corresponds to the square of the apparent numerical aperture ($sin(u)^2$) at a given field. Vignetting (i.e. truncation of the beam) decreases this value.

Effective NA Related to the projected solid angle and describes the effective numerical aperture at a given field.

Relative Illum. The product of transmittance and projected solid angle. A graphical representation of this value is obtained by the `RIRR` command (relative irradiance, see following section). The relative irradiance is dimensionless and is always referred to the first field.

17.4 Relative Irradiance

`RIRR [NUM]`	Plots relative irradiance at the image surface. Includes field dependent cosine effects and vignetting. Set `POL yes` to include air-glass losses and `TRA yes` to include material absorption losses. The optional parameter `NUM` outputs numerical data.

Plots the relative irradiance (also called relative illumination) in image space by determining the apparent size of the exit pupil in direction cosine space, including all effects like distortion, vignetting, pupil aberration, wavelength weighting and system transmission. The size of the exit pupil is calculated by tracing a bundle of rays through the optical system which fills the entire entrance pupil. NRD (number of rays across diameter) controls accuracy of the result as well as speed of calculation. The higher `NRD`, the more accurate the result will be, however, computation time increases quadratically with `NRD`.

The relative irradiance is the apparent off-axis pupil area divided by the pupil area of the first field defined in the system. Note that the apparent pupil area in $OpTaliX$ is expressed by the solid angle (in $sin(u)$ units) as seen from the image point. This approach is valid for any general optical system and not limited to rotationally symmetric systems. A detailed treatment of calculating relative illumination is found in [42].

Use `POL yes` and `TRA yes` to include transmission losses on air-glass interfaces (including coatings) and losses due to bulk absorption.

Note:
If the system is badly aberrated, the solid angle calculations obtained from ray trace may no longer provide accurate results for relative irradiance. In this case, accurate results are obtained by reversing the system with the image surface modelled as the object surface. Then the product of the transmittance and the projected solid angle in object space gives the relative irradiance with high accuracy, regardless of aberrations.

17.5 Colour Contribution Index

The colour code describes the influence of photographic lenses on the colour rendition of colour films. It is applicable only to the visible wavelength range, i.e. between approximately 370nm and 680nm and is only defined on-axis. Although the colour code is only defined at the optical axis, $OpTaliX$ calculates a colour code for all given fields, indicating possible colour shifts as a function of the field. This feature is particularly interesting in wide angle applications. This calculation also takes into account the effects of multilayer coatings, if attached to surfaces (see also section20 and how to attach coatings to optical surfaces).

The colour contribution index is calculated according to the following scheme (ISO 6728) :

Compute the spectral (wavelength-dependent) transmission $T(\lambda)$ in 10nm intervals in the range 370 - 680nm. The spectral transmission is then multiplied with the spectral sensitivity (weight) $W(\lambda)$ of a standard photographic film, as given in the following equation and in table17.1 :

$$T_{eff} = \sum T(\lambda) \cdot W(\lambda) \tag{17.2}$$

The total photographic responses, R_B, R_G, R_R, are expressed as Log_{10} values, i.e.

$$R_B = log_{10}\left(T_{eff\_blue}\right) \tag{17.3}$$

Likewise, R_G and R_R are determined. Finally, the smallest element of this three number designation is equaled to zero by subtracting it from all three log values.

Command syntax:

`CCI [AVG] [fi..j zi..j]`	Calculates the colour contribution index (CCI) according ISO 6728 (1983) for each field and zoom position, based on chief rays. The optional parameter `AVG` integrates over the aperture. Since many rays may be involved (depending on `NRD`) in evaluating an average transmission, the computing time may increase considerably. If necessary, reduce `NRD` to reduce computing time.

Weighting Factors for Standard Cameras					
$\lambda(nm)$	$W_{blue}(\lambda)$	$\lambda(nm)$	$W_{green}(\lambda)$	$\lambda(nm)$	$W_{red}(\lambda)$
370.00	1.00	470.00	1.00	550.00	1.00
380.00	1.00	480.00	1.00	560.00	1.00
390.00	3.00	490.00	1.00	570.00	1.00
400.00	7.00	500.00	2.00	580.00	2.00
410.00	10.00	510.00	4.00	590.00	3.00
420.00	12.00	520.00	5.00	600.00	4.00
430.00	12.00	530.00	8.00	610.00	6.00
440.00	13.00	540.00	15.00	620.00	8.00
450.00	13.00	550.00	25.00	630.00	12.00
460.00	12.00	560.00	13.00	640.00	19.00
470.00	8.00	570.00	13.00	650.00	22.00
480.00	4.00	580.00	9.00	660.00	16.00
490.00	2.00	590.00	2.00	670.00	4.00
500.00	1.00	600.00	1.00	680.00	1.00
510.00	1.00				

Table 17.1: Weighting factors for colour contribution index calculation of standard cameras

18

Polarization Analysis

Polarization analysis in *OpTaliX* uses an extension to the classical ray trace, such that vector properties are associated to rays. Interaction at surfaces in the optical system alter these vector properties, like the polarization state.

`POL yes|no` `POL y|n`	Activates/deactivates polarization ray trace `yes` : enables polarization ray trace for all subsequent analysis `no` : disables polarization ray trace
`POL LAM`	Polarization analysis vs. wavelength.
`POL APE`	Calculates the degree of polarization for all rays across the pupil.
`POL ELL`	Plots polarization ellipses for all rays across the pupil.
`POR`	Polarization raytrace with user-defined rays (e.g. those rays which have been previously defined by the `SET RAY` or `SET FAN` commands.)
`PA1 x1 y1 phase1`	Polarization amplitude and phase components of electric vector 1. The phase is given in radians.
`PA2 x2 y2 phase2`	Polarization amplitude and phase components of electric vector 2. The phase is given in radians. This vector is required to define unpolarized or partially polarized light. For strictly monochromatic (coherent) radiation, PA2 will not be used in polarization calculations.
`POLSTATE 0|1`	Polarization state of input radiation: 0 = unpolarized, uses both vectors `PA1` and `PA2`, 1 = completely polarized, uses vector `PA1` only.
`POLRAY [fi..j wi..j si..j zi..j]`	Polarization ray trace. See detailed description in sect. 18.1

18.1 Tracing a Polarization Ray

Polarization ray tracing is similar to tracing a single ray as given by theRSI and SIN commands (see page 232 for reference).

The commands `POLRAY` respectively `PRSI` trace a single ray through the optical system and output the polarization state (X/Y-amplitudes, phase, degree of polarization) associated with this ray. The input polarization is defined by the `PA1` and `PA2` commands (see also sect. 18.2).

The command syntax is:

```
prsi [ si..j | gk | wi..j | zi..j | fi..j ] ape_relX ape_relY
```

where `ape_relX` and `ape_relY` are the relative coordinates of the ray in the entrance pupil. Example:

`pa1 0 1 0`	! Linear input polarization, oriented along Y-axis
`polstate 1`	! Assume coherent (completely polarized) radiation
`prsi f1 si 0 0`	! Trace polarization ray at field f1, image surface si, for chief ray (relative pupil apertures 0/0)

A typical output in the text window is:

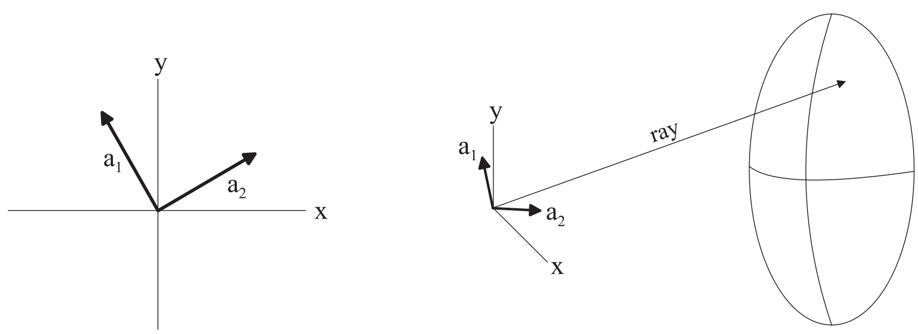
```
SINGLE RAY POLARIZATION COMPONENTS :
----------------------------------

Field  =  1      OBX =       0.00000   OBY =       0.00000
Colour =  1      WL  =       546.000 nm

  #  Pol.Degree         X1        Y1     Phase1
  1     1.00000   0.000000  1.000000       0.0
  2     1.00000   0.684547  0.728969      86.4
  3     1.00000   0.684547  0.728969      86.4
```

18.2 Defining Input Polarization

In order to perform polarization calculations, the polarization properties of the input beam must be fully specified. Any polarization state of input radiation may be expressed by two independent linearly polarized waves with their electric vectors vibrating in two mutually perpendicular directions at right angles to the direction of propagation. Fig. 18.1 shows the polarization vectors associated to a ray.



a) b)

Figure 18.1: Definition of polarization vectors, a) mutually perpendicular electric vectors, b) polarization vectors attached to a ray.

It is preferable to align the electric vectors a_1, a_2 along the (x,y) coordinate axes of an arbitrarily chosen coordinate system, typically the one which is used to describe the optical system. The polarization vectors are then $a_1 = (0, 1)$ and $a_2 = (1, 0)$. For coherent, i.e. strictly monochromatic radiation (`POLSTATE 1`), the polarization state is always 100% and one vector (a_1) is sufficient. a_2 will be ignored for this case.

The state of polarization is best represented by the coherency matrix **J** of the light wave as found for example in Born and Wolf [4]. The coherency matrix is defined as

$$\mathbf{J} = \begin{bmatrix} < a_1^2 > & < a_1 a_2 e^{i(\Phi_1 - \Phi_2)} > \\ < a_1 a_2 e^{-i(\Phi_1 - \Phi_2)} > & < a_2^2 > \end{bmatrix} = \begin{bmatrix} J_{xx} & J_{xy} \\ J_{yx} & J_{yy} \end{bmatrix} \tag{18.1}$$

where Φ is the phase difference between the components of each vector. The diagonal elements of **J** are real and are seen to represent the intensities of the components in the x- and y-directions. The non-diagonal elements are in general complex, but they are conjugates of each other. The form of the coherence matrix **J** can be expressed in a simple manner for some cases of particular interest:

18.2.1 Completely unpolarized (natural) light:

Light which is most frequently encountered in nature has the property that the intensity of its components in any direction perpendicular to the direction of propagation is the same. The coherence matrix of natural light of intensity I_0 is

$$\frac{1}{2} I_0 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{18.2}$$

18.2.2 Completely polarized light:

If we suppose that the light is strictly monochromatic, the amplitudes a_1 and a_2 and the phase factors Φ_1 and Φ_2 do not depend on the time. In particular, the matrices

$$I \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \qquad I \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

each represent linearly polarized light of intensity I, with the electric vector in the x-direction (a_2=0) and the y-direction (a_1=0) respectively. For circularly polarized light the coherency matrix is

$$\frac{1}{2} I \begin{bmatrix} 1 & \pm i \\ \mp i & 1 \end{bmatrix}$$

where I is the intensity of the light. The upper and lower sign is taken according whether the polarization is right- or left-handed.

18.2.3 Some equivalent representations:

We note some useful representations of *natural light*. The coherency matrix of natural light may always be expressed in the form

$$\frac{1}{2} I \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \frac{1}{2} I \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{2} I \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{18.3}$$

and this implies that a wave of natural light, of intensity I, is equivalent to two independent linearly polarized waves, each of intensity $\frac{1}{2}I$, with their electric vectors vibrating in two mutually perpendicular directions at right angles to the direction of propagation.

Another useful representation of natural light is

$$\frac{1}{2} I \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \frac{1}{4} I \begin{bmatrix} 1 & +i \\ -i & 1 \end{bmatrix} + \frac{1}{4} I \begin{bmatrix} 1 & -i \\ +i & 1 \end{bmatrix} \tag{18.4}$$

and implies that a wave of natural light of intensity I is equivalent to two independent circularly polarized waves, one right-handed, the other left-handed, each of intensity $\frac{1}{2}I$.

Thus, for the determination of the polarization behaviour of an optical system, **two** linearly polarized waves (represented by rays) according eq. 18.3 are traced independently through the optical system. The vibrating planes of this incident waves (represented by rays) can be defined by proper setting of the amplitudes a_1, a_2 and the phase difference δ between the components a_1, a_2 of each wave.

18.3 The Degree of Polarization:

The ratio of the intensity of the polarized portion of the total light intensity is called the *degree of polarization* $\mathbf{P}$ of the wave. Calculation of $\mathbf{P}$ requires two mutually perpendicular electric vectors as shown in Fig. 18.1. Two forms of expressing (calculating) $\mathbf{P}$ are shown below.

18.3.1 Polarzation expressed by Coherence Matrix

On the basis of the *coherence matrix* the degree of polarization is given by

$$\mathbf{P} = \frac{I_{pol}}{I_{tot}} = \sqrt{1 - \frac{4|\mathbf{J}|}{(J_{xx} + J_{yy})^2}} \tag{18.5}$$

where $|\mathbf{J}|$ is the determinant of the coherence matrix as given in eq. 18.1 :

$$|\mathbf{J}| = J_{xx}J_{yy} - J_{xy}J_{yx} \geq 0 \tag{18.6}$$

18.3.2 Polarization expressed by Stokes Vectors

The degree of polarization may also be expressed using *Stokes vectors*

$$P = \frac{\sqrt{s_1^2 + s_2^2 + s_3^2}}{s_0} \tag{18.7}$$

where the Stokes vector is defined by:

$$\begin{aligned} s_0 &= \langle a_1^2 \rangle + \langle a_2^2 \rangle \\ s_1 &= \langle a_1^2 \rangle - \langle a_2^2 \rangle \\ s_2 &= 2 \langle a_1 a_2 cos\delta \rangle \\ s_3 &= 2 \langle a_1 a_2 sin\delta \rangle \end{aligned} \tag{18.8}$$

18.4 Total Internal Reflection

The *Fresnel formulae* do not apply for total internal reflection. This is the case when light is propagated from an optically denser medium into one which is optically less dense and when the law of refraction

$$\sin\theta_t = \frac{\sin\theta_i}{n_{12}} n_{12} = \frac{n_1}{n_2}$$

does not give a real value for the angle of refraction θ_t. The intensity of light which is totally reflected for each component (TE- or TM-wave) is equal to the intensity of the incident light. But the two components are seen to undergo phase jumps of different amounts.

The changes of the phases δ_s , δ_t of the components of the reflected and the incident wave can be expressed as [4]

$$\tan\frac{\delta_s}{2} = -\frac{\sqrt{\sin^2\theta_i - n^2}}{n^2\cos\theta_i} \tag{18.9}$$

$$\tan\frac{\delta_t}{2} = -\frac{\sqrt{\sin^2\theta_i - n^2}}{\cos\theta_i} \tag{18.10}$$

where $n = n_2/n_1$. Linearly polarized light will in consequence become elliptically polarized on total reflection. The relative phase difference is $\delta = \delta_s - \delta_t$.

19

Optimization

Optimization of an optical system requires the solution of a highly nonlinear problem. It is the process by which the aberrations of a lens are minimized by changing selected lens data (*variables*). A *merit-function* is defined by commands relating to different classes of aberrations (e.g. spot diameter, distortion, etc) and constraints to be fulfilled exactly (e.g. focal length, overall length, etc). In order to optimize a system, both merit-function and variables must be defined. All entries in the merit-function must be computable functions of the variables.

Two types of optimization algorithms are available

KT -optimization, minimizes an error function by a damped-least-square (DLS) method subject to solving constraints using Lagrange multipliers and application of the Kuhn-Tucker optimality condition,

LM -optimization, minimizes an error function using a modified Levenberg-Marquardt algorithm,

A brief overview of the algorithms is given in sections 19.1 and 19.2. For a detailed understanding, the reader is referred to the references cited in the corresponding sections.

In order to set up an optimization, variables, targets and constraints must be defined. This is performed in several steps:

`VAR`: Define variables for non-zoomed and zoomed system. See sect. 19.3 for details.
`TAR`: Define target functions and constraints, as described in sect. 19.5.
`OPT`: Run the optimization (sect. 19.9).

19.1 KT-Optimization

The KT-optimization minimizes an error function by a damped-least-square (DLS) method subject to exactly solving constraints using Lagrange multipliers. The *Kuhn-Tucker*[1] optimality criteria are applied at each iteration to secure that the true local minimum is found within the domain of constraints given. The Kuhn-Tucker conditions are an extension to the classical DLS method. For further reading see Spencer [51] and Feder [11]. Closely following Spencer's treatment, the problem is stated as minimizing

[1] also known as Karush-Kuhn-Tucker condition

$$\sum_{m=1}^{M} w_m^2 \left(\sum_{j=1}^{J} a_{mj} q_j - d_m \right)^2 \tag{19.1}$$

while at the same time solving the set of linear equations

$$\sum_{j=1}^{J} b_{nj} q_j = e_n, \quad \text{for} \quad i = 1, \dots, N \tag{19.2}$$

with

$a_{mj} = \partial g_m / \partial p_j$	derivative on functions to be minimized,
$b_{mj} = \partial h_n / \partial p_j$	derivative on functions to be exactly solved,
q_j	= parameter increment,
d_m	= function aberration (minimize),
e_m	= constraint aberration (solve exactly),
w_m	= weight factors,

A solution to this problem, written in matrix form, is given by

$$\left(\mathcal{M}^T \mathcal{M} + \mathcal{C}\mathcal{I} \right) q - \mathcal{B}^T \lambda = \mathcal{M}^T r \tag{19.3}$$

with

$\mathcal{M} = \mathcal{W}\mathcal{A}$	= weighted derivative matrix (minimize)
$\mathcal{B}$	= derivative matrix (solve exactly)
$\mathcal{I}$	= identity matrix
$\mathcal{C}$	= dumping factor
$r = \mathcal{W}d$	= weighted aberration
λ	= Lagrange multipliers

At each iteration, that is after solving the set of DLS equations as given in eq. 19.3, the 1^{st} order (necessary) Kuhn-Tucker conditions, which satisfy the optimum solution of a non-linear problem subject to constraints, are then checked:

$$\begin{array}{lll} I & \dfrac{\partial L}{\partial p_j} = \dfrac{\partial g}{\partial p_j} - \lambda \dfrac{\partial h}{\partial p_j} = 0 & \text{stationary point} \\ II & h(p) \leq 0 & \text{feasibility} \\ III & \lambda h(p) = 0 & \text{complementary slackness} \\ IV & \lambda \geq 0 & \end{array} \tag{19.4}$$

19.2 LM-Optimization

The problem is solved subject to bounds on the variables using a modified Levenberg-Marquardt algorithm and a finite difference Jacobian [10, 30, 35]. The problem is stated as follows:

$$min_{x \in I\!R^n} \frac{1}{2} F(x)^T F(x) = \frac{1}{2} \sum_{i=1}^{m} f_i(x)^2 \tag{19.5}$$

where $m \geq n$ and $f_i(x)$ is the i-th component function of F(x). From a current point, the algorithm uses the trust region approach and a new point x_n is computed as

$$x_n = x_c - \left[J(x_c)^T J(x_c) + \mu_c I\right]^{-1} J(x_c)^T F(x_c) \tag{19.6}$$

$F(x_c)$ and $J(x_c)$ are the function values and the Jacobian evaluated at the current point x_c, respectively. This procedure is repeated until the stopping criteria are satisfied.

19.3 Editing Variables

In the command line, optimization variables may be added or deleted by the commands:

`VAR`	The VAR command without parameters invokes a dialog box for editing optimization variables (zoomed and non-zoomed) and targets/constraints. The dialog box contains the most commonly used types of optimization variables, however, variables not found in this dialog box must be set or deleted from the command line (see commands below).

`VAR si..j|sk vstr1 vstr2 ...`
 	
`VARZ si..j|sk vstr1 vstr2 ...`
	Add one or multiple variable(s) on surface(s) `si..j|sk` described by `vstr1`, `vstr2`, etc. The `VAR` command is used for single position (non-zoomed) variables, the `VARZ` form is used for zoomed variables. Multiple variables on a surface may be combined in a single line.
	Examples:
	`var s4 cuy` ! curvature (`CUY`) on surface 4 is variable
	`var s3..4 cuy thi` ! curvature and thickness on surfaces 3-4 are variable.

`DEL VAR si..j|sk vstr1 vstr2 ...`	Delete variable described in `vstr1`, `vstr2`, etc on surface(s) `si..j`. Example: `del var s3 thi` ! deletes thickness variable on surface 3.

From the main menu, *Optimization / Variables,Constraints*, edit variables/constraints in a spreadsheet-like dialog box. Optionally use the command `EDI VAR` or click on the VAR tool button in the main window to open the variables/targets dialog.

19.4 Definition of Variables (VAR)

Variables are defined and edited by the command "`VAR`". This command applies for both zoomed and non-zoomed variables. A dialog box will be opened.

In case of a multi-configuration (zoom) system, **n** variables will be created internally for each zoomed variable, if n is the number of positions.

Basically, any lens parameter, which can be changed on the command line, may be used as a variable in the optimization. A concise (but not complete) list of variables is given in the following table.

CUY	curvature
CUX	curvature X (toric deformation)

continued on next page

continued from previous page	
THI	thickness
THR	reference thickness
DEF	defocus
K	conic constant
A	aspheric parameter, h^4 for even asphere, h^2 for odd asphere
B	aspheric parameter, h^6 for even asphere, h^3 for odd asphere
C	aspheric parameter, h^8 for even asphere, h^4 for odd asphere
D	aspheric parameter, h^{10} for even asphere, h^5 for odd asphere
E	aspheric parameter, h^{12} for even asphere, h^6 for odd asphere
F	aspheric parameter, h^{14} for even asphere, h^7 for odd asphere
G	aspheric parameter, h^{16} for even asphere, h^8 for odd asphere
H	aspheric parameter, h^{18} for even asphere, h^9 for odd asphere
ADE	tilt around X-axis
BDE	tilt around Y-axis
CDE	tilt around Z-axis
XDE	X-decenter
YDE	Y-decenter
ZDE	Z-decenter
GZO	gradient Z-offset
DVO	Dispersion offset
DNO	Index offset
GLA	Combined variable, simultaneously makes DNO and DVO variable
H2	Hologram coefficient 2 (h-term for symmetric HOE, linear x-term for asymmetric HOE)
H3	Hologram coefficient 3 (h^2-term for symmetric HOE, linear y-term for asymmetric HOE)
H4	Hologram coefficient 4 (h^3-term for symmetric HOE, x^2-term for asymmetric HOE)
H5	Hologram coefficient 5 (h^4-term for symmetric HOE, $x \cdot y$-term for asymmetric HOE)
H6	Hologram coefficient 6 (h^5-term for symmetric HOE, y^2-term for asymmetric HOE)
H7	Hologram coefficient 7 (h^6-term for symmetric HOE, x^3-term for asymmetric HOE)
H8	Hologram coefficient 8 (h^7-term for symmetric HOE, $x^2 \cdot y$-term for asymmetric HOE)
H9	Hologram coefficient 9 (h^8-term for symmetric HOE, $x \cdot y^2$-term for asymmetric HOE)
H10 to H28	Hologram coefficients 10 to 28
HX1	x-coordinate of object point source for 2-point HOE
HY1	y-coordinate of object point source for 2-point HOE
HZ1	z-coordinate of object point source for 2-point HOE
HX2	x-coordinate of reference point source for 2-point HOE
HY2	y-coordinate of reference point source for 2-point HOE
HZ2	z-coordinate of reference point source for 2-point HOE
	continued on next page

continued from previous page	
Uxx	Coefficients of user-defined surfaces, SPS-ODD surfaces and SPS-XYP surfaces. 'xx' denotes the corresponding coefficient number. Example: `VAR s4 U7`
Zxx	Coefficients of Zernike surfaces. 'xx' denotes the corresponding coefficient number. Example: `VAR s4 Z7`

19.5 Target (Error) Function (TAR)

Optimization requires a set of targets and constraints which are minimized or solved. Targets are, for example, a minimum spot diameter (SPD) or minimum lateral chromatic aberration (LAC). A constraint is a parameter, which is held exactly or shall be greater or smaller than a specified value. For example, holding the focal length (EFL) to a precise value is a constraint.

The entity of the targets and constraints builds up the "*merit-function*". There is no built-in default merit function. The definition of targets is invoked on the command line by the command `TAR`. It opens the same dialog box as for the `VAR` command, since this dialog offers both settings for variables and for targets/constraints. To define targets and constraints (the merit function), almost any $OpTaliX$ command may be used. Entries to the merit function may be quite complex as arithmetic expressions (such as `2*sqrt(2)/3`), variables (such as `$x`) and lens database items (thickness, radius of curvature, etc.) may also be used for defining targets. The commands can be linked with operands and target values. Allowable operands are:

=	Constrains exactly to target value.
>	The target value of the constraint is defined as a minimum value, or lower boundary.
<	The target value of the constraint is defined as a maximum value, or upper boundary.

Target values to be minimized do not require an operand. A short example illustrates typical merit function definitions:

`EFL = 100.`	The focal length (`EFL`) shall be exactly 100 mm.
`SPD 0`	Minimizes spot diameter with target value 0. Since no field, wavelength or zoom parameters are specified, the spots are minimized for *all* wavelengths, fields and zoom positions.
`SPD f2..3 w4 0`	As above, minimizes spot diameter with target value 0. However, spots are minimized only for fields 2 to 3 and wavelength number 4.
`! This is a comment line`	Comments are indicated by the exclamation mark "!". The rest of the line is then ignored. In blank lines, the exclamation mark must be the first character of the line. This way, it is also possible to enable or disable selected target functions.
`WAV f1 0 ! wavefront`	Minimizes rms-wavefront at field 1. The comment right to the exclamation mark is ignored.
`SPD F3 Z2 0 ; wt = 0.7`	Minimizes spot diameter for field no.3 and zoom position 2. The target value is 0, the relative weight is 0.7.
`SPD F4 0`	Minimizes spot diameter for field no.4 and all wavelengths. Because no weight is specified, the default weight 1.0 is assumed.

From the list of target definitions, the merit function is then constituted by the weighted sum of "aberrations", i.e. the difference of actual value of the correction status and its specified target value. The actual value of the merit function can be printed by the `ERRF` command (see page358). Generally, a more detailed merit function will be required to fulfill specific needs.

19.5.1 Weights on Error Functions

All error function components (targets), except ">" or "<" constraints, can be assigned *weights* to express a relative importance among the various functions. Weights are arbitrary real numbers of positive value. Arithmetic expressions are not allowed in defining weights. If not specified, the default weight is 1. They can be explicitly overwritten by adding a "WT" qualifier to the specific error function component. For example,

```
spd 0 ; wt = 2
```

assigns the (relative) weight 2.0 to the spot diameter (SPD) function. This means that the relative importance of the spot diameter is two times higher than other functions (aberrations). Weight specifications **must** be separated from the error/target function specification by a semicolon ";".

The following examples explain the concept of "weights" and also show other advanced features:

`EFL = 100`	Constrains the focal length to exactly 100mm
`MFL s4 = 25`	Keep module focal length (defined at surface 4) to 25mm.
`bfl > 160.`	The (paraxial) back focal length shall be greater or equal to 160mm
	continued on next page

continued from previous page	
`et s3..4 12.0 > 5.`	The edge thickness between surfaces 3 and 4 at height 12mm shall be greater/equal 5mm. Note, that edge thickness (`ET`) is also available as a solve parameter. Although this constraint will work in optimization (provided there is no `ET`-solve at the corresponding surface), it is advisable to use the solve on `ET` in order to reduce computing load.
`spd f1 0 ; wt = 2`	Minimizes spot diameter at field 1. The weight is 2
`spd f2 0 ; wt = 1`	Minimizes spot diameter at field 2. The weight is 1
`spd f3 w1..3 0 ; wt = 0.5`	Minimizes spot diameter at field 3 for wavelengths 1 to 3
`disy f3 0.1`	Distortion in Y-direction is minimized to 0.1%. Since there is no weight given, the default weight is 1
`y f1 w1 s5 0 1 = 0`	Constrains the Y-coordinate of a marginal ray (relative pupil coordinates are $x_p = 0, y_p = 1$) at field number 1 and wavelength number 1 at surface 5 to zero. Note that all parameters are obligatory in order to specify one single ray only. For example, omission of the field qualifier (f1) would return Y-coordinates for all fields, which can hardly be solved.

19.5.2 Weighted Constraints

Weights can also be assigned to constraints which are solved exactly (=). The function is then included in the error function (minimized) instead of being exactly solved. This option should be used sparingly.

`WTC weight_on_constraint`	Include constraint in the error function (i.e. minimize) instead of solving it exactly. **Use only with equality constraints (=)**.

The smallest value that achieves control should be chosen. A low value will allow wider deviations from the target. A higher value will achieve a closer approach to the target but more strongly dominates the solution.

Using `WTC` is not the best way to optimize. It should only be used when targets are far from the present configuration or the exact solution demands a significant change in the optical design. In such cases it is recommended to switch temporarily to LM-optimization. After a sufficiently close point to the targets has been reached, constraints can be exactly solved using the KT-optimization. See also the notes on selecting the best optimization algorithm on page 359.

Examples on using weighted constraints (WTC):

```
efl = 100 ; wtc = 2
efl 100 ; wt = 2
```

Both forms yield identical results. Note the second form (EFL 100) without the 'equal' qualifier (=). Since it is omitted, the function will be minimized (with relative weight 2) instead of being exactly solved.

19.5.3 Include Targets from File

Targets may also be included from external files via the `#include` option. For example,

```
#include mytargets.txt
```

includes target definitions contained in `mytargets.txt` as if they were written directly in the targets/constraint editor. A file name without path is searched in the directory where the current system resides. Explicitly specify the path if the file to be included shall be searched in a different directory. Any extension is allowed to the file name. `#include` statements may appear at any place in the targets list, thus, mixed forms of target/constraints expressions and include file declaration are permitted. For example,

```
efl = 100
#include mytargets.txt
spd f1..3 0
```

There is no limit on the number of `#include` statements, however, nesting of `#include` is NOT permitted. That is, a file containing target/constraint definitions may not contain `#include` statements itself.

19.5.4 Targets using Lens Database Items

Targets may also be composed from *lens database items* (see sect. 27), which gives even greater flexibility. A few examples shall illustrate use of lens database items in defining targets/constraints:

`thi si-1 = [thi s5]`	Requires thicknesses `si-1` (the distance before the image surface) and thickness 5 to be equal. If `thi s5` is a variable, `thi si-1` will be dynamically adjusted as the optimization process evolves.
`thi s7 = [thi s5..6]`	The thickness on surface 7 shall be equal the sum of thicknesses of surfaces 5 to 6.
`cy s5 f1 w1 0 1 > -1/(2*[fno])`	Mix arithmetic instructions with lens database items to build complex targets.

It is advisable to check correctness of target constructions in the command line. For example, the target of the last example in the table above would be queried in the command line (using theEVAluate command, see sect. 26.9, page 446) as

```
eva -1/(2*[fno])
```

When no errors are issued in the text window, the target can be added to the optimization constraints. This example also illustrates that there is no functional difference in command syntax and constraints definition.

In this context it is important to note that square brackets `[ ]`, which indicate a *lens database item*, are only allowed on the right side of a constraint (i.e. the target to be evaluated). Basically, a lens database item is a function which returns a value. Thus, a constraint assignment such as `[thi s5] > 3*[thi s2]` would assign a number to the left part (`thi s5`), which would be a contradiction and therefore is not valid. The correct constraint syntax for this example would be: `thi s5 >`

```
3*[thi s2]
```

Notes:

Targets which invoke paraxial parameter should be used with care, for example `EFL`, `BFL`, `SAP`, ... and all third order aberrations. This applies particularly for zoom systems, where the target values will be computed for all zoom positions, if no other qualifier is present. For example, specifying a target "`EFL = 50`" in a zoom system with two positions used at two focal lengths (say 50 and 100mm), and omitting any other qualifier would attempt the optimization to solve focal length for *all* positions.[2] In such cases it is mandatory to specify the focal length for each zoom position separately. Thus, two distinct constraints must be specified: "`EFL z1 = 50`" and "`EFL z2 = 100`". The same logic applies for groups (surface ranges), e.g. `EFL s1..4 z3 = 50`.

19.5.5 User-defined Constraints

User-defined variables and user-defined functions may also be specified as part of the constraints list. See sections 26.11, 26.16 for the corresponding syntax. Note that user-defined variables must not be confused with optimization variables (such as curvatures, separations, etc.). User-defined variables are only used for storing calculation results and using them in other arithmetic expressions or constraints.

User-defined variables and functions allows the definition of complex constraints which are not found in the list of the built-in constraints. Variables and functions are dynamically updated as the optimization proceeds. For example,

`$x = 5`	! Variable assignment
`@xxx == [efl]+[bfl]-$x`	! Defines a complex function.
`@xxx = 100`	! Defines a constraint on the function. Note the single "=" sign.

On the examples given above, it is worth to emphasize the difference in using the "==" and "=" operators in optimization constraints. A function definition must use the "==" operator, however, it does not create an optimization constraint. A function statement using the "=" operator constitutes a constraint, i.e. the numeric result of a previously defined function is used as a parameter in the constraint definition.

Constraints on functions accept $(<, =, >)$ operators.

19.5.6 Default Constraints

If enabled, default constraints will automatically be added to the list of target (error) functions. Default constraints are useful for maintaining reasonable dimensions of lenses and air spaces during optimization. For example, default constraints ensure that edge thicknesses are always manufacturable (i.e. greater than a certain fraction of the lens diameter) and that lenses do not intersect (i.e. air edge separation is always positive).

Default constraints avoid the necessity to explicitly specify axial thickness constraints and edge thickness constraints in targets (merit) functions. Default constraints can be enabled or disabled via the DEFC command or in a dialog box, accessible from the main menu *Optimization –> Parameters* and then selecting the 'Default Constraints' tab (see Fig. 19.1, page 348).

Initially, default constraints are disabled. If required, default constraints must be enabled by checking the 'Enable default constraints' check box or by entering `DEFC Yes` in the command line prior to

[2] Absence of a zoom qualifier "`z`" implies **all** zoom positions).

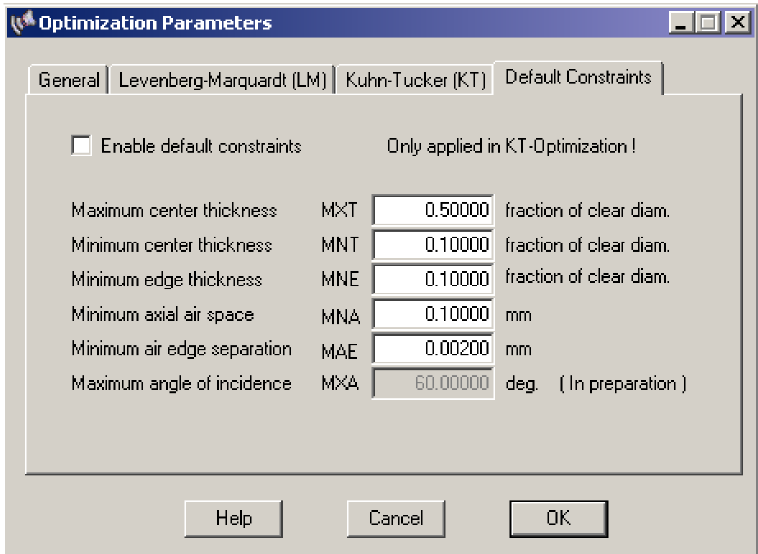


Figure 19.1: Dialog box for editing default constraints.

executing optimization. Note that default constraints currently only apply to the KT-optimization, they are ignored in the LM-optimization.

Default constraints differ from specific user constraints. Whereas a specific constraint must be explicitly defined and only applies to specific surfaces and/or zoom positions, the default constraints apply to all surfaces and all zoom positions. Default constraints cannot be given different values for different surfaces or different zoom positions. All default constraints are always imposed as bounds and never as equality constraints. default constraints are always controlled with the method of Lagrangian multipliers.

Note that default constraints are only applied to **variable** thicknesses/separations. Non-variable thicknesses are not included to the default constraints list. If a thickness/separation constraint is explicitly defined in the targets (error) function list, that constraint overrides the corresponding default constraint on that surface(s).

Default constraints settings are stored with the prescription data and optimization data for the current optical system in use. This allows individual settings of default constraints for each specific design.

`DEFC Yes|No`	Enable (Yes) or disable (No) default constraints handling.
`MXT max_ele_center_thi`	Constrain maximum center thickness of all variable thickness elements, unless overridden by `THI` or `ET` constraints on specific surfaces. MXT is given as a fraction of the maximum clear aperture. The default MXT value is 0.5 * maximum clear aperture.
`MNT min_ele_center_thi`	Constrain minimum center thickness of all variable thickness elements, unless overridden by `THI` or `ET` constraints on specific surfaces. The default `MNT` value is 1/10 minimum clear diameter.
`MNE min_ele_edge_thi`	Constrain minimum edge thickness of all variable thickness elements, unless overridden by `THI` or `ET` constraints on specific surfaces. The default `MNE` value is 1/10 minimum clear diameter.

continued on next page

continued from previous page	
MNA min_air_center_thi	Constrain minimum center thickness of all variable air spaces with 'negative' shape (thicker at edge than center), unless overridden by THI or ET constraints on specific surfaces. The default MNA value is 0.1mm.
MAE min_air_edge_thi	Constrain minimum edge thickness of all variable air spaces with 'positive' shape (thinner at edge than center), unless overridden by THI or ET constraints on specific surfaces. The default MAE value is 0.002mm.
MXA max_angle_inc	Constrain maximum angle of incidence (in degrees) for all active fields. The default MXA value is 60deg. **In preparation!**

The default constraints relating to element thickness and spacing are shown in Fig. 19.2. Note that default constraints are only active if the appropriate thicknesses are variable. If a thickness or spacing is frozen (not variable), default constraints on this surface are totally disabled, however, general thickness constraint violations can occur.

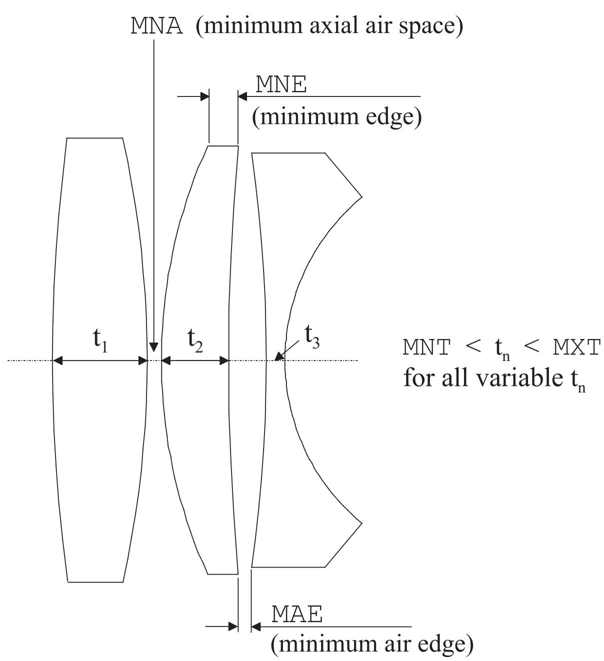


Figure 19.2: Default constraints on element thickness and spacings.

19.6 Targets/Constraints Overview

<table>
<tr><td>EFL [si..j | wi..j | zi..j]</td><td>Equivalent focal length</td></tr>
<tr><td>BFL [wi | zi]</td><td>Back focal length at used conjugate, wavelength number wi, zoom position zi</td></tr>
<tr><td>SYL [zi]</td><td>System length (from first surface to last surface, excluding image surface)</td></tr>
<tr><td>MAG [zi]</td><td>magnification</td></tr>
<tr><td>SAP [zi]</td><td>Location of exit pupil from last surface</td></tr>
<tr><td>THI si..j</td><td>Axial thickness (separation) at surfaces i to j. Example: thi s3..5 < 5.0</td></tr>
<tr><td>IMD [zk]</td><td>Image distance (THI si-1) at zoom position zk. If zk is omitted, IMD is calculated at the first zoom position.</td></tr>
<tr><td>IMC [zk]</td><td>Image clearance, the smaller distance (edge or axis) between surface i-1 and the image surface i. Only calculated at zoom position zk. If zk is omitted, the first zoom position is used.</td></tr>
<tr><td>RDY si..j</td><td>Radius of curvature at surfaces i to j. Example: rdy s5 > 100</td></tr>
<tr><td>OAL [si..j]</td><td>Overall length, which is the sum of the axial thicknesses/separations of surfaces i to j. In absence of a surface range specifier, OAL counts from the first surface to the image surface (not to be confused with SYL, which counts from the first surface to the last surface, excluding image surface). Example: oal s2..6 = 50</td></tr>
<tr><td>AOI sk fi zi wi rel_apeX rel_apeY</td><td>Angle of incidence of a ray at surface si, field fi, zoom position zi, wavelength wi. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. The result is in degree. Note that all parameters are obligatory. Example: aoi s3 f5 w1 0 1.</td></tr>
<tr><td>AOR sk fi zi wi rel_apeX rel_apeY</td><td>Angle of refraction (or reflection) of a ray with respect to the local surface normal. All parameters, surface sk, field fi, zoom position zi, wavelength wi are obligatory. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. The result is in degree. Example: aor s3 f5 w1 0 1 < 15.</td></tr>
<tr><td>AOE sk fi zi wi rel_apeX rel_apeY</td><td>Angle of exit of a ray with respect to the local surface normal. Note that this command is synonymous the the AOR command given above.All parameters, surface sk, field fi, zoom position zi, wavelength wi are obligatory. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. The result is in degree. Example: aoe s3 f5 w1 0 1 < 15.</td></tr>
<tr><td>X si fi zi wi rel_apeX rel_apeY</td><td>Ray X-coordinate at surface si, field fi, zoom position zi, wavelength wi. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. Note that all parameters are obligatory. Example: x s3 f5 w1 0 1 = 10.</td></tr>
<tr><td colspan="2">continued on next page</td></tr>
</table>

continued from previous page	
Y si fi zi wi rel_apeX rel_apeY	Ray Y-coordinate at surface si, field fi, zoom position zi, wavelength wi. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. Note that all parameters are obligatory. Example: y s3 f5 w1 0 1 = 10
Z si fi zi / wi rel_apeX rel_apeY	Ray Z-coordinate at surface si, field fi, zoom position zi, wavelength wi. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. Note that all parameters are obligatory. Example: z s3 f5 w1 0 1 = 10
CX si fi zi wi rel_apeX rel_apeY	Ray X-direction cosine at surface si, field fi, zoom position zi, wavelength wi. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. Note that all parameters are obligatory. Example: cx s3 f5 w1 0 1 = 0.1
CY si fi zi wi rel_apeX rel_apeY	Ray Y-direction cosine at surface si, field fi, zoom position zi, wavelength wi. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. Note that all parameters are obligatory. Example: cy s3 f5 w1 0 1 = 0.1
CZ si fi zi wi rel_apeX rel_apeY	Ray Z-direction cosine at surface si, field fi, zoom position zi, wavelength wi. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. Note that all parameters are obligatory. Example: cz s3 f5 w1 0 1 = 0.1
CXN si	X-direction cosine of vertex surface normal on surface si. Example: cxn s3 = 0.1
CYN si	Y-direction cosine of vertex surface normal on surface si. Example: cyn s3 = 0.1
CZN si	Z-direction cosine of vertex surface normal on surface si. Example: czn s3 = 0.9
XSC si	Vertex Y-coordinate of surface si. The coordinate returned is referred to the global coordinate system. If GLO sk—yes is defined, the X-coordinate is referred to the vertex coordinate of surface sk. Example: xsc s3
YSC si	Vertex Y-coordinate of surface si. The coordinate returned is referred to the global coordinate system. If GLO sk—yes is defined, the Y-coordinate is referred to the vertex coordinate of surface sk. Example: ysc s3
ZSC si	Vertex Z-coordinate of surface si. The coordinate returned is referred to the global coordinate system. If GLO sk—yes is defined, the Z-coordinate is referred to the vertex coordinate of surface sk. Example: zsc s3
XSG si	Vertex X-coordinate of surface si referred to the global coordinate system of the system. Use commands XSC and GLO, if reference to another (preceding) surface is required.
YSG si	Vertex Y-coordinate of surface si referred to the global coordinate system of the system. Use commands YSC and GLO, if reference to another (preceding) surface is required.
ZSG si	Vertex Z-coordinate of surface si referred to the global coordinate system of the system. Use commands ZSC and GLO, if reference to another (preceding) surface is required.
	continued on next page

continued from previous page

PATH si..j fi zi wi rel_apeX rel_apeY	Physical path-length along a ray between surfaces si..j, at field fi, zoom position zi, wavelength wi. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil.
OPL si..j fi zi wi rel_apeX rel_apeY	Optical path-length along a ray between surfaces si..j, at field fi, zoom position zi, wavelength wi. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. The optical path length is n·PATH where n is the index of refraction at the specified wavelength.
ET si..j|sk height_X height_Y	Edge thickness between surfaces si..j at surface coordinates (height_X, height_Y).
SPD [wi..j fi..j zi..j]	Spot diameter (rms).
SPX [wi..j fi..j zi..j]	Spot diameter (rms), X-section.
SPY [wi..j fi..j zi..j]	Spot diameter (rms), Y-section.
WAV [wi..j fi..j zi..j]	Wavefront aberration (rms).
SPA [zi]	Third order spherical aberration
COMA [zi]	Third order coma
ASTI [zi]	Third order astigmatism
PETZ [zi]	Third order Petzval Sum
DIST [zi]	Third order distortion
LCA [zi]	Third order longitudinal colour
TCA [zi]	Third order tranversal colour
LAC wi..j [fi..j zi..j]	real ray transversal colour
DISX [zi..j fi..j]	Distortion (in %) in X-direction
DISY [zi..j fi..j]	Distortion (in %) in Y-direction
FDISX [zi..j fi..j]	F-Theta distortion (%) in X-direction
FDISY [zi..j fi..j]	F-Theta distortion (%) in Y-direction
MTFA [wi..j zi..j fi..j]	Mean value of sagittal and tangential MTF, values range between 0 and 1. The MTF is computed at the spatial frequency defined by the MFR command. Note, that MTF is usually maximized, that is the target value is 1.
MTFT [wi..j zi..j fi..j]	MTF tangential, values range between 0 and 1. The MTF is computed at the spatial frequency defined by the MFR command. Note, that MTF is usually maximized, that is the target value is 1.
MTFS [wi..j zi..j fi..j]	MTF sagittal, values range between 0 and 1. The MTF is computed at the spatial frequency defined by the MFR command. Note, that MTF is usually maximized, that is the target value is 1.
UA [si..j zi..j] UMY [si..j zi..j]	Paraxial direction angle of the marginal aperture ray. Note: UA and UMY are synonymous.

continued on next page

continued from previous page

HA [si..j zi..j] HMY [si..j zi..j]	Paraxial height of the marginal aperture ray. Note: HA and HMY are synonymous.
UB [si..j zi..j] UCY [si..j zi..j]	Paraxial direction angle of chief ray. Note: UB and UCY are synonymous.
HB [si..j zi..j] HCY [si..j zi..j]	Paraxial height of chief ray. Note: HB and HCY are synonymous.
WEI [si..j]	Weight (in g/cm^2)
MFL	Module focal length
VIG [fk]	Vignetting factor relative to field 1. Values are returned between 0 (100% vignetting) and 1 (no vignetting). If fk is omitted, the maximum field is used.
TSF [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on test-plate fit. Note that TSF is the sensitivity on DLF tolerance. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLF or a description of test plate fit on page 393. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLF 2.0 (fringes) for calculating tolerance sensitivity TSF.
TSI [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on surface irregularity. TSI is the sensitivity on IRR tolerance. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command IRR on page 389. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes IRR 0.4 (fringes) for calculating tolerance sensitivity TSI.
TST [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on surface thickness (distance). Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLT on page 389. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLT 0.02 for calculating tolerance sensitivity TST.
TSN [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on index of refraction. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLN, page 389, for defining index tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLN 0.001 for calculating tolerance sensitivity TSN.
TSV [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on dispersion. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLN, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLV 0.008 (0.8%) for calculating tolerance sensitivity TSV.

continued on next page

continued from previous page	
TSX [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on X-decenter. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLX, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLX 0.02 (mm) for calculating tolerance sensitivity TSX.
TSY [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on Y-decenter. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLY, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLY 0.02 (mm) for calculating tolerance sensitivity TSY.
TSZ [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on Z-decenter. A Z-decenter is equivalent to a thickness tolerance. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLZ, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLZ 0.05 (mm) for calculating tolerance sensitivity TSZ.
TSA [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on tilt about X-axis. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLA, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLA 5 (arcmin) for calculating tolerance sensitivity TSA.
TSB [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on tilt about Y-axis. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLB, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLB 5 (arcmin) for calculating tolerance sensitivity TSB.
TSG [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on tilt about Z-axis. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLG, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLG 5 (arcmin) for calculating tolerance sensitivity TSG.
TSH [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on index homogeneity. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command HOM, page 389, for defining homogeneity tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes HOM 50 ($50 \cdot 10^{-6}$) for calculating tolerance sensitivity TSH.
	continued on next page

continued from previous page	
`TSR [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on radius change. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLR, page 389, for defining homogeneity tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes a radius change `DLR 0.0025` (mm) for calculating tolerance sensitivity `TSR`.

