

# pst-optexp

## A PSTricks package to draw optical experimental setups

Christoph Bersch <[usenet@bersch.net](mailto:usenet@bersch.net)>

2008/07/22    Version 1.3alpha

<b>Contents</b>	
<b>1 Introduction</b>	<b>2</b>
<b>2 Concept and General Behavior</b>	<b>2</b>
2.1 Concept . . . . .	2
2.2 General Options . . . . .	2
2.3 Positioning . . . . .	3
2.4 Labels . . . . .	3
2.5 Nodes For External Usage . . . . .	4
2.6 Connecting Objects . . . . .	5
<b>3 Free-Ray Objects</b>	<b>9</b>
3.1 Lens . . . . .	9
3.2 Optical Plate . . . . .	10
3.3 Retardation Plate . . . . .	10
3.4 Pinhole . . . . .	10
3.5 Crystal . . . . .	11
3.6 Box . . . . .	11
3.7 Detector . . . . .	12
3.8 Optical Diode . . . . .	13
3.9 Dove Prism . . . . .	13
3.10 Polarization . . . . .	13
3.11 Mirror . . . . .	14
3.12 Beamsplitter . . . . .	15
3.13 Optical Grid . . . . .	15
3.14 Prism . . . . .	16
3.15 Right-Angle Prism . . . . .	16
3.16 Penta Prism . . . . .	17
3.17 Custom Components . . . . .	17
<b>4 Fiber-Optical Objects</b>	<b>18</b>
4.1 Fiber . . . . .	18
4.2 Amplifier . . . . .	18
4.3 Mach-Zehnder Modulator . . . . .	18
4.4 Filter . . . . .	18
4.5 Polarization Controller . . . . .	19
4.6 Optical Switch . . . . .	19
4.7 Fiber Delay Line . . . . .	20
4.8 Fiber Collimator . . . . .	20
4.9 Coupler . . . . .	20
4.9.1 $2 \times 2$ Coupler . . . . .	20
4.9.2 WDM Coupler . . . . .	21
4.9.3 WDM Splitter . . . . .	21
4.10 Fiber Styles . . . . .	21
<b>5 Defining New Objects</b>	<b>23</b>
5.1 Customized Versions of Existing Macros . . . . .	23
5.2 Defining New Objects . . . . .	23
<b>6 Examples</b>	<b>25</b>
<b>7 Complete List of Parameters</b>	<b>28</b>
<b>8 Todo</b>	<b>31</b>
<b>9 Acknowledgements</b>	<b>31</b>
<b>10 History</b>	<b>31</b>

## 1 Introduction

The package `pst-optexp` is a collection of optical components that facilitate easy sketching of optical experimental setups. Mechanisms for proper alignment of different components are provided internally. This way the user does not have to care for proper orientation of the elements. Macros for convenient definition of new user-defined components are also provided.

## 2 Concept and General Behavior

This section introduces into the basic concepts of the package design and explains the parameters and commands which are supported by most optical objects.

### 2.1 Concept

The objects provided by `pst-optexp` can be differentiated into two different categories: free-ray and fiber-optical objects.

The free-ray units are subdivided in two different kinds: dipoles which require two reference points for alignment and do not alter the direction of passing light beams (e.g. lenses and retardation plates) and tripoles which work in reflection and require three reference points (mirrors, gratings, beamsplitters etc.).

For free-ray setups one usually has a few straight light paths in which several different objects are to be arranged. In this case it is very convenient to define only two nodes for each light path. The objects are placed on this light path using the different positioning parameters (see Sec. 2.3) of the package. After having arranged everything, the beams themselves are drawn. If objects with multiple internal reflections (e.g. prisms, see Sections 3.9, 3.14 – 3.16) or objects without internal beams (e.g. optical diodes, see Sec. 3.8) are involved. The different possibilities are explained in Sec. 2.6.

The fiber-optical objects can be classified as dipoles, tripoles and quadrupoles which have a corresponding number of fiber connections. Their handling differs in some aspects from the free-ray objects. The fiber optics are directly connected with the reference nodes. Every input and output fiber can be flexibly customized for each object (see Sec. 4.10). Positioning of the fiber dipoles is handled equivalently to the free-ray dipoles. Tripoles and quadrupoles can be found only as different coupler types. Their positioning mechanisms are a bit more involved and explained in Sec. 4.9.

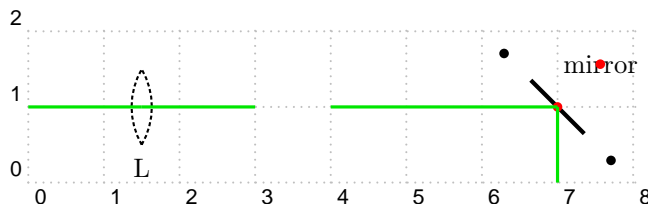
Some hybrid dipoles (laser, optbox, detector etc.) can be used both as fiber-optical or free-ray elements. The way they are treated regarding the connections to the reference points can be controlled by the parameters explained in Sec. 2.6.

### 2.2 General Options

**optional:** <boolean> (*default: false*)  
**showoptdots:** <boolean> (*default: false*)

`optional` can be used with every object and marks it as optional. The style of an optional element can be configured by changing the `psstyle OptionalStyle`.

`showoptdots` draws some internal nodes which are used to place the object and the label. The black points are used for positioning, the red points mark the label references.



```

1 \begin{pspicture}[showgrid=true](8,2)
2   \psset{beam}
3   \lens[optional](0,1)(3,1){L}
4   \mirror[showoptdots](4,1)(7,1)(7,0){mirror}
5 \end{pspicture}

```

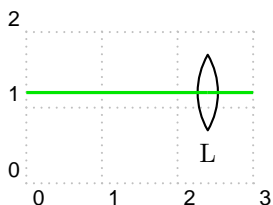
## 2.3 Positioning

**position:** <num> (*default:* {})

**abspos:** <num> (*default:* {})

**position** is equivalent to the `npos` parameter of `\ncput` (can be any number from 0 to 1) and controls the relative position of object between the two reference points. It is not available for the free-ray tripoles.

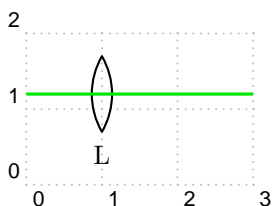
The parameter **abspos** allows absolute positioning between the two reference nodes. Its value is given in psunits.



```

1 \begin{pspicture}[showgrid=true](3,2)
2   \lens[beam, position=0.8](0,1.2)(3,1.2){L}
3 \end{pspicture}

```



```

1 \begin{pspicture}[showgrid=true](3,2)
2   \lens[beam, abspos=1](0,1.2)(3,1.2){L}
3 \end{pspicture}

```

## 2.4 Labels

**labeloffset:** <num> (*default:* 0.8)

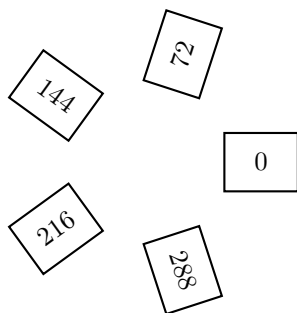
**labelangle:** <num> (*default:* 0)

**labelstyle:** <macro> (*default:* \small)

**labelalign:** <\rput ref string> (*default:* c)

**labelref:** relative|relgrav|global (*default:* relgrav)

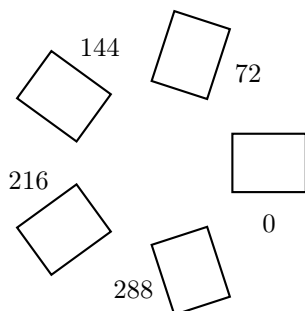
**labeloffset** specifies the offset from the label reference node of the object which is mostly the center. **labelstyle** defines the textstyle that is used to typeset the label and **labelalign** corresponds to the repoint of `\rput`. The parameter **labelref** sets the reference coordinate system for the **labelangle** and the orientation of the label text. The detailed behaviour is best illustrated looking at the following three examples.



```

1 \begin{pspicture}(-2,-2)(2.5,2)
2   \multido{\i=0+72}{5}{%
3     \optbox[endbox,
4       labelref=relative,
5       labeloffset=0,
6       optboxwidth=1](0,0)(1;\i){\i}
7   }
8 \end{pspicture}

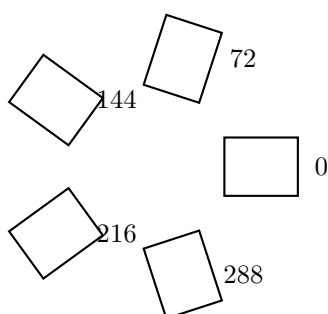
```



```

1 \begin{pspicture}(-2,-2)(2.5,2)
2   \multido{\i=0+72}{5}{%
3     \optbox[endbox,
4       labelref=relgrav,
5       optboxwidth=1](0,0)(1;\i){\i}
6   }
7 \end{pspicture}

```



```

1 \begin{pspicture}(-2,-2)(2.5,2)
2   \multido{\i=0+72}{5}{%
3     \optbox[endbox,
4       labelref=global,
5       optboxwidth=1](0,0)(1;\i){\i}
6   }
7 \end{pspicture}

```

## 2.5 Nodes For External Usage

**extnode:** <ref string> (*default:* {})

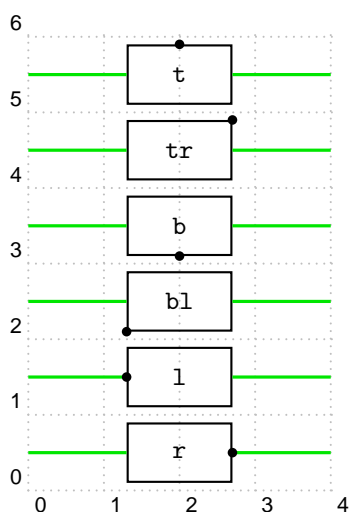
**extnodepostfix:** <string> (*default:* ExtNode)

Some of the objects can provide a supplementary node for additional connections. A laser diode may be connected for example to a frequency synthesizer (use package `pst-circ`) or a detector to a computer.

