

PSTricks

pst-optexp

A PSTricks package to draw optical experimental setups

2009/11/05

Package author(s):
Christoph Bersch

Contents

1	Introduction	4
2	Concept and General Behavior	4
2.1	Concept	4
2.2	General Settings	4
2.3	Using PSStyles	5
2.4	Positioning	6
2.5	Labels	6
2.6	Named Objects	7
2.7	Nodes For External Usage	8
2.8	Connecting Objects	9
3	Free-Ray Objects	14
3.1	Lens	14
3.2	Optical Plate	15
3.3	Retardation Plate	15
3.4	Pinhole	16
3.5	Crystal	16
3.6	Box	16
3.7	Detector	18
3.8	Optical Diode	18
3.9	Dove Prism	18
3.10	Polarization	19
3.11	Mirror	19
3.12	Beamsplitter	20
3.13	Optical Grid	21
3.14	Prism	22
3.15	Right-Angle Prism	22
3.16	Penta Prism	22
3.17	Custom Components	23
4	Fiber-Optical Objects	23
4.1	Fiber	23
4.2	Amplifier	24
4.3	Mach-Zehnder Modulator	24
4.4	Filter	24
4.5	Polarization Controller	24
4.6	Isolator	25
4.7	Optical Switch	25
4.8	Fiber Delay Line	25
4.9	Fiber Polarizer	25
4.10	Fiber Collimator	26
4.11	Coupler	26
4.12	Fiber Styles	28

5 Defining New Objects	29
5.1 Customized Versions of Existing Macros	29
5.2 Defining New Objects	30
6 Examples	32
7 Complete List of Parameters	38
8 Complete List of Styles	40
9 Requirements	40
10 Todo	41
11 Acknowledgements	41

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. 2.4) 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. 2.8.

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 to the reference nodes. Every input and output fiber can be flexibly customized for each object (see Sec. 2.3). 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. 4.11.

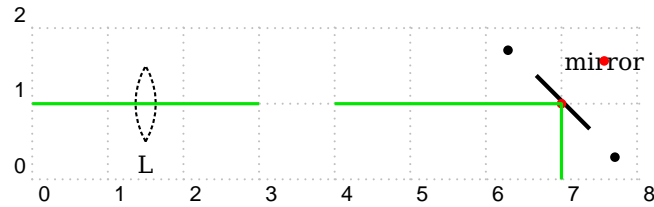
Some hybrid dipoles (`optbox`, `detector` etc.) can be used both as fiber-optical or free-ray elements. The way they are treated regarding the connections to the reference points can be controlled by the parameters explained in Sec. 2.8.

2.2 General Settings

`angle`: <degree> (*default*: 0)
`compshift`: <num> (*default*: 0)
`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.



```

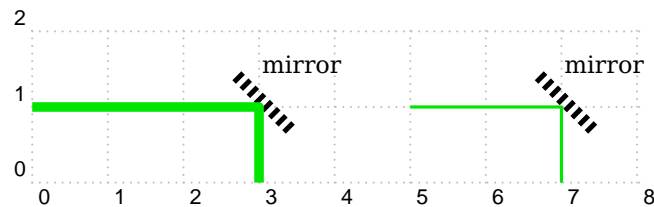
1 \begin{pspicture}[showgrid=true](8,2)
2   \psset{beam}
3   \lens[optional](0,1)(3,1){L}
4   \mirror[showoptdots](4,1)(7,1)(7,0){mirror}
5 \end{pspicture}

```

2.3 Using PSStyles

OptionalStyle: <psstyle>
 addtoOptComp: <psstyle> (default:)
 newOptComp: <psstyle> (default:)
 OptComp: <psstyle>

`OptComp` affects only the appearance of the optical components. This was introduced, because using only the standard graphics parameters changes also the connections that are drawn within the component.



```

1 \begin{pspicture}[showgrid=true](8,2)
2   \psset{beam}
3   \addtopsstyle{OptComp}{linestyle=dashed, dash=2pt 2pt}
4   % wrong, also beam width is changed
5   \mirror[linewidth=3\pslinewidth](0,1)(3,1)(3,0){mirror}
6   % correct result
7   \mirror[addtoOptComp={linewidth=3\pslinewidth}](5,1)(7,1)(7,0){mirror}
8 \end{pspicture}

```

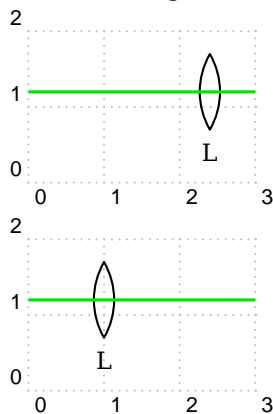
2.4 Positioning

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 only not available for the free-ray tripoles.