19.7 Controlling Contrast vs. Resolution

Optimizing for spot (`SPD`) or wavefront (`WAV`) alone is often not a sufficient criterion for achieving the desired result and a finer adjustment of the spot or wavefront shape may be necessary. In particular, emphasizing the central core of a spot will increase spatial resolution at the expense of a lowered contrast. The `WTA` command, as described below, allows the designer to balance performance between contrast and resolution.

`WTA [zk] aperture_weight`	Weight on aperture. Controls relative weight given to the center of each ray bundle (high values) vs. the edge. The effect of this parameter is to balance between contrast and resolution. Typical values: **weight** / **Conditions** 0.0 — High contrast, good resolution 0.5 — Good contrast, high resolution 1.0 — Low contrast, very high resolution. See also examples below.

The relative weight across the aperture follows the function

$$W = e^{-(WTA \cdot r)^2} \tag{19.7}$$

where r is the relative aperture radius and W is the relative weight (a number between 0 and 1) applied to the ray. This function is similar to the apodization function as described in section 7.3.6 (page 51). The main difference, however, is that `WTA` is *only* applied to spot or wavefront calculation in optimization, whereas pupil apodization is applied to *all* performance analyses. That is, pupil apodization -if defined- is always in effect, `WTA` is only used in optimization. Also note that Eq. 19.7 indicates arbitrary `WTA` values, however, for best performance $0 \leq \texttt{WTA} \leq 1$ is recommended.

Figs. 19.3 and 19.4 show the effect of `WTA` on spot (or wavefront) shape.

19.8 Glass Optimization and Glass Map Boundary Points

It is sometimes desirable to let glasses ”float” during optimization, i.e. the optimizer selects an appropriate glass in a continuous $n - \nu$ domain. To accomplish this, the DNO and/or DVO variables at a surface must be activated, which means that index and dispersion may vary during optimization and appropriate n and ν offsets are applied to the base glass. Internally, a glass with DNO/DVO offsets is modelled as a fictitious glass. It is, however, necessary to constrain the range in which index n and dispersion ν may vary, because otherwise n and ν will likely arrive at infeasible points.

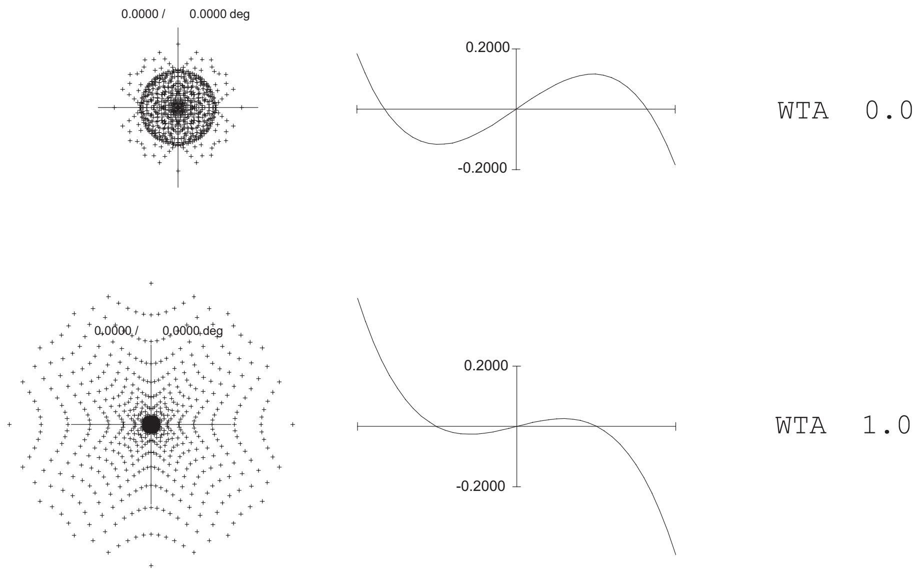


Figure 19.3: Effect of 'weight on aperture' (WTA) on spot shape (left) and transverse aberrations (right), by minimizing spot diameter (e.g. `spd f1 0`). High values emphasize the central core of the spots at the expense of a larger blur.

This range is defined by a **convex** polygon in the standard SCHOTT diagram, describing the outer boundaries of the allowable area in which the glasses must lie. Up to 20 polygon points may be specified. The following diagram shows the default glass polygon which encloses the majority of the SCHOTT glasses:

The error value of a fictitious (floating) glass is defined by the (vertical) distance of the fictitious $n - \nu$ coordinates from each boundary line. The error values must always be negative in order for the fictitious glass to stay within the glass map boundary polygon.

The glass map boundary ('glass polygon') is specified using the following command syntax:

`GLP corner1 corner2 ... n`
 	
or
 	
`GLP DEF`
	Define glass map corner points ("glass polygon"). The glass map boundary points can be specified by the following forms:
	xxx.yyy — Fictitious glass code. For example 514.643
	nnnnnnn — A six-digit glass code. For example 514643
	predefined glass — A 1- to 10-character alphanumeric code from the predefined glass catalogue.
	Mixed forms are permitted. Note that the polygon must be **convex** and corners must be specified in clockwise orientation in the $n - \nu$ diagram. Examples :
	`GLP 481.850 820.501 900.234 560.410 481.850`
	`GLP BK7 N-Lak9 SF6 F2 BK7`
	`GLP BK7 683542 SF6 531.422 BK7`
	The alternate form `GLP DEF` restores the default glass map boundary according to table 19.8.

`EDI GLP`	Edit glass map boundaries in a dialog.

The current setting of the glass map boundaries may be listed by the command `LIS GLP`. The default glass map boundaries are defined by a 7-point polygon in the $n - \nu$ domain (see also Fig. 19.5), to

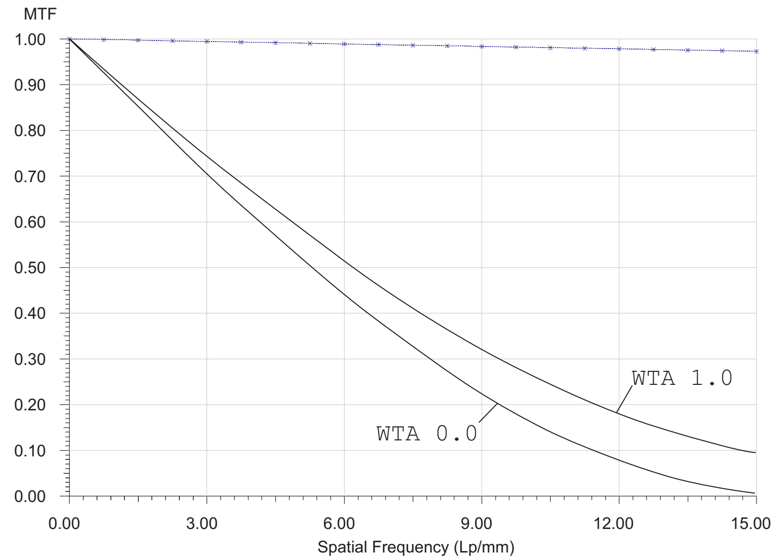


Figure 19.4: Effect of 'weight on aperture' (WTA) setting on MTF. High values improve the high-frequency components of MTF (i.e. high resolution), low values improve the low-frequency components of MTF (i.e. high contrast). Note that the curves above only show the case of improving high-frequency components.

match the domain of current SCHOTT glasses.

Notes:

The DNO and DVO variables are understood in a continuous $n - \nu$ domain, in contrast to the fixed properties of real glasses. Thus, n and ν offsets are fictitious additives to the currently selected glass. The dispersion offset is modelled as a fictitious MIL-glass which lies perfectly on the so-called Abbe-line ("normal" line).

A glass map polygon must be closed, that is, the last corner must be identical with the first corner.

Fictitious glasses obtained after an optimization run can be converted to a regular catalogue glass by the REG command (see also page 194). This option searches for the nearest catalogue glass on the basis of the DNO/DVO offsets and automatically replaces the continuous glass model by a fixed catalogue model. The REG option, however, does not eliminate DNO/DVO variables on that glasses.

Point	n_d	ν_d
1	87.00	1.4800
2	41.00	1.8900
3	20.00	1.9300
4	25.00	1.7700
5	37.00	1.5700
6	57.00	1.4900
7	87.00	1.4800

Table 19.8: Default glass map boundaries matched to SCHOTT glasses.

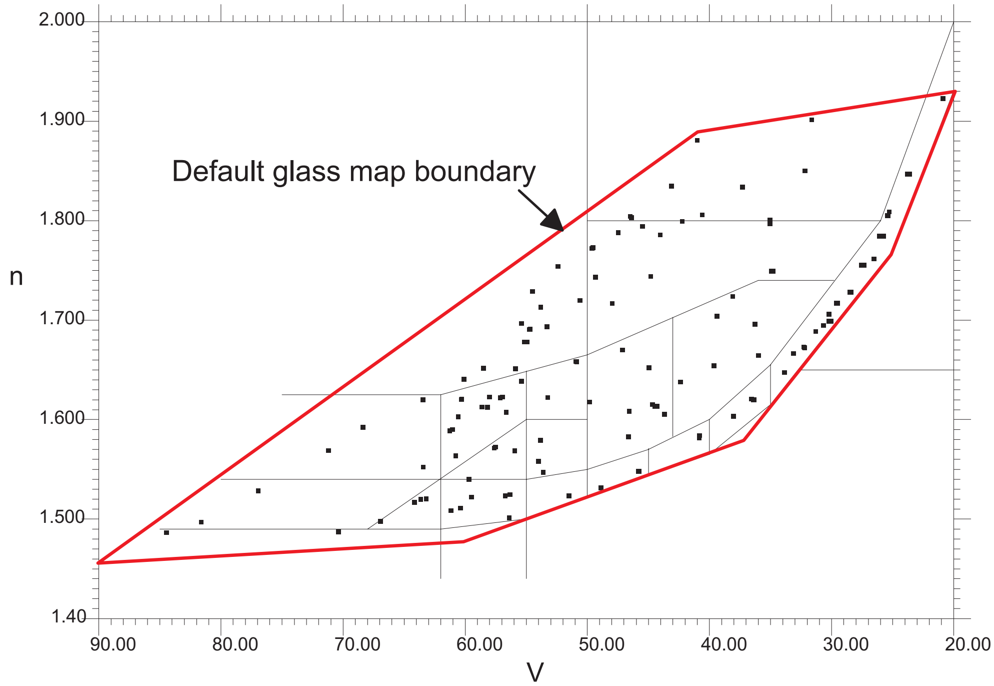


Figure 19.5: Definition of default glass map boundary.

19.9 Run the Optimization (`OPT`)

Once variables, targets and constraints are defined, the optical system can be optimized.

`OPT [LM	KT] [n_steps]`	Run the optimization. The optional parameters `LM`, and/or `KT` specify the algorithm to be used. See also the guidelines for selecting the appropriate algorithm. If neither `LM`, nor `KT` is specified, the selected method of the previous optimization run is repeated. Initially, KT-optimization will be used. `n_step` defines the maximum number of optimization steps (iterations). If no parameter is given, the default number of iterations is `n_steps = 10`.
`UNDO OPT`	Undo last optimization, i.e. it restores the state of the optical system before the optimization. This command is particularly useful if the optimization run failed to converge. For example, ill-conditioned or contradictory constraints will often lead to infeasible conditions. Undo is a one-step operation, i.e. only the last optimization can be undone.
`ERRF`	Print detailed error (merit) function including the error contributions of each constraint. This is a diagnostic tool to identify the most disturbing aberrations. It does not run the optimization.

Examples:

```
opt             ! initially uses KT-optimization, otherwise the method from the previous
                ! optimization run is repeated.
opt lm 5        ! uses LM-optimization, stop after 5 iterations.
opt lm kt 10    ! LM- and KT-optimization are executed successively, 10 iterations each,
opt kt          ! KT-optimization only.
```

19.9.1 Selecting the appropriate Optimization Method

As described in sections 19.1, 19.2, $OpTaliX$ provides two different optimization methods (KT- and LM optimization), and the question may arise which method is preferred under certain conditions. This section describes the pros and cons of each method and attempts to give recommendations for various cases.

The **Kuhn-Tucker (KT)** algorithm solves constraints (i.e. $=, >, <$ operations) exactly, while other functions are solved in a least-squares sense. It provides precise control of the constraints and it is not necessary to choose appropriate weights for each constraint and modifying it as the design process evolves. However, the user may (temporarily) overrule exact solving of equality constraints by the WTC command, which converts behaviour of the KT-optimization only for that specific constraint similar to properties of the LM-optimization (i.e. weighting that constraint).

If lens parameters are to be exactly controlled, for example object-image distance OAL, the KT-optimization gives *exact* solutions. Due to the highly non-linear nature of almost all aberrations in optical systems, it takes a few iterations to accurately control the desired parameters.

In the hands of an inexperienced user, however, the KT-optimization may cause difficulties, depending on the problem definition. For example, if a user inadvertently defines incompatible conditions, the resultant equations become indeterminate and optimization will not proceed. In such cases the program issues a warning message and prints the conflicting constraint(s).

Note that KT-optimization is the preferred (default) method in $OpTaliX$.

Basically, the **Levenberg-Marquardt (LM)** algorithm is an unconstrained damped least-squares algorithm. Constraints (i.e. $=, >, <$ operations) are handled like aberrations, except that higher weights are generated internally for these functions. This approach is preferable when the design is at an early stage of development and the optical performance is far from the design goal. In case of improperly defined or even incompatible constraints, it is unlikely that the LM-optimization will destroy the design. Contrary to the KT-optimization, the program will simply find the best compromise between the incompatible conditions. That is, it will rather 'squeeze' the design smoothly into a different form, which in almost all cases is still computable. Boundary conditions ($<, >$), for example, are not solved precisely, instead they are held very close to the desired target. One particular advantage is that constraints can be given large or small weight, depending on their importance. On the other hand it requires that constraint weights and target weights must be properly balanced to achieve the desired result.

Note that the optimization routines can only solve problems which have been specified by the user. In particular, they cannot

- Violate the law of optics,
- solve for more constraints than the number of variables you have provided,
- Solve for a constraint when there is no variable for it,
- add or remove elements or dramatically re-arrange the optical system,

- control aberrations that are uncorrectable (for examples astigmatism in doublets, distortion in eyepieces).

19.9.2 MTF Optimization

Using the modulation transfer function (MTF) directly as target in optimization often leads to unsatisfactory success, particularly to less experienced designers. One major problem with using MTF optimization is the fact that MTF values may oscillate significantly as a function of construction parameters. To illustrate the problem, consider the change of MTF as a function of defocus, i.e. when the image plane is moved forward and backward along the optical axis. Fig. 19.6 indicates the large MTF variation as the image plane is moved away from the optimum position (axial distance = 0). The success of the optimization will now depend on the initial starting point. Assume we have chosen staring point (1), which is at an axial distance $z \approx 0.6$, the side maximum will be found, because a locally optimizing algorithm cannot jump over adjacent minima/maxima.

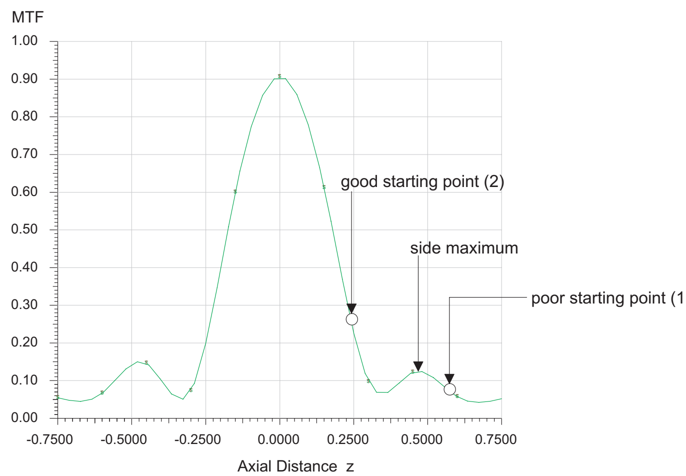


Figure 19.6: Variation of diffraction MTF for a perfect lens as a function of defocus.

A better starting point would be (2) where the optimization algorithm can find the 'true' MTF maximum without intermediate valleys. It is more realistic to use MTF optimization for systems which are close to the optimum and which can benefit from a final tuning. It is therefore good practise to run optimization using spot diameter (SPD) or wavefront variance (WAV) *prior* to optimizing MTF directly.

19.10 Optimizing for Tolerance Sensitivity

In the (iterative) design and optimization process it is often wanted not only to reduce aberrations, but also reducing sensitivity for parameters, such as decenter, tilt, thickness tolerances, etc. The driving force are manufacturing issues where manufacturing tolerances as large as possible are desired.

$OpTaliX$ helps you to simultaneously optimize for image performance and tolerance sensitivity on selected parameters, already in the design stage. Tolerance sensitivity is a measure for the change of performance $\Delta\Phi$ (aberration, merit function) given a certain perturbation Δx of a construction parameter. Therefore, $OpTaliX$ attempts to minimize the tolerance sensitivity function S

$$S = \sqrt{\sum_{i}^{N} \left(\frac{\Delta \Phi}{\Delta x} \right)^2} \tag{19.8}$$

where i is the surface number. The performance change $\Delta\Phi$ is always calculated on the basis of wavefront aberration (WAV) for each tolerance item. It should be noted that optimizing for **both** performance Φ and tolerance sensitivity S is a contradictory process. It often seems impossible to reduce tolerance sensitivity without sacrificing performance. Generally, a subtle balance between Φ and S must be selected. Finding this balance is the responsibility and skill of the optical designer. Further information on this subject is also given by Grey [16], and Isshiki et.al, [23].

19.10.1 Tolerance Sensitivity Items

$OpTaliX$ provides several commands to calculate tolerance sensitivity, TSF, TST, TSI, TSN, TSV, TSX, TSY, TSZ, TSA, TSB, TSG, as defined in section 19.6 (page 350). These tolerance sensitivity commands assume that an appropriate tolerance has been assigned in thetolerance editor (page 392). If tolerances on requested parameters are not available, respectively not defined in the tolerance option (sect. 22.5, page 399), the program assumes the following parameter changes (tolerances) Δx for calculating tolerance sensitivity:

Item	Effect	Default tolerance
TSF	Sensitivity on surface fit tolerance (DLF)	Δx = 2 fringes
TSI	Sensitivity on surface irregularity tolerance (IRR)	Δx = 0.4 fringes
TST	Sensitivity on axial thickness tolerance (DLT)	Δx = 0.1 mm
TSN	Sensitivity on refractive index tolerance (DLN)	Δx = 0.001
TSV	Sensitivity on dispersion tolerance (DLV)	Δx = 0.008
TSR	Sensitivity on radius of curvature tolerance (DLR)	Δx = 0.0025
TSX	Sensitivity on X-decenter tolerance (DLX)	Δx = 0.02 mm
TSY	Sensitivity on Y-decenter tolerance (DLY)	Δx = 0.02 mm
TSZ	Sensitivity on Z-decenter tolerance (DLZ)	Δx = 0.05 mm
TSA	Sensitivity on α-tilt (about X-axis) (DLA)	Δx = 5 arcmin
TSB	Sensitivity on β-tilt (about Y-axis) (DLB)	Δx = 5 arcmin
TSG	Sensitivity on γ-tilt (about Z-axis) (DLG)	Δx = 5 arcmin
TSH	Sensitivity on homogeneity tolerance (HOM)	$\Delta x = 50 \cdot 10^{-6}$

Commands for defining Tolerance Sensitivity Items	
TSF [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on test-plate fit. Assumes that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLF or a description of test plate fit on page 393. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLF 2.0 (fringes) for calculating tolerance sensitivity TSF.
TSI [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on surface irregularity. Assumes that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command IRR on page 389. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes IRR 0.4 (fringes) for calculating tolerance sensitivity TSI.

continued on next page

continued from previous page

`TST [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on surface thickness (distance). Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLT on page 389. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLT 0.02` for calculating tolerance sensitivity `TST`.
`TSN [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on index of refraction. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLN, page 389, for defining index tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLN 0.001` for calculating tolerance sensitivity `TSN`.
`TSV [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on dispersion. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLN, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLV 0.008` (0.8%) for calculating tolerance sensitivity `TSV`.
`TSX [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on X-decenter. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLX, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLX 0.02` (mm) for calculating tolerance sensitivity `TSX`.
`TSY [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on Y-decenter. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLY, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLY 0.02` (mm) for calculating tolerance sensitivity `TSY`.
`TSZ [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on Z-decenter. A Z-decenter is equivalent to a thickness tolerance. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLZ, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLZ 0.05` (mm) for calculating tolerance sensitivity `TSZ`.
`TSA [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on tilt about X-axis. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLA, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLA 5` (arcmin) for calculating tolerance sensitivity `TSA`.

continued on next page

continued from previous page	
`TSB [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on tilt about Y-axis. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLB, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLB 5` (arcmin) for calculating tolerance sensitivity `TSB`.
`TSG [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on tilt about Z-axis. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLG, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLG 5` (arcmin) for calculating tolerance sensitivity `TSG`.
`TSH [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on index homogeneity. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command HOM, page 389, for defining homogeneity tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `HOM 50` ($50 \cdot 10^{-6}$) for calculating tolerance sensitivity `TSH`.
`TSR [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on radius change. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLR, page 389, for defining homogeneity tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes a radius change `DLR 0.0025` (mm) for calculating tolerance sensitivity `TSR`.

19.10.2 Using Tolerance Sensitivity Items in Optimization

If optimizing (minimizing) for tolerance sensitivity, the various tolerance sensitivity items described in the previous section should be understood as *aberrations* added to the targets/constraints (merit function) list. The syntax for defining tolerance sensitivity in optimization is found in sect.27.1, page 465. Here is a typical example in the optimization targets/constraints list:

`efl = 100`	Focal length shall be exactly 100mm.
`spd 0`	Spot diameter (rms) shall be zero (minimized) for all fields, wavelengths, zoom positions.
`tsa s1..5 f1..2 w1 0`	Tolerance sensitivity on surface tilt about X-axis shall be minimized for surfaces 1-5, fields 1-2 and wavelength number 1.
`tsy 0`	Tolerance sensitivity on surface Y-decenter shall be minimized for **all** surfaces, **all** fields and **all** wavelengths defined in the system configuration.

Notes:

- Do not attempt to request a tolerance sensitivity item to become exactly zero, e.g. '`TSA = 0`' as this is impossible on elements/surfaces that have optical effect. Instead minimize it by omitting the equal '=' sign in the constraints definition, e.g. '`TSA 0`'.

- It is generally advisable to start with low weights on tolerance sensitivity constraints, for example
  `tsa sa f1 0 ; wt = 0.1`
  By gradually increasing the corresponding weight, an acceptable compromise between performance and general tolerance sensitivity is quickly found.

19.11 Description of Output

A typical output from an optimization run is shown below (load `\optalix\examples\double_gauss-2.otx` and change the target EFL to 60mm).

```
KT OPTIMIZATION:
  Number of variables            :    13
  Number of functions            :  2754
  Number of equality constraints   :     1
  Number of inequality constraints :     4
  Number of internal constraints   :     4

OPTIMIZATION PARAMETERS :
  Number of iterations          : min =  2   max =  15
  ORGR (Optimization Ray Grid)  : 16
  IMPR (Fractional Improvement) : 0.01000
  WTA  (Weight on Aperture)     : 0.00000
  DEFC (Default Constraints)    : Yes

Targets/Constraints                   Target       Function         Error  Violation
efl = 60.                          60.000000      49.999580    -10.000420  **
spd 0                               0.000000       0.009321      0.009321

Default Constraints                   Target       Function         Error
MAE S6                      >       0.002000       5.274917      5.272917
MXT S7                      <      10.147716       2.009000     -8.138716
MNT S7                      >       2.029543       2.009000     -0.020543  *
MNE S7                      >       2.029543       4.229104      2.199560

Iter          Min.          Equal.        Inequal.        DumpingF.     Improv.
   0       0.398957       3.162344       0.143329     1.000000
   1       6.332909       1.710547       0.000000     1.000000       -14.87367
   2       1.941585       0.635908       0.000000    0.6250000E-01     0.69341
   3       0.470827       0.452288       0.000000    0.2322369E-02     0.75750
   4       0.217870       0.145856       0.000000    0.1628259E-02     0.53726
   5       0.206532       0.144571       0.000000    0.1017662E-03     0.05204
   6       0.183684       0.066643       0.000000    0.1017662E-03     0.11063
   7       0.168225       0.075135       0.000000    0.1017662E-03     0.08416
   8       0.159436       0.158571       0.000000    0.5045660E-04     0.05224
   9       0.154823       0.011828       0.000000    0.8971902E-04     0.02893
  10       0.152053       0.022684       0.000000    0.1048387E-03     0.01789
  11       0.151615       0.013254       0.000000    0.7260012E-04     0.00288

Optimization stopped. Improvement is less than    0.01000 (1.00%)

Targets/Constraints                   Target       Function         Error  Violation
efl = 60.                          60.000000      60.000176      0.000176
spd 0                               0.000000       0.004111      0.004111

Default Constraints                   Target       Function         Error
MAE S6                      >       0.002000      11.715292     11.713292
MXT S7                      <      10.147716       5.052111     -5.095605
MNT S7                      >       2.029543       5.052111      3.022568
MNE S7                      >       2.029543       7.034676      5.005132
```

In the first section a listing of the number of variables and constraints is shown. Equality and inequality constraints are separately listed. Following this is a list of the user-defined constraints with the target-, function- and error-values of the starting system (i.e. prior to optimization).

The last column indicates violations on constraints (i.e. equal, less than or greater than), shown as a bar of asterisks (*) in steps of 10%. The maximum bar length is ten asterisks corresponding to 100% deviation.

If requested, default constraints are tabulated. These are constraints created internally by the program for all variable thicknesses in order to maintain reasonable minimum/maximum element, air-space and edge thickness dimensions. The DEFC command enables (Yes) or disables (No) default constraints.

Each iteration step outputs the merit functions on constraints to be minimized ('Min.' column), to be held exactly ('Equal.' column), and the inequality ('Inequal.' column) constraints together with the current dumping factor and a relative improvement compared to the previous iteration step. For example, a relative improvement factor 0.01 corresponds to a 1% improvement with respect to the previous iteration. Note that the improvement factor only applies to the KT (Kuhn-Tucker) optimization; it is ignored in the LM (Levenberg-Marquart) optimization.

Iteration terminates if the improvement factor is below a threshold defined by the IMPR command. The error function components of the refined optical system are listed.

19.11.1 List of Active Constraints

Inequality constraints are dynamically added or released during optimization, depending on whether they are violated by a solution or if they are in an acceptable region. When constraints are released they are allowed to drift into the acceptable region without affecting the solution. When constraints are added, the derivatives of the new constraints are calculated and added to the matrix. This causes additional 'minor' solution cycles to be calculated.

Active constraints are only reported if enabled in the Optimization Parameters dialog (there is currently no command line equivalent). From the main menu, select *Optimization* $--\,>$ *Optimization Parameters* and in the 'Kuhn-Tucker (KT)' tab check 'Show active constraints for each cycle'. A sample output would be

```
Active Constraints (  4)                  Value          Target            Cost
thi s3 > 8                              7.06120         8.00000    -0.415859E+01
thi s5 > 8                              7.50000         8.00000     0.319846E+00
MNE S3                                  0.63518         2.40000     0.227979E+02
MNE S5                                  1.65757         2.40000     0.359234E+01
```

The output includes target/boundary values, the actual value and the relative "cost" of imposing the constraints. The relative cost is the "pressure" that a constraint applies to the solution.

Inactive constraints are not included in the 'active constraints' listing. Only if a constraint becomes active, it shows up in the constraints listing.

19.12 Terminating Optimization

Optimization is terminated if

- the maximum number of iterations is reached, or
- the ESC-key has been pressed on the keyboard, or
- the fractional improvement of the merit function is below a certain limit value, or
- the number of ray trace errors (if any) has exceeded a certain limit.

The maximum number of iterations is set in the optimization parameter dialog or by the command MXC. See section 19.14 for further information.

Optimization can also be interrupted if ray trace errors occur and a certain number of ray errors has been exceeded. The limit of allowable ray trace errors is set by the OERR command. See sect. 19.14 for details.

The limit on fractional improvement of the merit function is set by the IMPR command. That is, if the improvement of the merit function is smaller than `IMPR`, optimization will be terminated.

If the ESC-key is pressed, a dialog box will be invoked asking the user whether to terminate or to continue optimization. Note that it may take a while for the dialog to appear because a running iteration step must first be finished. It is therefore recommended to press the ESC-key only once.

A prematurely terminated optimization leaves the optical system in the state of the last iteration step, that is, before the ESC-key was pressed. This state is most likely not the optimum condition (i.e. minimum aberrations), however there are numerous reasons to interrupt optimization (for example, convergence is low, inappropriate variables/constraint settings, time reasons, etc).

19.13 Undo Optimization

Optimization can be ”undone” by selecting from the main menu *Optimization –> Undo last optimization step*, or from the command line

```
UNDO OPT
```

Note that ”undo” only applies to the *last* optimization run. Multiple subsequent optimization cycles (prior to the last cycle) cannot be undone. It is recommended to save promising solutions in separate files.

19.14 Optimization Parameters

The following commands allow control of the optimization process.

EDI OPT	Edit operating parameters for optimization algorithms. Note that the command 'EDI OPR' is obsolete but still supported. Instead, use of the 'EDI OPT' command is encouraged.
MXC max_cycles	Maximum number of permitted cycles. The optimization will be terminated if that number of cycles is completed. Termination will probably occur before if the fractional improvement is less than the improvement factor (see IMPR command below).
MNC min_cycles	Minimum number of required cycles. Optimization will not exit earlier.
IMPR min_impr_factor	Fractional improvement. Optimization is terminated if the improvement of the error function is less than IMPR. Example: IMPR 0.01 corresponds to 1% improvement. Termination may occur before the maximum number of cycles (MXC) is reached.
ORGR num_opt_rays	Number of rays across pupil in optimization. Permissible values of num_opt_rays are 4, 8, 16, 32, 64, 128, 256 and 512. However, ORGR must always be smaller than NRD. See the notes below.
OERR error_limit	Error limit. Optimization is terminated if the number of ray trace errors (if any) exceeds error_limit. Enter OERR 0 for disabling this feature.

Dialog based editing of optimization parameters is accomplished from the main menu, *Optimization –> Parameters*. The dialog box as shown in Fig. 19.7 contains several tabs. In the main (general) tab, the optimization algorithms are selected. In addition, it controls the level of outputs generated for each optimization cycle.

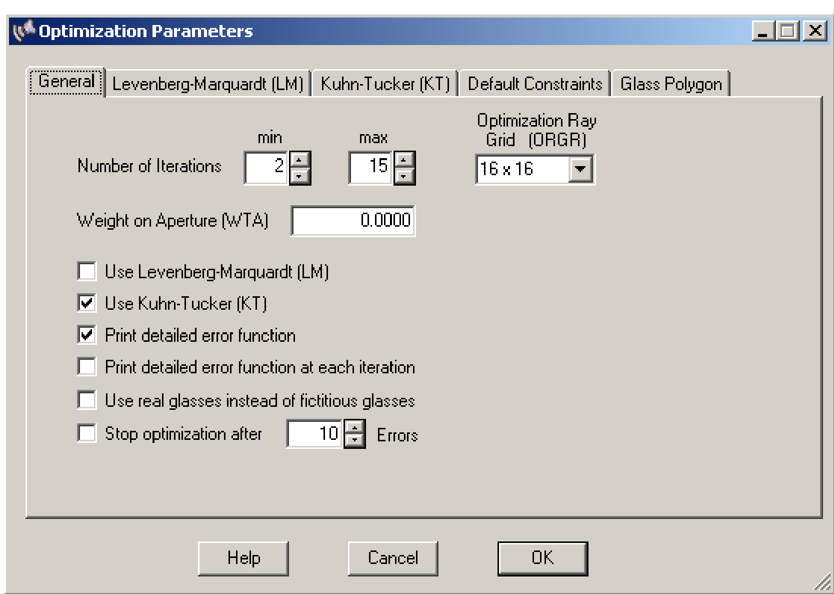


Figure 19.7: Optimization parameters main dialog.

Notes:

The optimization ray grid defines the number of rays across the pupil diameter during optimization. This setting must not be confused with the number of rays used for performance analysis (see NRD command). Setting the optimization ray grid (ORGR) to a value lower than NRD will only reduce the number of rays *during* optimization. For example, selecting ORGR 16x16 and NRD 32 will only use every second ray in the ray matrix during optimization. This accelerates the speed of optimization by a factor 4, whereas all performance analyses (e.g. spot, PSF, MTF, etc.) still use the 32x32 ray grid.

20

Coatings

Optical components are usually coated with thin layers of solid materials for the purpose of altering their physical or optical properties. Depending on the application, only one thin layer or a stack of as many as fifty to over hundred layers are deposited to produce the desired optical behaviour. The terms "*multi-layer*" respectively "*coating*" in the following sections are used as generic terms for single or multiple thin films on optical surfaces.

The design, analysis and optimization of multi-layer coatings (thin films) is seamlessly integrated to $OpTaliX$. Thus, it is not necessary to perform a multi-layer design in a separate program and then laboriously transfer (import) the data to $OpTaliX$.

One single coating can be loaded during a session. It will be stored in memory in parallel to the classical optical surface data and it can be modified, optimized and analyzed independently from the optical system. Once the performance is considered sufficient, it may be attached to a particular optical surface or a range of surfaces (see also section 20.5).

$OpTaliX$ also allows access to coating designs from other thin-film packages such as "The Essential MacLeod" and "Thin-Film-Calc (TFCalc)". See sect. 29 (page 473) on importing coating designs from these packages.

Nomenclature: In the commands and the options to follow, "`COA`" always refers to the single coating stored in the coating editor; it can be independently edited and optimized from the system prescription data. If "`MUL`" is indicated in a command syntax, it refers to the coating *attached to a surface*. Note that a coating attached to a surface cannot be modified, it can only be removed (DEL MUL) or overwritten (ATT COA) by another coating stored in a file or in the coating editor.

20.1 Editing Coating Data

Coating prescriptions may be edited either from the command line (sect. 6) or from the GUI via a spreadsheet editor giving access to all layer parameters. The coating editor is invoked by the command

`EDI COA`

Note that the coating editor only allows modification of layer data (layer material, layer thickness, etc.) of a coating stack. The conditions of use of the coating stack (e.g. incidence angle, plotting parameters) are defined in the coating configuration dialog.

20.2 Coating Configuration

The coating configuration data pertain to the use of thin-film multilayer coatings. For example, coating configuration data are reference wavelength, incident angle, plot or analysis wavelength, etc. A

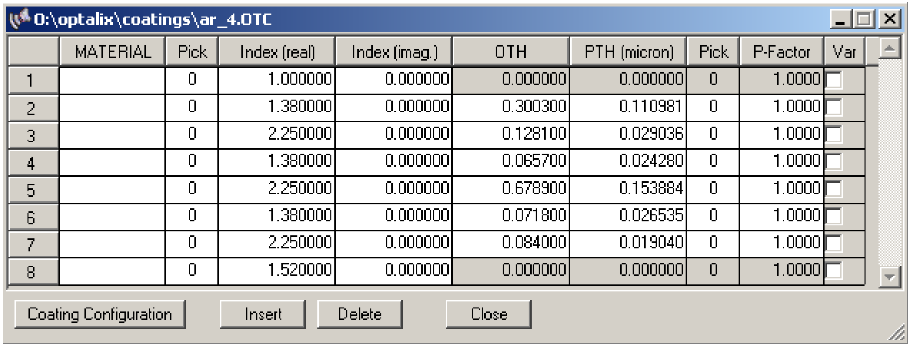


Figure 20.1: Coating editor, invoked by the command `EDI COA`.

dialog box for editing coating configuration data is invoked by the `EDI CCFG` command (see also command description in next section).

Important note: In this context, coating configuration data must not be confused with **system** configuration data (see `EDI CNF` command).

20.3 Coating Command Line:

`EDI CCFG`	Coating configuration dialog.

`RES COA [coating_name]`	Restore a coating from file and keep it in memory (in parallel to the lens data). The standard file extension is ".otc". In absence of the extension, it will be automatically added. If the optional parameter `coating_name` is missing, a dialog box will be opened. Once loaded into memory, the coating may be attached to an optical surface using the `ATT` command (see below). The file specified by `coating_name` must reside in the coating directory which is by default `$i\coatings`. Thus, it is not required to specify this path information explicitly. Examples of valid coating-file commands are:
	`res coa ar_coat.otc`
	`res coa ar_coat`

`SAV COA [coating_name]`	Save a coating to file "`coating_name`". The default directory where to the coating prescription is saved is `$i\coatings`. Do not modify this setting, because the stored file may not be loaded later (*OpTaliX* expects all coating files in this directory). In absence of `coating_name`, a dialog box is opened.
`LIS MUL [sk|si..j]`	Lists multilayer coatings attached to surfaces.
`DEL MUL [sk|si,,j]`	Delete multilayer coating on surfaces `sk|si..j`. The surface is then assumed uncoated. In subsequent polarization and transmission analyses, Fresnel equations are used.
`EDI COA`	Edit coating data using a spreadsheet.
`INV COA`	Invert a multilayer coating, including the incident/substrate media.

continued on next page

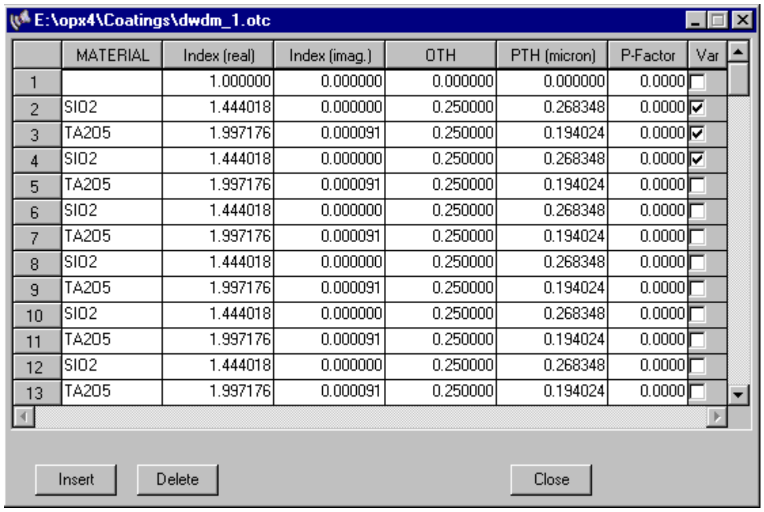
continued from previous page	
CREF coating_wavelength	Reference wavelength in μm of thin-film multilayer coating stack. The coating must have been loaded before (see RES COA command).
OTH li..j layer_thickness	Optical thickness (in wavelength units defined by the base wavelength). The physical thickness will be automatically evaluated according to the base wavelength.
PTH li..j phys_thick	Physical thickness (in mm) of the layer(s) lk|li..j . The optical thickness will be automatically evaluated according to the base wavelength.
INS li..j	Insert layer i to j
DEL li..j	Delete layer i to j
GLA li..j material	Material (glass) for layers i to j.
IND li..j real_index imag_index	Complex index of refraction of layer(s) i to j. Takes only effect, if no layer material (see GLA command above) is specified.
ATT sk|si..j [FILE coating_name | DEF]	Attach a multi-layer coating, stored in memory or in a file to surface(s) sk|si..j. The coating name refers to a file containing the coating prescription. The coating file MUST reside in the standard coating directory $OpTaliX$ (usually $i\coatings). If the option [FILE coating_name] is absent, the actual coating stored in memory will be attached. The optional parameter DEF assigns a 'default' coating, consisting of single quarter-wave thickness MgF2 layer to the designated surfaces.
MAN [R | T | A] [ANG incid_angle]	Numerical analysis of multi-layer performance. The analysis may be performed for : R = reflection, T = transmission. A = absorption If optional parameters (R or T) are omitted, all possible options (transmission, reflection, absorption) will be printed. An incidence angle (in degrees) can be optionally provided. In this case the ANG qualifier is obligatory. If ANG is omitted, the incidence angle specified in the coating configuration dialog (see EDI CCFG) is used.
COA LAM R|T|RP|TP	Plot reflection/transmission properties vs. wavelength (LAM = λ). R = reflection T = transmission RP = phase change on reflection TP = phase change on transmission
COA FLD R|T	Plot reflection/transmission properties vs. field (i.e. incidence angle). The wavelength used is the coating reference wavelength, which must not be confused with the reference wavelength in the optical system (see REF command). R = reflection T = transmission
	continued on next page

continued from previous page	
COA FLA R|T	Plot reflection/transmission properties vs. field (i.e. incidence angle) and wavelength as 2-dimensional surface plot. R = reflection T = transmission
COA GD R|T	Plot group delay vs. wavelength. R = reflection T = transmission
COA GDD R|T	Plot group delay dispersion (or group velocity dispersion) vs. wavelength. R = reflection T = transmission
FTAR	Define performance targets (see section 20.9.2 on page 378).
FOPT	Run the coating optimization.
CLS COA [colour...n]	Selects the colour list used for coating analysis plots corresponding to S, T and A (average). With no colours specified, colours are set to default settings. Examples: `cls coa red gre blu` ! defines red, green and blue for S, T and average plane. `cls coa` ! no colours specified, default coating colours are selected. See also names of predefined colours and their definition in sect. 28.1, page 471.
EXP COA R|T plane [fil filename]	Save (export) coating reflection/transmission performance to a file in ASCII format. `R|T` specifies reflection/transmission `plane` = polarization plane, `S` = s-plane, `P` = p-plane, `A` = average plane (S+P)/2 By default, output is directed to the text output screen. If a file name is specified ('`fil`' option), output is written to a file designated by 'filename'. This export option uses the parameters (max. angle, wavelength range, etc.) set in the general coating configuration (see also `EDI CCFG` command) **Example:** `exp coa R A fil c:\mycoat.txt`: exports reflection properties (R) for average polarization (A) to file `c:\mycoat.txt`.

Spreadsheet Entry:

The spreadsheet is invoked by the command `EDI COA` or from the main menu *Coatings –> Edit Layers*.

The meaning of the columns is:



	MATERIAL	Index (real)	Index (imag.)	OTH	PTH (micron)	P-Factor	Var
1		1.000000	0.000000	0.000000	0.000000	0.0000	☐
2	SIO2	1.444018	0.000000	0.250000	0.268348	0.0000	☑
3	TA2O5	1.997176	0.000091	0.250000	0.194024	0.0000	☑
4	SIO2	1.444018	0.000000	0.250000	0.268348	0.0000	☑
5	TA2O5	1.997176	0.000091	0.250000	0.194024	0.0000	☐
6	SIO2	1.444018	0.000000	0.250000	0.268348	0.0000	☐
7	TA2O5	1.997176	0.000091	0.250000	0.194024	0.0000	☐
8	SIO2	1.444018	0.000000	0.250000	0.268348	0.0000	☐
9	TA2O5	1.997176	0.000091	0.250000	0.194024	0.0000	☐
10	SIO2	1.444018	0.000000	0.250000	0.268348	0.0000	☐
11	TA2O5	1.997176	0.000091	0.250000	0.194024	0.0000	☐
12	SIO2	1.444018	0.000000	0.250000	0.268348	0.0000	☐
13	TA2O5	1.997176	0.000091	0.250000	0.194024	0.0000	☐

Figure 20.2: Editing coating data using a spreadsheet.

Material The material can be any glass/material name from the glass catalogue. If a blank name is specified, the complex index of refraction must be entered, which is always referred to the reference wavelength. This index is used for all wavelengths, hence material dispersion cannot be accounted for. For catalog glasses (i.e. a material name is given), dispersion will always be taken into account. New materials can be defined by the user with the material editor (see sect. 20.10).

Index (real) The real part n of the complex index of refraction, which is defined as $(n - ik)$.

Index (imag.) The imaginary part k of the complex index of refraction $(n - ik)$, also known as extinction coefficient.

OTH The optical thickness. It is the physical thickness PTH (as it would be measured by a ruler) multiplied by the refractive index of the material and divided by the reference wavelength, i.e. $OTH = n \cdot PTH/\lambda_0$. For example, 0.25 would be a quarter-wave layer, i.e. the optical path is exactly one quarter of a wave.

PTH The physical thickness as it would be measured by a ruler. The numbers in the column are always in microns.

P-Factor The P-factor describes the packing density, since materials in thin films seldom have bulk properties. Thin films usually exhibit a pronounced columnar morphology with pore-shaped voids between the columns. This reduces film packing density and in turn its optical properties. The P-factor is between 0 and 1. When P is 1, the whole void space is occupied by the material, this is equivalent to a bulk material. To model varying packing density, the refractive index of the layer is given by $n = (1-P)\left[(1-f) + f n_\nu\right] + P n_s$

Var A layer thickness can be made variable by checking the appropriate box. Variable layer thicknesses are required for coating optimization (refinement).

20.4 Composing a new Coating

New coating designs can be created using a shorthand notation on the basis of quarter-wave layers. This option requires specification of two different materials, which are represented by capital letters (symbols) such as **H**, **L**, **A**, **B**, etc. Commonly, the symbol H is used to represent a high-index material and L for a low-index material. The symbols can be combined into a formula using a sequence, such

Note the space following the exponent, which is required. If it is omitted, the formula will be rejected. Nesting of brackets is NOT permitted. Air and substrate need not necessarily be specified, as they are always automatically created.

Dialog based entry:

A dialog box is invoked from the menu *Coatings –> Compose new coating*. It allows entry of the material symbols and the corresponding materials, which are chosen from dropdown lists. Since each symbol represents an optical thickness of a quarter-wave, there is no option for thickness entry. Once the symbols have been defined the shorthand notation can be entered in the corresponding string field. In the example below, three materials are defined, which are represented by the symbols H, L and B.