`extnode` controls the position of the additional node. By default this parameter is empty (`{}`) and no node is created. Possible values are any combinations of `t` (top), `b` (bottom), `l` (left), `r` (right) and `c` (center).

The name of the new node depends on the `compname` parameter (see Sec. 2.6). It is composed by `compname` + `extnodepostfix`, that is if `compname` is empty the new node is named `ExtNode` by default and overwritten by following objects.

Table. 1 shows all objects which provide an external node. Some allow any combination for `extnode`, others have only one reasonable possibility (e.g. piezo mirror, see Sec. 3.11).



```

1 \begin{pspicture}[showgrid=true](4,6)
2   \psset{conn=o-o, labeloffset=0}
3   \optbox[extnode=t](0,5.5)(4,5.5){\texttt{t}}
4   \psdot(ExtNode)
5   \optbox[extnode=tr](0,4.5)(4,4.5){\texttt{tr}}
6   \psdot(ExtNode)
7   \optbox[extnode=b](0,3.5)(4,3.5){\texttt{b}}
8   \psdot(ExtNode)
9   \optbox[extnode=bl](0,2.5)(4,2.5){\texttt{bl}}
10  \psdot(ExtNode)
11  \optbox[extnode=l](0,1.5)(4,1.5){\texttt{l}}
12  \psdot(ExtNode)
13  \optbox[extnode=r](0,0.5)(4,0.5){\texttt{r}}
14  \psdot(ExtNode)
15 \end{pspicture}

```

Object	possible extnode positions	
<code>\optbox</code>	all (any combination of <code>t</code> , <code>r</code> , <code>l</code> and <code>b</code> )	
<code>\mirror</code>	one fix position (only for <code>mirrortype=piezo</code> )	
<code>\optdetector</code>	one (for <code>dettype=round</code> )	
<code>\optmzm</code>	all (for <code>dettype=diode</code> )	see <code>\optbox</code>
<code>\optfilter</code>	all	see <code>\optbox</code>
<code>\optswitch</code>	all	see <code>\optbox</code>
<code>\fiberdelayline</code>	all	see <code>\optbox</code>

Table 1: The objects which may provide an external node when parameter `extnode` is not empty. Some allow different positions of the node and for some only a fixed node makes sense.

## 2.6 Connecting Objects

`compname`: <string> (default: `{}`)

**conn:** i-i|o-o|f-f (*default:* -)

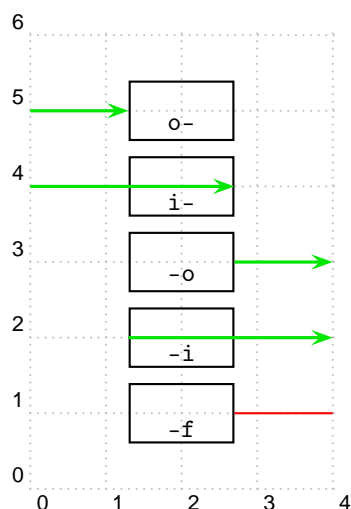
**fiber:** alias for **conn=f-f**

**beam:** alias for **conn=o-i**

Simple experimental setups with a few objects can usually be realized by defining some nodes, arranging the object inbetween and drawing the beams at the end. If, however, objects with multiple internal reflections (all the prisms) or without visible internal beams (optical diode) are involved, this simple method is not applicable anymore.

For this case several different possibilities of connecting objects are available: parameter **conn** specifies the kind of connections before and after the object. Its syntax is analogous to the PSTricks parameter **arrows**. By default it is set to - and no connections are drawn.

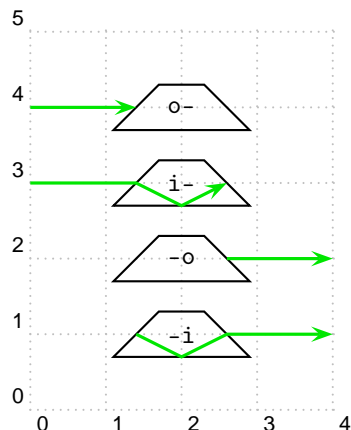
All fiber-optical units define **conn=f-f** which means that input and output connections are fibers (the boolean parameter **fiber** is an alias for **conn=f-f**). The other possibilities are **i** (inner node) and **o** (outer node). Their meaning should become clear looking at the next two code examples. The letter before the dash specifies the kind of connection from the first reference node to the object. A **conn=o-** connects the first reference node to the outer left node, a **conn=i-** connects the first reference node to the outer left node and then a line through all internal nodes up to the outer right node. Equivalently does **conn=-i** and **conn=-o** work for the second reference point. The boolean parameter **beam** is an alias for **conn=o-i**. The beam style is controlled by the **psstyle Beam** which can be changed using **newpsstyle** and **addtopsstyle**.



```

1 \begin{pspicture}[showgrid=true](4,6)
2   \addtopsstyle{Beam}{arrows=->, arrowscale=1.5}
3   \psset{labeloffset=0.1, labelalign=t}
4   \optbox[conn=o-](0,5)(4,5){\texttt{o-}}
5   \optbox[conn=i-](0,4)(4,4){\texttt{i-}}
6   \optbox[conn=-o](0,3)(4,3){\texttt{-o}}
7   \optbox[conn=-i](0,2)(4,2){\texttt{-i}}
8   \optbox[conn=-f](0,1)(4,1){\texttt{-f}}
9 \end{pspicture}

```

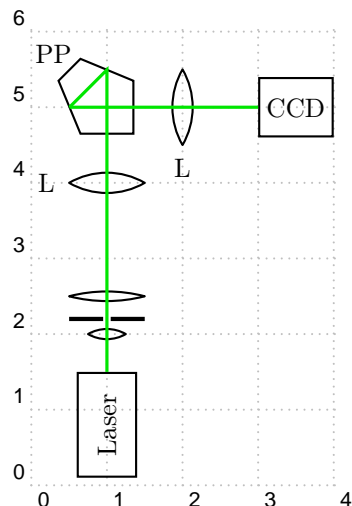


```

1 \begin{pspicture}[showgrid=true](4,5)
2   \addtopsstyle{Beam}{arrows=->, arrowscale=1.5}
3   \psset{labeloffset=0}
4   \doveprism[conn=o-](0,4)(4,4){\texttt{o-}}
5   \doveprism[conn=i-](0,3)(4,3){\texttt{i-}}
6   \doveprism[conn=-o](0,2)(4,2){\texttt{-o}}
7   \doveprism[conn=-i](0,1)(4,1){\texttt{-i}}
8 \end{pspicture}

```

The following example shows how this parameter can be used in some kinds of experimental setups using objects with internal reflections (here a penta prism). Instead of drawing the beam at the end with a `\psline`, the beams are created at definition time of the object with the `conn` parameter.



```

1 \begin{pspicture}[showgrid=true](4,6)
2   \pnode(1,1.5){A}\pnode(1,5){G}\pnode(3,5){B}
3   \optbox[endbox, labelref=relative,
4     labeloffset=0](G)(A){Laser}
5   \lens[lens=0.5 0.5 0.5, abspos=0.5](A)(G){}
6   \pinhole[abspos=0.7](A)(G){}
7   \lens[lens=2, abspos=1](A)(G){}
8   \lens[abspos=2.5, labelangle=180](A)(G){L}
9   \lens[abspos=1](G)(B){L}
10  \optbox[endbox, labeloffset=0, optboxwidth=1](G)(B){CCD}
11  \pentaprism[beam, labeloffset=1](A)(G)(B){PP}
12 \end{pspicture}

```

This method works as long as no objects without internal beams or with internal reflections (e.g. a Dove prism, see Sec. 3.9) are used in the setup. A possibility would be to create additional nodes, but this may be not very comfortable. Therefore, `pst-optexp` provides a macro `\drawbeam` which connects a named object with a node or another named object. A named object is a `pst-optexp` element with a non-empty `compname`.

```

1 \drawbeam[conn=...]{<from>}{<to>}

```

If `<from>` or `<to>` is a node, it must be written including the round braces. A call

```

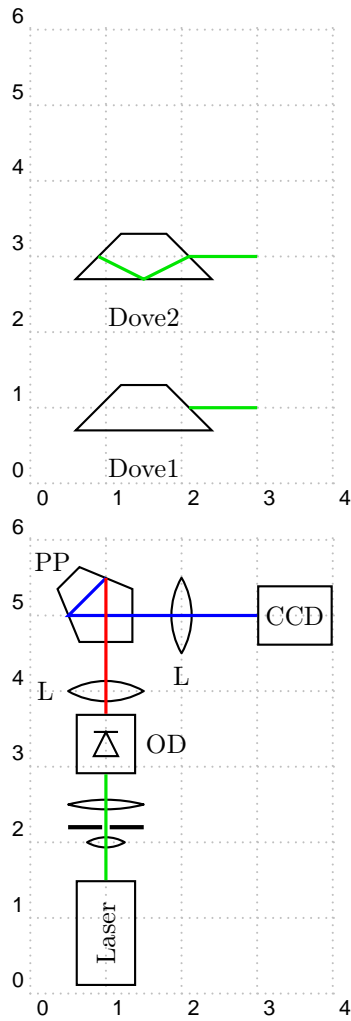
1 \drawbeam{Obj}{(1;45)}

```

connects an named object `Obj` to a node.