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



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

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

2.5 Labels

labeloffset: <num> (default: 0.8)

labelangle: <num> (default: 0)

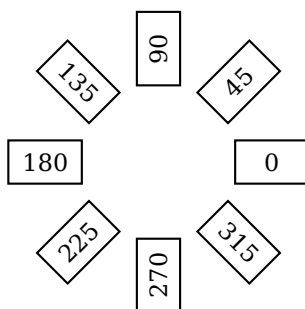
labelstyle: <macro> (default: \small)

labelalign: <ref string>¹ (default: c)

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

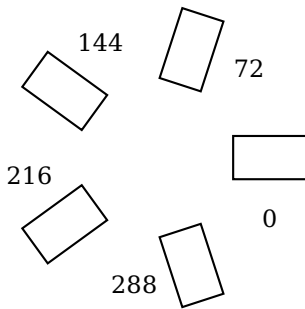
label: <offset> <angle> <ref string> <labelref> (default:)

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 refof 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.

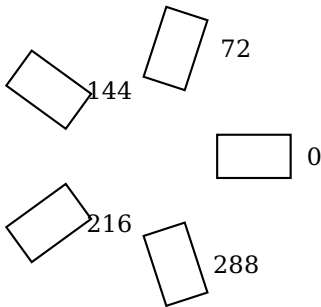


```
\begin{pspicture}(-2,-2)(2.5,2)
2 \multido{\i=0+45}{8}{%
3 \optbox[endbox,
4 labelref=relative,
5 labeloffset=0,
6 optboxwidth=1,
7 optboxheight=0.6](0,0)(1;\i){\i}
8 }
9 \end{pspicture}
```

¹ A <ref string> is any combination of c (center), t (top), b (bottom), l (left), r (right)

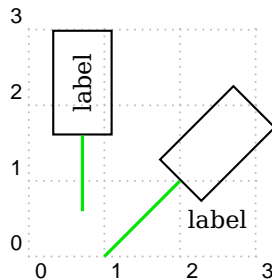


```
\begin{pspicture}(-2,-2)(2.5,2)
  \multido{\i=0+72}{5}{%
    \optbox[endbox,
      labelref=relgrav,
      optboxwidth=1,
      optboxheight=0.6](0,0)(1;\i){\i}
  }
\end{pspicture}
```



```
\begin{pspicture}(-2,-2)(2.5,2)
  \multido{\i=0+72}{5}{%
    \optbox[endbox,
      labelref=global,
      optboxwidth=1,
      optboxheight=0.6](0,0)(1;\i){\i}
  }
\end{pspicture}
```

`label` simplifies the simultaneous change of more than one label-related parameter. It takes up to four space-separated arguments. Unchanged arguments may be specified with a dot.



```
\begin{pspicture}[showgrid=true](0,0)(3,3)
  \psset{endbox, beam}
  \optbox[label=1 -45](1,0)(2,1){label}
  \optbox[label=0 . . relative](0.6,0.6)
    (0.6,1.6){label}
\end{pspicture}
```

2.6 Named Objects

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

Every `pst-optexp` object of an experimental setup can be assigned a name that is unique within one `pspicture` environment. The name is defined with the parameter `compname` which is definable only directly within a `pst-optexp` object:

```
1 \optbox[compname=MyBox](A)(B){Box} % valid use of 'compname'
2 \psset{compname=MyName} % not valid, gives an error
```

With this naming mechanisms one can access some special nodes of the component at any time after its definition:

node name	description
<compname>ExtNode	Node for external connections (<i>external node</i>)
<compname>Intern1	Node which should be connected to the first reference node. In the text we refer to this node as <i>left outer node</i>
<compname>Intern2	First internal node. As the nodes with higher numbers it is only available for objects with multiple internal beams (e.g. dove prism, see Sec. 3.9). They are called <i>internal nodes</i> .
⋮	
<compname>InternN	Node which should be connected to the second reference node. In the text this node is referred to as <i>right outer node</i>

Table 1: Naming conventions for special nodes which are created by named objects and can be accessed by the user after definition of the object.