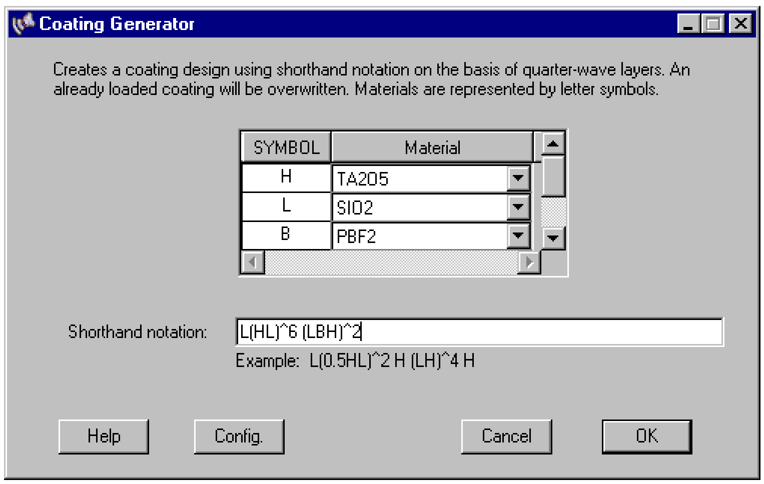


Figure 20.3: Dialog box to defining a new coating stack. Symbols (e.g. H or L) must first be assigned to materials, which can then be used in the shorthand notation, e.g. L(HL)^6 .

Command Line Entry:

`FCOMP 'formula'`	Film compose. Creates a new quarter-wave coating stack, which is described by a formula. Since the formula may contain blanks, it must be enclosed in quotation marks. **Example:** `fcomp 'L(HL)^3 B(HL)^6'`
`FSYM symbol material`	Assign a `symbol` to a `material`. For example, `FMAT H TIO2` assigns the symbol "H" to the material "TiO2". This makes the symbol "H" available to defining a coating formula using the command FCOMP (see above).

20.5 Specifying Coatings on Surfaces (Coating Attachment)

There are two methods to specifying coatings on optical surfaces:

1. Assign a coating, which is stored in a file, directly. This means specifying a coating name.
2. Load a coating into the coating editor and then view, analyse or optimize it. Once the performance is considered sufficient, attach it to a lens surface using the `ATT` command. Attach a

'default' coating (single quarter wave M_gF_2 layer) to optical surfaces by the "ATT sk—si..j DEF" command (see also comments below).

By default, air-glass surfaces are assumed uncoated. On reflecting surfaces (mirrors, see REFL) and total reflecting (TIR) surfaces 100% reflectivity will be assumed.

20.5.1 Default (Single Layer M_gF_2) Coating

In addition to user-defined coatings a 'default' coating may be assigned to optical interfaces in absence of any other information. A default coating consists of a single layer quarter wave M_gF_2 layer centered at the reference wavelength (see also section 17.1).

In the command line, a single layer (M_gF_2) coating is defined (i.e. attached to a surface) by

`ATT sk|si..j DEF`

In the surface editor, enter "DEFCOAT" in the column labelled "Coating" (see Fig. 20.4).

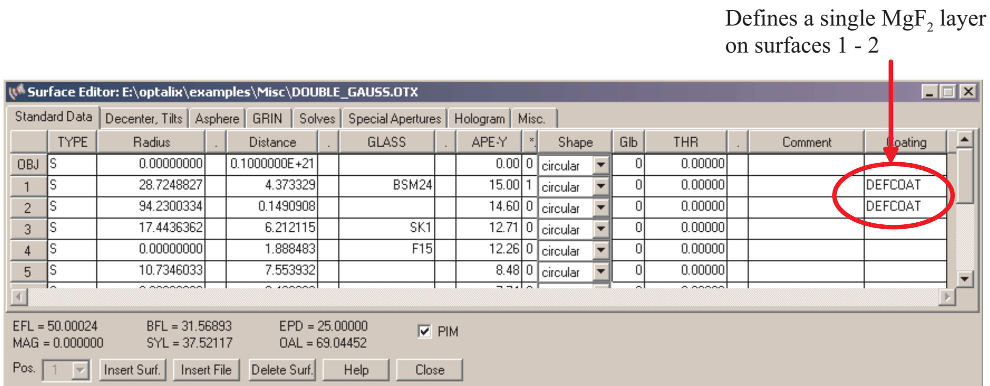


Figure 20.4: Defining 'default' coatings (i.e. single quarter wave layer M_gF_2) in the surface editor.

See also section 8.35.1 for more details.

20.6 Phase Changes introduced by Coatings

The phase change that occurs at a coating when polarization ray tracing is active (POL YES) is automatically considered in the optical path length. That is, the optical path difference introduced by the finite thickness of a coating attached to a surface is added to the optical path length (OPL). This may result in different optical path difference (OPD) and correspondingly different diffraction analysis results (MTF, PSF, etc), depending on whether coatings are attached to surfaces or not.

Phase changes that occur on coatings can normally be neglected, however, on high numerical aperture systems or wide-angle systems with steep incidence angles on optical surfaces coatings may have a noticeable effect on phase (=wavefront) response.

20.7 Coating Thickness Variation

Usually it is assumed that thicknesses of layers in thin film stack is uniform over the whole area of the lens surface. In practice, however, there may be special conditions for which this assumption is not valid. For example, steep curved surfaces are very hard to coat uniformly. Due to the deposition process the overall thickness of the coating stack at curved surfaces gets thinner in the outer zones of the lens surface. It is obvious, that the performance (reflectivity, transmissivity, phase) of the coating will be different at the surface vertex (where rays usually hit the surface at near normal incidence) compared to the rim of the lens.

The most prominent effect of coating thickness variations are seen on transmissivity and reflectivity. However, phase effects induced by variations of coating thickness may affect the overall performane of a system, e.g. in systems with strongly curved surfaces, wide angles, or diffraction limited systems.

In order to model this effect, the thickness profile of a coating can be specified by polynomial functions. Two forms are available:

- Radial thickness variation, i.e. coating thickness variation exhibits rotational symmetry,
- Non-rotational symmetry of coating thickness over surface.

Hint: Use the commands ”`POL Y`” and ”`TRA Y`” (without the quotes) to include variations of coating thicknesses in analyses, such as wavefront, PSF, MTF, etc.

20.7.1 Radial Thickness Variation

The overall coating thickness is described as a function of the *radial* coordinate on a surface by

$$s_c = a_1 + a_2 r^2 + a_3 r^4 + a_4 r^6 + a_5 r^8 \tag{20.1}$$

where s_c is the scaling factor for the nominal coating thickness and $r = \sqrt{x^2 + y^2}$ is the radial coordinate measured from the surface vertex. All layers of a given coating stack will be scaled by s_c. The scaling factor s_c is expected to be a number between 0 and 1. Negative values of s_c are not allowed, respectively are set to $s_c = 0$ in the analysis. The coefficients a_i are specified by the command

`CTV NO|RAD|XY sk|si..j ck|ci..j coeff_1 coeff_2 ...`	

	Coating thickness variation defined by either radial (RAD) or non-symmetrical (XY) polynomial. Enter the coefficients `coeff_1`, `coeff_2`, etc, as given in Eq. 20.1. Coating thickness variation is removed from a surface if all coefficients are zero.
	Examples:
	`ctv rad s3 c2 -0.002`
	`ctv rad s3 c2..5 0.01 0.02 0.03`
	`ctv xy s2..3 c4 -0.002`
	See also sect. 20.7.2 for a description of the non-symmetrical (XY) coating thickness variation.

`EDI CTV`	Edit coefficients of coating thickness variation in a spreadsheet editor.
`PLO CTV sk [style]`	Plot coating thickness variation (CTV) for a given surface `sk`. Plots can be made in various styles specified by the optional parameter `style`: `WIR` : wire-frame, `CON` : contour plot, `FAL` : false colour plot, `XY` : slices in X- and Y-direction. The default plot style is wireframe.
	continued on next page

continued from previous page	
`POL Y|N]`	Activate/deactivate polarization analysis. Turn on polarization analysis (`pol y`) if you want to analyze the effects of coating thickness variation on wavefront.

See also related commands:

`LIS MUL`	List multilayer coatings attached to optical surfaces,
PMA	Plot system pupil map (i.e. transmission in system exit pupil).
`POL Y`	Turn on polarization analysis to see CTV effects on wavefront.

Example:
We assume a decrease of the coating thickness by a radial quadratic function. The thickness of the coating stack at the rim of a lens reduces to 70% of the thickness at its vertex, i.e. the thickness scaling factor at the rim is 0.7. From Eq. 20.1 we have

$$0.7 = a_1 + a_2 r^2$$

Assuming furthermore a lens diameter of 50 mm ($r = 25mm$), we obtain

$$0.7 = a_1 + a_2 \cdot 25^2$$

Since the thickness scaling factor $s - c$ must be 1 at $r = 0$ (vertex), a_1 must be 1. Then, a_2 is calculated by

$$a_2 = \frac{s_c - 1}{r^2} = \frac{0.7 - 1}{25^2} = -0.00048$$

The commands for this example are then (assuming coating thickness variation at surface 3)

```
ctv s3 c1 1          ! a1 = 1
ctv s3 c2 -0.00048   ! a2 = -0.00048
```

20.7.2 Non-symmetrical Thickness Variation

Almost arbitrary (non-symmetrical) coating thickness variations can be modeled by a 2-dimensional polynomial of the form

$$\begin{aligned} s_c = \quad a_1 & + a_2 x + a_3 x^2 + a_4 x^3 \\ & + a_5 y + a_6 y^2 + a_7 y^3 \\ & + a_8 xy + a_9 x^2 y + a_{10} xy^2 \end{aligned} \tag{20.2}$$

where s_c is the scaling factor for the nominal coating thickness and x, y are the physical coordinates on the surface measured from the surface vertex. All layers of a given coating stack will be scaled by s_c. The coefficients a_1 to a_{10} are specified by the CTV command as given in the previous section 20.7.1, (page 376).

The coating thickness variation on specific surfaces can be plotted by the command `PLO CTV`. Set `POL Y` to see effects of coating thickness variation (CTV) on wavefront.

20.8 Accounting for the Phase in an Optical Coating

The wavefront in an optical system may be distorted by optical coatings, depending on the type of coating, the incidence angle and the wavelength. Optical coatings introduce additional phase effects in an optical system, and therefore may have a significant impact on the wavefront passing through an optical system, in particular if the thicknesses of the coating layers are not uniform over the area of deposition.

Unfortunately, there is no commonly accepted method to which surface the phase is referred to, it can be the incident surface of a coating or to the exiting surface of a coating. Depending on the definition, the geometrical thickness of a coating stack must be included or not in order to obtain a correct phase/wavefront representation.

In general, the phase introduced by an optical coating is expressed by

$$\Phi_{wf} = \Phi + \frac{2\pi n_0 \cdot d \cdot \cos(\theta_0)}{\lambda} \qquad (20.3)$$

where Φ is the phase as usually calculated by thin film codes, n_0 is the refractive index of the entrance medium, d is the total thickness of the coating, θ_0 is the angle of incidence on the coating.

$OpTaliX$ uses the convention defined in the MacLeod package where the geometrical term (i.e. the right term) in eq. 20.3 is already included in the phase result. Other thin film packages may use different definitions that must be carefully checked.

20.9 Thin Film Optimization (Refinement)

Optimization is a process for the improvement of design performance. It requires an already existing starting design. Optimization does not synthesize a coating design as it would be possible by other methods (e.g. building a system virtually from scratch by automatically adding layers, such as the so-called ”Needle” method, simulated annealing or ”Optimac”).

20.9.1 Variables

Variables are thicknesses of layers. They can be defined in the coating spreadsheet editor. If the appropriate box is checked, the layer thickness is variable during optimization, if it is unchecked, the thickness will not be changed in the optimization. See also page 373 for editing coating data.

20.9.2 Targets

Optimization (refinement) of coatings requires first of all the definition of a target performance. The actual performance is compared with the targets and the deviation of actual and required performance is expressed by the function of merit.

In coating optimization, targets are a series of reflectance or transmittance values at discrete wavelengths. Since there may be many targets required in complex designs, a dialog box supports the definition of targets. It is called from the main menu selecting *Coatings –> Targets*.

Targets are created by specifying a wavelength range and the number of wavelengths in that range. The target values in this range may be between 0 and 1, corresponding to 0% or 100% transmittance or reflectance, respectively. Targets can be referred to the S-plane, P-plane or an average value between S- and P-plane by selecting the appropriate radio buttons, as shown below:

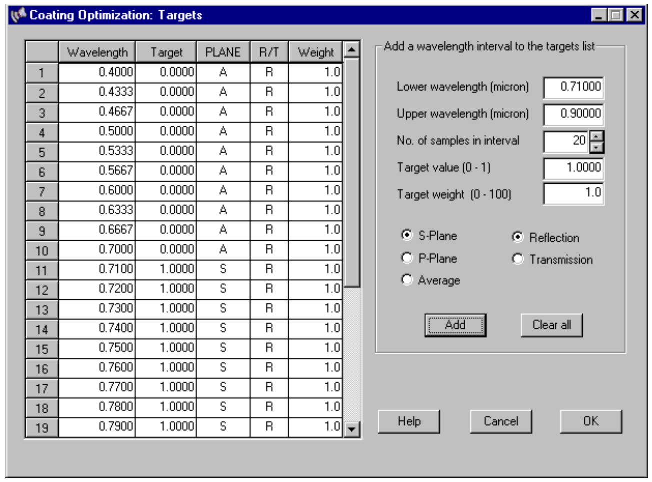
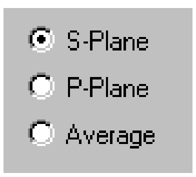
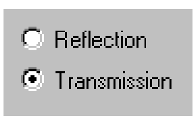


Figure 20.5: Targets dialog box.



S-plane: Targets are for S-plane (German ”senkrecht”) only,
P-plane: Targets are for P-plane (German ”parallel”) only,
Average: The arithmetic average (S+P)/2 is used.



Select whether transmittance or reflectance values shall be used.

Weights are usually set to 1, but they may be between 0 and 100. A weight 0 means, that this performance target does not contribute to the merit function. The higher a weight is, the more will the aberration (difference of actual performance from target) contribute to the merit function.

Pressing the **Add** button will create the targets. Several wavelength ranges with different targets (reflection, transmission, S- P- or average plane) can be combined to define more complex performance constraints.

Clear all: Pressing this button will clear all targets.

Deleting targets: Individual targets can be deleted by selecting a group of rows in the targets table. For example, deleting the variables (rows) numbered 2 to 3 is accomplished first by clicking onto the row label 2 (the whole row is marked), then holding the shift key and clicking onto row label 3. Rows 2 and 3 are now marked black. Pressing the **Del** button on the keyboard will delete the rows. Alternatively, **Ctrl-X** will also delete the rows and the contents of the deleted rows is additionally copied to the clipboard.

20.9.3 Run Coating Optimization

Having defined variables and performance targets, the coating can now be optimized (refined). This is accomplished in the command line by typing `FOPT` or from the main menu selecting *Coatings –> Optimize coating*.

`FOPT [n_iter]`	Thin film optimization, requires proper setting of targets and variables. The optimization stops after `n_iter` cycles, independent whether a local minimum has been reached. If `n_iter` is omitted, optimization stops at the apparent (local) minimum.

20.10 Coating Material Editor

The coating material editor manages a database of materials used in thin-films. $OpTaliX$ provides a library of predefined coating materials (which cannot be modified) and a library of private (i.e. user-defined) coating materials which can be modified (editing, adding new materials or deleting unnecessary materials).

Thin film materials are both dispersive and absorbing. This is the major distinction from "conventional" glasses used in ray tracing which are only modelled by their dispersive properties. "Conventional" glasses, like BK7, exhibit almost negligible absorption within the wavelength range for which dispersion coefficients are valid.

Unlike "conventional" glasses, thin-film materials are defined by the refractive index n **and** the extinction coefficient k (i.e. the imaginary part of the complex index of refraction) against wavelength λ (given in microns).

If necessary, the values are interpolated or extrapolated. Interpolation is linear. Extrapolation keeps the last value from the material table. A linear interpolation is used for calculating (n, k) pairs rather than dispersive formulae because of the wide range of different materials and conditions that are involved. Metals, for example, cannot be represented by the common normal dispersion formulae (such as Sellmeier or Herzberger equations) that are useful only for non-absorbing (dielectric) materials over a limited spectral region.

Private thin-film materials can be edited in the coating material editor which is invoked from the main menu by selecting *Coatings* $-->$ *Material Editor* or from the command line by

`EDI CMAT`	Edit coating (thin-film) materials. This command opens a dialog box as shown in Fig. 20.6. Each material can be defined by up to 100 (n, k) pairs. The wavelengths do not need to be equally spaced.

20.11 Coating Index Profile

Produces a plot of refractive index against thickness. In the index profile, the incident medium (typically Air) is on the left and the emergent medium, or substrate, on the right.

Refractive index profiles can be shown by real part, imaginary part or both components simultaneously.

20.12 Export Coating Performance Data

The performance of optical coatings (reflection, transmission, phase) can be exported to an Excel spreadsheet. From the command line, this is accomplished by the command

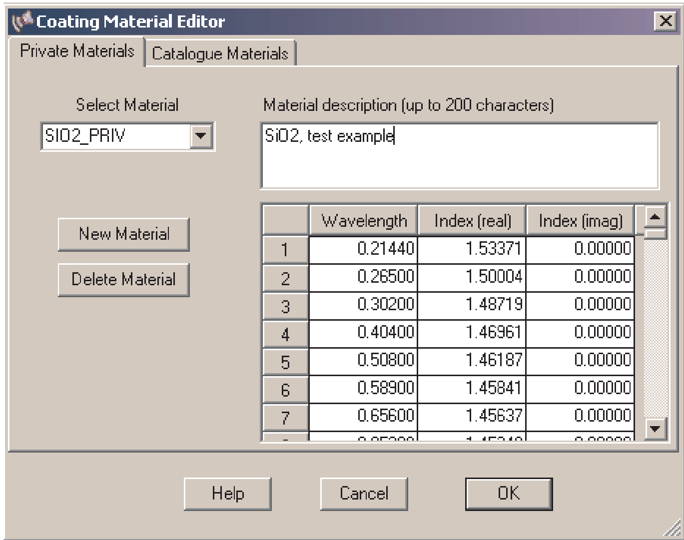


Figure 20.6: Editor for defining coating materials.

`MAN R|T XLS file_name`	Perform multilayer analysis and export the transmission/reflection/phase performance to an Excel spreadsheet. Example: `man r xls c:\temp\refl.xls`

From the menu, select *Coatings / Reflection / Numeric, as Excel file* (see Fig. 20.8):

20.13 Basic Relations

Generally, a thin film coating is a media, whose properties are constant throughout each plane perpendicular to a fixed direction and is called a *stratified medium*. The calculation scheme presented in this section follows the treatment by Macleod [32]. A similar treatment is found in Born and Wolf [4].

The electric field E and the magnetic field H at one boundary of a film are related to the fields E' and H' at the other boundary by two linear simultaneous algebraic equations, written in matrix form:

$$\begin{pmatrix} E \\ H \end{pmatrix} = M_j \cdot \begin{pmatrix} E' \\ H' \end{pmatrix} \tag{20.4}$$

where the M is the characteristic matrix for an individual layer j :

$$M_j = \begin{bmatrix} \cos(\delta_j) & -\frac{i}{p_j}\sin(\delta_j) \\ -ip_j\sin(\delta_j) & \cos(\delta_j) \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \tag{20.5}$$

For a multi-layer stack containing m layers, the calculation of reflectance, transmission and phase properties involves successive multiplication of the characteristics matrix

$$\begin{bmatrix} B \\ C \end{bmatrix} = \left\{ \prod_{j=1}^{m} \begin{bmatrix} \cos(\delta_j) & -\frac{i}{p_j}\sin(\delta_j) \\ -ip_j\sin(\delta_j) & \cos(\delta_j) \end{bmatrix} \right\} \cdot \begin{bmatrix} 1 \\ p_{sub} \end{bmatrix} \tag{20.6}$$

with

$k_0 = \frac{2\pi}{\lambda}$

$N_j = n - ik$ = complex refractive index of layer j. n is the real refractive index and k is known as the extinction coefficient. k is related to the absorption coefficient α by $\alpha = 4\pi k/\lambda$.

d_j = physical thickness of layer j

θ_j = refraction angle at boundary of layer j, given by Snell's law: $n_0 sin\theta_0 = n_j sin\theta_j$, the subscript 0 denoting the incident medium.

$\delta_j = 2\pi N_j d_j cos\theta_j/\lambda$

$i = \sqrt{-1}$

We obtain different characteristic matrices for TE- and TM-waves[1]. For a TE wave we set $p_j = N_j/cos\theta_j$. For a TM wave, the same equations hold, with p_j replaced by $q_j = N_j \cdot \cos\theta_j$. The reflection and transmission coefficients of the film are then obtained by:

$$r = \frac{m_{11}p_0 + m_{12}p_0 p_{sub} - (m_{21} + m_{22}p_{sub})}{m_{11}p_0 + m_{12}p_0 p_{sub} + (m_{21} + m_{22}p_{sub})} = \frac{p_0 B - C}{p_0 B + C} \tag{20.7}$$

$$t = \frac{2p_0}{m_{11}p_0 + m_{12}p_0 p_{sub} + (m_{21} + m_{22}p_{sub})} = \frac{2p_0}{p_0 B + C} \tag{20.8}$$

In terms of r and t, the *reflectivity* and *transmissivity* are:

$$\mathcal{R} = |r|^2 = \frac{(p_0 B - C)(p_0 B - C)^*}{(p_0 B + C)(p_0 B + C)^*} \tag{20.9}$$

$$\mathcal{T} = \frac{p_0}{p_{sub}}|t|^2 = \frac{4p_0 Real(p_{sub})}{(p_0 B + C)(p_0 B + C)^*} \tag{20.10}$$

The phase ϕ_r of r may be called the *phase change on reflection* and the phase ϕ_t of t the *phase change on transmission*. The phase change ϕ_r is referred to the first surface of discontinuity, whilst the phase change ϕ_t is referred to the plane boundary between the stratified medium and the last semi-infinite medium.

We have different phase changes for each plane of incidence (S and P) and we obtain for the phase changes on reflection and transmission:

$$\begin{aligned} \phi_r &= \phi_{r(S-plane)} - \phi_{r(P-plane)} \\ \phi_t &= \phi_{t(S-plane)} - \phi_{t(P-plane)} \end{aligned} \tag{20.11}$$

When a layer is a quarter-wave thick, particularly simple results can be obtained. A few special cases are summarized here (with n_0 = index of incident medium, n_s = index of substrate):

Single layer, zero reflectivity requires $n_1 = \sqrt{n_0 \cdot n_{sub}}$

[1]TE-wave = transverse electric wave: The electric vector is perpendicular to plane of incidence (S-plane, from German "senkrecht"). TM-wave = transverse magnetic wave: The magnetic vector is parallel to plane of incidence (P-plane, from German "parallel")

Double quarter, single minimum, zero reflectivity requires	$\frac{n_2}{n_1} = \sqrt{\frac{n_{sub}}{n_0}}$
Double quarter, double minimum, zero reflectivity requires	$n_1 \cdot n_2 = n_0 \cdot n_{sub}$
Triple Layer, Minimum reflectivity is accomplished for:	$n_1 \cdot n_3 = n_0 \cdot n_{sub}$ $n_2^2 = n_0 \cdot n_{sub}$

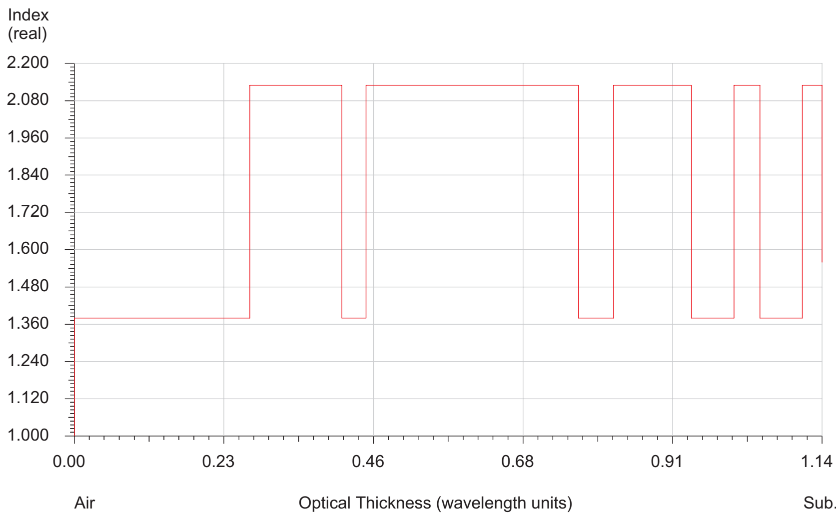


Figure 20.7: Coating Index Profile

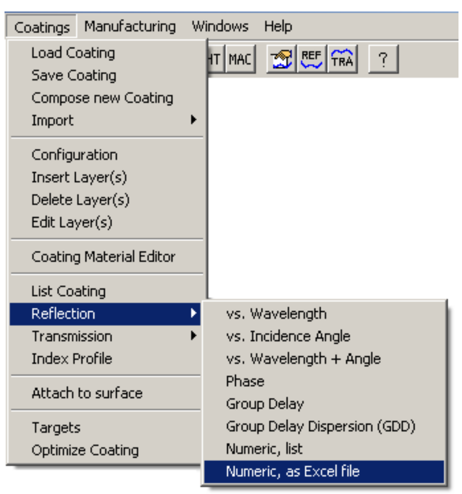


Figure 20.8: Menu for exporting coating performance to Excel.

21

Environmental Analysis

The environmental analysis takes into account the changes in lens data which result from changes in temperature and pressure. The changed system becomes the basis for all subsequent analyses, e.g. image evaluation. The changed system can be saved and also optimization can be performed to test active compensation schemes. The environmental parameters can be applied to the entire optical system or individual parts to model temperature and/or pressure gradients.

It is important to note the initial conditions for all lens data:

- The nominal temperature is 20$°C$,
- all spaces, including the object and image space, are filled with air at sea level pressure $(1013.25 \cdot 10^9$ Pa),
- the index of air is regarded to be 1.0. This is also the assumption made in glass catalogues. See also section 13.7.

These conditions need not to be entered explicitly, they are assumed as default. When temperature and/or pressure is altered, all data are converted from relative indices to absolute indices, relative to vacuum as 1.0. This conversion is automatically done and does not require user interaction. If no other environmental changes are made to the optical system (i.e. it remains at 20°C, 760 mm Hg), the same optical answers are given before and after this process. The only difference is, that indices are now referred to vacuum. For example, the command `TEM sa 20` assigns the temperature 20°C to all surfaces. This, however, is the initial default condition and the system must show the same optical performance. The surface listing (see `LIS`) then reports indices relative to vacuum. Air, for example, has an index of refraction of approximately 1.000273 in the visible spectrum. Air spaces will automatically be filled with the pre-stored ”material” `AIR` to account for the (small) dispersion of air.

21.1 Temperature

A temperature distribution can be assigned to a range of surfaces or to the entire lens system.

TEM si..j|sa temperature	Temperature at surface(s) si..j. The system data are changed immediately! Temperature gradients can be modelled by assigning different temperatures to individual surface ranges si..j. Example: TEM sa 30 ! sets temperature of all surfaces to 30°C
DEL TEM si..j|sa	Deletes temperature data for surfaces si..j or all surfaces (sa). The construction data are retained from the previous temperature state. For example, deleting temperature data on a lens at a higher temperature (say at 80°C), retains all construction data at the expanded temperature level. To restore the lens condition at room temperature (20°C), first apply the command TEM sa 20 and then delete temperature data (DEL TEM sa).
EXC si..j|sa expansion_coef , or CTE si..j|sa expansion_coef	Linear expansion coefficient for mount, glasses or surface(s). The assumed exponent is 10^{-6}.
EXM si..j|sa expansion_coef	Linear expansion coefficient for first surface mirror substrate. The assumed exponent is 10^{-6}. Values apply to the substrate for the designated surface(s) si..j or all surfaces sa.
EXR sk|si..j ref_expansion_coef	Linear expansion coefficients for globally referenced distances. See also a detailed explanation below (section 21.1.1).
DNDT si..j|wi..j dndt DNDT si..j dndt(w1) ... dndt(wn)	Enter *absolute* dn/dT coefficient explicitly, if unavailable in the glass catalogues. The assumed exponent is 10^{-6}. The second form expects data in the order system wavelengths are specified. Thus, for 3 wavelengths defined, 3 dndt-values must be entered. The dndt-values must correspond to the system wavelengths. If there are more wavelengths defined than dndt-values entered in the second command form, dn/dT = 0 is assumed for the remaining wavelengths. Example 1: dndt s3 w1..5 -1.5 Example 2: dndt s3 1.5 2.5 3.5 See also querying DNDT as a LDI item (sect. 27, page 459. Further information on absolute and relative dndT is given in sect. 13.2, page 216.

Changing the temperature causes all glass elements to expand or contract according to the expansion coefficient (EXC). Radii of curvature, axial thicknesses, aperture radii and aspheric coefficients change according to

$$L(T + \Delta T) = (1 + \alpha \cdot \Delta T) \cdot L_0 \qquad (21.1)$$

where L is a length at the changed temperature, T is the base temperature, ΔT is the change in temperature and α is the linear expansion coefficient.

All air spaces are changed by computing the change in the corresponding ***axial*** thicknesses and adding the thickness change to the axial separation. In the case of strongly bent surfaces the length of spacers may significantly differ from the axial air space. In this case the correct spacer expansion must me modeled by auxiliary surfaces with appropriate CTE assignments.

For surfaces, which are globally referenced to a preceding surface, the reference thickness (`THR`) is changed according to the linear expansion coefficient `EXR` of the reference surface (see also section 21.1.1).

The expansion coefficient of the mount materials must always be explicitly entered using the `EXC` command.

The linear expansion coefficient of front surface mirrors must be explicitly entered by the `EXM` command.

Refractive indices change with the corresponding dn/dT-coefficient of the glasses. The dn/dT-coefficient is unique for each glass/material and is taken from the glass catalogues, if available. If not available, it is set to zero or it may be explicitly entered using the `DNDT` command.

21.1.1 Expansion Coefficients on Global References

In order to fulfil certain requirements on thermal behaviour of an optical system, for example athermalization, it is sometimes required to apply special mounting techniques where single lenses or groups of lenses are mounted in separated housings. Quite often, housing materials with abnormal thermal expansion coefficients are used to maintain focus without any powered drive mechanism (passive athermalization).

When temperature changes, lenses (or lens groups) may move relative to another surface, typically a surface other than the immediately preceding one. The effect is that the change of the air space between two lenses is not dictated by the thermal expansion of the housing material, but follows a more complex relation.

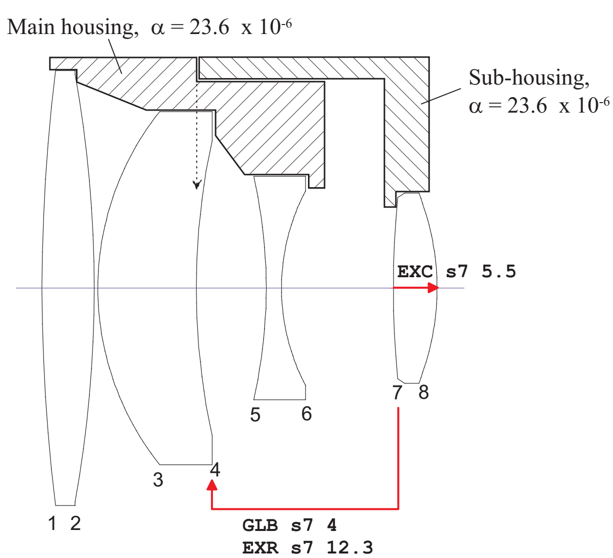


Figure 21.1: Modelling thermal expansion with globally referenced surfaces.

Fig. 21.1 indicates a simple optical system, where the last lens (surfaces 7-8) is mounted in a separate housing being attached to a flange on the main housing close to surface 4. If the main housing and the sub-housing for lens 4 are made of different materials, the air space between the third and fourth lens will change according to the expansion difference of the two materials involved.

In order to adequately model this optical-mechanical configuration, surface 7 is globally referenced to surface 4. See also section 8.21 (page 114) for a general description of global references.

Instead of specifying the expansion coefficient of the air space between surfaces 6 and 7, we directly specify the expansion coefficient for the reference length surface 7 to surface 4. This is `EXR`, which always refers to a surface *before* the current surface. In other words, `EXR` is the linear expansion coefficient of a reference thickness (THR).

21.2 Pressure

A pressure profile may be assigned to a range of surfaces or to the entire lens system. Inhomogeneous pressure profiles in axial direction may be accomplished by assigning different pressures to different surface ranges.

`PRE si..j|sa pressure`	Pressure in mm Hg at surface(s) `si..j` or all surfaces (`sa`). Example: `PRE sa 760`, sets the pressure to 760 mmHg (normal pressure).
`DEL PRE si..j|sa`	Deletes pressure data for surfaces `si..j` or all surfaces (`sa`).

22

Tolerancing

The goal of any tolerancing scheme is to determine the dimensional ranges of optical components that meets performance requirements. Tolerances are variations in design data related to fabrication considerations. Careful tolerancing is important for the designer to ensure that the performance will be maintained in the finished units. The various tolerances may be used in any combination to evaluate the impact of fabrication errors. The tolerance perturbations for system prescription data are always taken from the currently assigned values. Tolerances are automatically saved with the lens file.

The two most common effects in tolerancing an optical system are underspecification, that is incompletely describing of what is required, and overspecification, wherein much more severe tolerances are established than required. Thus, defining tolerances is a complicated process between the limits imposed by

a) the performance requirements of the optical system, and

b) the expenditure of money and time which is justified by the application.

As a guideline, tolerances should be established as large as the requirement for satisfactory performance of the optical system will permit. The tolerancing calculations available in $OpTaliX$ are divided into three separate categories:

- Sensitivity analysis
- Inverse tolerancing
- Monte Carlo analysis

All of these categories require the definition of *tolerance items* (section22.1, page 389) and *tolerance criteria* (section 22.2, page 396, which are described in the following two sections.

22.1 Surface Tolerance Items

Tolerance items assigned to surfaces can be edited by the command `EDI TOL`, which invokes a dialog box, or they may be directly specified in the command line as described below. A detailed definition of each tolerance item is given in the table below and in the following sections.

`EDI TOL`	Opens a dialog box for editing surface tolerances.
	continued on next page

continued from previous page	
DEL TOL [si..j]	Delete all types of existing tolerances on designated surfaces si..j. Example: del tol s1..3 ! Delete tolerances at surface 1 to 3. del tol sa ! Delete tolerances at ALL surfaces.
DLF si..j tol_testplate_fit	Tolerance on test-plate fit (in fringes at $\lambda = 546nm$) over the clear aperture. See also section 22.1.3 for more information. In ISO 10110 notation, DLF corresponds directly to the A-value, e.g. 3/ A (B,C) is synonymous to 3/ DLF (B,C). Note that sensitivity on test-plate fit may also be included in optimization using the TSF function (see page 353). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.
IRR si..j total_irregularity	Tolerance on cylindrical irregularity, in fringes at $\lambda = 546nm$. The irregularity of a spherical surface is a measure of its departure from sphericity. See sect. 22.1.4 for more details. Note that sensitivity on surface irregularity may also be included in optimization using the TSI function (see page 353). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.
SYM si..j symmetrical_irregularity	Tolerance on symmetrical aspherical irregularity, in fringes at $\lambda = 546nm$. In ISO 10110 notation, SYM corresponds directly to the C-value, e.g. 3/ A (B,C) is synonymous to 3/ A (B,SYM).
DLT si..j tol_thickness	Tolerance on axial thickness, in mm. Shows the effect of a change in the axial thickness between surfaces. Thickness tolerances applied to a surface will also move subsequent surfaces, except the subsequent surface(s) is/are globally referenced to any other preceding surface. See also sections 22.1.6 and 22.1.7 for more information. Note that sensitivity on surface irregularity may also be included in optimization using the TST function (see page 353). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.
DTR si..j tol_ref_thickness	Tolerance on (global) reference thickness (see THR), in mm. Shows the effect of a change in the reference thickness. This option is only applicable for surfaces, which are globally referenced to a preceding surface. See also sections 22.1.6 and 22.1.7 for more information.
	continued on next page

continued from previous page

`DLN si..j tol_index`	Tolerance on index of refraction, at the reference wavelength. The tolerance value `tol_index` is specified as absolute difference to the nominal index. Example: `dln s3 0.001` ! increases index of refraction by 0.001 Note that sensitivity on surface irregularity may also be included in optimization using the TSN function (see page 353). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.
`DLV si..j tol_V_number`	Tolerance on dispersion. The tolerance value is specified as a fraction of the nominal Abbe number ν_d. Example: `dlv s3 0.008` ! changes the Abbe number by 0.8% Note that sensitivity on dispersion may also be included in optimization using the TSV function (see page 353). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.
`DLR si..j tol_radius`	Tolerance on *absolute* radius, in mm.
`HOM si..j tol_homogeneity`	Tolerance on index homogeneity, in $10^{-6} units$. See also section 22.1.9, page 395 for details. Note that sensitivity on index homogeneity may also be included in optimization using the TSH function (see page 354). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.
`AXG si..j tol_axial_grin`	Tolerance on axial linear index gradient
`RAG si..j tol_radial_grin`	Tolerance on radial quadratic index gradient
`DLX si..j tol_x_decenter`	Tolerance on lateral displacement in X-direction, in mm. Note that sensitivity on X-displacement may also be included in optimization using the TSX function (see page 354). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.
`DLY si..j tol_y_decenter`	Tolerance on lateral displacement in Y-direction, in mm. Note that sensitivity on Y-displacement may also be included in optimization using the TSY function (see page 354). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.
`DLZ si..j tol_z_decenter`	Tolerance on longitudinal displacement in Z-direction, in mm. Note that `DLZ` is equivalent to a thickness tolerance. Also note that sensitivity on Z-displacement may be included in optimization using the TSZ function (see page 354). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.

continued on next page

continued from previous page	
`DLA si..j tol_a_tilt`	Tolerance on tilt about X-axis (α-tilt), in arcmin. Note that sensitivity on tilt about X-axis may also be included in optimization using the TSA function (see page 354). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.
`DLB si..j tol_b_tilt`	Tolerance on tilt about Y-axis (β-tilt), in arcmin. Note that sensitivity on tilt about Y-axis may also be included in optimization using the TSB function (see page 354). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.
`DLG si..j tol_c_tilt`	Tolerance on tilt about Z-axis (γ-tilt), in arcmin. Note that sensitivity on tilt about Z-axis may also be included in optimization using the TSG function (see page 354). This option allows minimization of tolerance sensitivity on this parameter, whenever possible.

22.1.1 Tolerance Editor

Editing of surface tolerance items, tolerance criteria and compensators is accomplished from the menu Edit –> Tolerances or by clicking on the TOL button in the toolbar. A dialog box as shown in Fig. 22.1 is invoked.

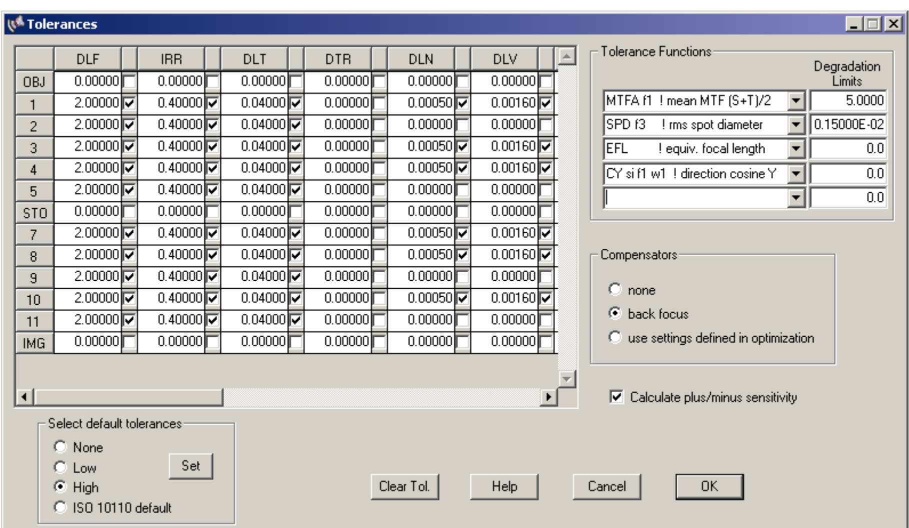


Figure 22.1: Spreadsheet for editing surface tolerance items and tolerance criteria.

Surface tolerances are entered in rows (surfaces) and columns (tolerance type). Each tolerance must be made active in the check box right to each tolerance field. If the field is unchecked, it is not used in subsequent tolerance analyses.

Default tolerances in various grades may be assigned to surfaces (see section22.1.2).

Up to eight performance criteria may be arbitrarily selected from the pull down menus. The example in Fig. 22.1 shows four performance criteria, which will be evaluated depending on surface or component tolerances:

`MTFA f1`	! mean MTF at field 1
`SPD f1`	! rms spot diameter at field 1
`EFL`	! equivalent focal length
`CY si f1 w1`	! direction cosine Y at the image plane, field number 1 and wavelength number 1. This function gives a good measure of boresight stability.

The default setting for compensators is none.

22.1.2 Default Tolerances

Default tolerances may be assigned to certain construction items. These tolerance values are taken from the ISO 10110-5 standard. Two other grades on tolerances are provided, ”low” and ”high”, which are intended for ”low”-performance and ”high”-performance systems respectively.

It is important to note, however, that these default tolerances may not be appropriate for your particular optical performance requirements. Therefore, the defaults should be considered as convenient starting points for examining the relative sensitivities of the various lens parameters. It is up to the user to deviate from the defaults and change the tolerances correspondingly.

22.1.3 Tolerance on Test-Plate Fit (DLF)

Shows the effect of a change in the radius of curvature of a surface. The perturbation is specified in terms of interference fringes[1] relative to test plate or interferometer fit at the *reference wavelength* used in the optical system. As default, ISO 10110-5 specifies $0.54607\mu m$ (e-line). If the reference wavelength differs from $0.54607\mu m$, the tolerance specification may be converted to another wavelength by

$$DLF_{\lambda 2} = DLF_{\lambda 1} \cdot \frac{\lambda_1}{\lambda_2} \tag{22.1}$$

where $DLF_{\lambda 1}$ and $DLF_{\lambda 2}$ are the numbers of fringe spacings at λ_1 and λ_2, respectively.

The number of fringe spacings corresponding to a dimensional radius tolerance, provided the radius change is small, is given by

$$DLF = \frac{2\Delta R}{\lambda}\left[1 - \sqrt{1 - \left(\frac{D}{2R}\right)^2}\right] \tag{22.2}$$

If the ratio D/R is small, Eq. 22.2 may be approximated by

$$DLF = \left[\frac{D}{2R}\right]^2 \frac{\Delta R}{\lambda} \tag{22.3}$$

Note that in ISO 10110-5 notation, DLF corresponds directly to the A-value, e.g. `3/ A (B,C)` is synonymous to `3/ DLF (B,C)`. More generally, `3/ A (B,C)` is equivalent in $OpTaliX$ to `3/ DLF (IRR,SYM)`.

[1] Due to the double pass of test plate or interferometer tests, fringes give twice the surface error measured in waves.

22.1.4 Tolerance on Irregular Surface Deviation (IRR)

Tolerance on cylindrical irregularity, in fringes at $\lambda = 546nm$. The irregularity of a spherical surface is a measure of its departure from sphericity, that is a difference in the radii of curvature between the X/Z and Y/Z meridians. The irregularity is applied by increasing the value of the X/Z radius by $\Delta R/2$ and by decreasing the value of the Y/Z radius by $\Delta R/2$.

In ISO 10110 notation, IRR corresponds directly to the B-value, e.g. `3/ A (B,C)` is synonymous to `3/ A (IRR,C)`.

In statistical tolerance simulations (TOL STAT command, see also sect. 22.7), the orientation (azimuth) of the cylindrical deformation is assumed always along the local Y-coordinate axis.

22.1.5 Tolerance on Symmetrical Aspherical Surface Deviation (SYM)

The SYM tolerance specifies the rotationally symmetrical (aspherical) surface irregularity according to the ISO 10110-5 norm. As such, the SYM tolerance is directly comparable to the C-value in ISO 10110-5. More generally, `3/ A (B,C)` is equivalent in $OpTaliX$ to `3/ DLF (IRR,SYM)`.

In $OpTaliX$, SYM is modeled by a Zernike deformation using coefficient 9 (spherical and focus, 3^{rd} order) to generate a surface deformation of SYM fringes. Example: SYM 1.0 (fringes) corresponds to a PV surface deformation of 0.000273 mm at the reference wavelength 546nm. A representation of this error form is given in Fig. 22.2.

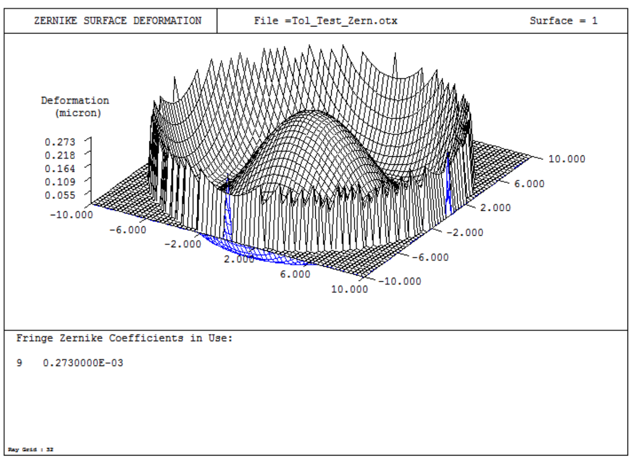


Figure 22.2: Symmetrical (aspherical) surface deformation representing the ISO 10110 C-value.

22.1.6 Tolerance on axial Thickness (DLT)

Axial thickness tolerances (DLT) change both, thicknesses of lens elements and of air spaces between lenses. The way DLT-tolerances affect the optical system depends on how subsequent surfaces are referenced. Fig. 22.3 shows the effects of DLT for two cases:

a) All surfaces are sequentially referenced, that is the position of a surface is defined with respect to its immediately preceding surface. A thickness tolerance of the first surface (DLT s1) will move the absolute position of all subsequent surfaces.

b) Surface 3 is globally referenced to surface 1. A thickness tolerance of the first surface does not change the absolute position of subsequent surfaces (here surfaces 3 and 4) and surface 2 now moves into the air space between the first and second lens.

Thus, in order to apply tolerance changes to the absolute position of surface 3, a DTR-tolerance must be assigned to this surface.

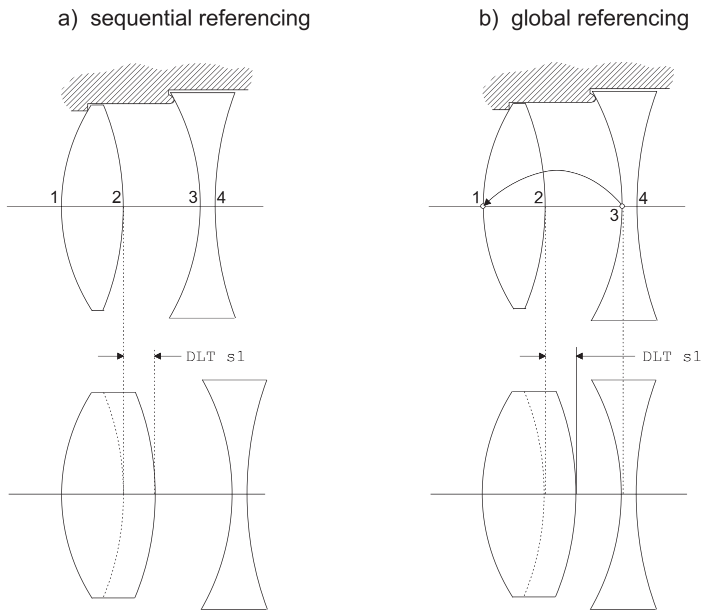


Figure 22.3: Axial thickness tolerance for different types of surface referencing.

22.1.7 Tolerance on global Thickness (DTR)

A DTR-tolerance changes the axial position of a surface, which is referenced to a preceding surface. This must not be confused with a DLT-tolerance at the same surface. As for the nominal value THR, which defines the separation *before* the surface vertex to the referenced surface, the DTR-tolerance changes the nominal THR value.