The type of beam connection is again controlled by the parameter `conn`. Almost every optical object does not have distinguished inputs and outputs and can be used in either directions. Therefore, it does not make sense to speak about ‘input’ and ‘output’ when referring to the object nodes, but rather about node A and node B. Consequently, the two letters of parameter `conn` can take the values `a`, `A`, `b` or `B` when used together with `\drawbeam`.

The first letter in `conn` refers to the `<from>` object, the second one to the `<to>` object.



```

\begin{pspicture}[showgrid=true](4,6)
  \doveprism[compname=Dove1](0,1)(3,1){Dove1}
  \drawbeam[conn=b-]{Dove1}{(3,1)}
  \doveprism[compname=Dove2](0,3)(3,3){Dove2}
  \drawbeam[conn=B-]{Dove2}{(3,3)}
\end{pspicture}

```

```

\begin{pspicture}[showgrid=true](4,6)
  \pnode(1,1.5){A}\pnode(1,5){G}\pnode(3,5){B}
  \optbox[endbox, labelref=relative,
    labeloffset=0](G)(A){Laser}
  \lens[lens=0.5 0.5 0.5, abspos=0.5](A)(G){}
  \pinhole[abspos=0.7](A)(G){}
  \lens[lens=2, abspos=1](A)(G){}
  \lens[abspos=2.5, labelangle=180](A)(G){L}
  \lens[abspos=1](G)(B){L}
  \optdiode[abspos=1.8, conn=o-, compname=OD](A)(G){OD}
  \optbox[endbox, labeloffset=0, optboxwidth=1](G)(B){CCD}
  \addtopsstyle{Beam}{linecolor=blue}
  \pentaprism[conn=-i, labeloffset=1,
    compname=PP](A)(G)(B){PP}
  \addtopsstyle{Beam}{linecolor=red}
  \drawbeam[conn=b-a]{OD}{PP}
\end{pspicture}

```



### 3 Free-Ray Objects

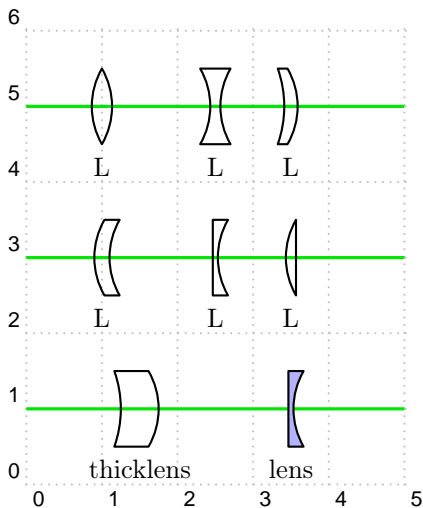
In the sections 3.1–3.17 the available free-ray components with their parameters are described.

In section 2 general parameters are described that are not proprietary to a specific unit but can be used for several different components. Finally, in section 2.4 the options for the positioning of labels are explained.

The appearance of all components can be changed with the corresponding standard PSTricks parameters such as `fillstyle` or `linestyle`. For some components changing only parts of the layout is possible (e.g. the extended part of mirrors). For those cases `psstyles` are provided that influence only the corresponding part of the components and can be redefined using `\newsstyle`.

#### 3.1 Lens

**lensheight:** <num> (default: 1)  
**lenswidth:** <num> (default: 0.3)  
**lensradius:** <num> [<num>] (default: {})  
**lensradiusleft:** <num> (default: 1)  
**lensradiusright:** <num> (default: 1)  
**lens:** <num> [<num> [<num> [<num>]]] (default: {})  
**thicklens:** <boolean> (default: false)



```
\begin{pspicture}[showgrid=true] (5,6)
% concave lenses
\node(0,5){A}\node(5,5){B}
\psline[style=Beam] (A) (B)
\lens[position=0.2] (A) (B){L}
\lens[lensradius=-1,position=0.5] (A) (B){L}
\lens[lens=-1.5 1,position=0.7] (A) (B){L}
% convex lenses
\node(0,3){A}\node(5,3){B}
\psline[style=Beam] (A) (B)
\lens[position=0.2,lens=1 -1] (A) (B){L}
\lens[lens=0 -1] (A) (B){L}
\lens[lens=1 0,position=0.7] (A) (B){L}
% thick lenses
\node(0,1){A}\node(5,1){B}
\psline[style=Beam] (A) (B)
\lens[position=0.3, lens=-1.5 1 1 0.5,
thicklens] (A) (B){thicklens}
\lens[lens=0 -1, position=0.7, fillstyle=solid,
fillcolor=blue!30!white] (A) (B){lens}
\end{pspicture}
```

The shape of a lens is defined by its two surface radii. A negative radius gives a concave, a positive radius a convex and a radius of 0 a plain surface. The parameters `lensradiusleft` and `lensradiusright` allow to define independent values for both surfaces. `lensradius` sets both curvatures to the same value. Usually only `lensheight` and the two radii are used to construct the lens. The thickness (or width) is determined automatically. Manually controlling

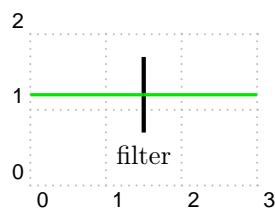
the thickness of the lens can be achieved by setting `thicklens` to `true`. Then `lenswidth` is used as width of the lens at its waist. Finally, the parameter `lens` allows the definition of all relevant lens parameters at once. It consists of one up to four space-separated numbers. The first one gives the left radius. If no further value is set, the right radius will be set to the same value and all other parameters are left unchanged. Using two numbers defines two different radii. The third optional value defines the `lensheight` and the fourth one the `lenswidth`.

**Compatibility:** The whole implementation of the lens was changed in version 1.2. It allows a much more flexible definition of different lens types. However, I could not get full compatibility with the older way to define lens using only `lensheight` and `lenswidth`. To use this old behaviour, you have to set the `lenstype` explicitly, but then you have no access to the new features! All users are encouraged to adapt their code to use the new parameters, as the old code will be removed in future versions.

### 3.2 Optical Plate

`plateheight`: <num> (*default*: 1)

`platelinewidth`: <num> (*default*: 2\pslinewidth)

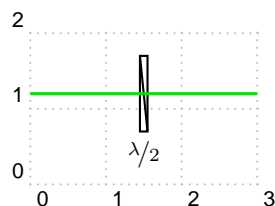


```
1 \begin{pspicture}[showgrid=true](3,2)
2   \optplate[beam](0,1.2)(3,1.2){filter}
3 \end{pspicture}
```

### 3.3 Retardation Plate

`plateheight`: <num> (*default*: 1)

`platewidth`: <num> (*default*: 0.1)



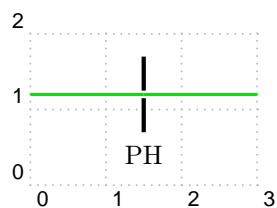
```
1 \begin{pspicture}[showgrid=true](3,2)
2   \pnode(0,1.2){A}
3   \pnode(3,1.2){B}
4   \optretplate[beam](A)(B){$\nicefrac{\lambda}{2}$}
5 \end{pspicture}
```

### 3.4 Pinhole

`outerheight`: <num> (*default*: 1)

`innerheight`: <num> (*default*: 0.1)

`phlinewidth`: <num> (*default*: 2\pslinewidth)



```

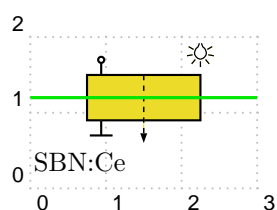
1 \begin{pspicture}[showgrid=true](3,2)
2   \pnode(0,1.2){A}
3   \pnode(3,1.2){B}
4   \pinhole[beam](A)(B){PH}
5 \end{pspicture}

```

### 3.5 Crystal

**crystalwidth:** <num> (*default:* 2)  
**crystalheight:** <num> (*default:* 0.8)  
**caxislength:** <num> (*default:* 0.6)  
**caxisinv:** <boolean> (*default:* false)  
**voltage:** <boolean> (*default:* false)  
**lamp:** <boolean> (*default:* false)  
**lampscale:** <num> (*default:* 0.3)  
**angle:** <degree> (*default:* 0)  
**rotateref:** <ref string> (*default:* c)

For a discussion of the `angle` and `rotateref` parameters see Sec. 3.6 about boxes.