If `compname` is empty, the external node has the name *ExtNode* and will be overwritten by any following object. The outer nodes are not accessible to the user and will also be overwritten by following object. The internal nodes are deleted after the object's definition.

These named objects are used to create permanent external nodes (see Sec. 2.7) and to connect objects after their definition (see Sec. 2.8).

2.7 Nodes For External Usage

`extnode: <ref string>2 (default: {})`

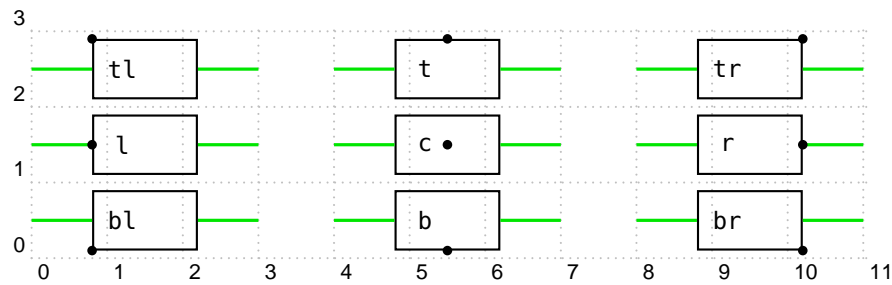
Some of the objects can provide a supplementary node for additional connections. A laser diode may be connected for example to a frequency synthesizer (use package `pst-circ`) or a detector to a computer.

`extnode` controls the position of the additional node and takes a `<ref string>` as its argument. By default this parameter is empty (`{}`) and no node is created.

The name of the new node depends on the `compname` parameter (see Sec. 2.6 for naming conventions). If `compname` is empty the new node is named *ExtNode* by default and overwritten by following objects.

Table. 2 shows all objects which provide an external node. Some allow any possible `<ref string>` for `extnode`, others have only one reasonable possibility (e.g. `piezo mirror`, see Sec. 3.11) which does not depend on the actual value of `extnode`.

² A `<ref string>` is any combination of c (center), t (top), b (bottom), l (left), r (right)



```

\begin{pspicture}[showgrid=true](11,3)
  \psset{conn=o-o, labelangle=-90, labeloffset=0.3}
  \optbox[extnode=tl](0,2.5)(3,2.5){\texttt{tl}}\psdot(ExtNode)
  \optbox[extnode=l](0,1.5)(3,1.5){\texttt{l}}\psdot(ExtNode)
  \optbox[extnode=bl](0,0.5)(3,0.5){\texttt{bl}}\psdot(ExtNode)
  \optbox[extnode=t](4,2.5)(7,2.5){\texttt{t}}\psdot(ExtNode)
  \optbox[extnode=c](4,1.5)(7,1.5){\texttt{c}}\psdot(ExtNode)
  \optbox[extnode=b](4,0.5)(7,0.5){\texttt{b}}\psdot(ExtNode)
  \optbox[extnode=tr](8,2.5)(11,2.5){\texttt{tr}}\psdot(ExtNode)
  \optbox[extnode=r](8,1.5)(11,1.5){\texttt{r}}\psdot(ExtNode)
  \optbox[extnode=br](8,0.5)(11,0.5){\texttt{br}}\psdot(ExtNode)
\end{pspicture}

```

Object	possible extnode positions	
<code>\optbox</code>	all (any combination of t, r, l and b)	
<code>\mirror</code>	one fixed position (only for <code>mirrortype=piezo</code>)	
<code>\optdetector</code>	one (for <code>dettype=round</code>)	
	all (for <code>dettype=diode</code>)	see <code>\optbox</code>
<code>\optmzm</code>	all	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 2: 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.