Since a surface may be globally referenced to another surface, which itself is globally referenced (i.e. a chain of global references), complex housings and interdependencies can be simulated. Referring to Fig. 22.3b, we see that surfaces 1 and 3 are directly attached to the housing. Since tolerances on mechanical distances are generally different from tolerances on lens thicknesses, also DLT and DTR tolerances will be different.

22.1.8 Tolerance on Surface Tilt (DLA, DLB, DLG)

Tolerances on surface tilts are expressed by DLA, DLB, DLG, representing the tilt around the x-axis, y-axis and z-axis, respectively. The tilt tolerances are defined in minutes of arc (arcmin). This unit has been chosen to directly relate to typical drawing specifications about tilt and lens wedge.

22.1.9 Tolerance on Homogeneity (HOM)

Homogeneity of refractive index (HOM) is modelled in $OpTaliX$ by a radially symmetric gradient, which cannot be completely cancelled by a focus compensator. The radial GRIN model used is

$$n = n_0 + c_t r^2 \tag{22.4}$$

where n_0 is the base (vertex) index of the glass, r is the radial distance from the optical axis and $\not{c}$ is calculated from the specified index tolerance $\Delta n = n - n_0$. Note that Δn must be specified in 10^{-6} units.

22.2 Tolerance/Performance Criteria

Once reasonable tolerances are entered, *tolerance criteria* are established to allow a sensitivity tolerance analysis based on any quality measure available in $OpTaliX$. Tolerance criteria are measures of system performance, whose sensitivities to changes in the construction parameters we wish to study. Thus, a tolerance function may be any arbitrary performance measure such as rms-spot diameter, MTF, Strehl ratio or boresight, to name a few. Anything that can be computed as an performance measure and that can be addressed in the optimization can also be used as a criterium in tolerance analysis. An overview of available performance functions is found in section 19.6, page 350. This approach provides the capability to ”tolerance on anything”.

`TOLC fcn_no fcn_string`	Tolerance criterium, i.e. the performance measure to be used in sensitivity analysis. Up to 5 tolerance criteria can be simultaneously defined for sensitivity analysis. `fcn_no` is the number of the function (criterium), which must be between 1 and 5. Tolerance criteria may also be edited in a dialog box, which is invoked by `EDI TOL`. Since tolerance criteria usually contain blank characters, `fcn_string` must be enclosed in apostrophes if entered in the command line. Examples: `tolc 2 'spd f3'` ! Defines rms spot diameter at field 3 as tolerance criterium. It is stored as 2^{nd} function. `tolc 3 'mtfa f3'` ! Defines average (mean) MTF at field 3 as tolerance criterium. Note, that MTF is always given in %, ranging between 0 and 100.
`TOCL fcn_no limit`	Limit on tolerance criterium, to be used in inverse sensitivity analysis. `fcn_no` is the number of the function (criterion), which must be between 1 and 5. Example: `tocl 3 5` ! In the second example of the `TOLC` command (`tof 3 'mtfa f3'`) a degradation limit of 5% is defined for mean MTF at field 3. Note that MTF is always specified in %.

22.3 Tolerance Compensators

Compensators are variable construction parameters that are changed after a tolerance has been applied. The most common compensator is the back focus to keeping the image plane always at best focus, but also any other parameter may be used to adjust for arbitrary performance measures.

The introduction of compensators prior to calculating tolerances is an important means for reducing tolerance sensitivity of an optical system. There are two basic compensation methods:

a) Adjusting the back focus only,

b) defining a complete optimization set, which may have multiple compensating variables.

Tolerance compensators can be specified by the command

`TOCM NO|BF|OPT`	Tolerance compensator method. `NO` disables compensator, `BF` uses back focal length as compensator (see section 22.3.1), `OPT` uses settings in optimization as compensator (see section 22.3.2).

22.3.1 Back Focus Compensator

Adjustment of the back focus is performed by the autofocus module. By default, minimum rms-spot size at all fields and wavelengths is used for finding the optimal focus. If focus adjustment for selectable fields, wavelengths or other performance criteria is desired, optimization shall be used as compensating module (see below).

22.3.2 Compensation using Optimization

Arbitrary construction parameter and target (performance) criteria may be selected when tolerance compensation is performed via the optimization module. This requires proper setting of variables and performance criteria. The optimization settings may be identical to the settings used for optimization of the system. Compensators are designated by optimization variables (i.e. thicknesses, radii of curvature, etc). However, it is preferable to setup special optimization settings, since generally only a few parameters (for example air spaces) will be used for tolerance compensation.

Before using the tolerancing routines, make sure that the current optimization variables correspond to those system parameters that you wish to use as compensators. See section 19, page 339 for defining optimization variables and performance functions (criteria).

Using optimization is much more powerful than simply adjusting the back focus, as any construction parameter, which can be edited, can be used as a compensator. There is also no limit in the number of compensator variables. Typical compensator variables used in tolerancing are air spaces and lens/group tilts or decentrations.

The functions (performance criteria) defined and used in the optimization module are completely independent from the tolerance criteria (section 22.2). Thus, it is possible to compensate (optimize) on wavefront and analyse tolerance sensitivity on MTF.

22.4 Sensitivity Analysis

This analysis provides information about the direct sensitivity of an optical system to fabrication and mounting errors. Each parameter is changed by its tolerance, and the changes in the requested performance measures are computed.

`TOL SEN`	Performs a sensitivity analysis based on surface tolerance items and tolerance criteria, both defined under `EDI TOL`

The variation of most performance measures is, in general, approximately quadratic with respect to changes of lens (construction) parameters. To model this variation, sensitivity is calculated for plus and minus tolerances and a quadratic function F as given in Eq. 22.5 is then calculated.

$$F = A \cdot T^2 + B \cdot T + C \qquad (22.5)$$

For each individual pair of tolerance and performance criterion a quadratic equation is calculated. For example, 5 types of tolerances at 10 surfaces and three tolerance/performance criteria will already create 5 x 10 x 3 = 150 quadratic functions.

Once surface tolerance items (section 22.1) and tolerance criteria (section 22.2) are established, a sensitivity analysis can be run. As an example, we use the Cooke triplet from the examples library `\optix\examples\misc\cooke.otx`. For the sake of simplicity, we only define tolerances on test-plate-fit, irregularity, axial thickness and x-decenter at the first three surfaces. The axial shift of the focal surface (back focus) is used as compensator. It is worthwhile to remember that back focus adjustment uses the autofocus module, which - by default - optimizes for minimum spot size over the entire field. This may or may not be appropriate for a specific application. Other compensators may be defined in the optimization settings (see sections22.3.2 and 19). We will also define three tolerance criteria, the on-axis MTF and the tangential and sagittal MTF separately at field number 2, which is at 70% of the maximum field. These are the system performance measures, whose sensitivities to changes in the construction parameters we wish to study.

```
TOLERANCE DATA :
   DLF  s1     3.0000
   IRR  s1     2.0000
   DLT  s1    0.10000
   DLX  s1    0.50000E-01
   DLF  s2     3.0000
   IRR  s2     2.0000
   DLT  s2    0.10000
   DLX  s2    0.50000E-01
   DLF  s3     3.0000
   IRR  s3     2.0000
   DLT  s3    0.10000
   DLX  s3    0.50000E-01

Compensator:  back focus.

Tolerance Criteria:
   MTFA f1  ! mean MTF (S+T)/2
   MTFT f2  ! tangential MTF
   MTFS f2  ! sagittal MTF
```

The sensitivity analysis is started with the command "SEN" or by selecting *Manufacturing –> Tolerances –> Sensitivity analysis* from the main menu.

```
TOLERANCE SENSITIVITY ANALYSIS

Compensator:  back focus (BFL)

                         MTFA f1     MTFT f2     MTFS f2   BFL-Change
Nominal value(s)        91.62532    47.32400    35.41631

Sur  Tol. (fringes)
  1  DLF  3.0000 (+)     0.19083    -1.31205     1.72375     0.00244
                 (-)     0.38478    -0.43768     1.98080     0.01047
  2  DLF  3.0000 (+)     0.37379     0.42367     1.40129     0.01046
                 (-)     0.13386    -1.94613     1.81060     0.00080
  3  DLF  3.0000 (+)    -0.05128    -1.11433     0.85838    -0.01033
                 (-)     0.51189    -0.77738     2.93915     0.02351

             RSS         0.77896     2.77957     4.63812

Sur  Tol. (fringes)
  1  IRR  2.0000 (+)    -0.14500    -3.56583    -3.22650     0.02389
                 (-)    -0.12578     1.52675     7.49339    -0.00928
  2  IRR  2.0000 (+)    -0.14084     1.75238     7.72477    -0.00927
                 (-)    -0.19613    -3.71870    -3.59981     0.02318
  3  IRR  2.0000 (+)    -1.64383    -6.54326    -7.60950     0.04369
                 (-)    -1.76839     4.15559    12.38967    -0.02792
```

```
              RSS         2.43403     9.59317    18.72432

 Sur  Tol. (mm)
   1  DLT  0.1000 (+)     0.33377    -1.69334     2.79460     0.01194
                  (-)     0.24226    -0.07476     0.85278     0.00087
   2  DLT  0.1000 (+)     0.48737     4.42504     2.10420     0.03149
                  (-)    -0.58395    -5.42251     0.79647    -0.02039
   3  DLT  0.1000 (+)    -0.56635    -7.47778     2.78950    -0.01111
                  (-)     0.52615     6.20604    -0.76605     0.01977

              RSS         1.16026    12.09503     4.68692

 Sur  Tol. (mm)
   1  DLX  0.0500 (+)     0.03605    -1.48804     1.98951     0.00730
                  (-)     0.03597    -1.48800     1.98886     0.00729
   2  DLX  0.0500 (+)     0.29881    -0.98564     1.93998     0.00706
                  (-)     0.29878    -0.98563     1.93977     0.00706
   3  DLX  0.0500 (+)    -3.09337    -2.01578    -1.17260     0.01254
                  (-)    -3.09320    -2.01357    -1.17472     0.01255

              RSS         4.39522     3.80646     4.26555

        Total RSS         5.21493    16.14106    20.30455
```

At the top of the sensitivity table (sometimes called change table) are the nominal values of the tolerance criteria, that is the performances of the undisturbed system. The output is grouped in the different types of tolerances (e.g. test-plate-fit, irregularity, etc) and within each group tabulated according surface numbers. Each column lists the *changes* in MTF for each tolerance item.

The changes in the back focus compensation are listed in the rightmost column under the label ”`BFL-Change`”. If more than one tolerance criterion is defined, the maximum value of back focus compensation is printed. The RSS values given for each column and each tolerance group is a ”statistical sum” of the performance perturbations ΔF and is defined as

$$RSS = \sqrt{\Delta F^2} \tag{22.6}$$

Tolerance sensitivities are usually given for plus and minus tolerances respectively. This is indicated by `(+)` and `(-)` in the sensitivity table.

22.5 Tolerance Sensitivity in Optimization

Typically, an optical designer needs to find the optimal compromise between optical performance, costs, volume constraints and manufacturing aspects. In particular, the latter requirement asks for an optical system that is insensitive to manufacturing tolerances to a maximum extent.

That is, optimizing for maximum (optical) performance alone will most likely not yield a design that fulfills all requirements mentioned above. Furthermore, considering the sequence of a typical design process, we have concept design, optimization, tolerancing and then, if needed, several re-iterations to achieving a design that can be economically manufactured.

This is a tedious process. $OpTaliX$ helps you in that it allows integration of tolerancing issues already during the optimization process. This means that you can specify certain surfaces (or all surfaces) whose sensitivity to alignment errors or manufacturing errors in general are to be minimized. Thus, in other words, $OpTaliX$ can simultaneously optimizes for both optimum image performance and minimum tolerance sensitivity.

See the commands TSF, TST, TSI, TSN, TSV, TSX, TSY, TSZ, TSA, TSB, TSG for defining tolerance sensitivity functions in optimization (sect. 19.10 and pages 353 to 354).

Tolerance sensitivity is calculated on the basis of wavefront aberration (WAV) for a given tolerance item. A basic introduction to the method used in $OpTaliX$ is given by Grey [16], and practical examples are given by Isshiki et.al, [23].

22.6 Inverse Tolerancing

Inverse tolerance analysis starts from a predefined change in system performance and determines the tolerance limit for each construction parameter. This analysis is based on the functional relationship between tolerances and performance measures, which is obtained during sensitivity analysis from the quadratic functions in Eq. 22.5. Then, using this data, the allowed tolerances for specified changes in performance (the tolerance criteria) are computed.

TOL INV	Performs an inverse tolerance based on tolerance criteria (TOLC) and limits on tolerance criteria (TOCL) , both defined under EDI TOL

22.7 Monte Carlo Analysis

The Monte Carlo tolerancing is a statistical approach to simulate production yields on the basis of predetermined surface/component tolerances. It allows prediction of freely definable performance metrics on the basis of statistical (random) perturbations of construction parameters within limits defined by the individual surface/element tolerances.

A successful statistical tolerance (yield) analysis is performed in several steps:

- Define the surface/component tolerances, e.g. in the tolerance editor (22.1.1), tab ”Tolerances”.
- Define the performance parameters you want to analyze, e.g. in the tolerance editor (22.2), tab ”Tolerance Functions”.
- Define the statistical distributions of tolerances and the statistical population (number of individual objectives), e.g. in the tolerance editor (22.7.1), tab ”Statistics”.
- Run the statistical analysis, e. g. by the command TOL STAT or from the tolerance dialog, tab ”Analyses”.

A central dialog allows defining and editing of all tolerance parameters and function. It is invoked by the command EDI TOL, or from the main menu ”Edit” -¿ ”Tolerance Editor”.

22.7.1 Statistical Parameters and Distributions

The parameter variation within a given tolerance can be differently distributed. Currently, three distribution forms are possible:

- Even distribution
- Gaussian distribution
- Beta distribution

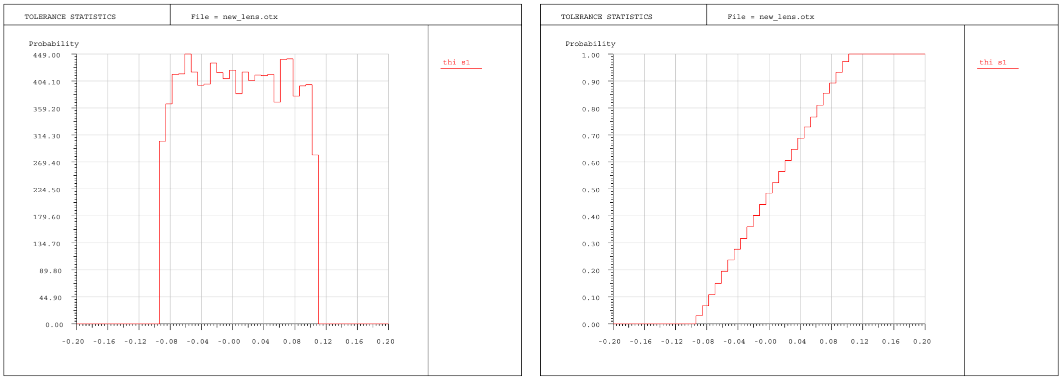


Figure 22.4: Even distribution with pseudo-random numbers. Left: linear plot, right: cumulative plot.

22.7.1.1 Even Distribution

This distribution form assumes that all parameter perturbations are evenly distributed within a given maximum tolerance. A graphical illustration is given in Fig. 22.4 below.

22.7.1.2 Gaussian Distribution

The Gaussian distribution is the most common form in statistical tolerancing. The statistical perturbations are based on normally (Gaussian) distributed pseudo-random numbers with zero mean. In $OpTaliX$ tolerancing, the Gaussian distribution accepts one parameter σ which denotes the standard deviation. For 1σ about 68% of all perturbations lie within the tolerance band defined, 2σ will include 95.4%, and 3σ include 99.7% of all tolerance perturbations.

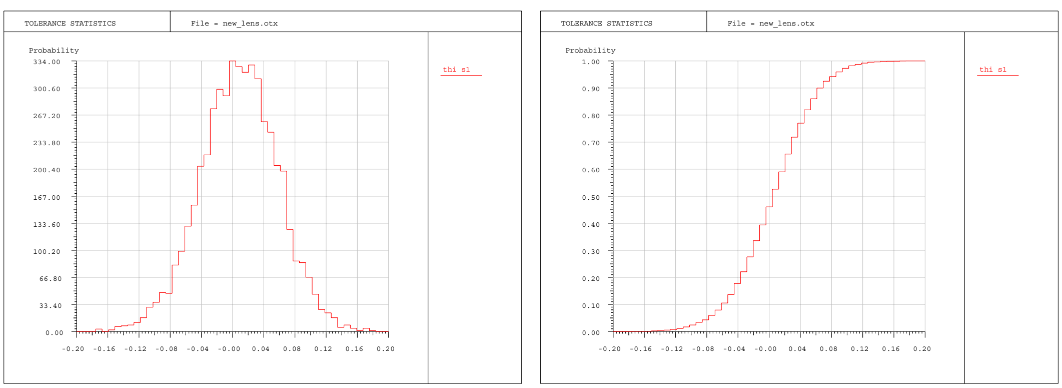


Figure 22.5: Gaussian distribution with 2σ variance on a $\pm 0.1mm$ tolerance. Left: linear plot, right: cumulative plot.

22.7.1.3 Beta Distribution

The Beta distribution is a special continuous probability distribution that allows simulation of special non-symmetrical distributions. It is currently implemented with fixed parameters ($\alpha = 2$, $\beta = 5$) that result in the distribution form given in Fig. 22.6.

This distribution is well suited to modeling fabrication specific effects. For example, polishing/grinding of lenses is typically stopped when the thickness of a specific lens is within a defined tolerance. Because this process always starts from a thicker blank and removal of material reduces the axial thickness, there is a tendency that axial thicknesses of lenses towards the upper tolerance interval, hence, the non-symmetrical thickness distribution.

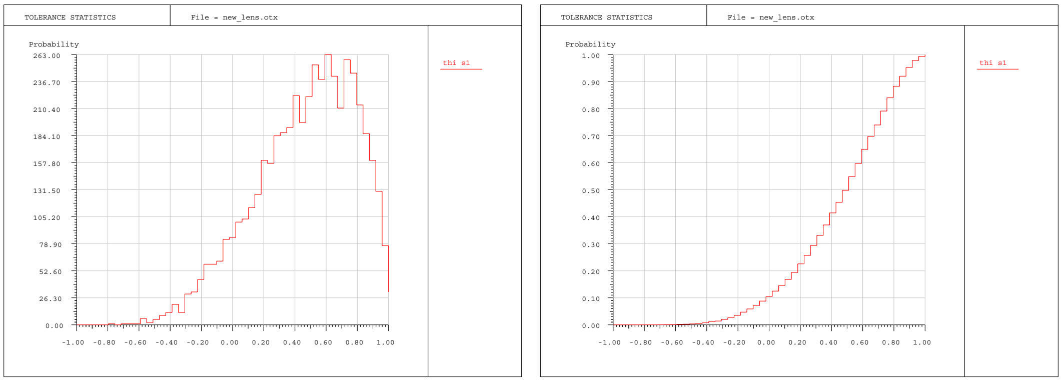


Figure 22.6: Beta distribution. Left: linear plot, right: cumulative plot.

23

Manufacturing Support

23.1 Footprint Analysis

The footprint option plots the boundaries of the light beams going through the optical system on a specified surface. This is done by calculating the intersection of the beam with the surfaces of interest. In case of curved surfaces, the beam intersections are plotted parallel to the local Z-axis onto the vertex tangent plane. All wavelength, activated fields and zoom positions are represented and the resulting plot is a composite of the used area of the surface. Vignetting is always taken into account. Note that rays are only vignetted if a fixed aperture (see FHY command, page 163) has been assigned to the designated surfaces. Internal obscurations are not taken into account in footprint analysis. They are, however, considered in the ray intersection analysis (page 14.1.8), which is equivalent to footprint analysis, where a ray grid is traced to the designated surface.

`FOO [sk | fi..j | plot_extent|NUM|? ]`	Plot the footprint on surface `sk` for fields `fi..j`. For 'zoomed' (multi-configuration) systems, the currently selected zoom position is used (see POS command). The parameter `plot_extent` is optional and defines the maximum displayed area. Absence of `plot_extent` or a zero value invokes automatic determination of the plot area on `sk`, respectively uses the previously entered value of `plot_extent`. The optional parameter NUM outputs additional data, such as enclosed area, center of gravity and maximum extensions of the beam footprints (see page 23.1). Examples: `FOO` — plots the footprint for all fields. Surface 1 is the default. `FOO ?` — invokes a dialog box to select surface, field and plot extents. `FOO s4 f4..6` — plot footprint on surface 4, fields 4 to 6. `FOO s4 25.0` — footprint with manual definition of `plot_extent`, all fields.

Like many options in *OpTaliX* , for footprint analysis the chief rays must be traceable, even if it is obscured. Boundary calculations are performed by a search algorithm moving from the chief ray outward in radial direction until the stop aperture or a fixed aperture on any other surface in the system is found. The algorithm is not designed to handle obscuring sub-apertures like spiders, which divide the pupil into three (or four) parts.

See also the ray intersection option (page 14.1.8), which plots the used area on surfaces based on a full grid of rays traced to the selected surface for each field bundle and zoom position.

In the following example (Fig. 23.1), a fold mirror has been added behind a Double Gauss lens. The footprint on the fold mirror shown for nine field points indicates how large it must be to avoid additional vignetting of the beams within the field of interest.

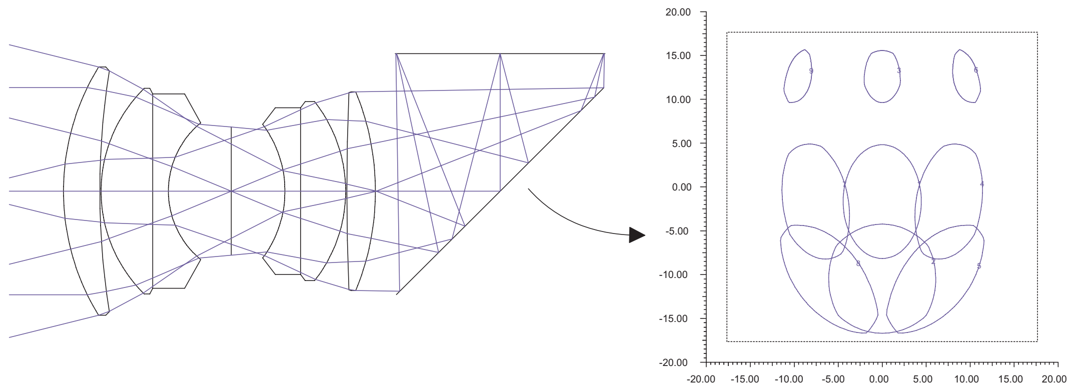


Figure 23.1: Beam footprints on fold mirror behind a Double Gauss lens.

NUM Option in Footprints:

The NUM option in footprint analysis outputs additional data, such as enclosed area, center of gravity and maximum extensions of each beam footprint, separated for field and zoom position. Note that this analysis does NOT include aperture obscurations on the designated surface. See the sample output below:

```
FOOTPRINT DATA on Surface  1

Pos.  Field     X-Center    Y-Center    Area (mm^2)        X-min       X-max       Y-min       Y-max
   1      1       0.0000      0.0000      313.57982        -9.994       9.994      -9.994       9.994
   1      2       0.0000     -4.1785      306.45888        -9.996       9.996     -13.943       5.578
   1      3       0.0000     -9.3060      251.48431       -10.006      10.006     -16.999      -2.024
                                                      ---------------------------------------------
                                                          -10.006      10.006     -16.999       9.994
```

23.2 Aspheric Deformation

The aspheric deformation option calculates the deviation of an aspherical (non-spherical) surface with respect to a perfect sphere. The radius of the perfect sphere is taken as a reference and can be selected according to different criteria.

Aspheric deformation is expressed as difference of the sag of the asphere to the sag of the perfect sphere (i.e. the reference sphere).

ASD sk [ref] [ref_rad] [?]	Aspheric deformation in radial direction. Plots and prints the sag of the asphere compared to a reference sphere which may be contacting at one or two zones on the air side of the surface. sk — surface number ref — reference describing the type of reference radius, where ref can be one of: VER : vertex radius CEN : center and rim zero RIM : only rim zero BFR : best-fit radius. ref_rad — spherical reference radius **Example:** ASD s3 CEN : Plots aspheric deviation with reference radius calculated for zero deviation at center and rim of surface.
ASD2 [sk|?] [ref_rad]	Aspheric deformation shown over full surface area. The deformation is based on the reference radius ref_rad. If ref_rad is omitted or is 0, the vertex radius of the designated surface is used.

23.2.1 Aspherization in radial Direction

Enter "ASD ?" in the command line or select from the main menu *Manufacturing − > Aspheric Deformation − > in radial direction*. Four options are selectable in a dialog box to determine the reference radius

1. the vertex radius is taken as the reference radius
2. the reference sphere contacts center and rim of the surface
3. only the rim of the surface is contacted by the reference sphere,
4. a "best fit" approach is attempted (the reference sphere touches the aspheric surface at 0.7 of the aperture radius.

Each of the options has its distinct advantages. The following treatment shall be a concise guide in selecting the optimum reference radius (see figure 23.2).

Option 1:
Vertex Radius: This option is probably the first and simplest choice as it directly reflects the mathematical definition of the asphere. However, for fabrication purposes, it is not reasonable as the amount of material to be removed is extremely large. In addition, it may lead to infeasible solutions for steep (conic) aspheres, as already shown in the drawing above.

Option 2:
Center + Rim Zero: The spherical reference radius is constructed such that the reference sphere has contact (touches) the asphere at two zones: The center (of revolution) and the rim (at the max. aperture). Thus, only in the intermediate zones, material must be removed.

Option 3:
Only Rim Zero: Here, the reference sphere touches th asphere at only one zone, the rim. Compared to option 2 (center and rim zero), much more material must be removed during grinding and polishing. The main advantage is, however, that the edge does not require further shaping during the subaperture grinding phase which generally avoids the ”turned down edge” problem.

Option 4:
Best Fit: This option is equivalent to option 3 (only rim zero) but differs in that the zone at which the reference sphere touches the asphere is at 0.7 of the maximum aperture radius. Much less material must be removed (compared to option 3) but the danger of turned down edges during polishing exist.

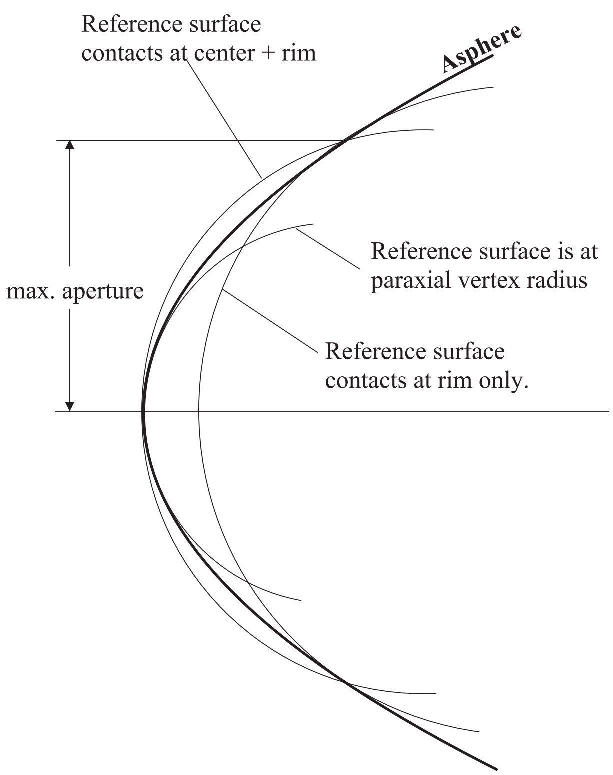


Figure 23.2: Construction of reference radius to an aspheric surface.

In addition to the aspheric deviation plot, numerical values are also printed at 21 positions along the Y-height of a surface. A typical output lists the surface parameters (curvature, conic constant, aspheric coefficients) and subsequently the Z-coordinates at various radial heights in Y-direction.

```
[h]
 ASPHERIZATION DATA:

    File    : F15_33.OTX
    Surface :  15
    Vertex Curvature  :    -0.17277313E-01
    Vertex Radius     :   -57.87937061E+00
    Conic Constant    :     0.00000000E+00
    A                 :    10.77564552E-06
    B                 :    23.69965431E-09
    C                 :   -53.48477648E-12
    D                 :   441.68107450E-15
    E                 :     0.00000000E+00
    F                 :     0.00000000E+00
    G                 :     0.00000000E+00
    H                 :     0.00000000E+00

    Asphierization is determined for zero deviation at center and rim.  Radius = -91.49306
```

Radial height (mm)	Z-Sphere (mm)	Z-Asphere (mm)	Difference (mm)	Slope (micron/mm)	-------- Surface Normal -------- CXN	CYN	CZN
0.00000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	1.000000
0.70000	-0.002678	-0.004231	-0.001553	-2.22	0.000000	0.012079	0.999927
1.40000	-0.010712	-0.016893	-0.006181	-6.61	0.000000	0.024069	0.999710
2.10000	-0.024103	-0.037897	-0.013794	-10.88	0.000000	0.035878	0.999356
2.80000	-0.042855	-0.067093	-0.024238	-14.92	0.000000	0.047410	0.998876
3.50000	-0.066969	-0.104261	-0.037291	-18.65	0.000000	0.058560	0.998284
4.20000	-0.096452	-0.149108	-0.052657	-21.95	0.000000	0.069219	0.997601
4.90000	-0.131306	-0.201261	-0.069955	-24.71	0.000000	0.079263	0.996854
5.60000	-0.171540	-0.260255	-0.088715	-26.80	0.000000	0.088557	0.996071
6.30000	-0.217159	-0.325522	-0.108363	-28.07	0.000000	0.096949	0.995289
7.00000	-0.268173	-0.396376	-0.128204	-28.34	0.000000	0.104264	0.994550
7.70000	-0.324589	-0.471991	-0.147401	-27.43	0.000000	0.110291	0.993899
8.40000	-0.386419	-0.551366	-0.164947	-25.07	0.000000	0.114768	0.993392
9.10000	-0.453673	-0.633286	-0.179614	-20.95	0.000000	0.117354	0.993090
9.80000	-0.526363	-0.716246	-0.189883	-14.67	0.000000	0.117587	0.993063
10.50000	-0.604502	-0.798349	-0.193847	-5.66	0.000000	0.114832	0.993385
11.20000	-0.688104	-0.877160	-0.189056	6.84	0.000000	0.108190	0.994130
11.90000	-0.777185	-0.949495	-0.172310	23.92	0.000000	0.096392	0.995343
12.60000	-0.871760	-1.011118	-0.139358	47.07	0.000000	0.077646	0.996981
13.30000	-0.971847	-1.056340	-0.084493	78.38	0.000000	0.049448	0.998777
14.00000	-1.077464	-1.077464	0.000000	120.70	0.000000	0.008382	0.999965

The meaning of the columns is:

`Z-Sphere`	Z-coordinate of the base sphere, respectively the reference sphere if fitting to the deviation at the rim or to the best-fit sphere (options 2-4, see above) is requested.
`Z-Asphere`	Z-coordinate of the aspheric surface
`Difference`	The deviation of the aspheric surface from a sphere (either base sphere or best-fit sphere)
`Slope`	The derivative of the aspheric deformation with respect to the base or reference sphere, as shown in Fig. 23.3.
`CXN, CYN, CZN`	Direction cosines of the surface normal.

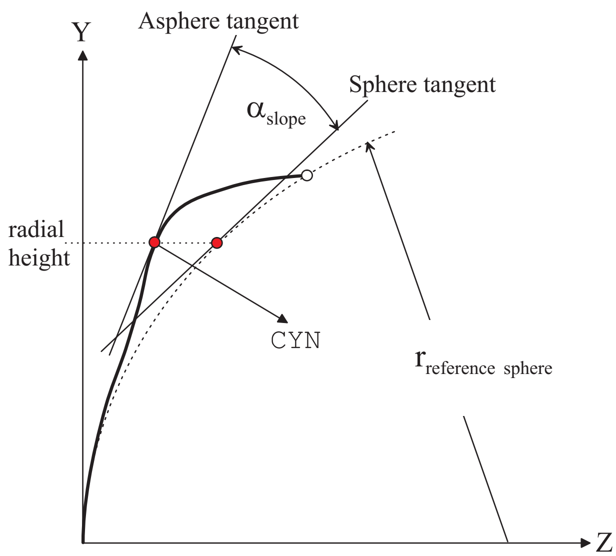


Figure 23.3: Slope of aspheric deformation based on the reference sphere.

23.2.2 Asphérization as 2D Surface Deformation

Enter "`ASD2 ?`" in the command line or select from the main menu *Manufacturing − > Aspheric Deformation − > as 2D-surface deviation* which invokes a dialog as shown in Fig. 23.4.

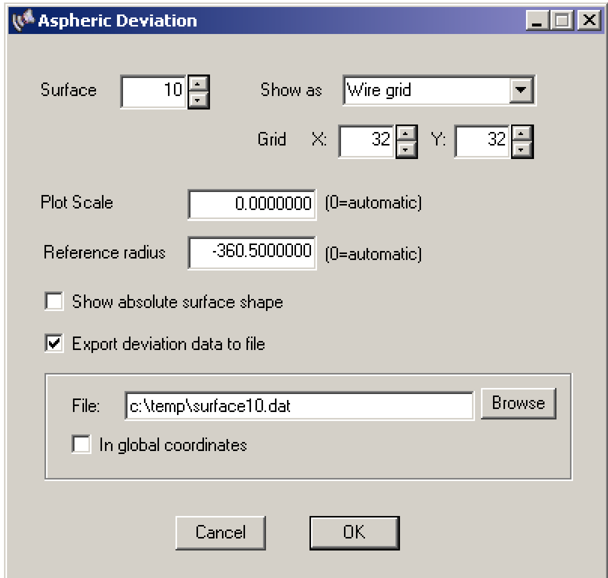


Figure 23.4: Dialog box for creating 2D surface deformation plots.

The program searches for the first aspheric surface in the optical system and displays the corresponding surface parameter in the dialog. The reference radius is always the vertex radius, however, it may be changed to any other arbitrary value.

2D aspheric deformation data may also be exported as X-Y-Z coordinates to a file in ASCII or Excel format. Note that this option is currently only available from the dialog.

23.3 Hologram Phase

This section displays the phase on diffractive surfaces, and indicates the required surface profile on a substrate according to the hologram coefficients.

`HPH [?]`	Plots phase on diffractive/holographic surfaces. Also plots the sag of the surface profile based on the corresponding hologram coefficients.
`HZO [?]`	**??** Calculates the radial zones of radially symmetric diffractive (HOE) phase profiles based on $2\pi = 1\lambda$ intervals. Output is only generated on ”H” and ”G” surface types. Otherwise, an error message is displayed.

23.3.1 Converting Symmetric Hologram Coefficients to other Programs

23.3.1.1 To Code V

On hologram surfaces with symmetric phase functions, the $OpTaliX$ hologram coefficients are converted to Code V by the following relation:

$$c_{CodeV} = \frac{c_{OpTaliX} \cdot \lambda_0}{1000} \tag{23.1}$$

Note that the factor in the denominator describes the conversion from micrometers ($OpTaliX$ default) to nanometers (Code V default).

23.3.1.2 To Zemax

Description to follow.

23.3.2 Hologram Zone Calculation

This section describes calculation of zones on diffractive structures (in absolute and 2π terms) with symmetrical phase profiles. The absolute phase is usually represented by a surface profile similar to Fresnel zones, where the steps are arranged at modulo (2π) phase intervals. Each interval corresponds to 1λ phase difference at the reference (design) wavelength. A typical cross-sectional representation of the phase profile is given in Fig. 23.5.

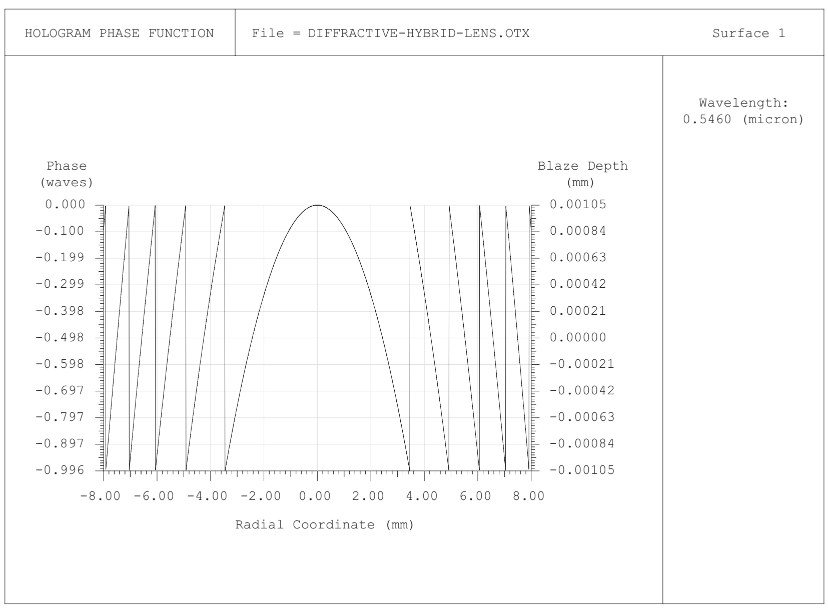


Figure 23.5: Modulo 2π zones on diffractive surface with radially symmetric phase function

The sagitta of the radial groove profile (i.e. modulo (2π) of the diffractive phase function), also commonly described as blaze depth d, is then calculated by [62],

$$d = \frac{\lambda_0}{n_0 - 1} \tag{23.2}$$

where λ_0 is the reference wavelength, and n_0 is the refractive index at the reference wavelength.

The radial coordinates of rotationally symmetric diffraction zones are calculated by theHZO command. Phase 2π steps are located at 1nm intervals which should be sufficiently accurate for all manufacturing aspects.

23.4 Edge Thickness

`ET si..j X_height Y_height`	Edge thickness of surface(s) `si..j` at surface coordinates `X_height`, `Y_height`. If `X_height`, `Y_height` are omitted, the clear aperture Y-height will be used. For tilted/decentered surfaces see the convention in sect.23.4.1 below.

23.4.1 Calculating edge thickness at tilted/decentered surfaces

If any surface within of the specified range `si..j` is tilted or decentered, edge thickness (`ET`) is calculated with reference to the local coordinate system of the first surface in the range given, i.e. `ET` is measured along the local Z-axis of the first surface.

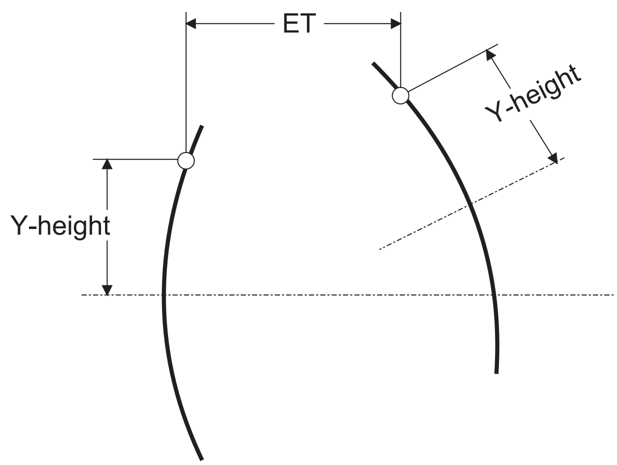


Figure 23.6: Edge thickness at tilted surfaces.

23.5 Test Plate Fitting

Performs automatic fitting of surface radii to a test plate list of a specific manufacturer. All test plate information is provided by the respective vendors.

`TPL [si..j | manuf]`	Find the nearest radius of curvature from a manufacturers test plate list and replace it against the existing radius. The expression `manuf` describes the manufacturer. The first three characters are significant. See table 23.1 below for a complete list of available test plate lists. If `manuf` is absent, a dialog box will be opened. Example: `tpl s4..7 ROD` selects test plates from Rodenstock and replaces the actual radii of surfaces 4 to 7.

`LIS TPL [manuf]`	Reports test plate list of `manuf`. The first three characters of the manufacturer string are significant to identify the list. If `manuf` is omitted, a dialog box will be invoked for selection of the appropriate manufacturer.
	Examples:
	`lis tpl mel`
	`lis tpl melles griot`

23.6 Adding a Test Plate List

Test plate lists (`TPL`) are stored in readable unformatted ASCII files, ending in the extension `TPL`. New lists may be added easily if the specific `TPL` file structure is preserved. A detailed description of the test plate file structure is given in section 32.6.

The file "`tplinfo.txt`" in the `./testplat` directory contains a summary of all available testplate files and a short description. New (user defined) testplate files must have an entry to this file. For each testplate list, two kinds of information must be entered (unformatted) in a single line, separated by at least one blank character:

The testplate filename (including extension) and a descriptive text to the testplate list, which also appears in the dialog combo box. If the descriptive text itself contains blanks, the text must be enclosed in quotation marks.

Example of `tplinfo.txt` file:

```
din.tpl "DIN (Deutsche Industrie Norm)"
kreischer.tpl Kreischer
s&h.tpl Spindler&Hoyer
kodak.tpl Kodak
liebmann.tpl Liebmann
lightnin.tpl Lightning
ofr.tpl OFR
optolyth.tpl Optolyth
```

23.7 ISO Element Drawing

Element drawings in accordance to the ISO 10110 standard can be generated from the lens prescription data. Such drawings are useful when a lens design is prepared for fabrication. The tolerances used in element drawings are taken from the previously entered or calculated tolerances.

Element drawings are created by the command

`ELE [ sk | ? ]`	Element drawing according to ISO 10110, starting from surface `sk`. Drawing items are taken from prescription data and (where available) from tolerance data. The optional question mark invokes a dialog box (see Fig. 23.7) for editing drawing items. The drawing can be immediately printed/plotted using the redirection symbol, for example `ELE s3 > plt`.

One drawing is generated for each element. Multiple elements must be printed separately. Single lenses or cemented doublets can be drawn. Only centered (axially symmetric) elements are drawn correctly. Tilts or decenter in an element are not reproduced.

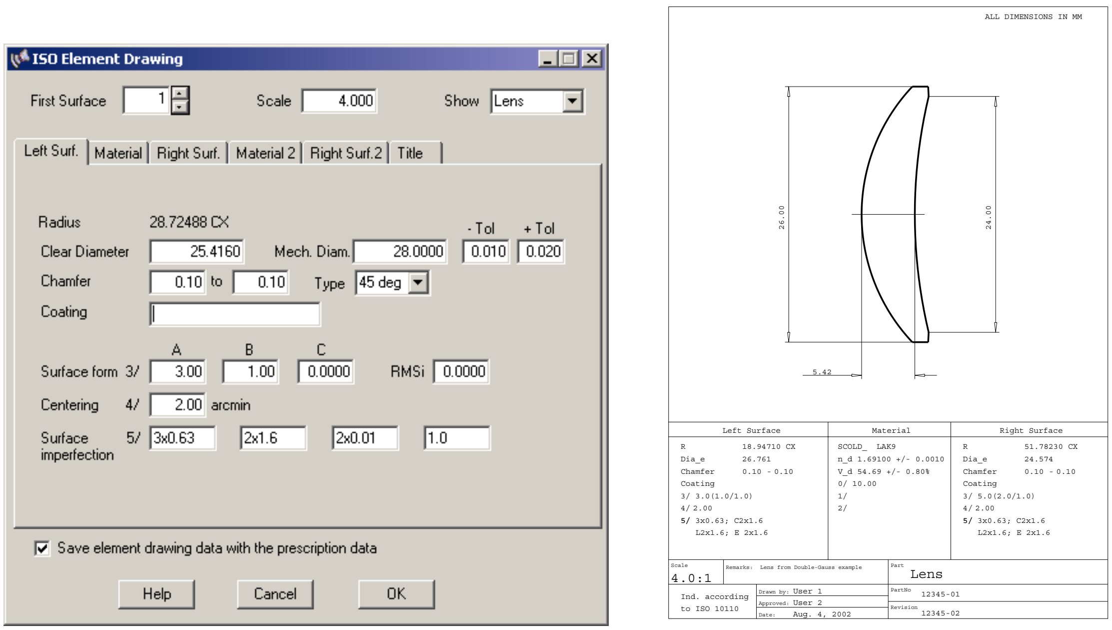


Figure 23.7: ISO element drawing dialog box for editing element drawing indications (left) and corresponding sample output (right). The dialog box is invoked from the command line by the command `ELE ?`.

The dialog box as shown in Fig. 23.7 is the central focus for editing and controlling the appearance of the element drawings. Changes take effect immediately and can be viewed interactively in the associated preview window, which remains open as long as the ISO element drawing dialog box is opened.

Data entry in the dialog box is grouped in six tabbed sections. The first three tabs belong to the first surface, the material and the last surface of a lens. Title information can be entered independently for each lens in the sixth tab. The fourth and fifth tab are reserved for cemented doublets and are activated only when doublet drawing is required (selected from the menu in the upper right corner of the dialog).

Tolerances in the ISO element drawing dialog are automatically taken from the current tolerance data if specified in the tolerance spreadsheet editor (see chapter 22), however, they can always be overwritten by manually entered tolerances.

Element drawing data is retained in the lens file if the appropriate check box ”*Save element drawing data with the prescription data*” in the dialog as shown in Fig. 23.7 is checked. Otherwise, element drawing data are lost on program exit or when a new optical system is restored (loaded).

The following description gives a concise overview about the meaning of all data entry fields in the

ISO element drawing dialog box. It does not replace a detailed study of the ISO 10110, Parts 1-11, specifications.

Radius: The radius of curvature is taken from the prescription data and cannot be changed in the element drawing dialog. In order to produce manufacturing ready drawings, it is assumed that the radii have been fitted to test plates (see section 23.5). Concave surfaces are denoted by ”CC” and convex surfaces are denoted by ”CX”.

Clear Diameter: Initially the clear diameter is taken from the prescription data and constitutes the effective optical diameter which is required by all defined ray bundles. Note that the clear diameter can be automatically determined by the command `SET MHT` (set maximum heights). The clear diameter can always be overwritten by the user.

Mech. Diameter: The outside diameter of the element can be specified with a $\pm$ tolerance. The diameter must be greater or equal to the clear diameter.

Chamfer: Minimum and maximum permissible widths of the protective chamfers. Pertains to all edges and corners that are not explicitly specified.

Coating: Coatings may be specified in a text field. No predefined form is given as coating specifications typically require separate specification documents. Usually, the coating indication contains a reference to the specification document.

Surface Form: Definition and specification of the surface form is given in detail in ISO 10110, Part 5. Surface form deviation is ”*the distance between the optical surface under test and the nominal theoretical surface, measured perpendicular to the theoretical surface, which shall be nominally parallel to the surface under test.*”
Surface form deviation is indicated in fringe spacing (one-half the wavelength of light at 546nm) in one of the three forms:

3/A(B/C)
3/A(B/C) RMSx ¡ D, where x is either t, i or a.
3/ - RMSx ¡ D

where
A is the maximum permissible sagitta error in fringes,
B is the maximum permissible value of irregularity expressed in fringe spacings,
C is the maximum permissible rotationally symmetric irregularity expressed in fringe spacings,
D is the maximum permissible value for rms residual deviation. Only RMSi values can be specified in the dialog box.

Centering: Indicates the maximum permissible tilt angle in minutes of arc.

Imperfections: Specifies surface imperfections (scratches, pits and coating blemishes) in the form

5/NxA; C N’xA’; L N”xA”; E A”’

where
NxA is the number and size of general surface imperfections,
C N’xA’ indicates coating blemishes, where **N’** is the number of allowed blemishes and **A’** indicates the grade number,
L N”xA” indicates the long scratch specification with **N”** being the number of allowed long scratches (>2mm) and **A”** is the maximum with of the scratches,
E A”’ is the edge chip specification where **A”’** specifies the maximum permissible extent of a chip from the physical edge of the surface.