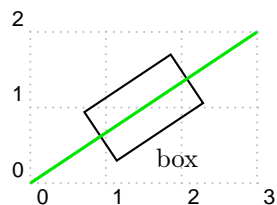
```

1 \begin{pspicture}[showgrid=true](3,2)
2   \pnode(0,1.2){A}
3   \pnode(3,1.2){B}
4   \crystal[crystalwidth=1.5, crystalheight=0.6, fillstyle=solid,
5     fillcolor=yellow!90!black, labelangle=-45, labeloffset=1.2,
6     voltage, lamp, beam](A)(B){SBN:Ce}
7 \end{pspicture}

```

### 3.6 Box

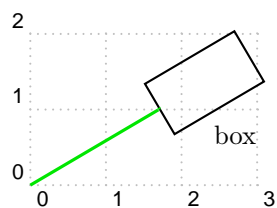
**optboxheight:** <num> (*default:* 0.5)  
**optboxwidth:** <num> (*default:* 1)  
**endbox:** <boolean> (*default:* false)  
**angle:** <degree> (*default:* 0)  
**rotateref:** <ref string> (*default:* c)  
**refractiveindex:** <num> (*default:* {})



```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optbox[beam](0,0)(3,2){box}
3 \end{pspicture}

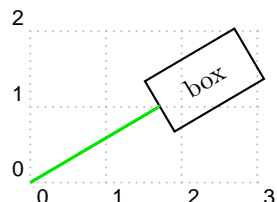
```



```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optbox[beam, endbox](0,0)(1.7,1){box}
3 \end{pspicture}

```

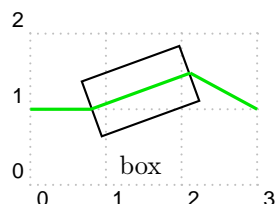


```

1 \begin{pspicture}[showgrid=true](3,2)
2   \node(0,0){A}
3   \node(1.7,1){B}
4   \optbox[beam, endbox, labelref=relative, labeloffset=0](A)(B){box}
5 \end{pspicture}

```

The parameter **angle** describes the tilt of the box relative to the reference line defined by the two reference nodes. The reference point for the rotation can be defined with **rotateref** which can take any combination of c, t, b, l and r (compare with **extnode** in Sec. 2.5). Note, that all connection-related nodes are also rotate, while the label is not affected.

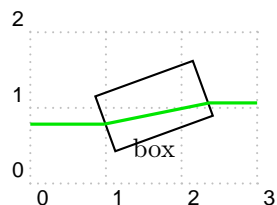


```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optbox[angle=20, beam, rotateref=l,
3     labeloffset=0.5](0,1)(3,1){box}
4 \end{pspicture}

```

Together with the parameter **refractiveindex** this can be exploited to sketch the refraction through a tilted homogeneous medium.



```

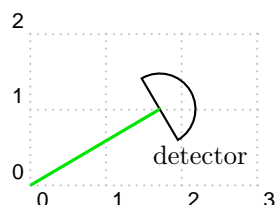
1 \begin{pspicture}[showgrid=true](3,2)
2   \node(0,1){A}
3   \node(3,1){B}
4   \optbox[angle=20, rotateref=t, labeloffset=0.5,
5     refractiveindex=2.3, compname=Box](A)(B){box}
6   \drawbeam[conn=-a]{(A|BoxIntern1)}{Box}
7   \drawbeam[conn=B-]{(Box){(B|BoxInternN)}}
8 \end{pspicture}

```

### 3.7 Detector

**detsize:** <num> (default: 0.5)

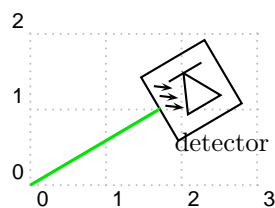
**detype:** round|diode (default: round)



```

1 \begin{pspicture}[showgrid=true](3,2)
2   \node(0,0){A}
3   \node(1.7,1){B}
4   \optdetector[beam](A)(B){detector}
5 \end{pspicture}

```



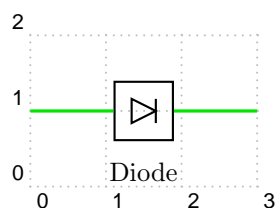
```

\begin{pspicture}[showgrid=true](3,2)
  \pnode(0,0){A}
  \pnode(1.7,1){B}
  \optdetector[beam, dettype=diode](A)(B){detector}
\end{pspicture}

```

### 3.8 Optical Diode

**optdiodesize:** <num> (*default:* 0.8)



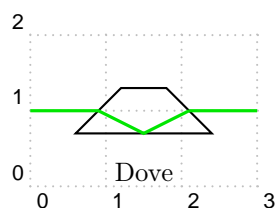
```

\begin{pspicture}[showgrid=true](3,2)
  \optdiode[conn=o-o](0,1)(3,1){Diode}
\end{pspicture}

```

### 3.9 Dove Prism

**doveprismsize:** <num> (*default:* 0.6)



```

\begin{pspicture}[showgrid=true](3,2)
  \doveprism[beam](0,1)(3,1){Dove}
\end{pspicture}

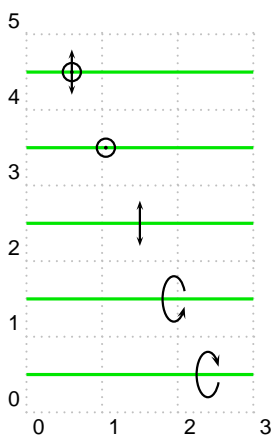
```

### 3.10 Polarization

**poltype:** parallel|perp|misc|lcirc|rcirc (*default:* parallel)

**polsize:** <num> (*default:* 0.6)

**pollinewidth:** <num> (*default:* 0.7\pslinewidth)



```

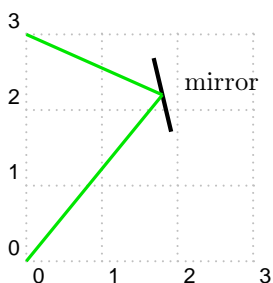
\begin{pspicture}[showgrid=true](3,5)
  \pnode(0,0.5){A1}\pnode(3,0.5){B1}\pnode(0,1.5){A2}
  \pnode(3,1.5){B2}\pnode(0,2.5){A3}\pnode(3,2.5){B3}
  \pnode(0,3.5){A4}\pnode(3,3.5){B4}\pnode(0,4.5){A5}
  \pnode(3,4.5){B5}\psset{style=Beam}
  \multido{\i=1+1}{5}{\psline(A\i)(B\i)}
  \psset{linecolor=black}
  \polarization[poltype=misc,position=0.2](A5)(B5)
  \polarization[poltype=perp,position=0.35](A4)(B4)
  \polarization[poltype=parallel,position=0.5](A3)(B3)
  \polarization[poltype=rcirc,position=0.65](A2)(B2)
  \polarization[poltype=lcirc,position=0.8](A1)(B1)
\end{pspicture}

```

### 3.11 Mirror

**mirrorwidth:** <num> (default: 1)  
**mirrorradius:** <num> (default: 0)  
**mirrorlinewidth:** <num> (default: 2\pslinewidth)  
**mirrortype:** normal|piezo|extended (default: normal)  
**mirrordepth:** <num> (default: 0.08)  
**variable:** <num> (default: false)

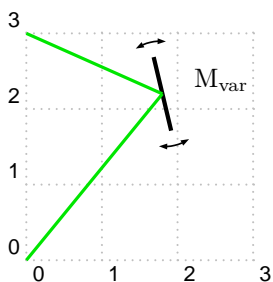
The parameter `mirrorradius` defines the curvature of the mirror. A value of 0 is for a plain mirror, a negative radius is for a concave mirror and a positive radius gives you a convex mirror. The style of the extended mirror is defined as a `psstyle ExtendedMirror` and can be changed using `\newpsstyle`. The appearance of the piezo mirror likewise can be changed by adapting the `psstyle PiezoMirror`.



```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,0){A}
3   \pnode(1.8,2.2){G}
4   \pnode(0,3){B}
5   \mirror[beam](A)(G)(B){mirror}
6 \end{pspicture}

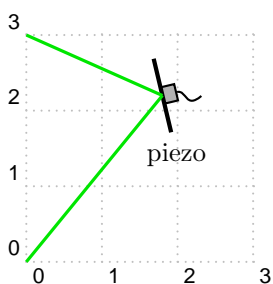
```



```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,0){A}
3   \pnode(1.8,2.2){G}
4   \pnode(0,3){B}
5   \mirror[beam, variable](A)(G)(B){M$_\mathrm{var}$}
6 \end{pspicture}

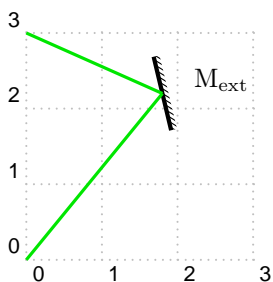
```



```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,0){A}
3   \pnode(1.8,2.2){G}
4   \pnode(0,3){B}
5   \mirror[beam, mirrortype=piezo, labelangle=-90](A)(G)(B){piezo}
6 \end{pspicture}

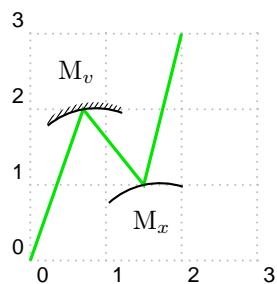
```



```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,0){A}
3   \pnode(1.8,2.2){G}
4   \pnode(0,3){B}
5   \mirror[beam, mirrortype=extended](A)(G)(B){M$_\mathrm{ext}$}
6 \end{pspicture}

```



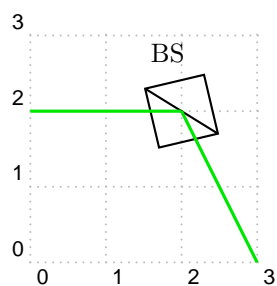
```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,0){A}\pnode(0.7,2){G1}
3   \pnode(1.5,1){G2}\pnode(2,3){B}
4   \psset{labeloffset=0.5}
5   \psline[style=Beam](A)(G1)(G2)(B)
6   \mirror[mirrortype=extended, mirrorradius=1](A)(G1)(G2){M$_v$}
7   \mirror[mirrorradius=-1](G1)(G2)(B){M$_x$}
8 \end{pspicture}