Material: The material (glass) name is taken from the prescription data and cannot be edited.

nd: The index of refraction at the d-line (587.6nm). Only the tolerance on refractive index can be specified. The default value is 0.001.

Vd: The Abbe number at the d-line (587.6nm). Only the tolerance on Abbe number can be specified. The default value is 0.8%.

Stress Birefringence: It is specified in terms of optical path difference, expressed in nm/cm. The default value is 10nm/cm.

Bubbles and Inclusions: The specification is indicated by **1/ NxA**, where **N** is the allowed number of bubbles and inclusions and **A** is a grade number. See ISO 10110 Part 3 for further reading.

Striae and Inhomogeneity: The specification is indicated by **2/ A;B**, where **A** is the inhomogeneity and **B** is the striae class. Inhomogeneity is characterized by the maximum permissible variation in refractive index, given in 10^{-6} units. Striae is defined in five classes where classes 1-4 are related to a density of striae. Class 5 is virtually free of striae and requires further information in a note. See ISO 10110 Part 4 for further reading.

Thickness: The tolerance on axial thickness.

Mirror Thickness: This field is only active on mirror surfaces. The mirror thickness is the center thickness to the back surface of a first-surface mirror. In the command line, this value is specified by the `THM` command.

Part: The element can be identified by a part name. Even though it is possible to enter a part name for every surface, only the part name of the leftmost surface of the element/doublet appears on the drawing.

Part No.: A number identifying the element. The field is limited to 64 characters.

Revision: Tracks version changes. The field is limited to 64 characters.

Remarks: A text field limited to 64 characters for entering additional notes.

23.8 CAM Calculation

The CAM option provides a table of parameters for constructing a precise relationship between movable parts (lenses or groups of lenses). This option is preferably used in constructing the cam for a mechanically compensated zoom lens, however, it is not restricted to calculate axial separations but allows *any* lens parameter to be included in the calculation. Thus, in $OpTaliX$ CAM may also be used for calculating relationships between tilt and decenter parameters (for example in scanning systems) or any other exotic combination of description parameters.

CAM generates cam data by optimizing the optical system at each step of the cam. This is done by successive passes through the optimization option incrementing the linear variable (stepping) parameter STE before each pass.

The CAM option does not primarily require a zoomed system, or that the system is 'dezoomed' prior to calculating cam tables. CAM mode is universally available for both zoomed and non-zoomed (fixed focus) systems.

In order to facilitate this capability, $OpTaliX$ provides two completely independent data areas to hold optimization variables, targets and constraints, which do not interfere. That way, 'normal' optimization and CAM calculation can be performed independently in the same setup.

Two modes of operation are provided, a 'normal' zoom mode and a CAM mode. Switching between those two modes is accomplished by the commands "`CAM Y`" and "`CAM N`".

In the description to follow we will concentrate on the most often required case of mechanically compensated zoom lenses, that is, the computation of a table of axial separations between moved groups.

In a zoomed system, simply switch to CAM mode, define a second optimization set and perform CAM calculation. Then the user may switch back to normal zoom/multi-configuration mode and continue optimization or analysis of the zoomed system. $OpTaliX$ saves both optimization sets with the prescription data. This allows continuation of 'normal' zoom optimization/analysis and/or CAM calculation from saved and restored systems.

Also note that due to the close relationship of CAM calculation and optimization settings, menu items to edit CAM parameters are found both in the *Optimization* and *Manufacturing* main menus.

When switching to CAM mode in a zoomed system, the program temporarily converts the system to a non-zoomed system (without losing the zoom data!) and calculates the cam. The previous zoomed state can always be restored by the "`CAM N`" command.

Commands:

`CAM Y|N|zk|RUN [XLS file.xls]`	Switch between CAM mode (Y) and normal zoom mode (N). Automatically dezooms a system to position 1. Specify `zk` to start CAM calculation from any other position `zk`. If in CAM mode, CAM calculation can be initiated by the `RUN` parameter. The `XLS` option exports the cam table to an Excel file. See also the notes on creating an Excel file (page 484). Examples: `CAM Y` ! switch to CAM mode starting with position 1, `CAM z2` ! switch to CAM mode starting with position 2, `CAM RUN` ! execute CAM calculation, `CAM N` ! switch back to normal zoom mode. `CAM RUN XLS c:\my_data.xls` ! execute CAM and export data to Excel file.

`STE sk|param`
or
`CAM STE sk|param`
	Designates the separation or parameter to be stepped linearly. If only a surface qualifier is specified, separation of that surface is assumed. That is, `sk` is implicitly understood as "`THI sk`". It is, however, possible to specify *any* prescription parameter, which is specified in the `param` string.
	For example,
	`STE s5` ! steps separation 5 (THI s5) linearly,
	`STE ADE s7` ! steps tilt about X-axis on surface 7 (ADE s7) linearly,
	`STE 'ADE s7'` ! as above but `param` provided as string.

`INC step_size`
or
`CAM INC step_size`
	Size of step to be taken in the separation or parameter target.

`LIM max_value`
or
`CAM LIM max_value`
	Stop the CAM calculation when the value of the stepped separation/parameter (given by `STE`) exceeds this value.

continued on next page

continued from previous page	
CAM OUT param_string1..10	Designates up to 10 parameters for which values are listed. The parameter definitions must be provided as strings, that is they must be enclosed in quotes. Parameter strings must be separated by at least one blank character. Parameter strings do not (yet) accept lens database items and arithmetic expressions. Example: `CAM OUT 'thi s5' 'thi s10' 'efl' 'oal'`
BAS offset or CAM BAS offset	Designates a constant value to be added to each of the listed parameters. Allows matching of table to reference points in the mechanical design.
LIS CAM	List CAM parameter and associated CAM optimization variables and constraints.
EDI CAM	Edit CAM parameter and associated CAM optimization variables and constraints in a dialog box.

Upon exit from a cam calculation in the CAM mode, the system is left in the configuration of the last cam step so that a continued run (with different parameters) may be made if desired. If the system is later switched to normal zoom mode (see `CAM N` command), the optical system is restored at zoom position 1.

Example:

The CAM calculations performed in this example are based on the design `CAM Example.otx` found in the `\optalix\examples\optimization` directory. In this design, thicknesses 5, 10 and 15 are variable to accomplish the movement of the groups. Thickness 5 will be linearly stepped through the allowable movement range (1mm - 50mm). The remaining thicknesses 10, 15 are optimized to fulfil a constant focus on the optical axis and a constant overall length (`OAL`).

We enter the CAM mode,

```
CAM Y
```

and define the linear stepping parameter

```
STE THI s5   ! Step thickness on surface 5
INC 2.0      ! Increment for surface 5
LIM 50.0     ! Maximum value of surface 5
```

The variables and targets/constraints for CAM calculation are defined in the same way as for normal optimization. Variables can be edited in a dialog (use `VAR ?` command) or directly from the command line:

```
VAR s10 THI
VAR s15 THI
```

The targets/constraints definition for CAM calculation is short and sweet:

```
spd f1 0       ! Minimize spot diameter at field 1 (axis),
oal = 121.5    ! Maintain overall length (OAL).
```

Finally we need to define the parameters to be listed. These are the thicknesses 10 and 15. In addition we want to monitor focal length (EFL) and the overall length (OAL).

CAM OUT 'thi s5' 'thi s10' 'efl' 'oal' 'spd f1'

Note that the parameters to be listed must be given as strings (that is enclosed in apostrophes) and parameter strings must be separated by at least one blank character.

Here is a summary of the whole story, obtained by the `LIS CAM` command:

```
CAM CALCULATION PARAMETERS:
  Linear stepping parameter    (STE) : THI S5
  Stepping increment           (INC) :        2.00000
  Maximum of stepped parameter (LIM) :       50.00000

       List Parameter             Offset
   1 : THI S10                    0.0000
   2 : THI S15                    0.0000
   3 : EFL                        0.0000
   4 : OAL                        0.0000
   5 : SPD F1                     0.0000

CAM VARIABLES :
  S10    THI
  S15    THI

CAM TARGETS AND CONSTRAINTS :
  spd f1 0
  oal = 121.5
```

The cam calculation is initiated by the command `CAM RUN`:

```
CAM CALCULATION

FILE = CAM_Example.otx

          THI S5      THI S10      THI S15          EFL          OAL       SPD F1
   1     1.00000     56.47201      1.12499      5.90331    121.50000      0.00367
   2     3.00000     54.37153      1.22547      6.25214    121.50000      0.00354
   3     5.00000     52.26517      1.33183      6.63267    121.50000      0.00343
   4     7.00000     50.15151      1.44549      7.04912    121.50000      0.00338
   5     9.00000     48.03185      1.56515      7.50546    121.50000      0.00330
   6    11.00000     45.90509      1.69191      8.00707    121.50000      0.00324
   7    13.00000     43.76938      1.82762      8.56059    121.50000      0.00325
   8    15.00000     41.62445      1.97255      9.17309    121.50000      0.00333
   9    17.00000     39.46936      2.12764      9.85319    121.50000      0.00348
  10    19.00000     37.30302      2.29398     10.61110    121.50000      0.00373
  11    21.00000     35.12420      2.47280     11.45900    121.50000      0.00411
  12    23.00000     32.93149      2.66551     12.41154    121.50000      0.00463
  13    25.00000     30.72328      2.87372     13.48641    121.50000      0.00529
  14    27.00000     28.49768      3.09932     14.70510    121.50000      0.00609
  15    29.00000     26.25255      3.34445     16.09395    121.50000      0.00700
  16    31.00000     23.98536      3.61164     17.68545    121.50000      0.00801
  17    33.00000     21.69320      3.90380     19.51995    121.50000      0.00913
  18    35.00000     19.37267      4.22433     21.64797    121.50000      0.01033
  19    37.00000     17.01982      4.57718     24.13323    121.50000      0.01159
  20    39.00000     14.63009      4.96691     27.05663    121.50000      0.01284
  21    41.00000     12.19831      5.39869     30.52153    121.50000      0.01397
  22    43.00000      9.71879      5.87821     34.66022    121.50000      0.01478
  23    45.00000      7.18589      6.41111     39.64128    121.50000      0.01493
  24    47.00000      4.59570      7.00130     45.67467    121.50000      0.01383
  25    49.00000      1.95113      7.64587     53.00207    121.50000      0.01075

INFORMATION: The system has been left at the last step in the CAM
             Enter "CAM N" to restore all zoom positions (Zoom systems only).
```

Short form	**Manufacturer**
app	Applied Optics
bm	B&M Optik
bef	Befort
ber	Bern Optics
br1	Brighten Optics, Shop 1
br2	Brighten Optics, Shop 2
br3	Brighten Optics, Shop 3
coa	Coastal Optical Systems
com	Computer Optics Inc.
con	Continental Optical Corp
ddo	DD-optik
din	DIN (Deutsche Industrie Norm)
gos	GOST Russian testplates
har	Harold Johnson Optical Lab.
ii-	II-IV Incorporated
jan	Janos
jlw	JLWood Optical Systems
kod	Kodak
kre	Kreischer
lig	Lightning
lie	Liebmann
lin	Linos
mel	Melles Griot
mod	Model Optics
med	MediVision
nee	Neeb Optik
new	Newport
oci	OCI (Optical Components Inc.)
ofr	OFR (Optics for Research)
ogf	OGF (Optico Glass Fabrication)
opt	Optimax
opl	opl Optolyth
pog	Praezisionsoptik Gera
pro	PRO (Pacific Rim Optical)
rmi	Rocky Mountain Instruments
rod	Rodenstock
sil	Sill Tools
spe	Special Optics
spc	Spectros
swi	SwissOptik
tel	Telic Optics
tro	Tropel Corp.
tuc	Tucson Optical Research Corp.
tow	Tower

Table 23.1: Available test plate lists and corresponding 3-letter short forms.

24

Glass Manager

OpTaliX contains a number of auxiliary tools to select, view and analyze optical properties of glasses.

24.1 Use of Glass Catalogs

This section describes the use of glass catalogs. Typically, one or multiple glass catalogs can be loaded for a particular optical system. The following commands support this feature:

`LOAD GCAT cat1 cat2 cat3 ...`	Load glass catalogues, designated by a sequence of catalogue names, e.g. `load gcat schott hoya oha` would load the glass catalogs from Schott, Hoya and Ohara. Only the first three characters are significant.
`LOAD GCAT ALL|?`	As above, loads glass catalogues. The parameter `ALL` loads all glass catalogs that are available in *OpTaliX* . The question mark "?" invokes a dialog box for interactive selection of catalogues.
`LIS GCAT`	Lists the currently loaded glass catalogues in the text output window.

Alternatively, interactive selection of glass catalogues is accomplished by from the main menu

Glass Manager −− > Select Glass Catalogs

A dialog box is invoked which allows selection of particular glass catalogs or all glass catalogs that are available in *OpTaliX* (see Fig. 24.1).

24.2 Glass Map

The glass map is a diagram of index of refraction versus Abbe number ν or versus dispersion $n_F - n_C$ as provided by most glass manufacturers. The collection of glass catalogues is selectable by the command `LOAD GCAT ?`.

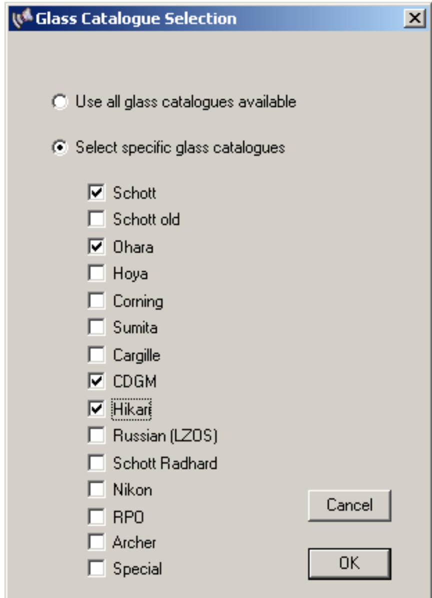


Figure 24.1: Selection of particular glass catalogs using the command ”`LOAD GCAT ?`” (without quotes).

`NNU`	Plot glass map, index of refraction vs. Abbe number
`NFNC [?]`	Plot partial dispersion diagram. Use the optional ”?” parameter to invoke a dialog for selecting glass catalogues and diagram options. See also sect. 24.3 below.

24.3 Partial Dispersion Plots

The partial dispersion plots are invoked by the `NFNC` command and show the deviation of the glass dispersion from the Abbe normal line (defined as a straight line connecting the Schott glasses K7 and F2). The selectable partial dispersions are $P_{g,F}$, $P_{C,s}$, two artificial partial dispersions for the spectral regions $1-2\mu m$ and $3-5\mu m$, and a plot of the partial dispersions $P_{g,F} - P_{d,C}$. For the latter, a similar plot is available using the Buchdahl coefficients η_1, η_2.

24.4 Athermal Map

The athermal map plots chromatic dispersive power versus thermal dispersive power, see Fig. 24.5. This is a useful tool for finding optical systems corrected for both chromatic aberrations and focus shift over temperature. See also section 24.5 for a more analytical approach to this subject.

For each material, chromatic dispersive power ω and thermal power ψ can be computed as

$$\omega = -\frac{(\partial n/\partial\lambda)\Delta\lambda}{n-1} \tag{24.1}$$

$$\psi = \frac{\partial n/\partial T}{n-1} - \alpha \tag{24.2}$$

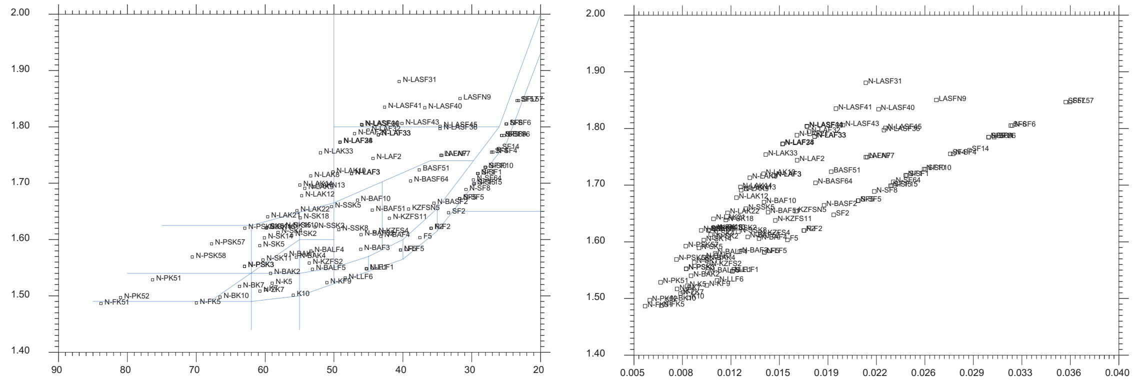


Figure 24.2: Glass maps, shown for Schott glasses. Left: index of refraction vs. Abbe number, right: index of refraction vs. dispersion $n_F - n_C$.

where α is the linear expansion coefficient. Note that the chromatic dispersive power ω is proportional to $1/\nu$, where ν is the Abbe number as defined in Eq. 13.6 (page 216). For the sake of simplicity, we consider a thin-lens doublet (i.e. two materials) only, which we want to achromatize (zero chromatic dispersive power) and athermalize (zero thermal power). This requires the solution of three linear equations,

$$\Phi = \Phi_1 + \Phi_2 = 1 \tag{24.3}$$

$$\Delta\Phi = \omega_1 \cdot \Phi_1 + \omega_2 \cdot \Phi_2 \tag{24.4}$$

$$\frac{d\Phi}{dT} = \psi_1 \cdot \Phi_1 + \psi_2 \cdot \Phi_2 \tag{24.5}$$

Referring to Fig. 24.5, this means that the two materials should lie on a straight line O-L intersecting the origin O in the thermal map. If no such material combination can be found, in particular when materials must transmit in a non-visible wavelength range (e.g. infrared glasses), three materials must be combined to accomplish the desired effect. For further reading see Tamagawa et.al. [55],[56].

Notes:
The athermal map does NOT take into account thermal effects of the housing structure (i.e. changes of air spaces under temperature), lens thicknesses and higher order ray aberrations. Therefore, in real systems, the athermal map can only be used as a guideline for selecting materials suitable for athermalization.

The following section 24.5 describes a method to include effects of housing expansion, at least in the paraxial domain.

24.5 Athermal Glass Selection

Tamagawa et.al. have devised a numerical method for athermalizing optical systems by combining optical materials with suitable lens powers and simultaneously fulfilling the achromaticity condition [54], [55], [56]. The method is based on determining both thermal and dispersive powers and calculating the corresponding lens powers, including the effects of thermal housing expansion.

Because it is difficult to find pairs of two glasses that lie on a straight line going through the origin of the athermal glass map (Fig. 24.5), accomplishing an athermal doublet is unlikely, albeit not

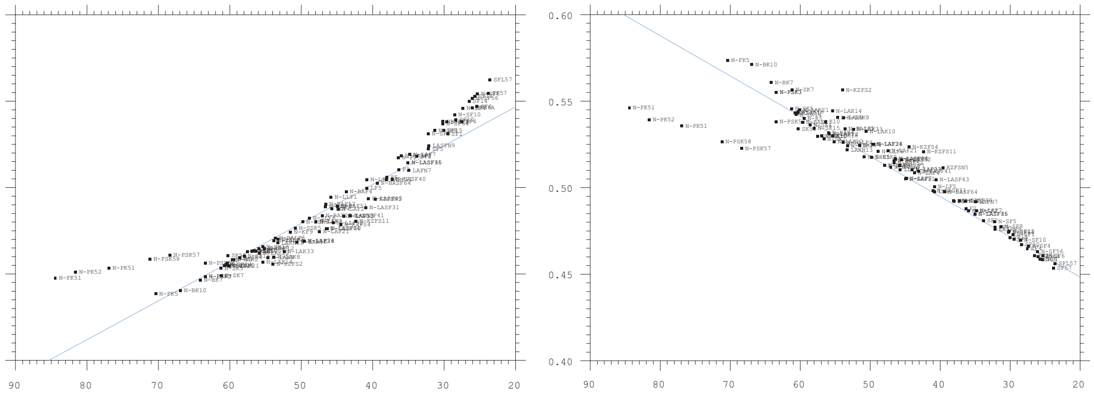


Figure 24.3: Partial dispersion plots, shown with Schott glasses. Left: index of refraction vs. $P_{g,F}$, right: index of refraction vs. $P_{C,s}$.

impossible. The following treatment focusses on a combination of three materials (triplet) which gives more flexibility and always allows to find suitable glass combinations for a given application.

With three glasses, we have three equations to be simultaneously fulfilled:

$$\Phi = \Phi_1 + \Phi_2 + \Phi_3 = 1 \tag{24.6}$$

$$\Delta\Phi = \omega_1 \cdot \Phi_1 + \omega_2 \cdot \Phi_2 + \omega_3 \cdot \Phi_3 \tag{24.7}$$

$$\frac{d\Phi}{dT} = \psi_1 \cdot \Phi_1 + \psi_2 \cdot \Phi_2 + \psi_3 \cdot \Phi_3 \tag{24.8}$$

These equations can be expressed in matrix form,

$$\begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \end{bmatrix} \cdot \underbrace{\begin{bmatrix} 1 & 1 & 1 \\ \omega_1 & \omega_2 & \omega_3 \\ \psi_1 & \psi_2 & \psi_3 \end{bmatrix}}_{M} = \begin{bmatrix} \Phi \\ 0 \\ -\alpha_h l \Phi \end{bmatrix} \tag{24.9}$$

The thermal expansion of the housing is considered by $-\alpha_h l\Phi$ where α_h is the linear expansion coefficient of the housing material and l is the length of the housing. The individual lens powers are then obtained by

$$\begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \end{bmatrix} = M^{-1} \begin{bmatrix} \Phi \\ 0 \\ -\alpha_h l \Phi \end{bmatrix} \tag{24.10}$$

It is important to note that the above equations refer to the paraxial domain. Solutions of Eq.24.10 do not necessarily result in systems with good aberration correction. It is therefore advisable to search for glass combinations with minimum individual lens powers Φ_1, Φ_2 and Φ_3.

Command Input:

`ATH3`	Find three-glass combinations for athermal and achromatic correction in the paraxial domain.

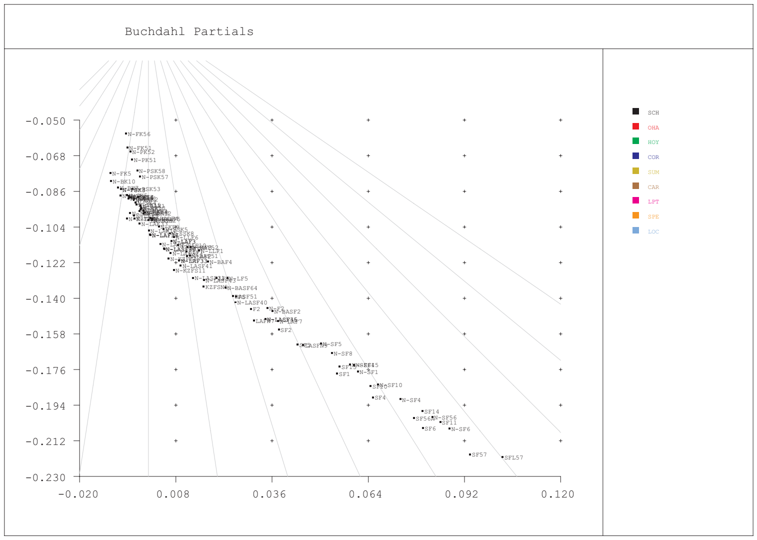


Figure 24.4: Partial dispersion plots with Buchdahl coefficients η_1, η_2, shown for Schott glasses.

24.6 Glass Selection for Thin-Lens Apochromats

This option is intended as an aid to selecting glass combinations, which are suitable for achieving apochromatic colour correction. Combinations of two and three glasses are supported. In finding such combinations, the program compares the dispersion properties of all glasses against a *base glass* and prints the required powers of the individual lenses.

The comparisons are based on Buchdahl's simplified equations for modeling dispersion by introducing a change in variables from wavelength λ to a chromatic coordinate ω. It is defined as

$$\omega = \frac{\lambda - \lambda_0}{1 + \frac{5}{2}(\lambda - \lambda_0)} \quad (24.11)$$

where λ_0 is the reference wavelength.

Using the chromatic coordinate, the index at any wavelength is expressed by the power series

$$n = n_0 + \nu_1 \omega + \nu_2 \omega^2 + ... + \nu_i \omega^i \quad (24.12)$$

where n_0 is the index at the reference wavelength λ_0 and the quantities ν_1, ν_2, ..., characterize the dispersion of the glass. This Taylor series converges very rapidly. The dispersive properties of glass are modelled with sufficient accuracy in the visible range (400-700nm) by a quadratic equation, and in the range 400 - 1000nm by a cubic equation.

It is important to note, that the above equations, if applied to real glasses and optical systems, are only valid in the paraxial domain. However, it may turn out that certain combinations will not perform as expected. In almost all cases, this is due to higher order monochromatic and chromatic spherical aberration, which is not covered by paraxial quantities.

24.6.1 Two-Glass Apochromats

`APO2 [ base_glass | ? ]`	Find two-glass combinations forming apochromatic correction in the paraxial domain.

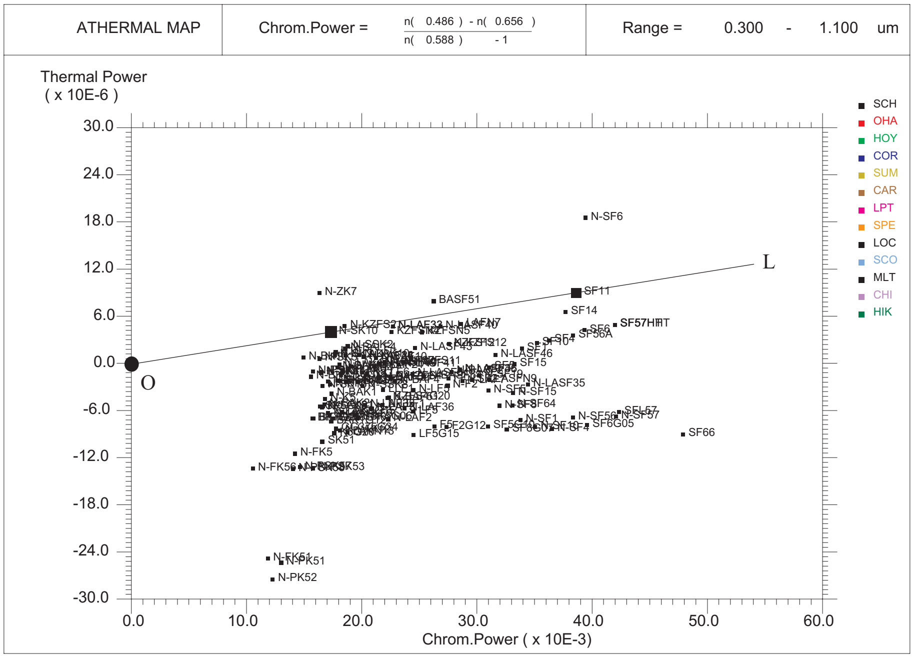


Figure 24.5: Athermal map, plotting chromatic dispersive power vs. thermal power for Schott glasses in the visible spectral range.

Example:

For a given base glass, the command APO2 selects glass combinations, where the ratio of the dispersion coefficients is as identical as possible to another glass.

The output gives a list of matching glasses (including their equivalent name) and the lens powers for a doublet of power = 1. The last column shows the expected rms-error of the longitudinal chromatic aberration (secondary spectrum) in the *paraxial* domain. Promising combinations are those with small lens powers (Phi1, Phi2) and small rms-error. However, even if the rms-error is small, high lens powers indicate large amounts of higher order chromatic aberrations (spherochromatism).

```
Glass dispersion coefficients based on Buchdahl chromatic coordinates :
 Baseglass  : KZFSN4
 Eta_1      :   -0.14080
 Eta_2      :    0.04012
 Ref.wavelength :       0.5500 micron

Glass          Equiv.Glass      Phi1       Phi2        RMS
SCH:LLF1       N-LLF1         -30.308     31.308     0.3855
SCH:N-BAF3     BAM3           -19.172     20.172     0.2716
SCH:N-BAF10    S-BAH10        -15.614     16.614     0.0964
SCH:N-BAF51    N-BAF51        -64.353     65.353     0.9895
SCH:N-KF9      N-KF9           -6.107      7.107     0.0631
SCH:N-KZFS4    N-KZFS4       -206.546    207.546     0.7215
SCH:N-KZFS11   N-KZFS11        23.824    -22.824     0.0012
SCH:N-LAF2     N-LAF2         -75.176     76.176     0.9451
.......
```

24.6.2 Three-Glass Apochromats

APO3 [base_glass | ?]	Find three-glass combinations forming apochromatic correction in the paraxial domain.

The following output is an example list for the base glass KZFSN4 from Schott:

```
Glass dispersion coefficients based on Buchdahl chromatic coordinates :
 Baseglass  : KZFSN4
 Eta_1      :   -0.14080
 Eta_2      :    0.04012
 Ref.wavelength :       0.5500 micron

Glass1          Glass2          Glass3                Phi1       Phi2       Phi3
SCH:KZFSN4      SCH:F2          SCH:N-FK51          -2.906      1.238      2.669
SCH:KZFSN4      SCH:F2          SCH:N-FK56          -2.387      1.074      2.313
SCH:KZFSN4      SCH:F2          SCH:N-PK52          -2.913      1.171      2.742
SCH:KZFSN4      SCH:F5          SCH:N-FK56          -2.568      1.294      2.274
SCH:KZFSN4      SCH:LAFN7       SCH:N-FK56          -2.533      1.105      2.428
SCH:KZFSN4      SCH:LASFN9      SCH:N-FK51          -2.458      0.756      2.701
SCH:KZFSN4      SCH:LASFN9      SCH:N-FK56          -1.993      0.655      2.337
SCH:KZFSN4      SCH:LASFN9      SCH:N-PK52          -2.489      0.715      2.774
SCH:KZFSN4      SCH:SF1         SCH:N-FK51          -2.359      0.613      2.747
.......
```

24.7 Gradient Index Profile

The profile of gradient index glasses shows the index of refraction as a function of the local z-coordinate. Currently, this plot is only available for pre-stored gradient index glasses with *axial* gradient. The plots are shown at the selected wavelengths.

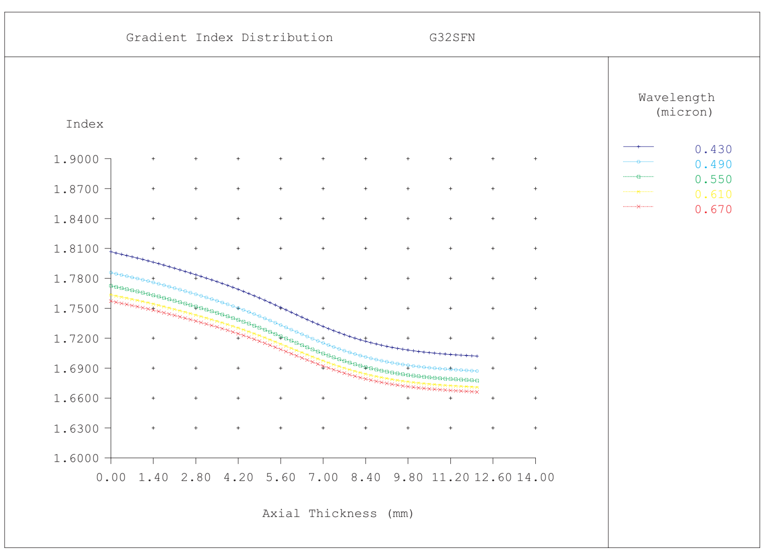


Figure 24.6: Gradient index profile, shown for five wavelengths.

24.8 View and Edit Glass Catalogues

<table>
<tr>
<td>GCAT [cat_name]</td>
<td>Invokes a spreadsheet containing glass data stored in the glass catalogues. The optional parameter cat_name is a three-character string designating the catalogue. The following catalogues are available:
<table>
<tr><th>cat_name</th><th>Glass manufacturer</th></tr>
<tr><td>SCH</td><td>Schott</td></tr>
<tr><td>SCO</td><td>Old Schott</td></tr>
<tr><td>OHA</td><td>Ohara</td></tr>
<tr><td>HOY</td><td>Hoya</td></tr>
<tr><td>COR</td><td>Corning</td></tr>
<tr><td>SUM</td><td>Sumita</td></tr>
<tr><td>CAR</td><td>Cargille liquids</td></tr>
<tr><td>LPT</td><td>LightPath Gradium</td></tr>
<tr><td>SPE</td><td>Specials catalogue (infrared, plastic, etc.)</td></tr>
<tr><td>HIK</td><td>Hikari</td></tr>
<tr><td>CHI</td><td>Chinese catalogue</td></tr>
<tr><td>MLT</td><td>Melts (user defined glasses)</td></tr>
</table>
Examples:
gcat
gcat sch</td>
</tr>
</table>

Only the melts catalogue (MLT) may be edited and saved whereas the data of all other catalogues can only be viewed. This is mandatory in order to preserve data integrity of glass catalogues during later updates.

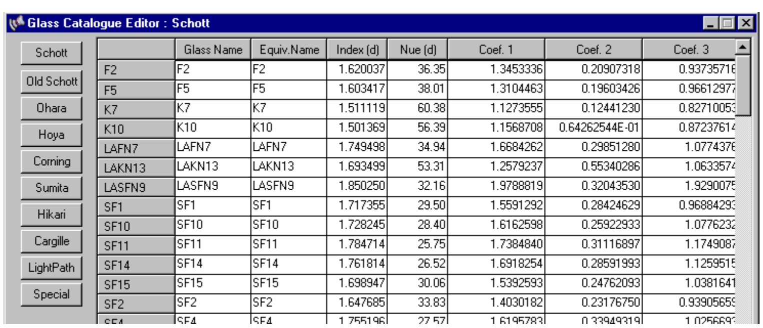


Figure 24.7: Spreadsheet for viewing and editing glass catalogue data. Only part of the dialog is shown.

The meaning of the columns is as follows:

Glass Name	The manufacturers glass name
Equiv.Name	Glass name of an equivalent glass. That is its optical properties are very similar. This can also be a glass from an other manufacturer.
Index(d)	Index of refraction at d-line
Nue (d)	Abbe number ν_d
Coef. 1-6	Dispersion coefficients. The type of dispersion formula is defined in the Column "Eq".
Eq.	Type of dispersion formula 0 = Old Schott formula, see Eq. 13.1 page 215. 1 = Sellmeier formula, see Eq. 13.2 page 215. 2 = Herzberger formula, see Eq. 13.5 page 215.
L-min	minimum wavelength in μm for which the dispersion coefficients are valid.
L-max	maximum wavelength in μm for which the dispersion coefficients are valid.
D0 D1 D2 E1 E2 LTK	Temperature coefficients dn/dT of index of refraction according to Eq. 13.2.
CTE	Thermal coefficient of expansion in 10^{-6} units.
Rho	Specific gravity ρ in g/cm^3.
RTI	Thickness in mm for which internal transmission data are defined.
2500 - 250	Internal transmission (excluding reflection losses) for a glass plate of thickness RTI at the wavelength (in nm) given in the column heading.

24.9 Melt Glasses

Manufactured optical glass and other materials as well vary slightly in refractive index from batch to batch as compared to the nominal or catalogue value. Typical tolerances for optical glass as supplied without any other specification are $n_d \pm 0.001$ and $\nu_d \pm 0.8\%$.

For critical applications such as long-focal-length high-resolution types, such (standard) tolerances are not sufficient and analysis with the exact measured refractive index data must be performed. To aid this process, glass manufacturers generally supply melt data sheets for each batch of glass, which allows adjustment of the values of radii, lens thicknesses or air spaces. Typically, the data is provided by the glass manufacturer at the wavelengths of a few selected spectral lines and some sort of fitting is required to obtain refractive index data at the wavelengths for which the optical system is designed. The interpolation uses the Sellmeier equation as described in equation13.2.

In order to use measured melt data, a new glass must be created on the basis of the manufacturer's melt data sheet and then added to the (melt) glass catalogue. Once created, the melt glass can be used like any ordinary catalogue glass.

This method is very general and can be used not only for melt glasses (i.e. glasses which deviate only slightly from a pre-stored catalogue glass) but also for creating entirely new glasses. Any feasible wavelength range may be entered, thus also "infrared" glasses or "UV" glasses may be created this way. It is, however, important to note that this scheme only applies for *homogeneous* glasses/materials. Inhomogeneous glasses such as gradient index cannot be created with this option.

Commands:

`MELT [?|fil melt_file_name]`	Create a melt glass from a set of discrete wavelength/index data pairs. Interpolation to Sellmeier coefficients is performed and the melt glass is then added to the "melts" catalogue. For command line input, the wavelength/index data pairs must be stored in an ASCII-file with extension "`.ind`". The melt glass file format is described in section 32.8. When used with the "`?`" option, a dialog box is invoked for interactive editing. **Examples:** `melt fil c:\optix\glasses\my_melt.ind` Fits index data contained in a file. `melt ?` Invokes dialog box for melt data editing

Dialog based Creation of Melt Glasses:

A particulary convenient method of creating and fitting melt glasses is using the dialog box. It is invoked by the command "`MELT ?`" or from the main menu *Glass Manager –> Create Melt Glass*.

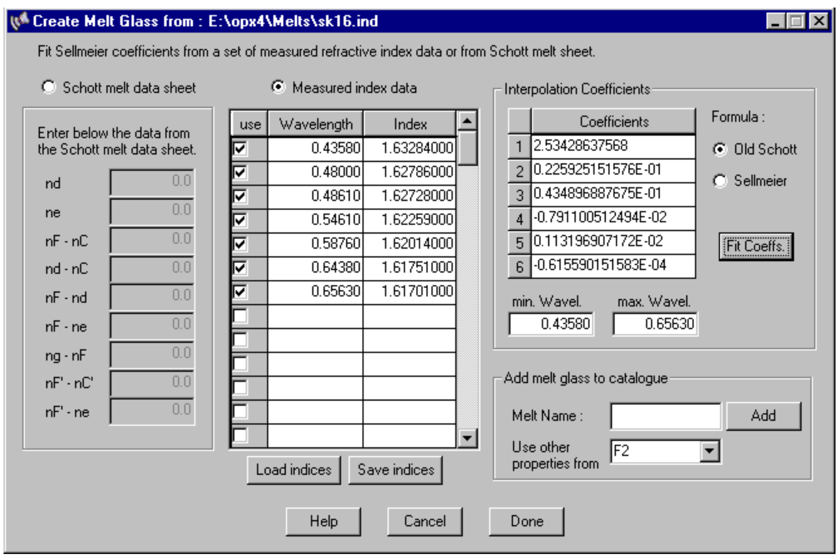


Figure 24.8: Dialog box for entering, fitting and creating melt glasses and new glasses respectively.

Two types of index data may be entered, either

- from the Schott melt data sheet (check the "*Schott melt data sheet*" radio button). The data must be entered manually into the dialog fields,
- or as pairs of wavelength/index data (check the "*Measured index data*" radio button. This data can be entered manually or can be restored from an "`.ind`" file, which should be preferably stored in the `\optix\melts\` directory (but may be any other).

Using the example dialog shown in Fig. 24.8, the steps to creating a melt glass are

1. Enter the wavelength/index pairs or load it to the dialog from an "`.ind`" file in the melts directory (click on the "load indices" button underneath the wavelength/index spreadsheet). Check those wavelengths, which shall be included into the fit. A maximum of 100 wavelength/index data pairs may be entered.

2. Select the formula to which the data shall be fitted. Currently, the old Schott equation (Eq. 13.1) and the Sellmeier equation (Eq. 13.2) are selectable.

3. Fit the data according to selected formula (click the ”fit coeffs.” button). The coefficients are then displayed in the rightmost table and are also reported (along with the accuracy of the fit) in the text window.

4. Enter a name for the new melt glass. A unique name (maximum 10 characters) must be given to identify the melt glass and distinguish it from the other catalogue glasses.

5. Select (or enter directly) a ”base” glass name, from which other glass properties (such as internal transmission, dn/dT, CTE, specific gravity, etc.) are taken and are also assigned to the new melt glass. In this way the melt glass possesses all properties of the base glass and behaves identically to the base glass (except index of refraction) for all subsequent analyses. Thus, analyses on transmission, thermal expansion, weight, etc. produce the same results for melt glass and base glass.

6. Add the fitted glass to the melts catalogue (press the ”Add” button).

7. Close the dialog box.

25

Printing and Plotting

Throughout this section, the term ”printing” is understood as printing text to the printer, i.e. all text and analysis output, which normally appears in the ”text window” on the screen. The term ”plotting” is denoted as ”printing” graphics to the printer using the Windows print manager. By default, all graphics and analysis output is directed to screen windows. To perform printing or plotting, the output device must be changed. Once an output unit is changed, all subsequent outputs are directed to the chosen device. To display the graphics and/or text output on the screen again, the corresponding output must be switched back to the screen. This concept works like a light switch, which is turned on and off. The currently selected output device (graphics or text) is displayed in the status bar of the main window as indicated in Fig. 25.1.

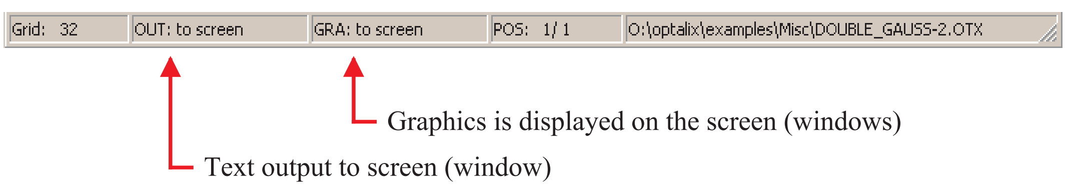


Figure 25.1: Print status shown in the status bar at the bottom of the main window.

In order to print/plot from the command line, you must switch the output devices manually as described in the following sections. From the GUI, switching output devices is done automatically in the background.

25.1 Printing and Plotting from the Command Line

`out prn	t	file file_name`	Direct text output to the default printer (`prn`) or the terminal/screen (`t`). Text output can be written to a file with the command `out fil file_name`. See examples below.
`gra prn|plt|t|file`	Direct graphics output to the printer (`prn`), plotter (`plt`), screen or text output window (`t`), or to a file (`fil`).
	continued on next page

continued from previous page	
`bmpx pixels_horiz`	Number of horizontal pixels in writing bitmaps (BMP, PCX). See sect. 25.2.2 for an example. The default width is 640 pixels.
`bmpy pixels_vert`	Number of vertical pixels for writing bitmaps (BMP, PCX). See sect. 25.2.2 for an example. The default height is 480 pixels.

For example, the following commands direct text output to the printer, a file or to the to the screen (text output window):

`out prn`	! output is directed to the default printer (output device is "prn")
`out t`	! output is directed (back) to the text output window (terminal)
`out fil 'c:\my output.txt'`	! write text output to file "`c:\my output.txt`".

In a similar way, changing the plot device (i.e. "printing" graphics) is accomplished by:

`gra prn`	! graphics output is directed to the default printer(output device is "prn")
`gra t`	! graphics output is directed back to the screen.

25.2 Printer and Plotter Device Units

The following output devices exist for printing text and plotting graphics:

`prn`	the default printer	text + graphics
`plt`	the plotter	graphics only
`t`	Screen (terminal)	text + graphics
`clp`	Clipboard	graphics only
`file`	Text/analysis output to a file	text only
`silent`	Disables text output (silent operation)	text only
`hpgl`	HPGL (Hewlett Packard Graphics Language)	graphics only
`dxf`	Graphics output to AutoCad DXF File	graphics only
`eps`	Graphics output to Encapsulated Postscript (EPS)	graphics only
`wmf`	Graphics output to Windows Metafile Format (WMF)	graphics only
`cgm`	Graphics output to Computer Graphics Metafile (CGM)	graphics only
`bmp`	Graphics output to Windows Bitmap format (BMP)	graphics only
`pcx`	Graphics output to Paitbrush file format (PCX)	graphics only
`png`	Graphics output to Portable Networks Graphics (PNG) format	graphics only
`svg`	Graphics output to Scalable Vector Graphics (SVG) format	graphics only

The following sections (25.2.1, 25.2.3) describe how printing/plotting is accomplished from the command line. Section 25.3 describes printing/plotting from the graphical user interface (GUI) directly.

25.2.1 Printing/Plotting Graphics

The default graphics output device is the screen. Other graphics output devices may be selected by the following commands:

```
gra dxf [file filespec]     ! redirect graphics to DXF-File
gra hpgl [file filespec]    ! redirect graphics to HPGL-File
gra bmp [file filespec]     ! redirect graphics to Windows bitmap (BMP) file
gra prn                     ! redirect graphics to default printer
gra plt                     ! redirect graphics to default printer, synonymous to gra prn
gra t                       ! redirect graphics to default screen
```

Other than for screen, printer and clipboard, graphics are always written to a file and, in this sense, redirecting a graphics output may be understood as ”exporting” the contents of a graphics window in the specified format.

For single plots, the graphics may be redirected to the printer/plotter temporarily by using the redirection symbol ”>” . For example,

```
fan > plt
vie > plt
```

redirect the ray-fan or lens layout plot immediately to the corresponding output unit, which is the Printer/Plotter ”plt”. Note, that the command entries must be separated by at least one single blank character. It is also important to note that the redirection is active only for one particular command, all subsequent commands appear on the previously selected device (usually the screen).

25.2.2 Controlling Bitmap Size

The size of graphics exported (printed) to bitmaps (BMP, PCX, Clipboard) can be controlled in two ways:

From the GUI:

The size of exported graphics to *raster image files* such as BMP, PCX, as well as to the clipboard corresponds to the size of the graphics window on the screen in pixel. That is, a small graphics window on screen will produce a small raster image file. The file size (and hence the number of pixels in horizontal and vertical direction) increases with increasing screen window size.

From the Command Line:

Specify the size of exported graphics by the commands BMPX, BMPY. The following example defines a lens layout plot (VIE command) as a bitmap of 800 pixels wide and 600 pixels high written to the file ”c:\my_graphics.bmp”:

```
gra bmp fil c:\my_grahics.bmp
bmpx 800
bmpy 600
vie
gra t
```

Note the logic of exporting graphics: In the command ”gra bmp ...” you define an output unit for the graphics (in this case, a file c:\my_graphics.bmp). Then additional commands can be added to define the property of the graphics such the bitmap size (BMPX, BMPY). Generate the type of graphics and then re-direct the graphics output back to the screen (windows) using the ”GRA T” command.

25.2.3 Printing Text Output

The default output device for text is the screen (terminal device). Other devices for text output may be selected by the following commands:

`out prn`	redirect all subsequent text output to default printer
`out file file_name`	redirect all subsequent text output to `file_name`.
`out t`	redirect all subsequent text output to default screen
`out silent`	disables text output (silent operation). Use one of the commands "`out t`" or "`out prn`" to enable text output again.

Once the output is directed to the printer (`out prn`), all subsequent text outputs will be printed on the default printer until the the text output is switched back to the screen (`out t`). Text output may be immediately redirected to the printer in a single command with the redirection symbol ">". For example,

`lis > prn`	! Listing is immediately printed on the default printer.
`rsi f1 w1 > prn`	! Single ray trace data is redirected to printer
`lis > xxx.txt`	! output to file `xxx.txt`

Note, that the command entries must be separated by at least one single blank character! The redirection is active only for one particular command, all subsequent outputs are written to the previously selected device (usually the screen).