```

### 3.12 Beamsplitter

bssize: <num> (default: 0.8)



```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,2){A}
3   \pnode(2,2){G}
4   \pnode(3,0){B}
5   \beamsplitter[beam](A)(G)(B){BS}
6 \end{pspicture}

```

### 3.13 Optical Grid

optgridcount: <integer> (default: 10)

optgridwidth: <num> (default: 1)

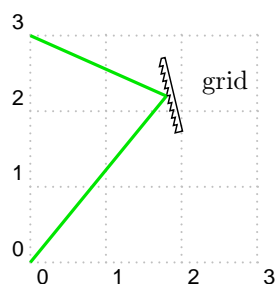
optgridheight: <num> (default: 0.1)

optgriddepth: <num> (default: 0.05)

optgridtype: blazed|binary (default: blazed)

optgridlinewidth: <num> (default: 0.7\pslinewidth)

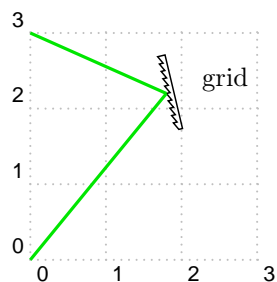
reverse: <boolean> (default: false)



```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,3){A}
3   \pnode(1.8,2.2){G}
4   \pnode(0,0){B}
5   \optgrid[beam](A)(G)(B){grid}
6 \end{pspicture}

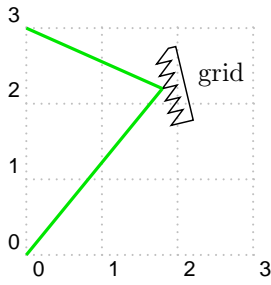
```



```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,3){A}
3   \pnode(1.8,2.2){G}
4   \pnode(0,0){B}
5   \optgrid[beam, reverse](A)(G)(B){grid}
6 \end{pspicture}

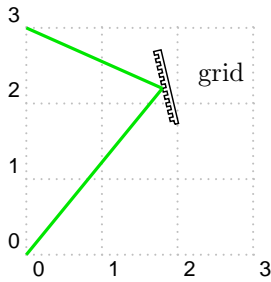
```



```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,3){A}
3   \pnode(1.8,2.2){G}
4   \pnode(0,0){B}
5   \optgrid[beam,%
6     optgridcount=6,%
7     optgriddepth=0.2,%
8     optgridheight=0.3](A)(G)(B){grid}
9 \end{pspicture}

```



```

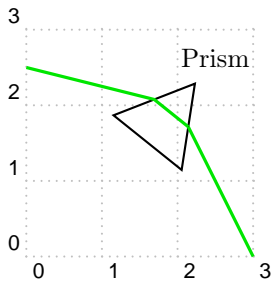
1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,3){A}
3   \pnode(1.8,2.2){G}
4   \pnode(0,0){B}
5   \optgrid[beam, optgridtype=binary](A)(G)(B){grid}
6 \end{pspicture}

```

### 3.14 Prism

**prismsize:** <num> (*default:* 1)

**prismangle:** <num> (*default:* 60)



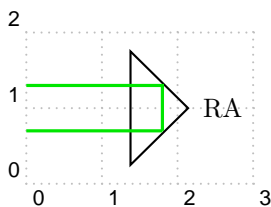
```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,2.5){A}
3   \pnode(2,2){G}
4   \pnode(3,0){B}
5   \optprism[beam](A)(G)(B){Prism}
6 \end{pspicture}

```

### 3.15 Right-Angle Prism

**raprismsize:** <num> (*default:* 1)



```

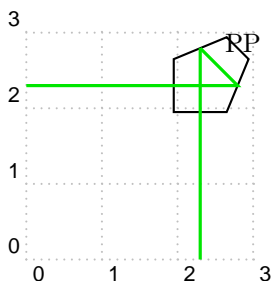
1 \begin{pspicture}[showgrid=true](3,2)
2   \pnode(0,1.3){A}
3   \pnode(1.8,1){G}
4   \pnode(0,0.7){B}
5   \rightangleprism[beam](A)(G)(B){RA}
6 \end{pspicture}

```



### 3.16 Penta Prism

pentaprismsize: <num> (default: 1)



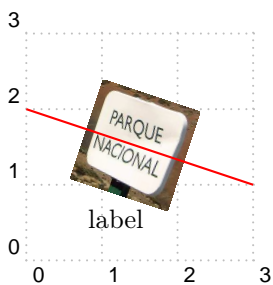
```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,2.3){A}
3   \pnode(2.3,2.3){G}
4   \pnode(2.3,0){B}
5   \pentaprism[beam](A)(G)(B){PP}
6 \end{pspicture}

```

### 3.17 Custom Components

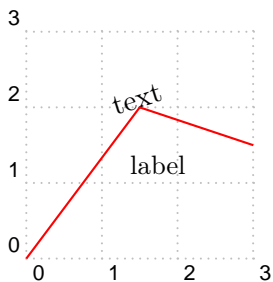
The macros `\optdipole` and `\opttripole` allow using everything as optical component. If you want to use a certain component several times, you should define it as a new component. For details see sec. 5.2.



```

1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,2){A}
3   \pnode(3,1){B}
4   \optdipole[labeloffset=1](A)(B){%
5     \rput(0,0){%
6       \includegraphics[scale=0.25]{parque-nacional}
7     }
8   }{label}
9   \psline[linecolor=red](A)(B)
10 \end{pspicture}

```



```

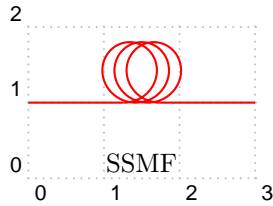
1 \begin{pspicture}[showgrid=true](3,3)
2   \pnode(0,0){A}
3   \pnode(1.5,2){G}
4   \pnode(3,1.5){B}
5   \opttripole(B)(G)(A){\rput[b](0,0){text}}{label}
6   \psline[linecolor=red](A)(G)(B)
7 \end{pspicture}

```

## 4 Fiber-Optical Objects

### 4.1 Fiber

**fiberloops:** <integer> (*default:* 3)  
**fiberloopradius:** <num> (*default:* 0.3)  
**fiberloopsep:** <num> (*default:* 0.3)



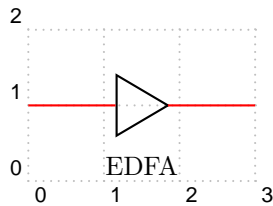
```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optfiber(0,1)(3,1){SSMF}
3 \end{pspicture}

```

### 4.2 Amplifier

**optampsize:** <num> (*default:* 0.8)



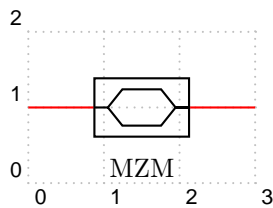
```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optamp(0,1)(3,1){EDFA}
3 \end{pspicture}

```

### 4.3 Mach-Zehnder Modulator

**optmzmsize:** <num> (*default:* 0.8)



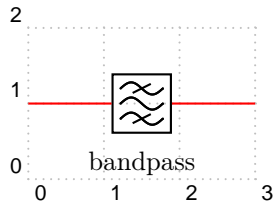
```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optmzm(0,1)(3,1){MZM}
3 \end{pspicture}

```

### 4.4 Filter

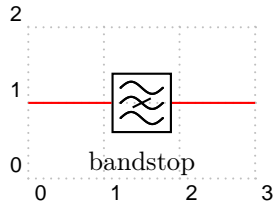
**filtersize:** <num> (*default:* 0.8)  
**filtertype:** bandpass|bandstop (*default:* bandpass)



```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optfilter(0,1)(3,1){bandpass}
3 \end{pspicture}

```



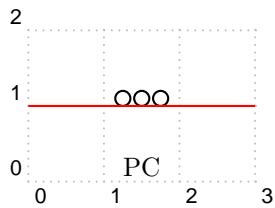
```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optfilter[filtertype=bandstop](0,1)(3,1){bandstop}
3 \end{pspicture}

```

#### 4.5 Polarization Controller

**polcontrolsize:** <num> (*default:* 0.1)



```

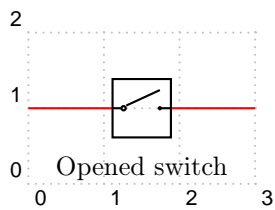
1 \begin{pspicture}[showgrid=true](3,2)
2   \polcontrol(0,1)(3,1){PC}
3 \end{pspicture}

```

#### 4.6 Optical Switch

**switchsize:** <num> (*default:* 0.8)

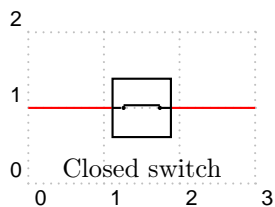
**switchstyle:** opened|closed (*default:* opened)



```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optswitch(0,1)(3,1){Opened switch}
3 \end{pspicture}

```



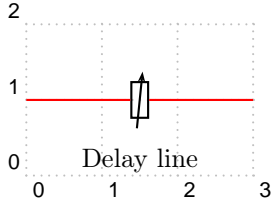
```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optswitch[switchstyle=closed](0,1)(3,1){Closed switch}
3 \end{pspicture}

```

## 4.7 Fiber Delay Line

fdlsize: <num> (default: 0.8)