25.3 Printing/Plotting from the GUI

The previous sections have shown how text/graphics can be printed/plotted from the command line. Whereas this is most useful in macros, for example to automate reports, there is an easier way for printing/plotting text and graphics.

25.3.1 Printing Text from the GUI

The entire text displayed in the text window or selected text can be printed.

Printing is then performed by clicking on the printer icon in the main window toolbar (Fig.25.3).

Note: If no text is selected, the contents of the entire window is printed. See also theCLS command for clearing the text window.

25.3.2 Printing Graphics from the GUI

Each graphics window has a toolbar to the left. Simply click on the printer icon to print the graphic contents of this window:

25.3.3 Examples

Send graphics to the clipboard:

```
gra clp
fan
gra t
```

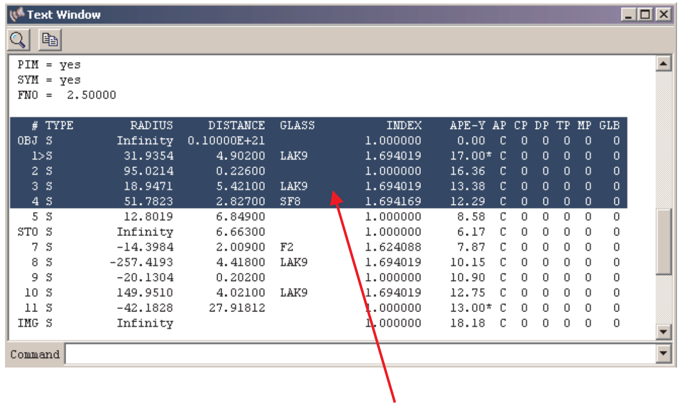


Figure 25.2: Select text in the text window. Printing of selected text is performed by clicking on the main menu printer icon (see Fig. 25.3). Note: If no text is selected, the contents of the entire window is printed. See also the CLS command for clearing the text window.

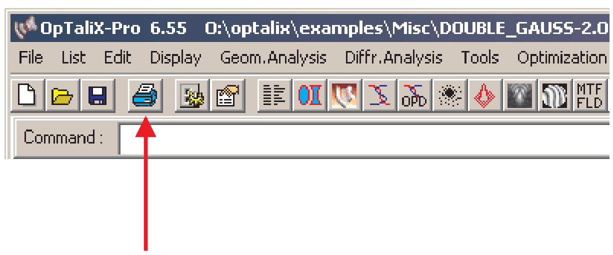


Figure 25.3: Print selected text from the text window. Note: For printing graphics, click on the printer icon at the left bar of each graphics window (see also Fig. 25.4).

Send graphics to a file:

```
gra bmp fil c:\graphics.bmp
fan
gra t
```

Send graphics with a specified size to a bitmap file:

```
gra bmp fil c:\graphics.bmp
bmpx 1200 !  horizontal with 1200 pixels
bmpy 800 !  vertical height 800 pixels
vie
gra t
```

Send text output to printer:

```
out prn
lis
out t
```

Send text output to printer (short form):

```
lis > prn
```

Send text output to a file:

```
out fil c:\text.txt
lis
out t
```

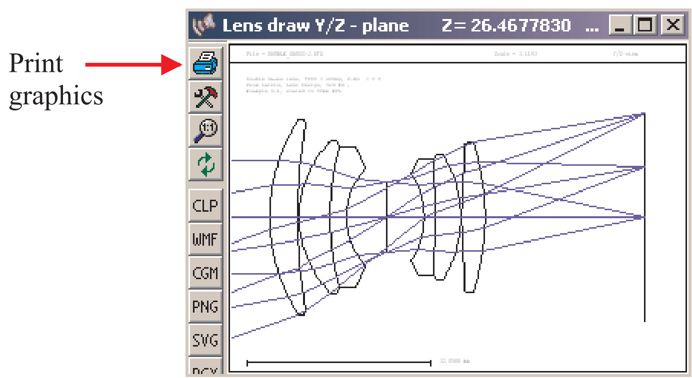


Figure 25.4: Print graphics.

26

Macro Language

A macro is a sequence of *OpTaliX* commands, arithmetic expressions and database item specifications stored in a file. Macro commands may also interactively entered and executed in the command line. There is no functional difference between commands in a command line or stored in a file.

Macros are written to summarize often repeated command sequences into one single command or to enhance the capabilities of *OpTaliX* with new user-defined or user-specific features.

Creating and executing a macro is a two step process. Macro commands to be used must first be entered in a text file, which has the preferred extension `.mac` (such as `test.mac`) but any other extension is also accepted. Editing can be done with any ASCII text editor available under the operating system. *OpTaliX* offers a built-in macro editor, which avoids the need to invoke an external editor. Up to 20 macros may be edited in the *OpTaliX* macro editor. The *OpTaliX* macro editor can be invoked by the command

```
EDI MAC
```

or from the menu *Edit –> Macro files*.

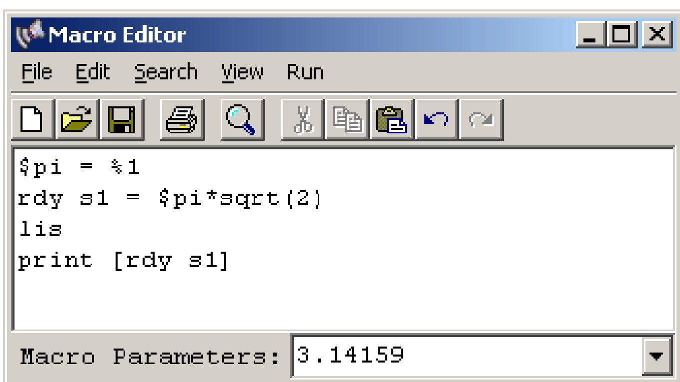


Figure 26.1: Macro Editor Window. This example passes one parameter (3.14159) to the macro which is interpreted in the macro script by the %1 token as the first parameter. In the first line of this script, a variable `$pi` is defined based on the passed parameter. The second line assigns the value of variable `$pi` to the radius of surface 1, multiplied by `sqrt(2)`. The third line list the prescription data and in line 4, radius of surface 1 is output (queried) from the lens database.

After editing the macro sequence, the macro can be immediately executed by clicking on the '**Run**' menu item. External parameters passed to the macro, if required, may be defined in the field labeled

"`Macro Parameters:`" at the bottom of the macro editor window (see Fig. 26.1). In case a requested %i parameter in the macro script is not available, respectively not defined, it is assumed zero (for numeric values) or blank (for character variables).

The macro editor offers several buffers to hold more than one macro sequence. Selecting the 'File' $-->$ 'New' option or by clicking on the icon  in the macro editor menu opens a new buffer. Buffers can be selected from the 'View' menu in the macro editor's main menu.

You will be asked at program exit whether to save unsaved buffers or not. Also on closing the macro buffer, either by selecting 'File' –¿ 'Exit' or by clicking on the icon  in the upper right corner, a dialog box will request saving of still unsaved macro sequences.

26.1 RUN Statement

From the macro editor, the macro can be immediately executed by clicking on the '**Run**' menu in the macro editor window. Alternatively, a saved macro file is executed by the command

```
run filename [parameter1...9]
```

This command reads in and executes the contents of a macro file (given with full path) where `[parameter1..9]` allows up to 9 expressions (numbers, strings or arithmetic expressions) to be passed to the macro as parameters. Each parameter expression is evaluated and the result (number or string) is substituted for a corresponding special symbol (%1, %2, ... %9) in the macro.

Suppose the following very simple example macro `example.mac`,

```
! Prints the root of a number
print 'The root of ' %1 'is ' sqrt(%1)
```

which is executed from the command line by

`run example.mac 2`

where the number 2 following the macro name is the first parameter to be passed to the macro. The output is

`The root of 2.000000000000000 is 1.414213562373095`

Note that parameters are not variables, they are essentially constants that are defined at runtime.

26.2 Arithmetic Expressions

An expression consists of operands and operators. Operands are constants, lens database items and user defined variables. Operators are

`+`	addition
`-`	subtraction
`*`	multiplication
`/`	division
`**`	exponentiation
`^`	exponentiation

There exist also an extensive set of intrinsic functions:

sin(r)	sine of angle in radians
cos(r)	cosine of angle in radians
tan(r)	tangent of angle in radians
exp(x)	e^x
log(x)	natural logarithm
log10(x)	common logarithm
logn(n,x)	logarithm base n
sqrt(x)	square root
acos(r)	arccosine
asin(r)	arcsine
atan(r)	arctangent
cosh(r)	hyperbolic cosine
sinh(r)	hyperbolic sine
tanh(r)	hyperbolic tangent
besj0(x)	Bessel function 1^{st} kind, order 0
besj1(x)	Bessel function 1^{st} kind, order 1
besjn(n,x)	Bessel function 1^{st} kind, order n
aint(x)	truncate to a whole number
anint(x)	real representation of the nearest whole number
abs(x)	absolute value
min(a,b)	minimum value
max(a,b)	maximum value
sech(x)	hyperbolic secant (= $1/cosh(x)$)
csch(x)	hyperbolic cosecant (= $1/sinh(x)$)
rand	random number

Numbers are all assumed to be real and are entered in the usual FORTRAN double precision way. The # sign represents an integer digit.

```
                     Example:
    #                1
    .#               .1
    #.#              1.2
    #.#d#            1.2d3
    #.#d-#           1.2d-3
    #.#d+#           1.2d+3
    #.#e#            1.2e3
    #.#e-#           1.2e-3
    #.#e+#           1.2e+3
    #.#D#            1.2D3
    #.#D-#           1.2D-3
    #.#D+#           1.2D+3
    #.#E#            1.2E3
    #.#E-#           1.2E-3
    #.#E+#           1.2E+3
```

Note that blank characters are not allowed in arithmetic expressions, except where enclosed in brackets. Valid arithmetic expressions are:

```
print 2+3
```

```
print (2 + 3)
print ([EFL] + 2)
```

Invalid arithmetic expressions:

```
print 2 + 3
print [EFL] + 2
```

26.3 Lens Database Items

Macro expressions may include lens database items, which are retrieved from the current optical system. Almost anything that can be entered in the command line has a corresponding lens database item (see also chapter 27 for a complete list of available lens database items). All references to lens database items must be enclosed in rectangular brackets [and], even if there are no qualifiers. The syntax for database items mirrors the syntax used for command line input.

For example,

```
rdy s1 43.5
```

specifies the curvature on surface 1. The same syntax, but now enclosed in square brackets, without the value 43.5, returns the curvature on surface 1

```
[rdy s1]
```

This syntax may be combined with other commands as given in the following examples:

```
thi s2  [EPD]       ! sets thickness s2 equal to entrance pupil diameter
cuy s3 -[cuy s4]    ! curvature on surface 3 is equal to minus the
                    ! curvature on surface 4
```

Note that the last example (`cuy s3 -[cuy s4]`) does NOT constitute a permanent functional relationship (or pickup) between the curvatures `cuy s3` and `cuy s4`, it occurs only at the moment of input or macro execution.

Lens database items can be combined with arithmetic operators to form an arithmetic expression anywhere a numeric data entry is expected.

```
fno [EFL]/[EPD]      ! sets F-number
thi s3 2*sqrt(3)*[thi s1]
```

As already expressed in section 26.2 above, arithmetic expression must not contain blank characters, except within lens database items or when enclosed in () brackets. For example,

valid:	fno [EFL]/[EPD]
valid:	fno ([EFL] / [EPD])
invalid:	fno [EFL] / [EPD]

26.4 PRINT Statement

The print statement is used to send data to an output unit (text output window or file). See also section 25 (page 431) for selecting output units and section 26.7 (page 445) for defining formatted output. The print command is followed by a list of expressions. For example,

```
print 'The entrance pupil diameter is' [epd]
```

generates the output

```
The entrance pupil diameter is   12.00000
```

Strings must be enclosed in quotation marks. Numeric data, being either arithmetic expressions or constants, are output in free floating format displaying full double precision (64 bit) accuracy. The output format can be controlled using the `format` option as described in section26.7 (page 445).

Arithmetic expressions are directly solved in print statements. Multiple expressions in an output list may be comma separated. The comma is then repeated in output. For example,

```
$pi = 3.14159
$diam = 10.0
print 'Area of a circle with 10mm diameter = ' $pi*($diam/2)**2 'mm^2'
print 'Some expressions:' 2*[EFL] , atan([NA]), 4*3.14159
```

results in

```
Area of a circle with 10mm diameter =   78.53975000000000 mm^2
Some expressions: 100.0000000000000 , -0.1566953104668687 , 12.56636000000000
```

Example of changing the output unit in a macro sequence:

```
out file c:\test.txt                ! directs output to file
print 'System focal length' [EFL]   ! prints EFL to file
out t                               ! redirects output to screen (terminal)
```

If several arithmetic expressions or database items shall be printed in one line, they can be separated by appropriate separators. Valid separators are ',' (comma) or any text enclosed in quotes ' '. Examples:

```
print 'Two expressions:'  [efl], 2*[bfl]
print 'Two expressions:'  [efl] 2*[bfl]
```

26.5 Formatted Output

The FORMAT statement, when used in conjunction with the print statement, provides explicit information how data and characters are displayed on output. The syntax for defining formatted output closely (but not entirely) follows the conventions of the FORTRAN programming language.

The major difference to the FORTRAN convention is that formatted output is defined by a character string enclosed in apostrophes and appended to the `print` statement.

Table 26.1: Format Definition

`'format format-items'`	Statements for defining the output format must always be enclosed in apostrophes (') or quotes ("). Typically, a format statement is given in conjunction with the print statement. The definition of output formats closely follows the FORTRAN convention. See examples below on how `format-items` are constructed.

Description of `format-items`:

`format-items` is a comma-separated list of data-edit-descriptors, (B, O, Z, F, D, E, EN, ES, G, L, A), and control-edit-descriptors (X). The different forms of edit descriptors are described as follows:

Table 26.2: Format Edit Descriptors

Edit Descriptor	**Interpretation**	**Type**
`Iw[.m]`	Displays value as integer number with field width of `w` and `m` digits. Example: `I3`	Integer
`Fw.d`	Displays decimal number with field width of `w` and `d` decimal places, no exponent. Example: The format `F8.5` prints the value 12.345 as `12.34500`.	Real
`Ew.d`	Displays decimal number with field width of `w` and `d` decimal places in exponential representation. Example: The format `E12.5` prints the value 12.345 as `0.12345E+02`.	Real
`ENw.d`	Displays decimal number with field width of `w` and `d` decimal places in **engineering notation**. Example: The format `EN12.5` prints the value 12.345 as `12.34500E+00`.	Real
`Gw.d`	Displays decimal number in *generalized* format width of `w` and `d` decimal places. The output format is adapted to optimally fit the output width. If necessary, exponential representation is used. Example: The format `G12.5` prints the value 0.012345 as `0.12345E-01`.	Real or Integer
`A[w]`	Displays alphanumeric field (text string) with a field width of `w`. Example: `A10` outputs the string 'This is another example' as "`This is an`" without the quotation marks. Longer strings are truncated to width `w`. Use the "A" format character without the width (`w`) descriptor if the length of the text output is unknown.	Character
`Zw[.m]`	Displays value as hexadecimal number with field width of `w` and `m` digits. Example: The format `Z4` prints the (decimal) value 43 as `2B` in hexadecimal notation.	Integer
`Ow[.m]`	Displays value as octal number with field width of `w` and `m` digits.	Integer
`nX`	Move `n` spaces right of current position. Inserts space of `n` (blank) characters.	None

Example 1:

```
print 'format F7.3,F10.1' 12.3 14.5
```

Prints the numeric values (12.3, 14.5) as floating numbers in the formats F7.3 and F10.1. The output is, where ␣ represents a blank character (space):

```
␣12.300␣␣␣␣␣␣14.5
```

Example 2:

```
print 'format F7.3,2X,A12' 12.3 'This is a long text'
```

prints

```
  12.300  This is a lo
```

because the format descriptor (`A12`) limits text output to 12 characters. If the length of the output string is not known, use the the generic `A` format as shown in example 3 below. It will not truncate text output, however, due to the unknown string length, formatted output is not predictable.

Example 3:

```
print 'format F7.3,2X,A,I4' 12.3 'This is a long text' 17
```

prints

```
 12.300  This is a long text  17
```

26.6 READ Statement

The READ statement transfers values from an input unit (typically a file) to the variables specified in an input list. Before reading variables from a unit, the input unit must be opened (see OPEN statement) and selected by the SELECT statement. Example:

```
open (unit=1, file='c:\temp\mac_read.txt')
select (1)
read $x $y
close (1)
```

26.7 Format Statements defined in Variables

Format definitions may also be stored in variables and re-used for printing data. An example of formatted output is given here:

```
$fmt1 = 'A4,F12.5'
print $fmt1 'formatted number' 4
```

In this example, the format definition is assigned to the variable `$fmt1`. This variable is then re-used in the print statement in line 2.

26.8 CONCATENATION of Strings

The character sequence '`//`' denotes the concatenation operator. In a command or macro statement, the concatenation operator joins two character strings end to end. For example the strings "sun" and "light" may be concatenated to give "sunlight".

Example 1: Concatenation of Two Strings:

`print 'abc'//'123'` outputs the string: 'abc123'

Example 2: Concatenation of String and Variable:

```
$x = 4
print 'abc'//$x
```

outputs the string 'abc4'

Example 3: Dynamic File Names:

```
   do $x = 1,5
      $file = 'test'//$x//'.dat'
   enddo
```

creates the file names

```
test1.dat
test2.dat
...
test5.dat
```

Multiple strings may be concatenated in one line, e.g.

```
print 'abc'//'def'//'ghi'
```

Note that blank strings are considered as "empty" strings according to the $OpTaliX$ syntax definition, i.e. they have no meaning. Accordingly the instruction `'my'//' '//'wife'` results in `mywife` and NOT `my wife`.

26.9 Evaluate Statement "EVA"

The evaluate statement `EVA` is functionally equivalent to the print statement (see above). It has been included for command compatibility with Code V. In addition to evaluating expressions, the `EVA` command also supports character strings. For example, the commands

```
print 'The half focal length is' [EFL]/2
eva 'The half focal length is' [EFL]/2
```

are equivalent. The EVA command also evaluates variables and functions such as

```
eva $x
eva @myfunc
```

26.10 File Inclusion

A file can be included with the command

```
#include filename
```

and the contents of the file ”filename” is executed as if it were entered directly in the macro file or on the command prompt. Nesting of included files is permitted to a depth of 10, i.e. an included file itself may call other files via the #include command. For example, consider the macro file ”macro1.mac” which calls (includes) the file ”macro2.mac”

```
! macro1.mac
#include macro2.mac
print 'Result' pi
```

and

```
! macro2.mac
$pi = 3.14159
```

On execution, they are executed as if all macro statements were entered in a single file:

```
! macro1.mac
$pi = 3.14159
print 'Result' pi
```

26.11 Variables

Variables are used for temporary storage of values. A variable may contain either a numeric value or a string of characters as data. The length of a variable name can be up to 60 characters. The type of a variable is the type of the data it contains. No distinction is actually made between integer or floating point numbers; all numbers are stored as double precision floating point values. The length of a variable *definition* (arithmetic expression) may be up to 128 characters. String data may also contain up to 128 characters.

Only scalar variables are permitted, that is, only a single value can be stored in a variable. The LVR command (list variables) may be used to display information about the currently defined variables.

LVR	List user-defined variable names and the numeric values associated.

The default value of an explicitly defined variable is zero (for numeric variables) or an empty string (for string variables).

A variable name **always** begins with a dollar character ($) followed by at least one alphabetic character, digits or underscores (_). Spaces are not allowed in variable names. Variable names are case insensitive, that is, $xy is equivalent to $XY. The following are examples of valid and invalid variable names.

valid	invalid	
`$x`	`$`	(at least one alphanumeric character required)
`$xy`	`$x y`	(space not allowed)
`$a_long_name`	`x`	(missing `$`)
`$1a`	`$a-b`	(arithmetic operators not allowed)

Variables are always declared 'global', that is, a variable is recognized during the entire run of *OpTaliX* , they can be accessed (set or queried) in all modules (e.g. macros, command line, user-defined graphics, etc) at any time they are required.

Variables may also be combined with qualifiers for surface, field, wavelength or zoom position. For example, a variable definition `$x = 2` may be reused for defining surface, field, wavelengths, zoom positions. With this example `s$x` would define surface `s2`. See section 6.2.3, page 34 for more details about this option.

26.11.1 Assignment Statement

The assignment statement is used to assign a value to a user-defined variable. The assignment operator (=) must have spaces preceding it and after it. The format of an assignment statement is as follows:

```
$user_var = expression
```

where

`user_var`	=	Specifies a user-defined variable name
`expression`	=	Specifies the value assigned to the variable

Examples:

`$x = 2`	Assigns the value 2.0 to the variable `$x`.
`$y = 3*$x`	Assigns the value `3*$x` to the variable `$y`. The variable `$x` must have been previously assigned.
`$z = 2*[efl]`	Assignment using a lens database item
`$glass = BK7`	Assigns the string `'BK7'` to the variable `$glass`

26.12 INPUT Statement

The INPUT statement interrupts the macro execution and prompts the user for numeric data or text data. A dialog box is displayed to enter up to five parameters. Input data is expected from the keyboard only. Up to five variables can be entered simultaneously in a single INPUT statement.

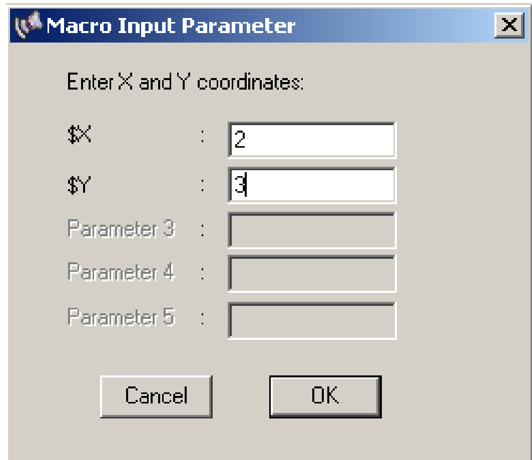


Figure 26.2: Input data

`INPUT 'text' $var1 [$var2 $var3 $var4 $var5]`	Input data in a macro sequence. The command interrupts execution of the macro, displays a dialog box for entering the variable(s) and then continues execution of the macro. The parameter `'text'` (enclosed in apostrophes or quotes) is a descriptive text displayed in the dialog. `'text'` is optional and can be omitted. At least one variable (`$var1`) must be specified/entered, otherwise the macro will be terminated.

Example 1:

```
input 'Enter x and Y coordinates:' $x $y
print $x $y
```

displays a dialog box for entering the variables `$x`, `$y`, as shown in Fig. 26.2:
Pressing the **OK** button continues execution of the macro, **CANCEL** terminates it.

Example 2:
The text field can be omitted, such as

```
input $x $y
print $x $y
```

26.13 OPEN Statement

The OPEN statement connects an external file to an input/output unit for subsequent read or write. The files are always opened in ASCII format. If a designated file does not exist, it is created.

`OPEN (unit = external-file-unit, file = 'filename')`	Opens an external file specified by '`filename`' and connects it to an input/output unit `external-file-unit`.
	The `external-file-unit` is a scalar INTEGER expression that evaluates to the input/output unit number of an external file. `external-file-unit` may be any INTEGER number greater than 0. The unit may also be defined in a variable and re-used in the OPEN statement (see example 2 below).
	`filename` is a scalar CHARACTER expression, enclosed in apostrophes, that evaluates to the name of a file, including the path specification. Files without path specification are assumed in the directory of the currently loaded optical system. The file name may also be specified in a variable, as shown in example 2 below.
	Syntax examples:
	`open(unit=3, file='c:\temp\test.txt')`
	`open (unit = 1 file = c:\temp\test.txt )`
	`open(unit=$unit, file=$file)`
	See also the corresponding statements CLOSE and SELECT.

Example 1: Writing data to a file:

```
open (unit=1, file='c:\temp\test.txt')
print 'format F10.4,2X,F7.4' EFL BFL
close (1)
```

Example 2: Using variables in OPEN statement:

```
$unit = 4
$file = 'c:\temp\my test file.txt'
open (unit=$unit, file=$file)
print 'some input/output follows'
close ($unit)
```

Note that the unit number 0 (zero) is reserved for the text output window and is not allowed in OPEN and CLOSE statements. See also the SELECT statement for re-directing output to the text output window.

26.14 CLOSE Statement

The CLOSE statement terminates the connection between a specified input/output unit number and an external file. The unit must have been opened previously by the OPEN statement.

`CLOSE (external-file-unit)`	Terminates the connection between a specified input/output unit number and an external file.
	Examples:
	`close (3)`
	`close($unit)`
	See also the corresponding statements OPEN and SELECT.

26.15 SELECT Statement

Selects an input/output unit that has been previously opened using the OPEN statement. This statement is particularly useful if more than one unit/file is opened and different operations (read, write) are performed on different files.

`SELECT (external-file-unit)`	Selects an input/output unit that has been previously opened using the OPEN statement. Example: `select (3)`

Example 1: Opening more than one unit and selecting the units:

```
open (unit=1, file='input.txt')
open (unit=2, file='output.txt')
select (1)
    print 'Writing text to unit 1'
select (2)
    read $x $y
close (2)
close (1)
```

Example 2: Selecting units with variables:

```
$unit = 1
$screen = 0
open (unit=$unit, file='input.txt')
print $x $y
select ($screen)    ! select screen/window (unit 0)
print $x $y         ! print to text window
close ($unit)
```

Note that unit number 0 (zero) is reserved for the text output window. Unit 0 is always opened and need not to be OPENed explicitly. By default, all outputs (PRINT command) are directed to unit 0 (text output window). Only when a output unit other than 0 has been selected and is in use, unit 0 must be explicitly selected in order to write to the text output window. Example:

```
open (unit=37, file='output.txt')
select (37)
```

```
    print 'Writing text to unit 37'
select (0)
    print 'other stuff'     ! output goes to text output window
close (37)
```

26.16 User-defined Functions

A user-defined function is the replacement of a defined name by its corresponding definition. A user-defined function name consists of an at-sign (@) followed by the name. The length of a function name can be up to 60 characters. The function name can have any number of alphabetic characters, digits, and underscores (_) following the at-sign (@). A special assignment operator (==) must be used for defining functions. The (==) assignment operator cannot have spaces separating the two = signs. A user-defined function assignment (i.e. definition) may include arithmetic expressions and operators (`+ - / * ** ^`), lens database items or intrinsic functions. The length of a function definition (arithmetic expression) may be up to 128 characters.

Examples:

`@my_fkn == 2*[efl]`	! Defines a function name "`my_fkn`" using a lens database item
`@123 == 12+sin(1)`	! Function names may contain digits

Invalid Function Definitions:

`@my_fkn = 2*[efl]`	! Function definition requires two = signs
`abc == 12+sin(1)`	! Function names must start with at-sign (@).

The function definitions may be listed by the `LFK` command:

LFC	List user-defined function names and the arithmetic definitions associated.

Note that the `#define` form is obsolete and should no longer be used.

26.17 Control Statements

Control statements allow the order of execution of statements to be changed. All control statements may be nested.

26.17.1 DO Construct

The `DO` construct specifies the repeated execution (loop) of a block of code. A `DO` statement begins a `DO` construct. An `ENDDO` statement ends the innermost nested DO construct. The maximum nesting depth of `DO-ENDDO` constructs is 20.

Syntax:

```
do $user_var = expr1, expr2 [,expr3]
   {statements}
```

```
enddo
```

where:

`$user_var`	Specifies a variable reference to contain the loop values.
`expr1`	Specifies the initial value of the loop variable `$user_var`.
`expr2`	Specifies the final value of the loop variable `$user_var`.
`expr3`	Optional. Specifies the increment/decrement value of the loop variable `$user_var`. If omitted, the default is +1.0. An increment value of 0 is not valid.
{statements}	Specifies the statement(s) to be executed within the `DO-ENDDO` environment.

Note: `expr1`, `expr2` and `expr3` may contain any valid arithmetic expression using variables, functions or lens database items.

Example 1:

A simple example indicating the use of arithmetic calculations.

```
do $x = 2,10,2
   $y = 2*$x
   print $x $y
enddo
```

Example 2:

This example alters the image surface thickness (the defocus) to step through a range of $\pm 0.1mm$ in increments of $0.02mm$. The coupling efficiency (`CEF`) is printed at the various focal positions.

```
do $x = [thi si]-0.1, [thi si]+0.1, 0.02
   thi s2 = $x
   print $x [cef]
enddo
```

Example 3:

This example uses macro parameters passed from the command line to the macro. For example the command '`RUN my_macro.mac 2 10 2`' passes the parameter values to be used for %1, %2 and %3 in the following DO-loop:

```
do $x = %1, %2, %3
   print $x
enddo
```

26.17.2 WHILE Construct

The `WHILE` construct specifies the repeated execution (loop) of a block of code until a condition is true. A `WHILE` statement begins a `WHILE` construct. An `ENDWHILE` or `ENDDO` statement ends the innermost nested WHILE construct. The maximum nesting depth of `WHILE-ENDWHILE` constructs is 20.

Syntax:

```
while (while_expr)
   {statements}
```

`endwhile`

In a WHILE loop-control, `while-expr` is evaluated and if false, the loop terminates. `while_expr` may contain any valid arithmetic expression using variables, functions or lens database items.

Example 1:

```
$x = 0
while ($x < 10)
   $x = $x+1
   print $x
endwhile
```

Example 2:

```
$x = 0
while ([thi s1] < 5)
   $x = $x+1
   thi s2 $x
   print [mtfa f1]
endwhile
```

26.17.3 IF Construct

The IF construct controls whether a block of statements will be executed based on the value of a logical expression. The syntax of IF constructs is:

`IF` (*expr*) `THEN`
    {statements}
`ELSEIF` (*expr*) `THEN`
    {statements}
`ELSE`
    {statements}
`ENDIF`

where *expr* is a scalar LOGICAL expression. The *statements* are evaluated in the order of their appearance in the construct until a true value is found, or an `ELSE` statement or `ENDIF` statement is encountered. If a true value is found, the block immediately following is executed. *Statements* in any remaining `ELSEIF` statements of the `IF` construct are not evaluated.

If none of the evaluated expressions is true, then the block of code following the `ELSE` statement is executed. The ELSE statement and its *statements* must be the last block in the `IF` construct.

The characters accepted for enclosing `IF/ELSEIF` expressions are parenthesis ( ) or braces { }.

Logical expressions may include arithmetic expressions (e.g. `2*sqrt($x)`) or database items or a combination of both (such as `2*[efl]`).

IF constructs my be nested. The maximum nesting depth of `IF-ELSEIF-ELSE-ENDIF` constructs is 20.

Rules for constructing Logical Expressions:

- Logical expressions must be enclosed in () or {} brackets.

- Logical expressions must have a logical operator, such as =, ==, /=, >, >=, <, <=.
- Blank characters are allowed within logical expressions, except within arithmetic expressions. That is,
  `IF (2*2 > 3)` is correct, whereas
  `IF( 2 * 2 > 3)` is not accepted (blanks within arithmetic expression).

Operators in IF Expressions:

The intrinsic operators in `IF` expressions are:

=	equal to
==	equal to
/=	not equal to
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to

Example 1:

```
$x = 0
if($x > 3) then
   print '$x is greater than 3'
elseif ($x > 0 ) then
   print '$x is greater than 0 but less than 3'
elseif ($x < 0) then
   print '$x is less than zero'
else
   print '$x is zero'
endif
```

Example 2:

```
$x = 0
if( [bfl] <= sqrt(100)) then
   $r = 0.5*[rdy s1]
   rdy s3 $r
   print 'Radius at s3 has been adjusted to ' $r
else
   print 'BFL is greater than 10'
endif
```

Example 3:

```
if ([gla s2]='n-bk7') then
   print 'true'
else
   print 'false'
endif
```

26.18 Return

The `return` statement passes one or more values from a macro to its caller. A `return` statement without variables has no effect. Arithmetic expressions are not allowed in the return statement.

Example 1:

```
$x = sqrt(2)
return $x       ! pass the value of $x to the caller
```

Example 2:

```
$x = sqrt(2)
$y = sin(1)
return $x $y    ! pass the values of $x and $y to the caller
```

Example 3:

```
return              ! statement has no effect (variables missing)
return 3*($x+2)     ! arithmetic expressions not allwed in return statement!
```

26.19 Comments

The character `!` indicates a comment except where it occurs in a character context. Examples:

```
$a = 3    ! this is a comment, which is not processed
print 'variable $a ' $a   ! this prints the variable
```

26.20 Logical Line Separation

The character `;` separates logical lines on a single physical line. For example,

```
THI s1..3 12 ; LIS; fan
```

is processed as if the following lines were entered separately

```
THI s1..3 12
LIS
fan
```

26.21 Logical Line Continuation

The character `&` as the last non-blank character of a line signifies that the logical line is continued on the next physical line. [1] If a character context in a macro file is being continued, the `&` may not be followed by a comment. If the first non-blank character is `&`, then the continuation begins at the character position immediately following the `&`; otherwise it begins in column 1.

Example:

```
The first line will be &
continued by a second line
```

[1]Note that the `&` character continues lines *only* in macro files. It has a different meaning in the command line, where it invokes option dialog boxes for commands.

is interpreted as a single line:

```
The first line will be continued by a second line
```

27

Lens Database Reference

This chapter summarizes the available lens database items. Almost all commands have a corresponding lens database item. The syntax for lens database items is identical to the syntax used in the command line. Unless otherwise noted, the returned quantity is a numeric value.

When specifying lens database items, the same mnemonics and syntax is used in the command line, in a macro file or as constraint/target in the definition of the optimization merit function. Lens database items must always enclosed in square brackets, [and]. Examples of valid and invalid lens database items are

```
[thi s3]           valid
thi s3             invalid, brackets missing
[EFL]              valid
[EFL   ]           valid
[E F L ]           invalid, keywords must not include blanks
```

Lens database items can also be used in arithmetic expressions such as

```
thi s3 sqrt(2*[SYL]+3.14159)
```

Lens database items can be printed via the `print` command. For example,

```
print 'Radius = ' [rdy s3]
```

outputs the radius of curvature on surface 3.

Lens database items accept variables in conjunction with qualifiers (for surface, field, wavelength, zoom, etc), such as

thi s$var 10.5

where `$var` is the integer value of variable `$var`. Assuming `$var = 3`, this syntax may be understood as concatenating "s" (without the quotes) and the integer value of `$var` to form the string "`s3`".

Configuration Data:	
`REF [zk]`	Reference wavelength number
`WL wk [zk]`	Wavelength at wavelength number `wk`, zoom position `zk`

continued on next page

continued from previous page

`XAN fi [zk]`	X-angle (in degree) for field number `fi` and (optional) zoom position `zk`. Note: If `XAN` is not the field specification value, for example when XIM defines the X-field, `XAN` returns the paraxial equivalent to the field specification. XAN = tan^{-1}(XIM/EFL).
`YAN fi [zk]`	Y-angle (in degree) for field number `fi` and (optional) zoom position `zk`. See also the note given for XAN.
`XOB fi [zk]`	X-object height for field number `fi` and (optional) zoom position `zk`. See also the note given for XAN.
`YOB fi [zk]`	Y-object height for field number `fi` and (optional) zoom position `zk`. See also the note given for XAN.
`XIM fi [zk]`	X-image height (paraxial) for field number `fi` and (optional) zoom position `zk`. See also the note given for XAN.
`YIM fi [zk]`	Y-image height (paraxial) for field number `fi` and (optional) zoom position `zk`. See also the note given for XAN.
`FNO [zk]`	Paraxial F-number
`NA [zk]`	Numerical aperture in image space
`NAO [zk]`	Numerical aperture in object space
`EPD [zk]`	Entrance pupil diameter
`APD [zk]`	Exit pupil diameter [1]
`PUI`	Intensity apodization across pupil
`PUX`	Apodization relative X-pupil coordinate at which PUI is reached
`PUY`	Apodization relative Y-pupil coordinate at which PUI is reached
Paraxial Data:	
`EFL [zk]`	Equivalent focal length, Y/Z-cross section, default
`EFLX [zk]`	Equivalent focal length, X/Z-cross section
`PWR [zk]`	Optical power = 1/EFL
`MFL sk`	Module focal length
`BFL [wk] [zk]`	Back focal length, if `wk` is absent, reference colour is used.
`OAL [si..j] [zk]`	Overall length between surface vertices `si` to `sj`
`SYL [si..j] [zk]`	Overall length between surface vertices `si` to `sj`. Without surface qualifier, first surface to image plane is returned.
`SH1 [zk]`	Position of front principal plane measured from vertex of first surface.
`SH2 [zk]`	Position of rear principal plane measured from vertex of last surface.
`OAL [zk]`	Overall length (= object-image distance for finite conjugates, respectively first surface to image for infinite object distance)
`OID [zk]`	Object to image distance
`MAG [zk]`	Magnification
`RED [zk]`	Reduction factor (= -MAG)
`EPD [zk]`	Entrance pupil diameter
`SAP [zk]`	Location of exit pupil from last surface
`SAPI [zk]`	1/SAP
`SEP [zk]`	Location of entrance pupil from first surface

continued on next page

[1] `APD` is derived from the German word 'Austrittspupillendurchmesser' = exit pupil diameter.

continued from previous page	
PRD [zk] PRDI [zk]	Pupil relay distance (distance of entrance pupil to exit pupil) 1/PRD
UMY sk [zk] UA sk [zk] HMY sk [zk] HA sk [zk] UCY sk [zk] UB sk [zk] HCY sk [zk] HB sk [zk]	Paraxial direction angle of the marginal aperture ray same as UMY Paraxial height of the marginal aperture ray same as HMY Paraxial direction angle of chief ray same as UCY Paraxial height of chief ray same as HCY
Surface Data:	
so	Number of object surface, returns an integer value. Example: eva [so]
ss	Number of stop surface, returns an integer value. Example: eva [ss]
si	Number of image surface, returns an integer value. Example: eva [si]
THI sk [zk]	Thickness on surface sk, zoom position zk
THR sk [zk]	Reference thickness on surface sk
IMD [zk]	Image distance (THI si-1) at zoom position zk
IMC [zk]	Image clearance, the smaller distance (edge or axis) between surface i-1 and the image surface i.
IND sk wk	Index of refraction at surface sk, wavelength wk.
CUX sk [zk]	Curvature in X/Z plane
CUY sk [zk]	Curvature in Y/Z plane
RDX sk [zk]	Radius of curvature in X/Z plane
RDY sk [zk]	Radius of curvature in Y/Z plane
ADE sk [zk]	Tilt angle (in degree) around X-axis
BDE sk [zk]	Tilt angle (in degree) around Y-axis
CDE sk [zk]	Tilt angle (in degree) around Z-axis
XDE sk [zk]	X-decenter
YDE sk [zk]	Y-decenter
ZDE sk [zk]	Z-decenter
AADE sk [zk]	Tilt angle (in degree) of array cells around local X-axis
ABDE sk [zk]	Tilt angle (in degree) of array cells around local Y-axis
ACDE sk [zk]	Tilt angle (in degree) of array cells around local Z-axis
XSG sk [zk]	Global vertex X-coordinate of surface sk. Coordinates are **always** referred to the global system.
YSG sk [zk]	Global vertex Y-coordinate of surface sk. Coordinates are **always** referred to the global system.
ZSG sk [zk]	Global vertex Z-coordinate of surface sk. Coordinates are **always** referred to the global system.

continued on next page

continued from previous page

`XSC sk [zk]`	Global vertex X-coordinate of surface `sk`. Coordinates are referred to the coordinate system defined by the GLO sk command. Compare with XSG command above.
`YSC sk [zk]`	Global vertex Y-coordinate of surface `sk`. Coordinates are referred to the coordinate system defined by the GLO sk command. Compare with YSG command above.
`ZSC sk [zk]`	Global vertex Z-coordinate of surface `sk`. Coordinates are referred to the coordinate system defined by the GLO sk command. Compare with ZSG command above.
`CXG sk [zk]`	global X-direction cosine of surface normal
`CYG sk [zk]`	global Y-direction cosine of surface normal
`CZG sk [zk]`	global Z-direction cosine of surface normal
`A sk [zk]`	4^th order aspheric constant
`B sk [zk]`	6^th order aspheric constant
`C sk [zk]`	8^th order aspheric constant
`D sk [zk]`	10^th order aspheric constant
`E sk [zk]`	12^th order aspheric constant
`F sk [zk]`	14^th order aspheric constant
`G sk [zk]`	16^th order aspheric constant
`H sk [zk]`	18^th order aspheric constant
`K sk [zk]`	Conic constant
`SAG sk x_height y_height`	Surface sag at surface `sk`. `x_height` and `y_height` are the local coordinates at the tangent plane of surface `sk`.
`DEF`	Defocus
Arrays:	
`ARX sk`	Array surface X-spacing
`ARY sk`	Array surface Y-spacing
`ARXO sk`	Array surface X-offset of entity of array channels
`ARYO sk`	Array surface Y-offset of entity of array channels
`AMX sk`	$\pm$ limit for grid in X-direction
`AMY sk`	$\pm$ limit for grid in Y-direction
`AADE sk`	α-tilt angle (in degree) of each array cell.
`ABDE sk`	β-tilt angle (in degree) of each array cell.
`ACDE sk`	γ-tilt angle (in degree) of each array cell.
Grating/Hologram:	
`GRO sk`	Grating order
`GRX sk`	Grating frequency X (grooves per mm)
`GRY sk`	Grating frequency Y (grooves per mm)
`HWL sk`	Hologram design wavelength (in μm)
Materials Data:	
`GLA sk [zk]`	Returns string with glass name
`GL1 sk [zk]`	Returns string with glass name, equivalent to `GLA`
`GL2 sk [zk]`	Returns string with glass name on ”right” side of surface
`EXC sk [zk]`	Linear expansion coefficient $\cdot 10^6$
`DNO sk [zk]`	Offset on refractive index

continued on next page

continued from previous page

`DVO sk [zk]`	Offset on Abbe number (V-number)
`GADE sk [zk]`	Tilt of GRIN profile around X-axis
`GBDE sk [zk]`	Tilt of GRIN profile around Y-axis
`GCDE sk [zk]`	Tilt of GRIN profile around Z-axis
`GXDE sk [zk]`	X-decenter of GRIN profile
`GYDE sk [zk]`	Y-decenter of GRIN profile
`ABBE sk`	Abbe number at surface `sk`.
`DNDT sk wk [TEMP PRE]`	Absolute dndT at surface `sk`, wavelength `wk`. The absolute dndT is referred to vacuum (the default in $OpTaliX$ if temperature calculations are concerned). Optional parameters are Temperature `TEMP` (in °C) and pressure `PRE` (in mmHg). If not specified, `TEMP` defaults to 20°C and `PRE` to 760 mmHg.
`ADNDT sk wk [TEMP PRE]`	Absolute dndT at surface `sk`, wavelength `wk`. The absolute dndT is referred to vacuum. `ADNDT` is a complementary command to `DNDT` (see above).
`RDNDT sk wk [TEMP PRE]`	Relative dndT at surface `sk`, wavelength `wk`. `RDNDT` is referred to air. See also the relation between absolute dndT and relative dndT in section 13.2, page 216.
`EXC sk`	Linear expansion coefficient. Unit = $*10^6$. at surface `sk`.
`EXM sk`	Linear expansion coefficient of mirror substrate at surface `sk`. Unit = $*10^6$.
`SPG sk`	Specific gravity [g/cm^2] at surface `sk`.
`RHO sk`	Specific gravity, alternative command to `SPG`, at surface `sk`.
Apertures:	
`CIR sk pk [zk]`	Circular aperture radius of surface `sk`, pupil number `pk`, zoom position `zk`
`REX sk pk [zk]`	Rectangular aperture, X-extension
`REY sk pk [zk]`	Rectangular aperture, Y-extension
`ELX sk pk [zk]`	Elliptical aperture, half X-axis
`ELY sk pk [zk]`	Elliptical aperture, half Y-axis
`ADX sk pk [zk]`	Aperture decenter X, `pk` = pupil number
`ADY sk pk [zk]`	Aperture decenter Y, `pk` = pupil number
`ARO sk pk [zk]`	Aperture rotation (in degree)
`SD sk [fi..j] [zi..j]`	Maximum semi-diameter on surface sk. In absence of field and zoom qualifiers, value is calculated at all fields and zoom positions.
`WTA [zk]`	Weight on aperture (used in optimization only)
Environmental Data:	
`TEM sk [zk]`	Temperature (in $^\circ C$)
`PRE sk [zk]`	Pressure (in mm Hg)
Ray Data:	
`AOI sk fi zi wi [zk] rel_apeX rel_apeY`	Angle of incidence of a ray at surface `si`, field `fi`, zoom position `zi`, wavelength `wi`. The values `rel_apeX`, `rel_apeY` are the relative coordinates in the entrance pupil. The result is in degree. Note that all parameters are obligatory. Example: `aoi s3 f5 w1 0 1 < 15`.

continued on next page

continued from previous page

AOR sk fi zi wi [zk] rel_apeX rel_apeY	Angle of refraction (or reflection) of a ray with respect to the local surface normal. All parameters, surface sk, field fi, zoom position zi, wavelength wi are obligatory. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. The result is in degree. Example: aor s3 f5 w1 0 1 < 15.
AOE sk fi zi wi [zk] rel_apeX rel_apeY	Angle of exit of a ray with respect to the local surface normal. **Note that this command is synonymous the the AOR command** given above. All parameters, surface sk, field fi, zoom position zi, wavelength wi are obligatory. The values rel_apeX, rel_apeY are the relative coordinates in the entrance pupil. The result is in degree. Example: aoe s3 f5 w1 0 1 < 15.
X sk wk fk rx ry [zk] [gk]	X-intersection coordinate of ray on surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
Y sk wk fk rx ry [zk] [gk]	Y-intersection coordinate of ray on surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
Z sk wk fk rx ry [zk] [gk]	Z-intersection coordinate of ray on surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
XGR wi..j fk [zk]	X-coordinate of spot gravity center on the image surface for wavelength range wi..j, field fk
YGR wi..j fk [zk]	Y-coordinate of spot gravity center on the image surface for wavelength range wi..j, field fk
CX sk wk fk rx ry [zk] [gk]	X-direction cosine of ray on surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
CY sk wk fk rx ry [zk] [gk]	Y-direction cosine of ray on surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
CZ sk wk fk rx ry [zk] [gk]	Z-direction cosine of ray on surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
CXG sk wk fk rx ry [zk]	Global X-direction cosine of ray on surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
CYG sk wk fk rx ry [zk]	Global Y-direction cosine of ray on surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
CZG sk wk fk rx ry [zk]	Global Z-direction cosine of ray on surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
CXN sk wk fk rx ry [zk]	X-direction cosine of surface normal on intersection of ray at surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
CYN sk wk fk rx ry [zk]	Y-direction cosine of surface normal on intersection of ray at surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
CZN sk wk fk rx ry [zk]	Z-direction cosine of surface normal on intersection of ray at surface sk, wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry
NRAYS wk fk [zk]	Number of rays traced at wavelength wk, field fk, and optional zoom position zk.
Polarization Data:	

continued on next page

continued from previous page	
POLX fk wk sk rel_apeX rel_apeY	Polarization amplitude component X for a single ray at field fk, wavelength wk, surface sk.
POLY fk wk sk rel_apeX rel_apeY	Polarization amplitude component Y for a single ray at field fk, wavelength wk, surface sk.
POLP fk wk sk rel_apeX rel_apeY	Polarization phase (difference) for a single ray at field fk, wavelength wk, surface sk. The polarization phase is given in radians.
POLD fk wk sk rel_apeX rel_apeY	Degree of polarization for a single ray at field fk, wavelength wk, surface sk.
Tolerance/Sensitivity Data:	
TSF [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on test-plate fit. Assumes that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLF or a description of test plate fit on page 393. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLF 2.0 (fringes) for calculating tolerance sensitivity TSF.
TSI [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on surface irregularity. Assumes that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command IRR on page 389. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes IRR 0.4 (fringes) for calculating tolerance sensitivity TSI.
TST [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on surface thickness (distance). Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLT on page 389. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLT 0.02 for calculating tolerance sensitivity TST.
TSN [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on index of refraction. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLN, page 389, for defining index tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLN 0.001 for calculating tolerance sensitivity TSN.
TSV [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on dispersion. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLN, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLV 0.008 (0.8%) for calculating tolerance sensitivity TSV.
TSX [fk|fi..j wk|wi..j] sk|si..j	Tolerance sensitivity on X-decenter. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLX, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes DLX 0.02 (mm) for calculating tolerance sensitivity TSX.
	continued on next page

continued from previous page

`TSY [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on Y-decenter. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLY, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLY 0.02` (mm) for calculating tolerance sensitivity `TSY`.
`TSZ [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on Z-decenter. A Z-decenter is equivalent to a thickness tolerance. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLZ, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLZ 0.05` (mm) for calculating tolerance sensitivity `TSZ`.
`TSA [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on tilt about X-axis. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLA, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLA 5` (arcmin) for calculating tolerance sensitivity `TSA`.
`TSB [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on tilt about Y-axis. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLB, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLB 5` (arcmin) for calculating tolerance sensitivity `TSB`.
`TSG [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on tilt about Z-axis. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command DLG, page 389, for defining dispersion tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `DLG 5` (arcmin) for calculating tolerance sensitivity `TSG`.
`TSH [fk|fi..j wk|wi..j] sk|si..j`	Tolerance sensitivity on index homogeneity. Requires that a tolerance has been defined on the corresponding surface in the tolerance editor. See the command HOM, page 389, for defining homogeneity tolerances. If a tolerance on this parameter has not been defined in the tolerance editor, the program assumes `HOM 50` ($50 \cdot 10^{-6}$) for calculating tolerance sensitivity `TSH`.
Geometric Analyses:	
`SPD fk wk [zk]`	Spot diameter (rms)
`SPX fk wk [zk]`	Spot diameter (rms), only X-direction
`SPY fk wk [zk]`	Spot diameter (rms), only Y-direction
`SPDPV fk wk [zk]`	Spot diameter (PV)
`SPXPV fk wk [zk]`	Spot diameter (PV), in X-direction
`SPYPV fk wk [zk]`	Spot diameter (PV), in Y-direction
`LAC fk [wi..j] [zk]`	Lateral colour
`LAX fk wk [zk] ape_relX ape_relY`	Logitudinal aberration X

continued on next page

continued from previous page	
LAY fk wk [zk] ape_relX ape_relY	Logitudinal aberration Y
SSR [wi..j] [zi..j]	Secondary spectrum, weighted rms-value.
SPA [zk]	3^{rd} order spherical aberration
COMA [zk]	3^{rd} order coma
ASTI [zk]	3^{rd} order astigmatism
PETZ [zk]	3^{rd} order petzval sum (field curvature)
PTZ [zk]	synonymous to PETZ, 3^{rd} order petzval sum (field curvature), for Code V compatibility only.
DIST [zk]	3^{rd} order distortion
DST [zk]	synonymous to DIST, 3^{rd} order distortion, for Code V compatibility only.
LCA [zk]	3^{rd} order longitudinal colour
TCA [zk]	3^{rd} order transversal colour
AX [zk]	synonymous to TCA, 3^{rd} order longitudinal colour, for Code V compatibility only.
DISX fk [zk]	Distortion, X-direction
DISY fk [zk]	Distortion, Y-direction
FDISX fk [zk]	F-theta distortion, X-direction
FDISY fk [zk]	F-theta distortion, Y-direction
VIG [fk] [zk]	Vignetting factor relative to field 1. Values are returned between 0 (100% vignetting) and 1 (no vignetting).
ECG fk [wi..j] diam_x diam_y	Encircled energy (geometric) contained in image area X = diam_x, Y = diam_y
GMTFT [fk zk]	Tangential geometric MTF at field fk, zoom position zk.
GMTFS [fk zk]	Sagittal geometric MTF at field fk, zoom position zk.
GMTFA [fk zk]	Average geometric MTF at field fk, zoom position zk. GMTFA = 0.5(GMTFT + GMTFS)
ASTT fk wk rx ry [zk]	Tangential astigmatism along a single ray defined by wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry. Astigmatism is always measured at the image surface. If wk is omitted, the RMS value over all wavelengths is returned.
ASTS fk wk rx ry [zk]	Sagittal astigmatism along a single ray defined by wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry. Astigmatism is always measured at the image surface. If wk is omitted, the RMS value over all wavelengths is returned.
ASTD fk wk rx ry [zk]	Astigmatic difference along a single ray defined by wavelength wk, field fk, relative x-pupil rx, relative y-pupil ry. Astigmatism is always measured at the image surface. If wk is omitted, the RMS value over all wavelengths is returned.
Transmission Analyses:	
TRA fk wk sk zk pupil_X, pupil_Y	Mean (average) transmission along a single ray, defined at field fk, wavelength number wk, zoom position zk. The data pair (pupil_x, pupil_Y) defines the relative coordinates in the entrance aperture.
	continued on next page

continued from previous page

TRAS fk wk sk zk pupil_X, pupil_Y	S-pol transmission along a single ray, defined at field fk, wavelength number wk, zoom position zk. The data pair (pupil_x, pupil_Y) defines the relative coordinates in the entrance aperture.
TRAP fk wk sk zk pupil_X, pupil_Y	P-pol transmission along a single ray, defined at field fk, wavelength number wk, zoom position zk. The data pair (pupil_x, pupil_Y) defines the relative coordinates in the entrance aperture.
Diffraction Analyses:	
CEF [fk wk zk]	Fiber coupling efficiency
CEFDB [fk wk zk]	Fiber coupling efficiency in decibel
STREHL fk [wi..j] [zk]	Strehl ratio
DMD [fk|fi..j wk|wi..j] ape_x ape_y	Conrady D-d sum at field fk weighted over wavelengths wi..j. See also sect. 14.2.12.
MTF fk [wi..j] [zk]	Mean MTF = 0.5*MTF(Sag+Tan). Equivalent to the MTFA command (see below).
MTFA fk [wi..j] [zk]	Average (mean) MTF = 0.5*MTF(Sag+Tan)
MTFS fk [wi..j] [zk]	Sagittal MTF
MTFT fk [wi..j] [zk]	Tangential MTF
WAV fk wk [zk]	Wavefront aberration (rms)
WAVPV fk wk [zk]	Wavefront aberration, peak-to-valley (PV)
WAVZ fk wk [zk]	Wavefront aberration (rms), with selected Zernike terms subtracted. Define Zernike terms by the ZWACT command, see page 145.
PSDX fk [zk] [threshold]	PSF width-x at intensity-threshold at field fk.
PSDY fk [zk] [threshold]	PSF width-y at intensity-threshold at field fk.
PSE fk [zk] [threshold]	Ellipticity of PSF, ratio of PSDX/PSDY at intensity-threshold at field fk.
ECE fk diam	encircled energy within diameter (diam) at field fk.
EQE fk diam	ensquared energy within diameter (diam) at field fk.
KEFS fk	Knife Edge Function (KEF) in the sagittal orientation at field fk.
KEFT fk	Knife Edge Function (KEF) in the tangential orientation at field fk.
Gaussian Beams:	
WRX [sk]	Gaussian beam waist radius X (in mm) at surface sk
WRY [sk]	Gaussian beam waist radius Y (in mm) at surface sk
ZWX [sk]	Location of Gaussian beam waist X relative to surface sk
ZWY [sk]	Location of Gaussian beam waist Y relative to surface sk
RCX [sk]	Radius of X-curvature of Gaussian beam waist at surface sk
RCY [sk]	Radius of Y-curvature of Gaussian beam waist at surface sk
SRX [sk]	Spot size of Gaussian beam in X/Z-plane at surface sk
SRY [sk]	Spot size of Gaussian beam in Y/Z-plane at surface sk
GDX [sk]	Divergence of Gaussian beam in X/Z-plane at surface sk. Must have the Gaussian source parameters WRX, WRY, RCX, RCY properly set.

continued on next page

continued from previous page	
GDY [sk]	Divergence of Gaussian beam in Y/Z-plane at surface sk. Must have the Gaussian source parameters WRX, WRY, RCX, RCY properly set.
RRX [sk]	Rayleigh range of Gaussian beam in X/Z-plane at surface sk.
RRY [sk]	Rayleigh range of Gaussian beam in Y/Z-plane at surface sk.
Fiber Data:	
FSR [zk]	Source fiber mode field radius (in mm)
FSD [zk]	Source fiber far-field divergence (in rad)
FSA [zk]	Fiber source α-tilt in degree
FSB [zk]	Fiber source β-tilt in degree
FSN1 [zk]	Source fiber, index of refraction n_1 of core material
FSN2 [zk]	Source fiber, index of refraction n_2 of cladding material
FSCR [zk]	Source fiber, core radius in mm
FRR [zk]	Receiving fiber mode field radius (in mm)
FRD [zk]	Receiving fiber far-field divergence (in rad)
FRA [zk]	Receiving fiber α-tilt in degree
FRB [zk]	Receiving fiber β-tilt in degree
FRX [zk]	Receiving fiber x-offset (in mm) with respect to the chief ray
FRY [zk]	Receiving fiber y-offset (in mm) with respect to the chief ray
FRN1 [zk]	Receiving fiber, index of refraction n_1 of core material
FRN2 [zk]	Receiving fiber, index of refraction n_2 of cladding material
FRCR [zk]	Receiving fiber, core radius in mm
Illumination Source Data:	
SUSE sk 0|1	Use illumination source k (0=no, 1=yes). Example, enabling source 2: suse s2 1
SPWR sk pwr	Source power
SXEX sk x_extension	Source X-extension (full width)
SYEX sk y_extension	Source Y-extension (full width)
SXDE sk x_dec	Source X-decenter
SYDE sk y_dec	Source Y-decenter
SZDE sk z_dec	Source Z-decenter
SADE sk alpha	Source tilt (α) about X-axis
SBDE sk beta	Source tilt (β) about Y-axis
SCDE sk gamma	Source tilt (γ) about Z-axis
SARAY sk analysis_rays	Source: Number of analysis rays
SPRAY sk plot_rays	Source: Number of plot rays
SGREF sk O|G	Source reference: O = object, G = global
SDIVX sk x_div	Source divergence X (in degrees), full width
SDIVY sk y_div	Source divergence Y (in degrees), full width
SOFA sk x_offs	Source emittance angular offset in Y direction (in degrees)
SOFB sk y_offs	Source emittance angular offset in X direction (in degrees)
Illumination Analysis Data:	
RPWR	Return received power, including all activated sources
EPWR	Return emitted power, all activated sources
NILR	Return number of successfully received rays at target surface, including all active sources.

continued on next page

continued from previous page	
Miscellaneous Functions	
RAIS	Ray aiming maximum step relative to entrance pupil (default = 1).
RAIT	Ray aiming tolerance relative to entrance pupil (default = 0.001).
TIT	Returns 80 character string containing lens title.
COM sk	Returns the comment string for surface sk
DAT	Returns 12-character string with current date in the format DD MMMM JJJJ
TIM	Returns 8-character string with current time in the format HH:MM:SS
POX [zk]	Plot offset X in paper units
POY [zk]	Plot offset Y in paper units
POZ [zk]	Plot offset Z in paper units
WEI [si..j]	Weight (in grams)
SPG [sk]	Specific gravity (in g/cm^3)
PLANCK wavel T	Calculate radiance of a black body source according to Planck's law. wavel is the wavelength in μm, T is the temperature in Kelvin.
LADX [fk wk dlam]	Lateral dispersion in X-direction, given at the image surface. Preferably used in spectrometric systems. Calculates the spread of a wavelength interval dlam (in μm) at the image surface. Example: ladx f1 w2 0.01 calculates the spatial extension (spread) of the wavelength interval $\Delta\lambda = 0.01\mu m$ in the image. The resulting unit is $\mu m/mm$.
LADY [fk wk dlam]	Lateral dispersion in Y-direction, given at the image surface. Preferably used in spectrometric systems. Calculates the spread of a wavelength interval dlam (in μm) at the image surface. Example: lady f1 w2 0.01 calculates the spatial extension (spread) of the wavelength interval $\Delta\lambda = 0.01\mu m$ in the image. The resulting unit is $\mu m/mm$.
RAND [SEED num]	Random number. Optionally the seed can be set by "RAND SEED num", where num is any arbitrary number.

28

Colour Names

This chapter describes names of predefined colours in *OpTaliX* to be used in most graphical output. Currently colours can be separately defined for fields, coatings and encircled energy geometric (ECG). In later versions this will also be possible for wavelengths and zoom positions.

Colours for various plot/analysis types are specified by the `CLS` command. For a detailed description see the individual sections on page 45 (fields), page 372 (coatings).

Note that colour settings are preserved for a specific optical design. On loading (restoring) a new design, colours are set to their default values unless user-defined colours are specified in the new file.

28.1 Predefined colours

Predefined colours are designated by names. The first three characters are significant in specifying colour names.

	Short name	colour	RGB - value
	`RED`	red	`255,0,0`
	`GRE`	green	`0,255,0`
	`BLU`	blue	`0,0,255`
	`MAG`	magenta	`255,0,255`
	`CYA`	cyan	`0,255,255`
	`YEL`	yellow	`255,255,0`
	`BLA`	black	`0,0,0`
	`BRO`	brown	`185,92,0`
	`ORA`	orange	`255,128,0`
	`GRY`	grey	`192,192,192`
	`VIO`	violet	`192,128,255`
	`TUR`	turquoise	`0,194,194`
	`SAL`	salmon	`255,128,128`

28.2 Default Colours in Field Plots

The default sequence of colours for field is RED, GREEN, BLUE, MAGENTA, and CYAN. This sequence is repeated up to the last field for systems with more than 5 fields. Use the CLS FLD command (see page 45) to specify your own field colours.

28.3 Default Colours in Coating Analysis

Default colours used in coating analysis plots are RED GREEN BLUE.Use the CLS COA command (see page 372) to specify your own colours.

28.4 Default Colours in Encircled Energy Geometric (ECG) Analysis

Default colours used in encircled energy geometric (ECG) analysis are RED and GREEN.

29

Importing Lens and Coating Data

The following section describes how lens data from other design packages or from lens catalogues can be imported. Currently supported are optical design packages from CODE-V, ZEMAX, OSLO, MODAS, ATMOS, WinLens, as well as designs from standard catalogue lenses. It is, however, important to note that due to constant improvements in software development, only a subset of the individual design packages will be successfully translated. *OpTaliX* attempts to recognize a maximum amount of commands and features stored in external lens design files.

Import is accomplished by the generic "`IMP`" command with optional parameters.

29.1 Import of CODE-V Sequential Files

The import of CODE-V sequential files is accomplished by:

`imp seq|codev file_spec`	Import CODE-V sequential file from `file_spec`. Example: `imp seq c:/codev/dblgauss.seq`

29.2 Import of ZEMAX Files

From the command line:

`imp zmx|zemax file file_spec`	Import ZEMAX file from `file_spec`. The correct file extension `.ZMX` must be added Example: `imp zmx file c:/zmx_examples/dblgauss.zmx`

From the menu, select

FILE / IMPORT / ZEMAX which opens a file selection box.

29.3 Import of OSLO Files

From the command line :

`imp osl[o] file file_spec`	Import Oslo file from `file_spec`. The correct file extension `.LEN` must be added Example: `imp oslo file c:/oslo_examples/dblgauss.len`

from the menu, select:

FILE / IMPORT / OSLO which opens a file selection box.

29.4 Import of MODAS Files

MODAS (Modern Optical Design and Analysis Software) is an amateur program, written by Ivan Krastev.

`imp mod[as]as file file_spec`	Import Modas file from `file_spec`. The correct file extension `.dsg` must be added. Example: `imp modas file c:/modas_examples/cassegr.dsg`

from the menu, select:

FILE / IMPORT / MODAS which opens a file selection box.

Note on aspheric surfaces: MODAS uses an additional quadratic term A_2h^2 to the aspheric definition in Eq. 8.1 (page 67). This term describes a parabola, which is equivalently modeled by the conic constant $K = -1$. Since MODAS only allows either a pure conic surface or a higher-order asphere, but not both simultaneously, a simple relation for converting coefficients can be established:

$$c = 2 \cdot A_2 \tag{29.1}$$

Thus, on import MODAS aspheres, the conic constant K will be set to -1 (parabola) and the curvature is set to c. The inverse procedure is applied on export to MODAS.

29.5 Import of ATMOS Files

ATMOS is an amateur program, written by Massimo Riccardi, Italy.

`imp atm[os] file file_spec`	Import Atmos file from `file_spec`. The correct file extension `.atm` must be added Example: `imp atmos file c:/modas_examples/cassegr.atm`

from the menu, select:

FILE / IMPORT / ATMOS which opens a file selection box.

29.6 Import of WinLens Files

From the command line:

`imp winl[ens] file file_spec`	Import WinLens file from `file_spec`. The correct file extension `.spd` must be added Example: `imp winl file c:/examples/dblgauss.spd`

From the menu, select

FILE / IMPORT / WinLens which opens a file selection box.

29.7 Import of Accos Files

From the command line:

`imp acc[os]`	Import lens system in Accos format. This command opens a dialog box for selecting optical designs from library files. Accos stores lenses in lens libraries of roughly 2 Mbyte each. Each library may contain 98 lenses, called lens library blocks, plus a lens in working storage. Lenses have limits imposed in terms of number of surfaces, clear apertures etc.

From the menu, select
FILE / IMPORT / Accos which opens a file selection box.

29.8 Import of Sigma Files from Kidger-Optics

From the command line:

`imp sigma|sigmapc file file_spec`	Import Kidger-Optics Sigma file from `file_spec`. The following formats are supported
	Sigma-PC, which is identified by the file extension `.DAT`
	Sigma 2000, which is identified by file extension `.LEN`
	Examples:
	`imp sigma file c:/examples/dblgauss.len`
	`imp sigmapc file c:/examples/dblgauss.dat`

From the menu, select
FILE / IMPORT / Kidger Optics / Sigma which opens a file selection box.

29.9 Import Coatings from ”The Essential MacLeod” Thin-Film Package

From the command line:

`imp macl file file_spec`	Import coating design file in the ”Essential MacLeod” format from `file_spec`. Example: `imp macl file c:/ar_coat.dds`

From the menu, select
COATINGS / IMPORT / MacLeod which opens a file selection box.

29.10 Import Coatings from the ”TFCalc” Thin-Film Package

From the command line:

`imp tfc file file_spec`	Import coating design file in the ”TFCalc” format from `file_spec`. Example: `imp tfc file c:/ar_coat.dds`

From the menu, select
COATINGS / IMPORT / TFCalc which opens a file selection box.

29.11 Import Coatings from the ”Optilayer” Thin-Film Package

From the command line:

`imp opti file file_spec`	Import coating design file in the ”Optilayer” format from `file_spec`. Example: `imp opti file c:/ar_coat.ods`

From the menu, select

COATINGS / IMPORT / Optilayer which opens a file selection box.

29.12 Import from Lens Catalogs

$OpTaliX$ has the capability to read and extract lens systems from lens catalogues of various manufacturers and distributers (e.g. Melles Griot, Newport, Linos, etc).

From the main menu, extract a particular lens from a catalogue by

FILE / IMPORT / Catalogues, or
FILE / Catalog Lenses

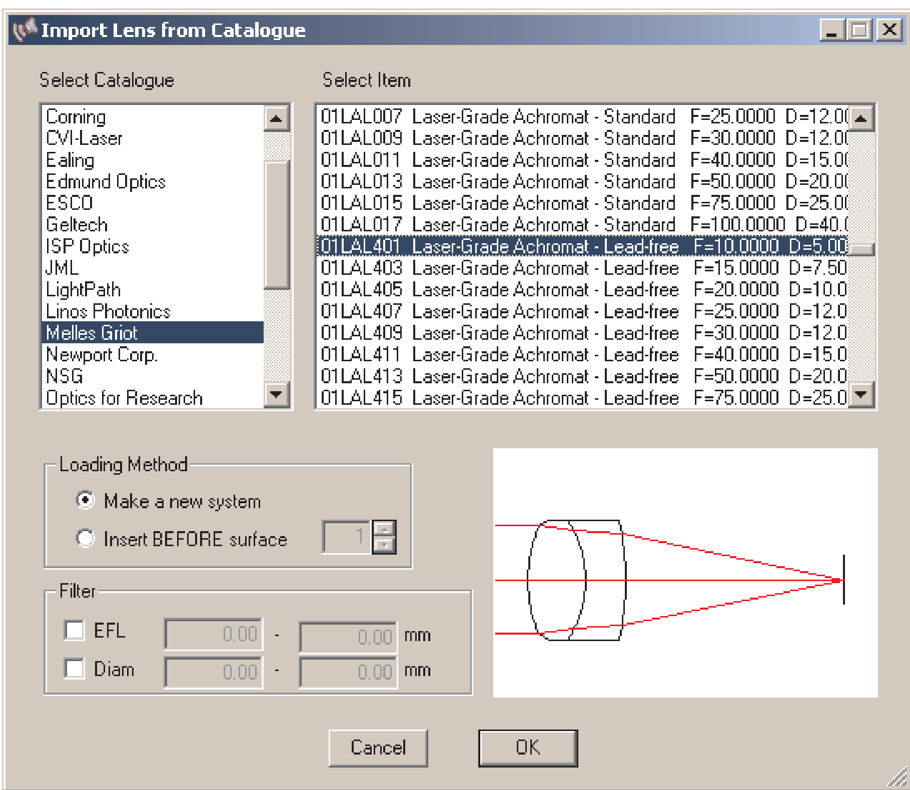


Figure 29.1: Dialog for selecting and importing lenses from vendor catalogs.

From the command line, extract a file from a catalogue by the command:

`imp cat [cat_ident code_string] [sk]`

The lens is identified by `code_string` in the catalogue described by `cat_ident`. If neither `cat_ident` nor `code_no` is specified at the command line, a dialog box is opened to select vendor and code number. If surface `sk` is provided, the system is inserted to the existing system **before** surface `sk`, otherwise a new system is built.

`cat_ident` is a short form of the vendor name, specify one of (only the first three respectively four characters are significant):

`ARCH`	Archer OpTx
`COHE`	Coherent Scientific
`CORN`	Corning
`CVI`	CVI-Laser
`EAL`	Ealing
`EDMU`	Edmund Optics
`ESCO`	Esco
`GELT`	Geltech
`ISP`	ISP-Optics
`JML`	JML
`LPT`	LightPath Inc.
`LINO`	Linos Photonics
`MELL`	Melles Griot
`NEWP`	Newport Corporation
`NSG`	Nippon Sheet Company
`OFR`	Optics for Research
`OPTO`	OptoSigma
`PHIL`	Philips
`QUAN`	Quantum
`ROLY`	Rolyn Optics
`ROSS`	Ross Optical
`SIGM`	Sigma-Koki, Japan
`SPEC`	Special Optics
`THOR`	ThorLabs
`3M`	3M Precision Optics

`Examples:`

`imp cat melles lpx027`
`imp cat mell lpx027`
`imp cat` ! invokes a dialog box
`imp cat linos 322286 s4` ! inserts Linos achromat before surface 4.

30

Exporting Lens Data

The following section describes how $OpTaliX$ lens data can be exported to other optical design packages. It is important to note that due to constant improvements in software development, only a subset of the options respectively commands provided by the individual design packages can be successfully translated. However, $OpTaliX$ attempts to recognize a maximum amount of commands and features provided by other packages. The capabilities of OpTaliX for converting features are constantly improved.

Export is accomplished by the generic ”`EXP`” command with additional parameters.

30.1 Export to Code V

From the command line :

`exp seq file file_spec`	Export to CODE-V sequential file. Example: `exp seq c:/temp/dblgauss.seq`
`wrl file_spec`	Writes lens data to Code V sequential (`.seq`) file.

From the menu, select : *FILE / EXPORT / CODE-V* which opens a file selection box.

30.2 Export to ZEMAX

From the command line:

`exp zmx file file_spec`	Export to Zemax file . The correct file extension `.ZMX` must be added Example: `exp zmx file c:/temp/dblgauss.zmx`

From the menu, select *FILE / EXPORT / ZEMAX* which opens a file selection box.

30.3 Export to OSLO

From the command line :

`exp osl|oslo file file_spec`	Export to Oslo file. The correct file extension `.LEN` must be added Example: `exp oslo file c:/temp/dblgauss.len`

All glasses used in the system are written to a private glass catalogue file in a format expected by OSLO. If required, the glasses contained in the file `\optalix\temp\oslo_private.glc` can be merged with the OSLO private catalogue using an ASCII text editor.

From the menu, select:

FILE / EXPORT / OSLO which opens a file selection box.

By default, $OpTaliX$ also exports glass data to a separate file being compatible with the OSLO private glass catalog. This file is found at `$i\temp\oslo_private.glc`. This feature is particularly useful for glasses not found in OSLO, for glasses with n, ν offsets and for exact transfer of fictitious glasses. These glasses may then copied/added to your OSLO private glass catalogue.

30.4 Export to ASAP

ASAP, optical modelling software, is a software package distributed by Breault Research Organization [5].

`exp asap file file_spec [RAY]`	Export to ASAP. The correct file extension `.INR` must be added. The file specification (path + file name) must be enclosed in quotes if `file_spec` contains blank characters or other special characters (-, &). The optional parameter `RAY` exports ray sets corresponding to the field points defined in the system. **Examples:** `exp asap fil c:/temp/dblgauss.inr` `exp asap file c:/temp/dblgauss.inr RAY !` exports rays as well `exp asap fil 'c:/temp/my-dbl gauss.inr' !` contains special characters

30.4.1 Exporting Special Surfaces to ASAP

Special surfaces which do not have an equivalent representation in ASAP must be modelled using the USERFUNC option. This requires definition of a user-function in the ASAP script.

If special surfaces exist in an optical system $OpTaliX$ adds appropriate commands to the exported ASAP script (`*.INR`). For example, an anamorphic surface (AAS) would be exported as

```
$READ BICONIC_FUNC.INR

USERFUNC EXPLICIT 0 0 0 BICONIC_FUNC 0.03125 0.031313 0.003 0.001,
         0.1E-06 0.0 0.0 0.0 0.0 0.0 0.0 0.0
```

where the corresponding function definition is provided with $OpTaliX$ and is found in the directory `$i\usersur\asap`. With the example given above you may wish to copy the "`BICONIC_FUNC.INR`" file to your ASAP working directory.

30.5 Export to MODAS

MODAS (Modern Optical Design and Analysis Software) is an amateur program, written by Ivan Krastev. From the command line :

exp mod|modas file file_spec	Export to Modas file format. The correct file extension .dsg must be added Example: exp modas file c:/temp/cassegr.dsg

from the menu, select:

FILE / EXPORT / MODAS which opens a file selection box. See also the notes in section 29.4 on exporting aspheres.

30.6 Export to ATMOS

ATMOS is an amateur program, written by Massimo Riccardi. From the command line :

exp atm|atmos file file_spec	Export to Atmos file format. The correct file extension .atm must be added Example: exp atmos file c:/temp/cassegr.atm

from the menu, select:

FILE / EXPORT / ATMOS which opens a file selection box.

30.7 Export of Wavefront to ABERRATOR

”Aberrator”[1] is a freeware program written by Cor Berrevoets, Netherlands, that generates star-testing images in order to show the effects of aberrations. It computes the diffraction PSF from the exported wavefront and displays it as a gray-coded bitmap, in a similar way as obtained in $OpTaliX$ via the PSF DF or PSF FF commands. At the command line enter :

exp wav [fi wi] file file_spec	Export wavefront to ”Aberrator” file format. The correct file extension .opd must be added Example: exp wav file c:/temp/wavefront.opd

from the menu, select:

FILE / EXPORT / Wavefront to Aberrator which opens a file selection box.

30.8 Export to Persistence of Vision (POV)

”Persistence of Vision” (POV) is a freeware general rendering and animation software which may be used to create almost photo-realistic images of the optical design.

From the command line:

exp pov file file_spec [ray]	Export to Persistence of Vision (POV) file . The correct file extension ”.POV” must be added. In absence of path information, the file will be stored in the current working directory. The optional parameter ray exports the user defined rays as defined by the SET FAN command. Example: exp pov file c:/pov_examples/dblgauss.pov

From the menu, select: *FILE / EXPORT /POV* which opens a file dialog box.

In order to write files in the POV-format, it is not required to have POV installed on the same machine. However, for testing purposes and to check whether the optical system has been successfully transferred, a working installation of POV is recommended. See also section 10.1, page 183 on how to interface *OpTaliX* with POV.

Note: A similar mechanism is used in the rendering option of the lens draw section (see `REN` command). The major difference is that the renderer (POV) is directly called.

30.9 Export to IGES

Exchanges optical surface models as 3D geometry to other computer-aided design (CAD) programs in the IGES 5.3 (Initial Graphics Exchange Specification) format. Exported models may include trimmed surfaces, rays, apertures and lens edges. A pure wire-frame option is also available.

`exp igs [sur ray wir ape edg all] [si..j|sk] [zk] [?] file file_spec`	Export optical system to IGES. The correct file extension `.igs` must be added. IGES output is controlled by the optional parameters:
	`sur` export surfaces (=default)
	`ray` export rays as defined in the VIE option
	`wir` export a wire-frame model (similar to 3D lens view)
	`ape` export aperture bounds
	`edg` export lens edges
	`all` export all (surfaces + rays + edges + wire-frame)
	Absence of any option defaults to `SUR`, for all surfaces, at zoom position 1. Examples:
	`exp igs sur ray file c:/temp/test.igs`
	`exp igs ape ?`

30.9.1 Illustration of IGES Export Options

This section illustrates the export options `SUR`, `RAY` and `WIR`. Note that the colour rendering may vary, depending on your preferred CAD system.

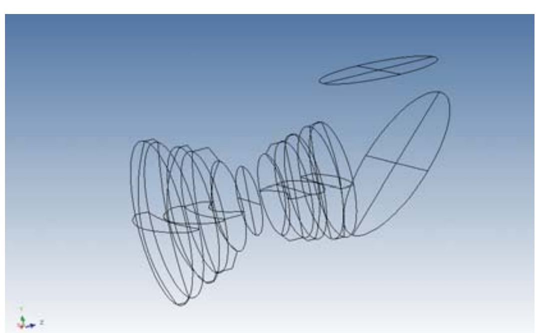


Figure 30.1: IGES export with wire frame only option (Command: '`exp igs wir`')

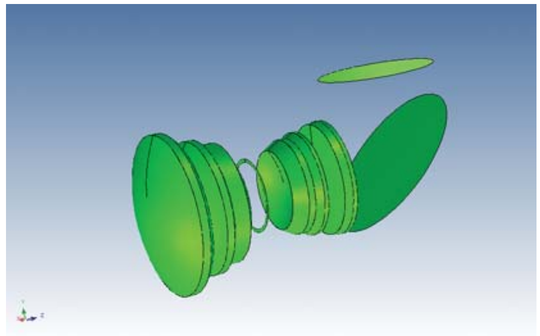


Figure 30.2: IGES export with surface only option (Command: ’`exp igs sur`’)

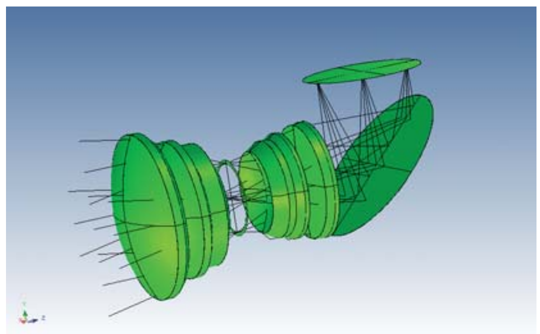


Figure 30.3: IGES export with surface and ray only options (Command: ’`exp igs sur ray`’, alternatively use `exp igs all`)

30.9.2 Supported IGES Entities

Entity Type Number	Description	Comment
102	Composite curve	
106	Copious data	Form number 12
108	Plane	
110	Line	
112	Parametric spline curve	
114	Parametric spline surface	
120	Surface of revolution	
124	Transformation Matrix	
128	Parametric B-Spline surface	In preparation
142	Curve on parametric surface	
144	Trimmed parametric surface	

30.9.3 IGES Export Limitations

$OpTaliX$ tries to export as many construction features as possible. However, not all properties could be supported in the current version.

- Non-rotationally symmetric surfaces (such as cylinders, toroids or free-form surfaces) are represented by a grid of curves, instead of a continuous parametric surface representation as in rotationally symmetric surfaces.

- Only circular and rectangular surface apertures are supported. Elliptical and polygon apertures will be added in future releases.
- Export of edges is not supported for elliptical or polygon apertures, and for decentered circular apertures.

30.9.4 IGES Trouble Shooting

Converting CAD data is a complex process. The quality of the translation depends on the diligence and understanding of the people involved, on both sides of the exchange.

IGES is a standard almost 20 years old, now in its sixth revision. Its successor is known as STEP (Standard for Exchange of Product information). After release 5.1, IGES was supposed to metamorphose gracefully into STEP 1.0. But it hasn't worked out that way. There are simply too many active IGES users and too few STEP users to shut IGES down completely. This is also the reason why $OpTaliX$ offers an IGES interface.

The major problem with IGES is that it mostly creates problems! At least it does not work perfectly, not for all people, and not all the time. A complete list of problems people encounter with 3D IGES files would fill a book, so let us identify the general categories of problems.

- The 'law' written into the IGES specification is subject to interpretation and it contains loopholes. Over the years, different brands of CAD companies have interpreted different parts of IGES in uniquely different ways, creating incompatibilities and "flavours".
- There is a large number of ways IGES data can be written. For example, users can export analytic surfaces such as cones and planes as spline surfaces before exporting. Some CAD systems would prefer the the analytic version, others the Spline representation. Also, a cubic spline may be presented as IGES entity 112 or 126 or even as a polyline of points (entity 106).
- Tolerances, accuracy, and resolution: The IGES problem this creates is when IGES files are moved between two CAD/CAM products using different accuracies. Moving a coarse toleranced IGES file to a fine toleranced system produces curves that don't close and surfaces that have gaps and overlaps. Moving a fine toleranced IGES to a coarse toleranced system loses detail for the opposite reason.
- Entity 108 (cubic spline) may not be supported by your preferred CAD system. This entity is often used (also by $OpTaliX$) for general (2D or non-rotationally symmetric) surfaces.
- Much trouble is caused with raw spline curve and surface geometry (entities 126 and 128).
- Pay special attention to trimmed surfaces (IGES entity 144). The trimming curves can be misplaced or are self intersecting.
- Be sure to look for curves or lines that extend beyond their required limits.
- In general, check if the entities written by $OpTaliX$ (see section30.9.2, page 483) are suported (recognized) by your CAD system.

30.10 Export to Microsoft^{TM} Excel File

Certain output data can be exported to a format compatible with Microsoft Excel^{TM}. This is not a general output switch (such that it would be available on *any* text output) because it is only available for a particular set of data which can be provided as gridded (or tabulated) data.

The ability to provide calculation data in Excel format is based on the installation of Microsoft's ODBC drivers. This requirement is fulfilled if Excel is installed on the target system. Alternatively, it is sufficient to install the "Microsoft Access Database Engine 2010 Redistributable" which may be downloaded from the Microsoft website free of charge, for example
`http://www.microsoft.com/en-us/download/details.aspx?displaylang=en&id=13255`.
Since export to Excel is based on the ODBC drivers, the export is also bound by the limitations inherent to the ODBC interface. These are namely,

- New data can only be added. It is not possible to address specific cells.
- Only data types NUMBER, DATETIME, TEXT, CURRENCY and LOGICAL are supported. It is not possible to transfer arithmetic equations or other formats.
- Text formatting (colour, font, etc.) is not possible.
- The maximum length of column names is limited to 63 characters.

Exported data from $OpTaliX$ is found in a sheet labelled "Data" as shown in the figure below (Fig. 30.4):

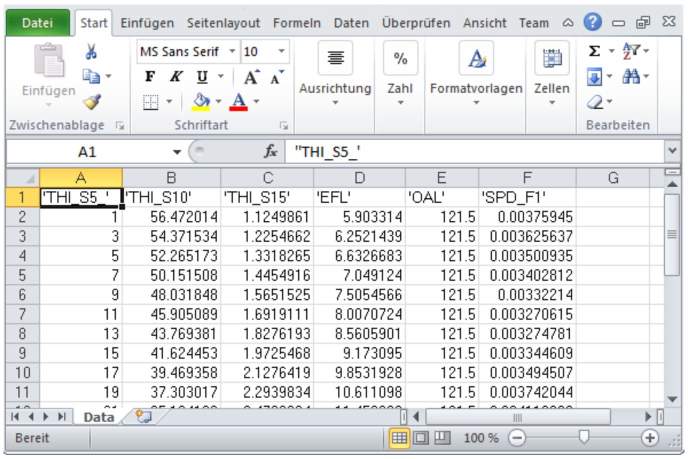


Figure 30.4: Example export to Excel^{TM} from a zoom CAM calculation.

31

Examples Library

$OpTaliX$ provides an extensive library of starting designs, comprising more than 500 designs from publications and patent literature. This also includes the complete libraries from Arthur Cox, Warren Smith, and from the Wiley ”Handbook of Optical Design, Vol 4”.

The example designs are stored during installation of $OpTaliX$ in the folder

`$i\examples`

In the program, the example files can be browsed from the command line

EXAMP	Invokes a dialog box for selecting various example designs.

or from the main menu

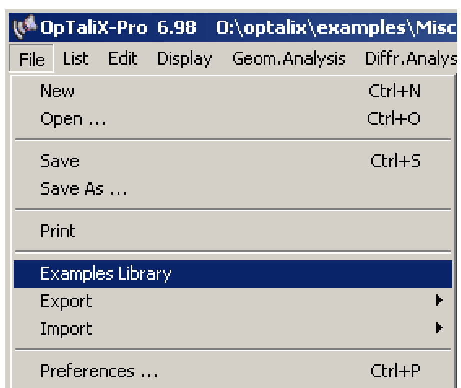


Figure 31.1: Menu entry for selecting the Examples Library

A typical dialog box is shown in Fig. 31. Select the design category and the design file in the tree-view to the left. Pressing OK loads the selected design. CANCEL resumes to the previously loaded design.

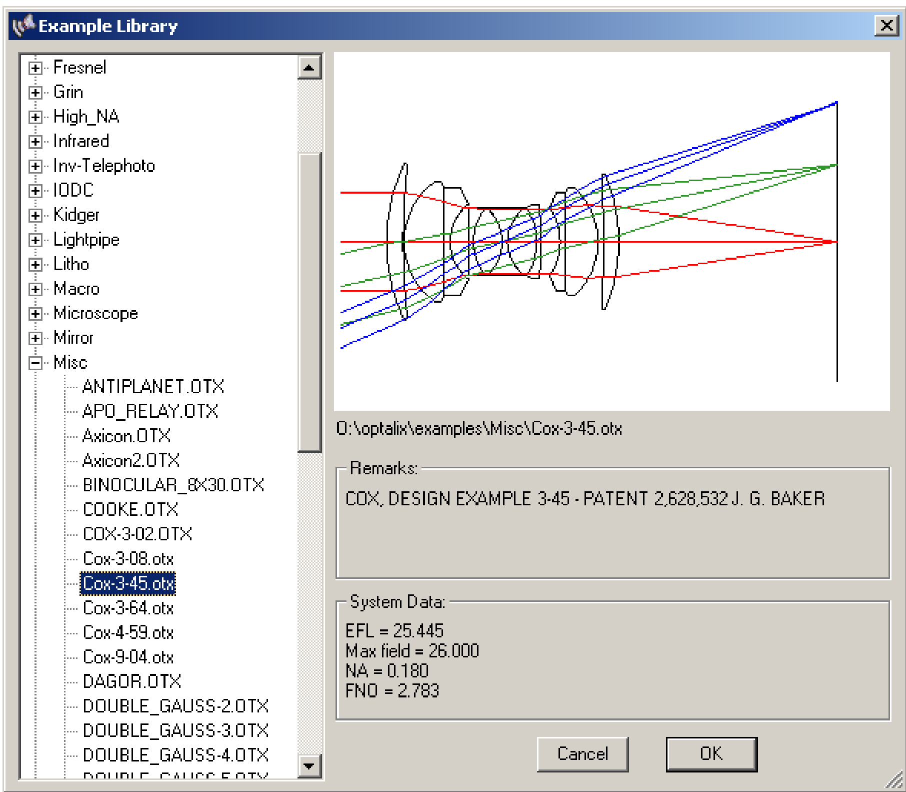


Figure 31.2: Selecting an example design from the library

32

File Formats

All files used or created by *OpTaliX* are plain ASCII files which may be edited by any text editor.

32.1 *OpTaliX* Configuration File "optix.cfg"

The *OpTaliX* configuration file "`optix.cfg`" stores a number of settings (mainly path information) which are used during each session. The file must reside in the *OpTaliX* installation (home) directory. The information is stored in free-form ASCII format and thus, may be read and edited by any text editor.

All entries are separated by at least one blank, multiple blanks as separator are permitted. The exclamation character "!" is identified as comment.

Qualifiers and parameters are separated by the equal " = " character. The qualifiers and its corresponding parameters are:

`RENDER = path_string`	Path to an external rendering program for generation of shaded perspective 3-dimensional views of the lens layout. To use this feature, the official version of the "Persistance of Vision" (POV) raytracer must be installed separately.
`HTML = path+exe_string`	Path to an external HTML browser. This path is mandatory to have access to the online help manual. This entry will be created during installation. Modify it if a different browser shall be used.
`GLASSES = path_string`	Path to glass catalogues. This entry is commented by default and should not be modified (except if you exactly know what you are doing).
`COATINGS = path_string`	Path to coatings files.
`TEMP = path_string`	Path to temporary working directory
`MACRO = path_string`	Path to macro files and user defined graphics definitions.
`SAVDEFAULTONEXIT = int`	Save the current system on program exit. `int` is an integer number. 0 = don't save, 1 = save.
`SAVWINONEXIT = int`	Save window settings (position, size) on program exit, 0=no, 1=yes
`TEXTFOREGR = int`	Put text output window to foreground each time new output is generated, 0=no, 1=yes

An example of an *OpTaliX* configuration file is:

```
! Optix configuration file
! Entries must be separated at least by one blank character
```

```
! Characters are case insensitive
! Path names containing blanks must be enclosed in quote character (")
!
HTML =
RENDER = "f:\pov31a\bin\pvengine.exe"
!
! Uncomment and edit the following lines only if you wish a
! different search path for glasses,coatings or temp.
!
! GLASSES =  "e:\optix\GLASSES\"
! COATINGS = "e:\optix\coatings\"
! TEMP   =   "e:\optix\temp\"
```

As can be seen from the example above, some qualifiers (GLASSES, COATINGS, ..) are commented. The default paths are used instead (i.e. below the $OpTaliX$ installation directory).

32.2 Lens Prescription Format ".otx"

The lens data are stored in standard unformatted ASCII file with the extension "`.otx`". In each line, the lens prescription parameters are identified by a keyword. All entries are separated at least by one blank, multiple blanks as separator are permitted. The exclamation character "!" is identified as comment.

The keywords and the possible (allowed) parameters are described in alphabetical order in the following table. The type of the variables is indicated by "`int`" for an integer value, "`real_val`" for a real value and "`char`" for a character string.