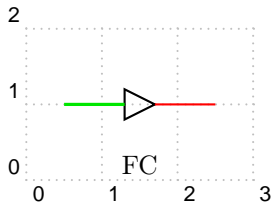
```

1 \begin{pspicture}[showgrid=true](3,2)
2   \fiberdelayline(0,1)(3,1){Delay line}
3 \end{pspicture}

```

## 4.8 Fiber Collimator

fibcolsize: <num> (default: 0.4)



```

1 \begin{pspicture}[showgrid=true](3,2)
2   \fibercollimator(0.5,1)(2.5,1){FC}
3 \end{pspicture}

```

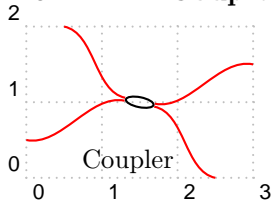
## 4.9 Coupler

couplersize: <num> (default: 0.1)

couplersep: <num> (default: 0.1)

couplertype: none|elliptic (default: elliptic)

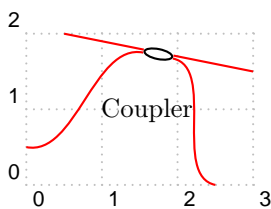
### 4.9.1 $2 \times 2$ Coupler



```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optcoupler(0.5,2)(0,0.5)(3,1.5)(2.5,0){Coupler}
3 \end{pspicture}

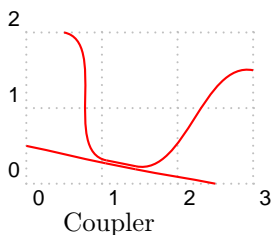
```



```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optcoupler[align=top](0.5,2)(0,0.5)(3,1.5)(2.5,0){Coupler}
3 \end{pspicture}

```

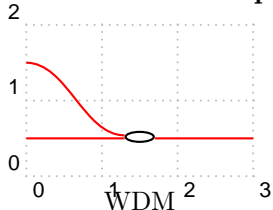


```

1 \begin{pspicture}[showgrid=true](3,2)
2   \optcoupler[align=bottom,
3     couplertype=none](0.5,2)(0,0.5)(3,1.5)(2.5,0){Coupler}
4 \end{pspicture}

```

## 4.9.2 WDM Coupler

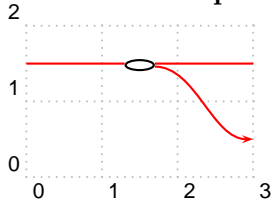


```

1 \begin{pspicture}[showgrid=true] (3,2)
2   \wdmcoupler[align=bottom] (0,1.5) (0,0.5) (3,0.5){WDM}
3 \end{pspicture}

```

## 4.9.3 WDM Splitter



```

1 \begin{pspicture}[showgrid=true] (3,2)
2   \newpsstyle{FiberOut2}{style=Fiber, arrows=->}
3   \wdmsplitter[align=top] (0,1.5) (3,1.5) (3,0.5){}
4 \end{pspicture}

```

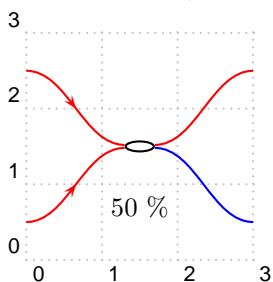
## 4.10 Fiber Styles

**usefiberstyle:** <boolean> (default: true)  
**Fiber:** <psstyle> (default: linecolor=red)  
**FiberIn:** <psstyle> (default: style=Fiber)  
**FiberIn1:** <psstyle> (default: style=FiberIn)  
**FiberIn2:** <psstyle> (default: style=FiberIn)  
**FiberOut:** <psstyle> (default: style=Fiber)  
**FiberOut1:** <psstyle> (default: style=FiberOut)  
**FiberOut2:** <psstyle> (default: style=FiberOut)

All these psstyles control the appearance of the fiber parts before and after each components. The styles can be redefined with `\newpsstyle` or changed with `\addtopsstyle`. For optical systems it is not possible to define a unique input and a unique output as most components can be used bidirectionally. Therefore, I refer to the input as the part from the first node to the component and to the output as the part from the component to the second node.

The basic style is **Fiber** which is the parent of all other styles. **FiberIn** inherits from **Fiber** and defines the style of the input fiber. Analogously **FiberOut** controls the style of the output fiber. If you want to change the input and output fiber styles you should use `\addtopsstyle` as then the inheritance from the parent style **Fiber** remains.

The other styles are used by the fiber couplers (`\optcoupler`, `\wdmcoupler` and `\wdmsplitter`). **FiberIn1** affects the upper input fiber, **FiberIn2** the lower input fiber, **FiberOut1** the upper output fiber and **FiberOut2** the lower output fiber. If the object has only one input (e.g. `\wdmsplitter`), **FiberIn** is used.



```

1 \begin{pspicture}[showgrid=true] (3,3)
2   \addtopsstyle{FiberIn}{ArrowInside=->, arrowscale=1.2}
3   \addtopsstyle{FiberOut2}{linecolor=blue}
4   \optcoupler (0,2.5) (0,0.5) (3,2.5) (3,0.5){50~\%}
5 \end{pspicture}

```

In addition to the `psstyles` there exist corresponding `new...` and `addto...` parameter keys for each of them.

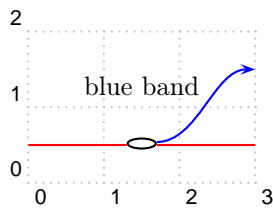
```
1 \psset{addtoFiberIn={arrows=->, arrowscale=1.3}}
```

is equivalent to

```
1 \addtopsstyle{FiberIn}{arrows=->, arrowscale=1.3}
```

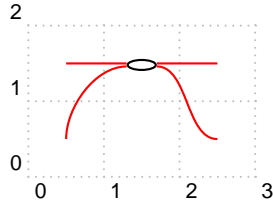
Accordingly `newFiberIn` corresponds to `\newpsstyle{FiberIn}{...}`.

At first glance these keys make no sense. The reason why I introduced them was to be able to define special couplers with `\newpsobject`. This is only possible if all modifications can be expressed as parameter keys. Consider for example a WDM splitter which only couples out a certain spectral range of the input and you want to mark the output with an arrow:



```
1 \begin{pspicture}[showgrid=true](3,2)
2   \newpsobject{mywdmsplitter}{wdmsplitter}{addtoFiberOut1={arrows=->,
3     arrowscale=1.3, linecolor=blue}, labelangle=180, align=bottom}
4   \mywdmsplitter(0,0.5)(3,1.5)(3,0.5){blue band}
5 \end{pspicture}
```

Or if you need a coupler with a particular input angle you can do it by extending the appropriate fiber style:

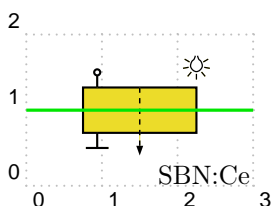


```
1 \begin{pspicture}[showgrid=true](3,2)
2   \newpsobject{mycoupler}{optcoupler}{addtoFiberIn2={angleA=90},
3     align=top}
4   \mycoupler(0.5,1.5)(0.5,0.5)(2.5,1.5)(2.5,0.5){}
5 \end{pspicture}
```

## 5 Defining New Objects

### 5.1 Customized Versions of Existing Macros

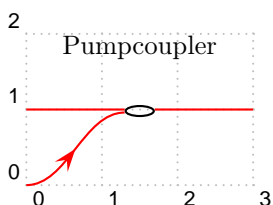
The easiest way to define your own components is to use the `\newpsoobject` macro. With this you can define a new component using predefined objects with a set of options. These options serve only as default values and can be overridden. The following examples defines a new object `\sbn` for the special crystal used in Sec. 3.5.



```

1 \newpsoobject{sbn}{crystal}{crystalwidth=1.5, crystalheight=0.6,
2   voltage, lamp, labelangle=45, labeloffset=1.2, fillstyle=solid,
3   fillcolor=yellow!90!black}
4 \begin{pspicture}[showgrid=true] (3,2)
5   \sbn(0,1) (3,1){SBN:Ce}
6   \psline[style=Beam] (0,1) (3,1)
7 \end{pspicture}

```

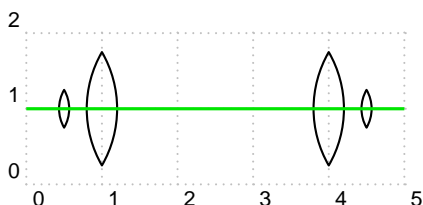


```

1 \newpsoobject{pumpcoupler}{wdmcoupler}{align=top, labelangle=180,
2   addtoFiberIn2={ArrowInside=->, arrowscale=2}}
3 \begin{pspicture}[showgrid=true] (3,2)
4   \pumpcoupler(0,1) (0,0) (3,1){Pumpcoupler}
5 \end{pspicture}

```

If you need more than one type of lenses in your setup it is very cumbersome to specify all parameters every time.



```

1 \newpsoobject{MOLensIn}{lens}{lens=0.5 0.5 0.5}
2 \newpsoobject{MOLensOut}{lens}{lens=1.5 1.5 1.5}
3 \begin{pspicture}[showgrid=true] (5,2)
4   \pnode(0,1){A}\pnode(5,1){B}
5   \MOLensIn[abspos=0.5] (A) (B){}
6   \MOLensOut[abspos=1] (A) (B){}
7   \MOLensOut[abspos=4] (A) (B){}
8   \MOLensIn[abspos=4.5] (A) (B){}
9   \psline[style=Beam] (A) (B)
10 \end{pspicture}

```

### 5.2 Defining New Objects

Since version 1.2 `pst-optexp` provides some high-level macros to allow very convenient definition of completely new components. The macro `\newOptexpDipole` generates all organizing code for a new free-ray component. All you have to do is to define a new "drawing" macro `\mycomponent@iii` which contains all drawing code. Analogously `\newOptexpDipoleNolabel` defines a new free-ray object which takes no label (like `\polarization`) and `\newOptexpTripole` defines a new reflective component.