`AAP int`	Asymmetric aperture (for lens cross sectional plot only) `int = 0` : full surface aperture is plotted `int = 1` : only the section used by the light beam is plotted
`ADE real_val`	Surface tilt around X-axis, in degree
`AFO int`	Afocal switch, `int = 1` : system is afocal.
`APE int val1 val2 val3 val4 val5 int2 int3 int4`	Aperture definition int = pupil number (default = 1) val1 = semi aperture in X val2 = semi aperture in Y val3 = X-offset of aperture from surface vertex val4 = Y-offset of aperture from surface vertex val5 = rotation angle (in degree) int2 = pupil type (1=circular, 2=rectangular, 3=elliptical, 4=polygon) int3 = logical operator (0=base pupil, 1= logical and, 2=logical or) int4 = transmission properties (0=inside, 1=obstruct, 2=hole)
`APEC int val1 int2 int3`	Circular aperture int = pupil number (default = 1) val1 = semi aperture in Y int2 = logical operator (0=base pupil, 1= logical and, 2=logical or) int3 = transmission properties (0=inside, 1=obstruct, 2=hole)

continued on next page

continued from previous page	
`AFR real_val`	Autofocus spatial frequency in line pairs. This is the spatial frequency, at which the MTF-autofocus is determined.
`ASP val1 val2 .... val7`	Aspheric coefficients, val1 = conic constant val2 ... val7 = polynomial coefficients
`ARX real_val`	Array X-spacing of channels
`ARY real_val`	Array Y-spacing of channels
`ARXO real_val`	Array X-offset
`ARYO real_val`	Array Y-offset
`AXG real_val`	Tolerance: axial linear gradient
`BDE real_val`	Surface tilt around Y-axis, in degree
`BIR val1 ... val11`	Refractive index of birefringent material
`CDE real_val`	Surface tilt around Z-axis, in degree
`COA string`	File name of coating, attached to current surface
`COM string`	Comment per surface
`CON string`	Optimization constraints
`CTV icoeff real_val`	Coating thickness variation coefficient, `icoeff` is the coefficient number between 1 and 5, `real` contains the coefficient.
`CUX real_val`	X-curvature
`CUY real_val`	Y-curvature
`DEF real_val`	Defocus of real image plane from paraxial focus
`DLA real_val`	Tolerance: alpha tilt (about X-axis)
`DLB real_val`	Tolerance: beta tilt (about Y-axis)
`DLG real_val`	Tolerance: gamma tilt (about Z-axis)
`DLF real_val`	Tolerance: Test plate fit in fringes
`DLN real_val`	Tolerance: index of refraction
`DLR real_val`	Tolerance: absolute radius in mm
`DLT real_val`	Tolerance: axial thickness in mm
`DLV real_val`	Tolerance: dispersion (Abbe number) in %
`DLX real_val`	Tolerance: X-decenter
`DLY real_val`	Tolerance: Y-decenter
`DLZ real_val`	Tolerance: Z-decenter
`DTR real_val`	Tolerance: reference thickness in mm
`DNO real_val`	Δn - Offset
`DVO real_val`	$\Delta \nu$ - Offset
`EPD real_val`	Entrance pupil diameter
`EXC real_val`	Linear expansion coefficient in 10^{-6} units
`FACT i_active1 i_active2 ...`	Field activation. A particular field point may be excluded from analysis, i.e. it is not active. `i_active` is an integer number (0 = inactive, 1 = active) and counts from 1 to the maximum number of fields (defined by `FLDX` and `FLDY`)
`FH int`	Fixed aperture height, int=0 : aperture does not limit/truncate light beam int = 1 : aperture defines/truncates light beam
`FIBS string`	Specify source fiber by product (e.g. by manufacturers type number).
	continued on next page

continued from previous page	
FIBR string	Specify receiving fiber by product (e.g. by manufacturers type number).
FILE string	File name (optional)
FNO real_val	F-Number
FLDX val1 ... val11	Field coordinate in X.
FLDY val1 ... val11	Field coordinate in Y.
FLD int x_field y_field weight active	Alternative form of specifying field points. Use either FLDX/FLDY or FLD entry. int = field number x_field = X-field coordinate, meaning depends on FTYP y_field = Y-field coordinate, meaning depends on FTYP weight = field weight active = 0/1, defines whether field point is used in analysis.
FRES val1 val2	Fresnel parameter val1 = X-tilt of fresnel facets val2 = Y-tilt of fresnel facets
FRA alpha_tilt	Receiving fiber α-tilt in degree.
FRB beta_tilt	Receiving fiber β-tilt in degree.
FRD real_val	Far-field divergence of receiving fiber (in rad).
FRN1 real_val	Receiving fiber, index of refraction n_1 of core material
FRN2 real_val	Receiving fiber, index of refraction n_2 of cladding material
FRCR real_val	Receiving fiber, core radius in mm.
FRR mode_radius	Receiving fiber, mode-field radius in mm.
FRX x-offset	Receiving fiber, x-offset (in mm).
FRY y-offset	Receiving fiber, y-offset (in mm).
FSA alpha_tilt	Fiber source α-tilt in degree.
FSB beta_tilt	Fiber source β-tilt in degree.
FSD div_x div_y	Far-field fiber source divergence (in radians) in X- and Y-direction.
FSN1 real_val	Source fiber, index of refraction n_1 of core material
FSN2 real_val	Source fiber, index of refraction n_2 of cladding material
FSCR real_val	Source fiber, core radius in mm.
FSR rad_x rad_y	Fiber source radius in X- and Y-direction (in mm).
FTH f_thick	Fresnel thickness
FTYP int	Field type int = 1 : Field coordinates are defined by field angle int = 2 : fields are defined by object coordinates int = 3 : fields are defined by paraxial image coordinates int = 4 : fields are defined by real image coordinates
FWGT int1 ... int10	Field weights
GIC val1 ... val50	Gradient index coefficients. The number of coefficients is defined by NGIC.
GIS real_val	Gradient index step, the integration distance in gradient index material
GIT string	Gradient index type (e.g. SEL, AXG, LPT, URN,...)
GLA string	Glass name (up to 10 characters)
	continued on next page

continued from previous page	
GL1 string	Glass name, defines material left to surface (only applicable for NSS)
GL2 string	Glass name, defines material right to surface (only applicable for NSS)
GRO real_val	Grating order
GRX real_val	Grating constant in X-direction, applicable only for a straight-line ruled grating
GRY real_val	Grating constant in Y-direction, applicable only for a straight-line ruled grating
GTILT val1 ... val6	Gradient profile tilt/decenter val1 ... val3 : X,Y and Z decenter of gradient profile val4 ... val6 : α, β, γ - tilts around X-, Y-, and Z-axis respectively
GZO real_val	Gradient Z-Offset of profile definition from surface vertex (applicable only for axial profiles from LightPath).
HWL real_val	Hologram design wavelength, in microns
HCO icoeff real_val	Hologram coefficient, icoeff is the coefficient number between 1 and 28.
HOM real_val	Tolerance: index homogeneity
HOR order	Hologram diffraction order
HOT int	Hologram type, int = 0 for a straight-line ruled grating, 1 for a symmetrical phase function, 2 for an asymmetrical (2d) phase function
HX1 obj_source_x	X-coordinate of object point source for holographic surface.
HY1 obj_source_y	Y-coordinate of object point source for holographic surface.
HZ1 obj_source_z	Z-coordinate of object point source for holographic surface.
HX2 ref_source_x	X-coordinate of reference point source for holographic surface.
HY2 ref_source_y	Y-coordinate of reference point source for holographic surface.
HZ2 ref_source_y	Z-coordinate of reference point source for holographic surface.
IRR real_val	Tolerance: irregularity in fringes
KLDR	For internal use only, not required (controls plot appearance)
LINK int1 int2 int3 int4	Link(pickup) surface (curvature, thickness,tilt,material)
LMOD val1 ... val5	Lens module (ideal lens) val1 = focal length val2 ... val5 : not yet defined
M2 val	quality factor M^2
MFR real_val	Maximum spatial frequency (for MTF calculation)
MPRS string	Mode profile, source. "string" may be any of GAU for Gaussian mode profile, STE for step-index, FIL for user defined profile loaded from file.
MPRR string	Mode profile, receiver. "string" may be any of GAU for Gaussian mode profile, STE for step-index, FIL for user defined profile loaded from file.
MXH int	Maximum hits (of rays at a non-sequential surface).
NA real_val	Numerical aperture, in image space
NAO real_val	Numerical aperture, in object space
	continued on next page

continued from previous page

`NGIC int`	Number of GRIN-coefficients
`NSS int`	Non-sequential surface int = 0 : sequential, int = 1 : NSS-surface
`NTOF int`	Number of tolerance functions.
`OSP spectrum_name`	Optical spectrum. The spectrum names are defined in the file `osp.dat`
`PCO real_val`	Partial dispersion P(C,s)-Offset
`PGO real_val`	Partial dispersion P(g,F)-Offset
`PLSC ...`	For internal use only. (Plot scaling)
`POL int`	Polarization switch int = 0 : polarization is ignored int = 1 : polarization is taken into account.
`POL1 val_x val_y val_ph`	Polarization state of input wave 1 val_x = X-amplitude val_y = Y-amplitude val_ph = Phase
`POL2 val_x val_y val_ph`	Polarization state of input wave 2 val_x = X-amplitude val_y = Y-amplitude val_ph = Phase
`PRI val1 ... val11`	Private glass. val1 ... val11 are the indices of refraction at the wavelengths defined in WL.
`PRE real_val`	Pressure in mmHg
`PUI real_val`	Pupil intensity (to be used in combination with PUX, PUY).
`PUX real_val`	Relative X-coordinate (refered to entrance pupil radius) for PUI value
`PUY real_val`	Relative Y-coordinate (refered to entrance pupil radius) for PUI value
`RAG real_val`	Tolerance: radial quadratic gradient
`RAY string val1 ... val5`	User defined ray coordinates at entrance pupil. string = ray type val1 = X-coordinate of ray val2 = Y-coordinate val3 ... val5 = X, Y, Z direction cosines
`RAIM int`	Ray aiming method int = 0 : rays are aimed to paraxial entrance pupil (no iteration) int = 1 : rays are aimed to real stop, iteration is performed. int = 2 : telecentric ray aiming
`RAIT real_val`	Ray aiming tolerance. The tolerance (in mm) during ray iteration to the real stop surface.
`RCX val`	Radius of curvature of wavefront at object plane in x-direction
`RCY val`	Radius of curvature of wavefront at object plane in y-direction
`REF int`	Reference wavelength number
`REM int string`	Remarks, ”int” is the surface number, ”string” containes the remark text (up to 80 characters)

continued on next page

continued from previous page	
`SREF iref val1 ... val7`	Surface reference `iref` : reference surface `val1` : reference thickness (THR) `val2 ... val4` : X,Y and Z decenter wrt. reference surface `iref` `val5 ... val7` : α, β, γ - tilts around X-, Y-, and Z-axis respectively
`SPLR icoeff rad z_deform`	Radial spline deformation. `icoeff` is the running number of the deformation point, `rad` is the radial component, `z_deform` is the deformation (in mm).
`SUR int`	Surface identifier. Increments the surface counter.
`SUT string`	Surface type
`STO`	Surface is aperture stop
`TEM real_val`	Temperature in degree Celsius
`TGR int`	Transformation grid size
`THI real_val`	Thickness (axial separation) to next surface.
`TILT val1 ... val6`	Surface tilt/decenter val1 ... val3 : X,Y and Z decenter val4 ... val6 : α, β, γ - tilts around X-, Y-, and Z-axis respectively
`TLM int`	Tilt mode
`TOLC fkn_tol string`	`fkn_tol` = limit on tolerance criterium, `string` = Tolerance criterium string
`TOCM int`	Tolerance compensation method. int = 0 : no compensator int = 1 : back focus int = 2 : use setting in optimization.
`TOPM int`	Compute plus/minus tolerance sensitivity (0 = no, 1 = yes).
`TRA int`	Transmission switch int = 0 : transmission is ignored int = 1 : transmission is taken into account.
`VERS real_val`	Version number
`VAR ...`	Optimization variables
`VARZ ...`	Zoom variables for optimization
`WL val1 ... val11`	Wavelengths in micron.
`WRX` val	Waist radius in X-direction, given in mm.
`WRY` val	Waist radius in Y-direction, given in mm.
`WTW int1 ... int11`	Wavelength weight, integer numbers between 0 and 100
`XDE real_val`	Surface X-Decenter
`YDE real_val`	Surface Y-Decenter
`ZDE real_val`	Surface Z-Decenter
`ZOO ....`	Zoom parameter string
`ZPOS int`	Number of zoom positions
`ZRN val1 ... val40`	Zernike coefficients
`ZWX val`	Location of beam waist relative to object plane in x-direction
`ZWY val`	Location of beam waist relative to object plane in y-direction

32.3 Multilayer File Format ".otc"

Multilayer coatings are typically stored in the directory `$i/coatings` where `$i` is the installation directory (i.e. where the *OpTaliX* executable resides). It is, however, possible to specify a different coatings directory by modification of the COATING entry in the "`optix.cfg`" file.

The coating prescription is stored in standard unformatted ASCII file with the extension "`.OTC`". In each line, the coating parameter is identified by a keyword. The keywords and the allowed parameters are described as follows:

`VERS`	Version number of *OpTaliX* which created the coating file.
`COM string`	Comment string, enclosed in quotation marks, e.g. `COM "AR-Coating for visible"`. The comment string may be up to 256 characters.
`NLY real_val`	Number of layers (excluding top and bottom medium (typically air and substrate)
`LAM0 real_val`	reference wavelength, in microns
`LAM1 real_val`	minimum wavelength, needed for plotting purposes only
`LAM2 real_val`	maximum wavelength, needed for plotting purposes only
`TSMIN TSMAX`	Minimum and maximum of transmission plot range. The parameter is between 0 and 1. Required for plotting purposes only.
`RSMIN RSMAX`	Minimum and maximum of reflection plot range. The parameter is between 0 and 1. Required for plotting purposes only.
`ANGLE real_val`	Incidence angle (in degree). Required for plotting purposes only.
`PLOT_S int_val`	Plot the S-component. 0 = no, 1 = yes.
`PLOT_T int_val`	Plot the T-component. 0 = no, 1 = yes.
`PLOT_A int_val`	Plot the A-component (average). 0 = no, 1 = yes.
`LOG int_val`	Select logarithmic display (0=no, 1=yes). Use in conjunction with `FLOOR`.
`FLOOR real_val`	Floor for logarithmic display. For example `FLOOR -3.0` defines 0.001 as the lowest value displayed in plots.
`SHOWTARG int_val`	Show refinement targets in transmission/reflection plots (0=no, 1=yes).
`PLOT_COL col1 col2 col3`	Defines colours of curves in transmission/reflection plots, for S-, T- and Average components. The colour numbers are integer values and are calculated in a 24-bit RGB colour space as red + green*256 + blue*256**2.
`LAY`	Layer number. Increments the layer. Numbering starts with the incident medium (layer 1) and ends with the substrate (NLY + 2).
`GLA`	The layer "glass" (material name). A character string up to 64 characters is accepted. Blank characters and control characters (carriage return, end-of-file, tab, etc.) are not allowed. The glass name may be any of the standard catalogue glasses (e.g. BK7). If not specified, i.e. the glass name is empty (blank characters), the refractive index as defined in the IND command will be used instead. A glass (material) name is mandatory if dispersion shall be taken into account.
`OTH`	Optical thickness, in wavelength units defined at the reference wavelength `LAM0`. `OTH` is interpreted in the normal direction to the stratified layer.

continued on next page

continued from previous page	
PTH	Physical thickness, in mm. This is an optional parameter, as the thickness of a layer is primarily defined by the optical thickness. Only in case optical thickness (OTH) is not specified in the input file, optical thickness is calculated from the physical thickness (PTH).
IND [layer_num]	Complex refractive index. This index will be used for all wavelengths, that is, material dispersion effects are ignored unless a glass is specified for this layer. `layer_num` is optional, because the key word `LAY` alone will increment numbering of the layers. `layer_num` is only written for better readability of the coating prescription file (*.otc).
PICKUP i_mat i_thi	Picks material and/or thickness properties from a previous layer. `i_mat` is the layer number for material pickups, `i_thi` is the layer number for thickness pickups.
PFAC real_val	Packing factor describing layer density. A value between 0 and 1. Currently not used.

Normally, thin-film layer materials are defined in the catalogue files `coat.asc` (for pre-defined catalogue materials) and `coatp.asc` (privately defined layer materials). The definition of layer materials may be embedded with the coating (multi-layer) prescription file `*.otc`. The syntax for describing layer material properties within the coating file is defined in the following table32.4:

Table 32.4: **Embedding layer materials in coating files:**

BEGIN MATERIAL mat_name DATA lam n k DATA lam n k ... END MATERIAL	The environment BEGIN MATERIAL / END MATERIAL defines material properties as part of the coating prescription, i.e. material properties (n,k) are embedded in the the coating file (`*.otc`). The material name `mat_name` is a string of max. 64 characters wide. The `DATA` statement describes the triple (`lam`, `n`, `k`), where `lam` is the wavelength (in μm), and (`n`, `k`) is the the complex index of refraction.

Example Coating File:

```
VERS =  2.82
COM = "Antireflection coating for visible range"
NLY =     4
LAM0 =  .5460000
LAM1 =  .4000000
LAM2 =  .8000000
TSMAX = .0000000e+00
TSMIN = .0000000e+00
RSMAX = .5000000e-01
RSMIN = .0000000e+00
ANGLE = .0000000e+00
PLOT_S = 1
PLOT_T = 1
PLOT_A = 1
```

```
LAY =   1
   GLA =
   OTH =  0.00000000e+00
   PTH =  0.00000000e+00
   IND =   1.0000000     0.00000000e+00
LAY =   2
   GLA =  mgf2
   OTH =  0.24819737
   PTH =  0.98300005e-04
   IND =   1.3785938     0.00000000e+00
LAY =   3
   GLA =
   OTH =  0.50558242
   PTH =  0.12960001e-03
   IND =   2.1300000     0.00000000e+00
LAY =   4
   GLA =
   OTH =  0.20545055
   PTH =  0.68400003e-04
   IND =   1.6400000     0.00000000e+00
!
BEGIN MATERIAL NewMat
   DATA 0.45 1.50 0.0001
   DATA 0.55 1.48 0.0002
   DATA 0.65 1.46 0.0003
END MATERIAL
```

Note:
Keywords and parameters may be separated by an equal sign "=". The separator for multiple parameters in a single line can be a comma "," or at least one blank character. $OpTaliX$ correctly interprets formats like:

```
IND 1.521 0.0d0
IND = 1.521 0.0d0
IND = 1.521,0.0d0
```

32.4 Zernike Deformation File Format ".zrn"

Reading Zernike coefficients from a file is rather straightforward. The coefficients are stored in a free formatted ASCII file where each line contains the number of the coefficient and the coefficient itself:

`coeff_no coefficient`

The entries are separated by at least one blank, multiple blanks as separator are permitted. The exclamation character "!" is identified as comment. An example of a valid Zernike coefficient file is

```
! Zernike coefficients at surface 1
     ! here follows more descriptive text
1   0.0003
3 1.743E-5
14        0.1        ! this is coefficient no. 14
   16 -2.345d-12
! end of Zernickes
```

Coefficients for different surfaces must be stored in different files. The standard file naming convention is the 8.3 DOS standard. Longer file names must be enclosed in parenthesis, e.g.

`"this is my file.txt"`

32.5 Radial Spline Deformation File Format

Reading radial Spline deformation coefficients from a file is rather straightforward. The coefficients are stored in a free formatted ASCII file where each line contains two real numbers:

`radial_distance deformation`

where :

radial_distance	is the distance in radial direction of the sample point,
deformation	is the deformation at the sample point with respect to the base surface.

The entries are all separated by at least one blank, multiple blanks as separator are permitted. The exclamation character "!" is identified as comment. As an example, a valid Spline deformation file is

```
! Spline deformation at surface 1
     ! here follows more descriptive text
1.234   0.0003
3.5 1.743E-5
4.56       0.1        ! deformation is +0.1mm at 4.56mm radial height
   5.9 -2.345d-12
! end of deformations
```

Coefficients for different surfaces must be stored in different files. The standard file naming convention is the 8.3 DOS standard. Longer file names must be enclosed in parenthesis, e.g.

`"this is my file.txt"`

32.6 Test Plate File Format ".tpl"

Test plate lists (`TPL`) are stored in unformatted ASCII files. Each test plate radius is stored in a single line which contains four entries:

```
plate_ID     RADIUS    MAX_DIAM     CVCX
```

where:

PLATE_ID	A unique identification string
RADIUS	Radius of curvature (in mm)
MAX_DIAM	Maximum test plate diameter
CVCX	Availability of test plate: -1 = only concave radius available 0 = convex and concave radius available 1 = only convex radius available

All entries are separated by at least one blank character. Comment lines in a `TPL` file begin with an "!" (exclamation mark). Each entry is separated by at least one blank character. Tabs are allowed and are interpreted as a single blank character. There is no limit on the number of comment lines.

The first lines of a valid test plate file are:

```
! My Company Inc.
!
10000-1 1.00000 1.96 0
14330-1 1.43220 2.81 0
15679-1 1.56800 3.07 0
20833-1 2.08320 4.08 0
21288-1 2.12880 4.17 0
```

32.7 Glass Catalogue File Format ".csv"

Optical glasses from vendor catalogues are stored in ASCII files (hence the extension .asc) which can be read and modified by any text editor that handles ASCII files properly, such as NOTEPAD. We explicitly discourage use of Windows-Word for editing glass catalogues.

Data for each glass type are stored in a single line where the parameters are separated by commas "," or semi-colons ";".

The first line is obligatory and must contain the string "!GLASSV2" as the first characters. The rest of the line is not significant.

The second line is obligatory and may contain any arbitrary text. Note that the first and second line are not used in reading glass data.

The third line and all subsequent line lines contain glass parameters, one line for each glass type.

The glass catalogue file is ended by an empty line followed by a carriage return (CR) and line feed (LF) character.

Example file:

```
!GLASSV2
!Manufact.,Name      ,EqName    ,Code   ,B1             ,B2
SCHOTT    ,N-BAF3    ,S-BAM3    , 583466,1.34859634E+00,1.07644240E-01, ....
SCHOTT    ,N-BAF4    ,S-BAM4    , 606437,1.42056328E+00,1.02721269E-01, ....
SCHOTT    ,N-BAF10   ,S-BAH10   , 670471, 1.5851495E+00, 1.4355939E-01, ....
SCHOTT    ,N-BAF51   ,N-BAF51   , 652450,1.51503623E+00,1.53621958E-01, ....
SCHOTT    ,N-BAF52   ,N-BAF52   , 609466,1.43903433E+00,9.67046052E-02, ....
SCHOTT    ,N-BAK1    ,S-BAL11   , 573576, 1.1236566E+00, 3.0927685E-01, ....
SCHOTT    ,N-BAK2    ,S-BAL12   , 540597, 1.0166215E+00, 3.1990305E-01, ....
SCHOTT    ,N-BAK4    ,S-BAL14   , 569560,1.28834642E+00,1.32817724E-01, ....
SCHOTT    ,N-BALF4   ,H-E-BALF4 , 580539,1.31004128E+00,1.42038259E-01, ....
SCHOTT    ,N-BALF5   ,          , 547536,1.28385965E+00,7.19300942E-02, ....
....
....
```

Note that this file format is compatible with Microsoft Excel CSV files. Glass catalogue data can easily be imported into Excel, manipulated, and subsequently written to a file with extension ".csv".

The sequence of glass parameters in each line is as follows:

Data type	**Description**
Manufacturer	Manufacturer's name. The first three characters are significant.
Name	Glass name as defined by manufacturer. Limited to 64 characters.
Equivalent name	Name of equivalent glass from alternative manufacturer. Limited to 64 characters.
Code	MIL code as described in sect. ... A six-digit number.
B1	Schott Sellmeier dispersion coefficient B1
B2	Schott Sellmeier dispersion coefficient B2
B3	Schott Sellmeier dispersion coefficient B3
C1	Schott Sellmeier dispersion coefficient C1
C2	Schott Sellmeier dispersion coefficient C2
C3	Schott Sellmeier dispersion coefficient C3
Equation type	Integer number, describing type of dispersion equation. 0 = old Schott equation, 1 = new Schott equation (Sellmeier equation).
Lambda_min	minimum wavelength supported by the dispersion equation.
Lambda_max	maximum wavelength supported by the dispersion equation.
Availability (Lv)	Glass availability. 1 = highest melt frequency, 6 = lowest melt frequency, 0 = unknown.
D0	dn/dT coefficient 1
D1	dn/dT coefficient 2
D2	dn/dT coefficient 3
E0	dn/dT coefficient 4
E1	dn/dT coefficient 5
LTK	dn/dT coefficient 6, (λ_{TK})
DRT	thickness for internal transmission ("Reintransmission") data, (mm)
τ_{2500}	internal transmission at 2500nm, at DRT
τ_{2325}	internal transmission at 2325nm, at DRT
τ_{1970}	internal transmission at 1970nm, at DRT
τ_{1530}	internal transmission at 1530nm, at DRT
τ_{1060}	internal transmission at 1060nm, at DRT
τ_{700}	internal transmission at 700nm, at DRT
τ_{660}	internal transmission at 660nm, at DRT
τ_{620}	internal transmission at 620nm, at DRT
τ_{580}	internal transmission at 580nm, at DRT
τ_{546}	internal transmission at 546nm, at DRT
τ_{500}	internal transmission at 500nm, at DRT
τ_{460}	internal transmission at 460nm, at DRT
τ_{436}	internal transmission at 436nm, at DRT
τ_{420}	internal transmission at 420nm, at DRT
τ_{404}	internal transmission at 404nm, at DRT
τ_{400}	internal transmission at 400nm, at DRT
τ_{390}	internal transmission at 390nm, at DRT
τ_{380}	internal transmission at 380nm, at DRT
τ_{370}	internal transmission at 370nm, at DRT
τ_{365}	internal transmission at 365nm, at DRT
τ_{350}	internal transmission at 350nm, at DRT
τ_{334}	internal transmission at 334nm, at DRT

continued on next page

continued from previous page	
τ_{320}	internal transmission at 320nm, at DRT
τ_{310}	internal transmission at 310nm, at DRT
τ_{300}	internal transmission at 300nm, at DRT
τ_{290}	internal transmission at 290nm, at DRT
τ_{280}	internal transmission at 280nm, at DRT
τ_{270}	internal transmission at 270nm, at DRT
τ_{260}	internal transmission at 260nm, at DRT
τ_{250}	internal transmission at 250nm, at DRT
no data	intentionally left blank
no data	intentionally left blank
Chemical constants (CC)	
α_1	Linear constant of thermal expansion (CTE), -30°C to +70°C
α_2	Linear constant of thermal expansion (CTE), +20°C to +300°C
ρ	Specific density (g/cm^3)
RelPrice	Relative price (BK7 = 1.0).

32.8 Melt Glass File Format ".ind"

Pairs of wavelength and measured refractive index are stored in a standard ASCII-file with extension ".ind" (required). Each pair is stored in a separate line. Wavelengths must be given in μm. All entries are separated by at least one blank, multiple blanks as separator are permitted. The exclamation character "!" is identified as comment. A typical example of a melt data file is

```
!  wavel.    index
  0.435800  1.825150
  0.480000  1.816510
  0.486100  1.815500
  0.546100  1.807510
  0.587600  1.803390
  0.643800  1.799020
  0.656300  1.786080
!
! Data for Schott Lasfn30, batch no. 123456-1
```

32.9 GRIN Dispersion Coefficients File Format

Dispersion data for gradient index (GRIN) materials are stored in the file `grindisp.asc` in the GLASSES directory. Dispersion coefficients are assigned a name, which can be used by the GDISP command to associate that dispersion characteristics to a surface.

The `grindisp.asc` file contains blocks of 10 lines each. The file format has the following structure:

```
Dispersion name
min_wavelength   max_wavelength
ref_wavelength
K_max  L_max
K11  K12  K13  K1K_max
K21  K22  K23  K2K_max
K31  K32  K33  K3K_max
L11  L12  L13  L1L_max
```

```
 L21  L22  L23  L2L_max
 L31  L32  L33  L3L_max
```

Multiple materials may be defined by adding blocks of 10 lines one after the other. Blank lines between the blocks are not permitted.

Note that dispersion coefficients defined by a dispersion name require the glass name `GRIN` on a surface. Predefined gradient index materials will ignore user defined dispersion coefficients. Currently only profiles from LightPath (LPT) and the general URN (University of Rochester) profile accept these coefficients.

Sample grindisp.asc file containing two dispersion profiles "GLAK" and "GSF":

```
GLAK
0.365 0.725
0.58756
4 1
0.00522664 0.0206983 -0.00450304 0.006873
0.0472841 0.0429402 -0.00724884 -0.0445419
0.988601 0.057962 0.0941671 0.152672
0.0421634
0.0368588
110
GSF
0.38 2.2
0.58756
6 3
-0.0683636 -0.0323639 -0.0286748 -0.0169163 0.00256909 0.0174719
-0.00109783 0.0334663 0.0388098 0.0370413 0.017429 -0.0405421
0.931075 -0.0306245 -0.0392756 -0.0423487 -0.0256629 0.0437821
0.00498103 0.000410271 2.44E-05
0.082168 0.0343531 -0.0337717
110 0.000285988 0.000362547
```

32.10 GRIN Catalogue Glasses File Format (grin.asc)

Index profiles and dispersion of predefined gradient index (GRIN) glasses are stored in the file `$i\glasses\grin.asc`. The file format is plain ASCII. All data items are stored in free-format, each item is separated by at least one blank character. Multiple blanks have no effect.

Warning and Disclaimer: The data in `grin.asc` have been carefully compiled by Optenso to ensure validity and correctness of the results. Modification of this file is NOT recommended. If a user alters data in this file, he is doing this at his own risk. In case of improper data, the program may crash or hang or produce incorrect results.

The first line in `grin.asc` is a comment line and is ignored. Each subsequent line contains index profile and dispersion coefficients of an individual GRIN material. The first 12 data items in each line are common for *all* GRIN materials and have the following meaning:

Item No.	Description
1	GRIN type.
2	Material name
3	Equivalent name
4	Equation type
5	Number of K_{ij} coefficients
6	Number of L_{ij} coefficients
7	Reference wavelength, in microns
8	Minimum wavelength (in μm)
9	Maximum wavelength (in μm)
10	not used
11	Specific gravity, in g/cm^3
12	Linear coefficient of thermal expansion (CTE)
13 - 70	Profile and dispersion coefficients (see below)

Data items numbered 13 and higher store a stream of profile and dispersion coefficients. Profile coefficients are stored first, followed by the dispersion coefficients. Since number and definition of coefficients vary among GRIN types, there is no fixed location for a specific coefficient. For example, the SELFOC profile is described by 2 coefficients (n and $\sqrt{A}$) whereas the LightPath profile uses 11 coefficients.

Hence, the SEL *profile* coefficients are stored on places 13 - 14 (that is 12+1 and 12+2), followed by SEL *dispersion* coefficients, which start at item number 15.
Likewise, the LPT profile coefficients are stored at item numbers 13 - 23. LPT dispersion coefficients start at item number 24.

32.11 INT File Format ”.int”

Interferometric deformations are stored in ASCII files with the extension ”.int”. INT files describe gridded surface deformations, wavefront perturbations, intensity apodizing filters, radial deformations or Zernike polynomial coefficients. $OpTaliX$ supports a subset of these options: surface deformations, wavefront perturbations and intensity apodizing filters can be specified as two-dimensional (gridded) data.

INT files consist of a series of records, each of up to 80 characters followed by a carriage return. Each file consists of three major sections:

1. **Title**. This is a single record (80 characters) with descriptive information. It must NOT start with ”!”.

2. **Parameters**. A single record containing codes and data for interpreting the subsequently following data. The syntax for rectangular (gridded) data is:

   `GRD x_size y_size SUR|WFR|FIL WVL wavelength SSZ scale_size [NDA no_data_value]`

   The meaning of each entry is given as follows:

   `GRD x_size y_size` : The qualifier ”GRD” is required for gridded data. `x_size` and `y_size` are the number of grid points in X- and Y-directions.

   `SUR` : Specifies surface deformation.

   `WFR` : Specifies wavefront perturbation.

   `FIL` : Specifies intensity apodization filter.

`SSZ scale_size` : Defines the value of input data corresponding to one wave of deformation.

`WVL wavelength` : Wavelength in microns at which the interferogram was measured.

`NDA no_data_value` : Value of the input data which will be interpreted as missing data. Rays are blocked in these areas.

3. **Data**. Values for grid data are integers in the range -32768 to 32768. For each record, 10 values are entered, using enough records to enter all data. The number of entered values must match the product `x_size` · `y_size`.

Example of grid format:

```
0019-002-009 Time: 10:58:22 Date: 02/13/01
GRD 368 240 SUR WVL 0.632800 SSZ 24131 NDA 32767 XSC 0.857143
  32767  32767  32767  32767  32767  32767  32767  32767  32767  32767
  32767  32767  32767  32767  32767  32767  32767  32767  32767  32767
  32767  32767  32767  32767  32767  32767  32767  32767  32767  32767
    |
    |
   4763   4722   4723   4674   4621   4619   4583   4305   4204   4225
   4140   4017   3945   3834   3693   3723   3605   3515   3548   3461
   3442   3477   3333   3275   3167   3154   3035   2886   2767   2767
   2619   2619   2505   2436   2449   2392   2366   2099   1927   1927
    |
    |
  -4844  -4844  -4829  -4756  -4685  -4672  -4567  -4536  -4483  -4427
  -4319  -4205  -4113  -4018  -3908  -3818  -3774  -3684  -3589  -3501
  -3400  -3318  -3226  -3170  -3089  -3000  -2936  -2810  -2680  -2559
    |
    |
  32767  32767  32767  32767  32767  32767  32767  32767  32767  32767
  32767  32767  32767  32767  32767  32767  32767  32767  32767  32767
  32767  32767  32767  32767  32767  32767  32767  32767  32767  32767
  32767  32767  32767  32767  32767  32767  32767  32767  32767  32767
```

32.12 PSF File Format

Intensity distributions resulting from PSF calculations may be written to plain ASCII files. The files consist of a square matrix of data arranged in N columns and N rows. N is strictly dependent from NRD (number of rays across diameter) and is calculated by

`N = 4*NRD`

That is, calculating PSF using a grid of 32 x 32 rays in the entrance pupil yields a 128 x 128 matrix describing the PSF at the image surface. Hence, the file written consists of a matrix of 128 columns and 128 rows.

The ASCII-file only contains PSF-intensity data. No headers or control commands are written. An excerpt of the data structure is given below:

```
0.0027 0.0047 0.0061 0.0069 0.0072 0.0072 0.0072 0.0069 0.0061 0.0047 0.0027 0.0010
0.0067 0.0079 0.0078 0.0071 0.0064 0.0061 0.0064 0.0071 0.0078 0.0079 0.0067 0.0043
0.0073 0.0059 0.0041 0.0030 0.0026 0.0026 0.0026 0.0030 0.0041 0.0059 0.0073 0.0071
0.0040 0.0028 0.0038 0.0065 0.0091 0.0102 0.0091 0.0065 0.0038 0.0028 0.0040 0.0061
0.0035 0.0083 0.0161 0.0238 0.0290 0.0308 0.0290 0.0238 0.0161 0.0083 0.0035 0.0032
0.0119 0.0235 0.0336 0.0394 0.0417 0.0423 0.0417 0.0394 0.0336 0.0235 0.0119 0.0041
0.0259 0.0363 0.0387 0.0369 0.0358 0.0357 0.0358 0.0369 0.0387 0.0363 0.0259 0.0119
0.0363 0.0371 0.0335 0.0401 0.0565 0.0655 0.0565 0.0402 0.0335 0.0371 0.0363 0.0235
0.0387 0.0335 0.0491 0.1088 0.1872 0.2240 0.1872 0.1088 0.0491 0.0335 0.0387 0.0336
0.0369 0.0401 0.1088 0.2684 0.4501 0.5313 0.4501 0.2684 0.1088 0.0402 0.0369 0.0394
0.0358 0.0565 0.1872 0.4501 0.7338 0.8579 0.7338 0.4502 0.1872 0.0565 0.0358 0.0417
```

```
0.0357 0.0655 0.2240 0.5313 0.8579 1.0000 0.8580 0.5314 0.2240 0.0655 0.0357 0.0423
0.0358 0.0565 0.1872 0.4501 0.7338 0.8579 0.7338 0.4502 0.1872 0.0565 0.0358 0.0417
0.0369 0.0401 0.1088 0.2684 0.4501 0.5313 0.4501 0.2684 0.1088 0.0402 0.0369 0.0394
0.0387 0.0335 0.0491 0.1088 0.1872 0.2240 0.1872 0.1088 0.0491 0.0335 0.0387 0.0336
0.0363 0.0371 0.0335 0.0401 0.0565 0.0655 0.0565 0.0402 0.0335 0.0371 0.0363 0.0235
0.0259 0.0363 0.0387 0.0369 0.0358 0.0357 0.0358 0.0369 0.0387 0.0363 0.0259 0.0119
0.0119 0.0235 0.0336 0.0394 0.0417 0.0423 0.0417 0.0394 0.0336 0.0235 0.0119 0.0041
0.0035 0.0083 0.0161 0.0238 0.0290 0.0308 0.0290 0.0238 0.0161 0.0083 0.0035 0.0032
0.0040 0.0028 0.0038 0.0065 0.0091 0.0102 0.0091 0.0065 0.0038 0.0028 0.0040 0.0061
0.0073 0.0059 0.0041 0.0030 0.0026 0.0026 0.0026 0.0030 0.0041 0.0059 0.0073 0.0071
```

32.13 Ray File Format

This section describes the file format for ray sources, that is, volume sources defined by a collection of rays. Rays may be written to a file using one of the following commands:

RAYLOG	write (log) ray trace data on a specific surface to a file (ASCII only).
VIE SRC	The source viewer also allows export of ray data in ASCII or binary format.

32.13.1 General Ray Format

Ray data are written as coordinate triples (X,Y,Z), direction cosine triples (CX, CY, CZ), the associated ray intensities I_s, I_p in the S- and P-planes, and the current wavelength (in micrometers) at which the ray is traced. (Int):

X,Y,Z	XYZ-coordinates of the ray impinging at surface sk
CX,CY,CZ	Direction cosines of the rays impinging at surface sk
Int_p	Relative ray intensity in P-plane
Int_s	Relative ray intensity in S-plane
Lam	Ray wavelength in micrometers.

Ray data (X,Y,Z,CX,CY,CZ,Int_p,Int_s,Lam) are written as single lines, one line per ray. Data are formatted column-wise, separated by blanks, tabs or commas.

32.13.2 Ray Data in ASCII Format

Ray data stored in ASCII files should have the preferred file extensions "*.txt" or "*.dat". The first few lines of a ray source file defined in ASCII format, including one header line, is given below (number of digits reduced in print):

```
!      X        Y        Z       CX          CY       CZ    Int_p    Int_s        Lam
   0.000    0.000    0.000    0.000   0.000E+00    1.000    1.000    1.000    1.02400
   0.000   -1.067    0.000    0.000   0.300E-04    0.999    1.000    1.000    1.02400
   0.000   -1.029    0.000    0.000   0.228E-04    0.999    1.000    1.000    1.02400
   0.000  -0.9899    0.000    0.000   0.123E-04    0.999    1.000    1.000    1.02400
   0.000  -0.9499    0.000    0.000   0.211E-05    1.000    1.000    1.000    1.02400
   0.000  -0.9086    0.000    0.000  -0.610E-05    1.000    1.000    1.000    1.02400
   0.000  -0.8659    0.000    0.000  -0.115E-04    0.999    1.000    1.000    1.02400
   0.000  -0.8217    0.000    0.000  -0.143E-04    0.999    1.000    1.000    1.02400
   0.000  -0.7763    0.000    0.000  -0.146E-04    0.999    1.000    1.000    1.02400
   0.000  -0.7295    0.000    0.000  -0.132E-04    0.999    1.000    1.000    1.02400
   0.000  -0.6817    0.000    0.000  -0.106E-04    0.999    1.000    1.000    1.02400
   0.000  -0.6328    0.000    0.000  -0.748E-05    1.000    1.000    1.000    1.02400
......
```

An arbitrary number of header lines may precede the data lines. In ASCII files, the first character in a header line must be an exclamation mark "!". The numerical values in each line must be separated by at least a single blank character (ASCII decimal value 32), a horizontal tab character (ASCII decimal value 9) or the may be comma separated (ASCII decimal number 44). Multiple space/tab characters are allowed. This implies that the ray data need not be formatted. The only necessary information between data items are blank, tab or comma separators.

32.13.3 Ray Data in Binary Format

Binary files generally allow significantly smaller file sizes, however, they are dependent on the operating system. Ray data in binary files are always stored in single precision accuracy and are similar to ASAP binary source files (`*.dis` extension). A header line of 140 bytes is obligatory and precedes the data lines.

The preferred file extension for binary source files is "`*.dis`", however, any other extension is allowed if the user is aware about the file encoding (binary or ASCII).

Parameter	Bytes	Description
`Header`	140	Header line preceding the data lines. The header accepts arbitrary data, including blanks.
`X`	8	X-coordinate of ray with respect to reference system.
`Y`	8	Y-coordinate of ray with respect to reference system.
`Z`	8	Z-coordinate of ray with respect to reference system.
`CX`	8	Direction cosine of the ray in X-direction
`CY`	8	Direction cosine of the ray in Y-direction
`CZ`	8	Direction cosine of the ray in Z-direction
`Int`	8	Ray intensity

Note that the ASCII and binary file formats of ray data are different in contents. The ASCII format writes the S- and P-intensities plus the ray wavelength, whereas in the binary format only the mean ray intensity is written. Compatibility with the ASAP ray format was the driving factor.

The following FORTRAN code is a template to write (respectively read) ray data in the $OpTaliX$ binary format.

```
! Declarations:
        real                  :: dx,dy,dz,dcx,dcy,dcz,di
        character (len=140)   :: header
        integer               :: nrays, iunit = 12

! Open unit:
        open(iunit, file=filename, access='SEQUENTIAL', &
             form='BINARY', status='UNKNOWN', action='WRITE')

! Write header:
        header = 'OpTaliX ray data'
        write(iunit,'(A)') header    ! 140 bytes for header

! Write ray data:
        do k = 1,nrays
           write(iunit, err=600) dx,dy,dz,dcx,dcy,dcz,di
        enddo

! Close unit:
  600   close(iunit)
```

Bibliography

[1] Aberrator: http://www.aberrator.astronomy.net/

[2] J.A. Arnaud, ”Hamiltonian Theory of Beam Mode Propagation”, Progress in Optics, Vol.11, E. Wolf Ed. (Amsterdam 1973)

[3] M.Berek, ”Grundlagen der praktischen Optik”, Walter de Gruyter & Co. , Berlin 1970, Nachdruck der 1.Auflage 1930

[4] M.Born, E.Wolf, ”Principles of Optics”, 6^{th} edition, Pergamon Press

[5] Breault Research Organization, http://www.breault.com/

[6] E.Oran Brigham. ”FFT, Schnelle Fouriertransformation”, Oldenburg Verlag, München,Wien 1982

[7] H. A. Buchdahl, ”Many-color correction of thin doublets”, Applied Optics, Vol. 24, No. 12, June 1985

[8] Cargille Laboratories Inc., 55 Commerce Rd., Cedar Groove, N.J. 07009, USA

[9] A.E. Conrady, Mon. Not. Royal astr. Soc., 64 (1904) 182

[10] J.E. Dennis Jr., R.B. Schnabel, ”Numerical Methods for Unconstrained Optimization and Nonlinear Equations”, Prentice-Hall, Englewood Cliffs, New Jersey (1983)

[11] Donald P. Feder, ”Automatic optical design”, Appl. Opt., Vol.2, No.12, Dec. 1963, pp. 1209

[12] W.G.Driscoll, W.Vaughan, ”Handbook of Optics” Mc.Graw Hill, 1978, ISBN 0-07-047710-8

[13] A. Ghatak, K. Thyagarajan, ”Optical Electronics”, Cambridge University Press, 1989, ISBN 0 512 30643 4

[14] P.E. Gill, W.Murray, M.A. Sanders and M.H. Wright, ”Model building and practical aspects of nonlinear programming”, Computational Mathematical Programming (edited by K. Schittkowski), NATO ASI series, 15, Springer-Verlag, Berlin, Germany (1985)

[15] D. Gloge, ”Weakly Guiding Fibers”, Appl.Opt., Vol.10, No.10, Oct.1971, pp.2252-2258

[16] D. S. Grey, ”The inclusion of tolerance sensitivities in the merit function for lens optimization”, SPIE Vol. 147, Computer-Aided Optical Design (1978).

[17] J.W. Goodman, ”Introduction to Fourier Optics”, McGraw-Hill, New York, (1988), $2^{nd} edition$

[18] H.Haferkorn. ”Optik, Physikalisch technische Grundlagen und Anwendungen”. Harri Deutsch, Thun, Frankfurt/M. 1981

[19] H.Haferkorn, ”Bewertung optischer Systeme”, VEB Deutscher Verlag der Wissenschaften, Berlin 1986, ISBN 3-326-00000-6

[20] R.Herloski, S.Marshall, R.Antos, ”Gaussian beam ray-equivalent modeling and optical design”, Appl.Opt. 22, No.8, pp. 1168-1174, April 1983

[21] J.L.Houghton, US Patent 2,350,112 (30.May 1944)

[22] R.A. Hull, ”Transmission of light through a pile of parallel plates”, Proc.Phys.Soc., 574-575 (1936)

[23] M.Isshiki et.al., ”Lens Design: Global optimization of both performance and tolerance sensitivity”, SPIE, Vol. 6342, International Optical Design Conference 2006

[24] M.J. Kidger, ”Intermediate Optical Design”, SPIE Press monograph, PM 134, ISBN 0-8194-5217-3

[25] T.Kita et.al., Appl.Opt., Vol. 22, No.4, pp. 512

[26] W.A. Kleinhans, ”Aberrations of curved zone plates and Fresnel lenses” Appl. Opt. 16, pp. 1701-1704, (1977)

[27] H.W. Kogelnik, T.Li, Appl.Opt. 5, 1550 (1966)

[28] F. Kohlrauch, ”Praktische Physik”, 1968, Vol.1, p.408.

[29] M. Laikin, ”Lens Design”, 3^{rd} edition, Marcel Dekker Inc., (2001)

[30] K. Levenberg, ”A method for the solution of certain problems in least squares”, Quarterly of Applied Mathematics, **2**, 164-168

[31] R.J.Lurie, ”Anastigmatic Catadioptric Telescopes”, Journal of the optical society of America (JOSA), Vol.65(3), 1975, p261-266,

[32] H.A. Macleod, ”Thin-film optical filters”, 2^{nd}Ed., Adam Hilger Ltd., Bristol ISBN 0-85274-784-5

[33] D.Malacara, ”Diffraction Performance Calculations in Lens Design”, SPIE, Vol.1354, June 1990, pp.2-14

[34] D.Malacara, ”Handbook of Lens Design”, Marcel Deccer, Inc., (1994), ISBN 0-8247-9225-4

[35] D.Marquardt, ”An algorithm for least-squares estimation of nonlinear parameters” SIAM Journal on Applied Mathematics, **11**, 431-441

[36] M.Mansuripur, ”Classical Optics and its Applications”, Cambridge University Press, (2002), ISBN 0 521 80093 5

[37] R. I. Mercado, ”Design of apochromats and superachromats”, SPIE Critical Review Vol. CR41, Lens Design, ed. Warren J.Smith (Jan 1992).

[38] R.C.Millard, G.Seaver, ”An index of refraction algorithm for seawater over temperature, pressure, salinity, density and wavelength”, Deep-Sea Research, Vol. 37, No.12, pp.1909-1926, (1990)

[39] NSG Website, www.nsgeurope.com/technical.shtml

[40] OpsiraGmbH, Leibnitzstr. 20, D-88250 Weingarten, Germany, http://www.opsira.de

[41] Matthew P. Rimmer, ”Ray tracing in inhomogeneous media”, SPIE Proc. 399, p.339, (1983)

[42] Matthew P. Rimmer, ”Relative illumination calculations”, SPIE Proc. 655, p.99, (1986)

[43] M.J.D. Powell, ”A fast algorithm for nonlinearly constrained optimization calculations”, Numerical Analysis Proceedings, Dundee 1977, Lecture Notes in Mathematics (edited by G.A. Watson), 630, pp.144-157, Springer-Verlag, Berlin, Germany

[44] Radiant Imaging, http://www.radimg.com/index.htm

[45] J.L. Rayces, L.Lebich, ”Modeling of diffractive optical elements for optical design”, SPIE Vol. 2000, pp. 43, 0-8194-1249-X/93

[46] P.N.Robb, R.I.Mercado, ”Calculation of refractive indices using Buchdahl’s chromatic coordinate”, Appl.Optics, Vol.22, No.8, April 1983, pp.1198

[47] J.B.Scarborough, ”Numerical Mathematical Analysis”, Johns Hopkins U.P., Baltimore, 1966; Chap.13, Article 116.

[48] SCHOTT Glaswerke, ”Dispersionsformel fuer die Temperaturkoeffizienten der Brechzahlen von Glaesern”. Technische Information Nr. 19 (4/1988) der Schott Glaswerke, Postfach 2480, D-6500 Mainz.

[49] A.Sharma, D.V.Kumar, A.K.Ghatak, ”Tracing rays through graded-index media: a new method”, Applied Optics, Vol.21, No. 6, March 1982

[50] A.E. Siegman, Proc. SPIE 1224, 2, 1990

[51] Gordon H. Spencer, ”A flexible automatic lens correction procedure”, Appl. Opt., Vol.2, No.12, Dec. 1963, pp. 1257

[52] W.C. Sweatt, ”Describing holographic optical elements as lenses” J.Opt.Soc.Am. 67, pp.803-808, 1977

[53] W.C. Sweatt, ”Mathematical equivalence between a holographic optical element and an ultra-high index lens” J.Opt.Soc.Am. 69, No.3, pp.486-487, 1979

[54] Y.Tamagawa, S.Wakabayashi, T.Tajime, ”New design method for athermalized optical systems”, SPIE, Vol. 1752 (1992)

[55] Y. Tamagawa et.al., ”Multilens system design with an athermal chart”, Appl.Opt., Vol. 33, No. 34, pp. 8009, Dec. 1994

[56] Y. Tamagawa, T. Tajime, ”Expansion of an athermal chart into a multilens system with thick lenses spaced apart”, Opt. Eng. 35(10), pp. 3001, Oct. 1996

[57] R.E. Wagner, W. J. Tomlinson, ”Coupling efficiency of optics in single-mode fiber components”, Appl.Optics, Vol.21, pp. 2671-2688 (1982)

[58] W.T.Welford, ”A vector raytracing equation for hologram lenses of arbitrary shape”. Optics Communications, Vol.14, No. 3, July 1975, p.322

[59] E.Waluschka, ”Polarization Ray Trace”, Optical Engineering, Feb. 1989, Vol.28, No.2

[60] J.Y.Wang, D.E.Silva, ”Wave-front interpretation with Zernike polynomials”, Applied Optics, Vol. 19(9), pp. 1510-1518, May 1980

[61] R.N. Wilson, ”Reflecting Telescope Optics I”, Springer Verlag Berlin-Heidelberg 1996, ISBN 3-540-58964-3

[62] T.R.M.Sales, G.M.Morris, ”Diffractive-refractive behaviour of kinoform lenses”, Appl.Opt., Vol.36, No.1, January 1997, pp. 253

[63] G.W.Forbes, ”Shape specification for axially symmetric optical surfaces”, OSA, Vol.15, No.8, April 2007, Optics Express 5218

[64] G.W.Forbes, ”Robust and fast computation for the polynomials of optics”, OSA, Vol.18, No.12, June 2010, Optics Express 13851

[65] C.K. Carniglia, ”Accounting for the phase of an optical coating in an optical system”, OSA, Opticsinfobase

[66] MacLeod: Optical Thin Films, www.thinfilmcenter.com

Index

Symbols

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z