New fiber-optical components can be defined using `\newOptexpFiberDipole`. This macro differs from its free-ray analogous only in that it presets `fiber` and hence directly connects the

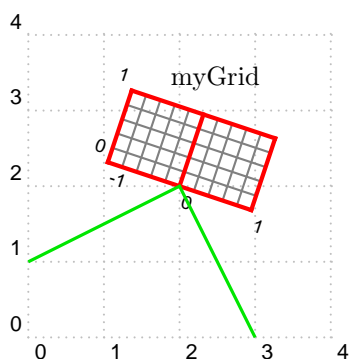
component with the nodes. The first node in the parameter list gets connected with a node `tempNode@A@`, the second node with a node `tempNode@B@`. These two internal nodes are preset to (0,0) and can be overwritten within the drawing macro.

The syntax of the macros is

```
1 \newOptexpDipole[fixed options]{name}{default options}
2 \newOptexpDipoleNolabel[fixed options]{name}{default options}
3 \newOptexpTripole[fixed options]{name}{default options}
4 \newOptexpFiberDipole[fixed options]{name}{default options}
```

The `default options` are simply a list of PSTricks parameters which are taken as defaults for the new component. The optional argument allows setting of parameters which cannot be overridden later.

This is illustrate a bit more in the next code snippet, which also shows how the coordinate system is handled withing the `\mycomponent@iii` macro.



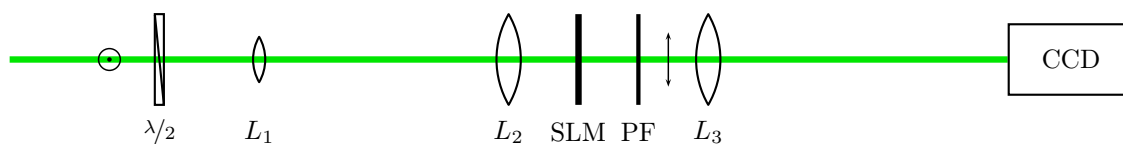
```
1 \newOptexpTripole{mygrid}{subgriddiv=5, griddots=0,
   subgridwidth=\pslinewidth, gridwidth=2\pslinewidth}
2 \makeatletter
3 \def\mygrid@iii{% put here all PSTricks drawing code
4   \psgrid(-1,0)(1,1)
5 }%
6 \makeatother
7 \begin{pspicture}[showgrid=true](4,4)
8   \pnode(0,1){A}\pnode(2,2){G}\pnode(3,0){B}
9   \mygrid[gridcolor=red,labeloffset=1.5](A)(G)(B){myGrid}
10  \psline[style=Beam](A)(G)(B)
11 \end{pspicture}
```

The default position of the label reference point is (0,0). If you want to change this, you have to define a new pnode named `tempNode@Label` in the `\mycomponent@iii` macro.

If you create a new component, please send it to me then I can incorporate this in a new released version.



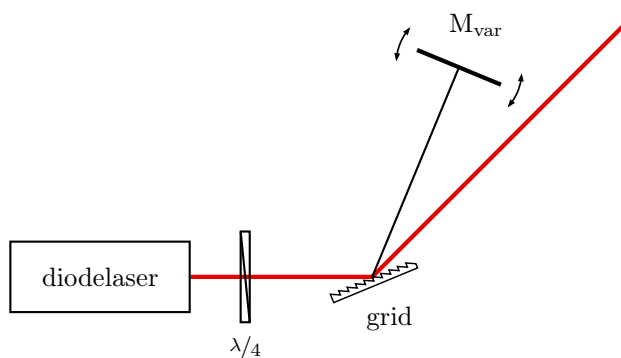
## 6 Examples



```

1 \begin{pspicture}(0,0.2)(12,1.8)
2 \pnode(0,1.2){Start}\pnode(11,1.2){CCD}
3 \psline[linewidth=2\pslinewidth,style=Beam](Start)(CCD)
4 \polarization[poltype=perp,position=0.1](Start)(CCD)
5 \optretplate[position=0.15](Start)(CCD){$\frac{\lambda}{2}$}
6 \lens[lensheight=0.5,
7       lensradius=0.5,
8       position=0.25](Start)(CCD){$L_1$}
9 \lens[position=0.5](Start)(CCD){$L_2$}
10 \optplate[position=0.57, platelinewidth=3\pslinewidth](Start)(CCD){SLM}
11 \optplate[position=0.63](Start)(CCD){PF}
12 \polarization[position=0.66](Start)(CCD)
13 \lens[position=0.7](Start)(CCD){$L_3$}
14 \optbox[endbox,labeloffset=0](Start)(CCD){CCD}
15 \end{pspicture}

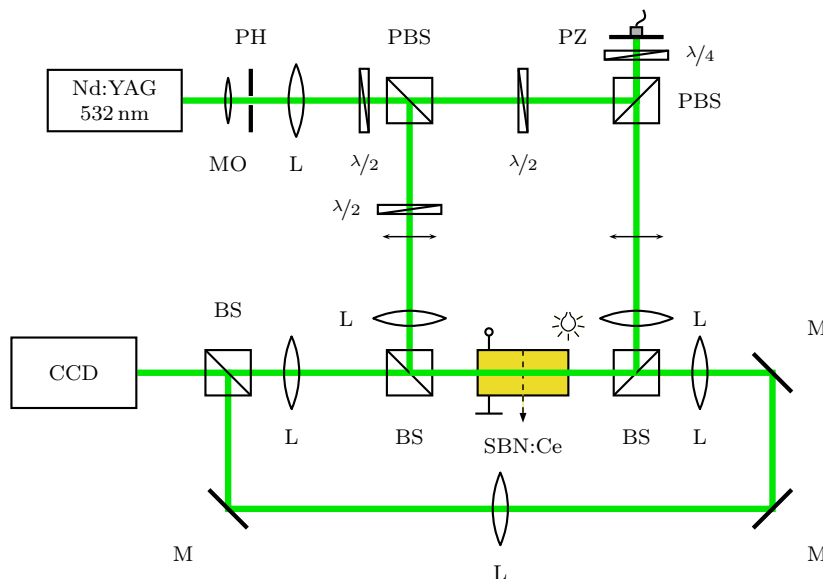
```



```

1 \begin{pspicture}(-4,-1)(3,3)
2 \psset{labeloffset=0.5}
3 \pnode(-2,0){LaserOut}
4 \pnode(0,0){Grid}
5 \pnode(4;45){Out}
6 \pnode(2.5;67.5){Mvar}
7 \psline[linewidth=2\pslinewidth,
8         linecolor=red!90!black](LaserOut)(Grid)(Out)\psline(Grid)(Mvar)
9 \optbox[endbox, optboxwidth=2, labeloffset=0](Grid)(LaserOut){diodelaser}
10 \optretplate[position=0.3, labeloffset=0.8](LaserOut)(Grid){$\frac{\lambda}{4}$}
11 \optgrid(LaserOut)(Grid)(Out){grid}
12 \mirror[variable](Grid)(Mvar)(Grid){M$_{\mathrm{var}}$}
13 \end{pspicture}

```

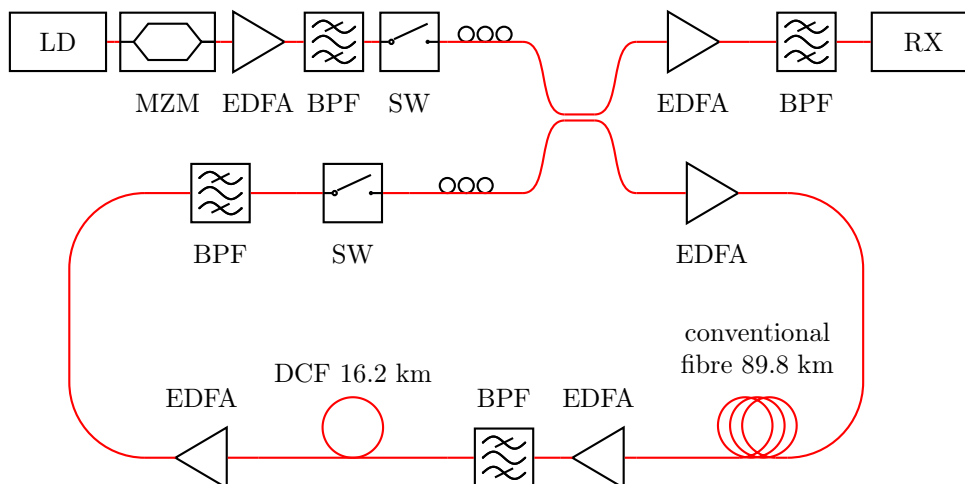


```

\begin{pspicture}(0,-0.4)(9,6)
\pnode(1.5,5){Laser}\pnode(4,5){PBS}\pnode(6.5,5){PBS2}
\pnode(6.5,5.7){piezo}\pnode(4,2){BSFwd}\pnode(6.5,2){BSBwd}
\pnode(2,2){BS4f}\pnode(2,0.5){M4f3}\pnode(8,2){M4f1}
\pnode(8,0.5){M4f2}\pnode(1,2){CCD}
\psline[style=Beam,linewidth=2\pslinewidth]{%
(Laser)(PBS2)(piezo)(BSBwd)(M4f1)(M4f2)(M4f3)(BS4f)(CCD)
\psline[style=Beam,linewidth=2\pslinewidth](PBS)(BSFwd)(BS4f)
\psset{mirrorwidth=0.6,plateheight=0.7,outerheight=0.7,labeloffset=0.7,
labelstyle=\scriptsize,lensheight=0.8,lenswidth=0.2,bssize=0.5}
\optbox[endbox,optboxwidth=1.5,optboxheight=0.7,labeloffset=0]{%
(PBS)(Laser){\parbox{1.5cm}{\centering Nd:YAG\ 532\,nm}}
\lens[lensheight=0.5,position=0.2](Laser)(PBS){MO}
\pinhole[position=0.3,labelangle=180](Laser)(PBS){PH}
\lens[position=0.5](Laser)(PBS){L}
\optretplate[position=0.8](Laser)(PBS){$\nicefrac{\lambda}{2}$}
\beamsplitter(Laser)(PBS)(BSFwd){PBS}
\optretplate[position=0.4](PBS)(BSFwd){$\nicefrac{\lambda}{2}$}
\polarization(PBS)(BSFwd)\polarization(PBS2)(BSBwd)
\lens[position=0.8](PBS)(BSFwd){L}
\optretplate(PBS)(PBS2){$\nicefrac{\lambda}{2}$}
\beamsplitter(PBS)(PBS2)(piezo){PBS}
\optretplate[abspos=0.5](PBS2)(piezo){$\nicefrac{\lambda}{4}$}
\mirror[mirrortype=piezo,labelangle=90](PBS2)(piezo)(PBS2){PZ}
\lens[position=0.8,labelangle=180](PBS2)(BSBwd){L}
\beamsplitter(PBS)(BSFwd)(BSBwd){BS}
\beamsplitter[labelangle=-90](PBS2)(BSBwd)(BSFwd){BS}
\crystal[crystalwidth=1,crystalheight=0.5,voltage,lamp,fillstyle=solid,
fillcolor=yellow!90!black,labeloffset=0.8](BSFwd)(BSBwd){SBN:Ce}
\mirror(BSBwd)(M4f1)(M4f2){M}\mirror(M4f1)(M4f2)(M4f3){M}
\lens[labelangle=180](M4f2)(M4f3){L}\mirror(M4f2)(M4f3)(BS4f){M}
\beamsplitter(M4f3)(BS4f)(CCD){BS}\optbox[endbox,labeloffset=0](BS4f)(CCD){CCD}
\lens[abspos=0.7](BS4f)(BSFwd){L}\lens[abspos=0.7](BSBwd)(M4f1){L}
\psline[style=Beam,linewidth=2\pslinewidth](BSFwd)(BSBwd)
\end{pspicture}

```

The following schematic configuration of a recirculating loop was adapted from the publication N. Kikuchi, S. Sasaki and K. Sekine, "10 Gbit/s dispersion-compensated transmission over 2245 km conventional fibres in a recirculating loop", Electron. Lett. 31 (5), 375 (1995).



```

1 \psset{unit=1cm}
2 \begin{pspicture}(0.5,4)(13.2,10.5)
3   \laser[labeloffset=0](2.1,10)(2,10){LD}
4   \optmzm(2.1,10)(3.5,10){MZM}
5   \optamp(3.5,10)(4.5,10){EDFA}
6   \optfilter(4.5,10)(5.5,10){BPF}
7   \optswitch(5.5,10)(6.5,10){SW}
8   \polcontrol(6.5,10)(7.5,10){}
9   \optcoupler[couplertype=none,couplersep=0.2](7.5,10)(7.5,8)(9,10)(9,8){}
10  \optamp(9,10)(10.5,10){EDFA}
11  \optfilter(10.5,10)(12,10){BPF}
12  \optbox[endbox, labeloffset=0, conn=f-f](12,10)(12.1,10){RX}
13  % loop
14  \optamp(9,8)(11,8){EDFA}
15  \psline[linear=1,style=Fiber](11,8)(12,8)(12,4.5)(11,4.5)
16  \optfiber[labelalign=b, labeloffset=-1,
17    position=0.8](9,4.5)(11,4.5){\begin{tabular}{c}conventional\\fibre 89.8~km\end{tabular}}
18  \optamp(9,4.5)(8,4.5){EDFA}
19  \optfilter(8,4.5)(6.5,4.5){BPF}
20  \optfiber[fiberloops=1, labeloffset=-1, labelalign=b](4,4.5)(6.5,4.5){DCF 16.2~km}
21  \optamp(4,4.5)(2.5,4.5){EDFA}
22  \psline[style=Fiber,linear=1](2.5,4.5)(1.5,4.5)(1.5,8)(2.5,8)
23  \optfilter(2.5,8)(4.5,8){BPF}
24  \optswitch(4.5,8)(6,8){SW}
25  \polcontrol(6,8)(7.5,8){}
26 \end{pspicture}

```

## 7 Complete List of Parameters

parameter	values	default
<b>abspos</b>	<value>	\@empty
<b>angle</b>	<value>	0
<b>beam</b>	<boolean>	false
<b>bssize</b>	<num>	0.8
<b>caxisinv</b>	<boolean>	false
<b>caxislength</b>	<num>	0.6
<b>compname</b>	<string>	{}
<b>conn</b>	i-i   o-o   f-f	-
<b>couplersep</b>	<num>	0.1
<b>couplersize</b>	<num>	0.1
<b>couplertype</b>	none elliptic	elliptic
<b>crystalheight</b>	<num>	0.8
<b>crystalwidth</b>	<num>	2
<b>detsize</b>	<num>	0.5
<b>detstyle</b>	round   diode	round
<b>doveprismsize</b>	<num>	1
<b>endbox</b>	<boolean>	false
<b>extnode</b>	<ref string>	\@empty
<b>extnodename</b>	<string>	Ext
<b>fdlsize</b>	<num>	0.8
<b>fibcolsize</b>	<num>	0.4
<b>fiber</b>	<boolean>	false
<b>Fiber</b>	<psstyle>	linecolor=red
<b>FiberIn</b>	<psstyle>	style=Fiber
<b>FiberIn1</b>	<psstyle>	style=FiberIn
<b>FiberIn2</b>	<psstyle>	style=FiberIn
<b>fiberloopradius</b>	<num>	0.3
<b>fiberloops</b>	<integer>	3
<b>fiberloopsep</b>	<num>	0.3
<b>FiberOut</b>	<psstyle>	style=Fiber
<b>FiberOut1</b>	<psstyle>	style=FiberOut
<b>FiberOut2</b>	<psstyle>	style=FiberOut
<b>filtertype</b>	bandpass bandstop	bandpass
<b>innerheight</b>	<num>	0.1
<b>labelalign</b>	<\rput ref string>	c
<b>labelangle</b>	<num>	0
<b>labeloffset</b>	<num>	1
<b>labelref</b>	relative relgrav global	relgrav
<b>labelstyle</b>	<macro>	\small
<b>lamp</b>	<boolean>	false

parameter	values	default
lampscale	<num>	0.3
lens	<num> [<num> [<num> [<num>]]]	\@empty
lensheight	<num>	1
lensradius	<num> [<num>]	\@empty
lensradiusleft	<num>	1
lensradiusright	<num>	1
lenswidth	<num>	0.3
mirrordepth	<num>	0.08
mirrorlinewidth	<num>	2\pslinewidth
mirrorradius	<num>	0
mirrortype	normal piezo extended	normal
mirrorwidth	<num>	1
optampsize	<num>	0.8
optboxheight	<num>	0.5
optboxwidth	<num>	1
optdiodesize	<num>	0.8
optgridcount	<integer>	10
optgriddepth	<num>	0.05
optgridheight	<num>	0.1
optgridlinewidth	<num>	0.7\pslinewidth
optgridtype	blazed binary	blazed
optgridwidth	<num>	1
optional	<boolean>	false
optmzmsize	<num>	0.8
outerheight	<num>	1
pentaprismsize	<num>	1
phlinewidth	<num>	2\pslinewidth
plateheight	<num>	1
plateheight	<num>	1
platelinewidth	<num>	2\pslinewidth
platewidth	<num>	0.1
polcontrolsiz	<num>	0.1
polline	<num>	0.7\pslinewidth
polsiz	<num>	0.6
poltype	parallel perp misc lcirc rcirc	parallel
position	<value>	\@empty
prismangle	<num>	60
prismsize	<num>	1
raprismsize	<num>	1
reverse	<boolean>	false
showoptdots	<boolean>	false
switchsize	<num>	0.8
switchstyle	opened closed	opened

parameter	values	default
<b>thicklens</b>	<boolean>	false
<b>usefiberstyle</b>	<boolean>	true
<b>variable</b>	<num>	false
<b>voltage</b>	<boolean>	false

## 8 Todo

The next thing I will add are components with multiple internal reflections, as right-angle, penta and dove prisms. The code is almost ready, I just need to think a bit more about the best way to provide access to the nodes that are newly defined within the components.

Drawing of extended beams with focusing and so on could be integrated to some extent in future versions. But as the topic is rather difficult if you want to do it properly (components should be placed above the beam, but the new nodes are available only when the component is drawn) it could take very long until this feature will be implemented.

## 9 Acknowledgements

I thank all the people of the PSTricks mailinglist for the continuous help, especially Herbert Voß.

## 10 History

For the package history see file Changes that is distributed with the package.