2.8 Connecting Objects

`conn`: <conn definition> (*default*: -)
`fiber`: alias for `conn=f-f`
`beam`: alias for `conn=o-i`
`connjoin`: <int> (*default*: 1)

Simple experimental setups with a few objects can usually be realized by defining some nodes, arranging the object in between and drawing the beams at the end. If, however, objects with changed internal optical path (all the prisms) or without

conn style	description	scope
i	Draw beam from the first reference node to its assigned outer node and then through all internal nodes and end at the other outer node.	optexp objects
o	Draw beam from the first reference node to its assigned outer node.	
f	Draw fiber from the first reference node to its assigned outer node.	
a	Left outer node	
A	Connect left outer node to all internal nodes and then to the right outer node	\drawbeam macro
b	Right outer node	
B	Connect right outer node to all internal nodes and then to the left outer node	

Table 3: All possible values for the <conn definition>, their detailed description and scope.

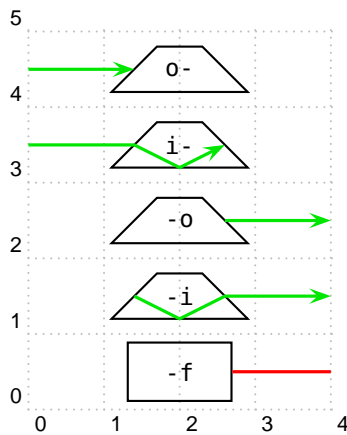
visible internal beam (optical diode) are involved, this simple method is not applicable anymore.

For this case several different possibilities of connecting objects are available: conn specifies the kind of connections in front of and behind the object. Its syntax is analogous to the PSTricks arrows parameter. By default it is set to - and no connections are drawn. Tab. 3 lists all possible values and their scope for the <conn definition>.

The first letter (before the dash) in the <conn definition> refers to which object node the first reference node should be connected to, the second letter (after the dash) affects the connection from the object to the second reference node. Tab. 3 lists all possibilities for conn within an object: f draws a fiber connection and o a beam connection to the appropriate outer node. i draws a beam connection to the appropriate outer node and then through all internal nodes and end at the other outer node. 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.

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 fiber connection style can be changed by adapting the Fiber* styles (see Sec. 4.12).

The way how to really use this kind of connections should become more clear after looking at the following examples in this section.

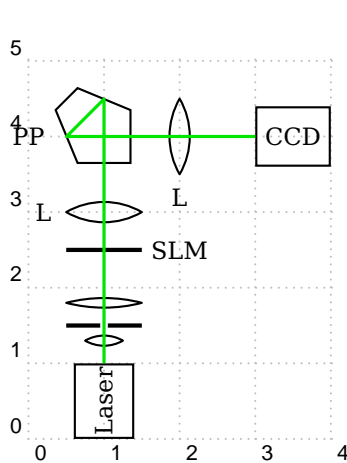


```

1 \begin{pspicture}[showgrid=true](4,5)
2   \addtopstyle{Beam}{arrows=->, arrowscale=1.5}
3   \psset{labeloffset=0}
4   \doveprism[conn=o-](0,4.5)(4,4.5){\texttt{o-}}
5   \doveprism[conn=i-](0,3.5)(4,3.5){\texttt{i-}}
6   \doveprism[conn=-o](0,2.5)(4,2.5){\texttt{-o}}
7   \doveprism[conn=-i](0,1.5)(4,1.5){\texttt{-i}}
8   \optbox[conn=-f](0,0.5)(4,0.5){\texttt{-f}}
9 \end{pspicture}

```

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



```

1 \begin{pspicture}[showgrid=true](4,5)
2   \node(1,1){A}\node(1,4){G}\node(3,4){B}
3   \optbox[endbox, labelref=relative, labeloffset
4     =0, optboxwidth=1](G)(A){Laser}
5   \lens[lens=0.5 0.5 0.5, abspos=0.3](A)(G){}
6   \pinhole[abspos=0.5](A)(G){}
7   \lens[lens=2, abspos=0.8](A)(G){}
8   \lens[abspos=2, labelangle=180](A)(G){L}
9   \optplate[abspos=1.5, labeloffset=1](A)(G){SLM}
10  \lens[abspos=1](G)(B){L}
11  \optbox[endbox, labeloffset=0, optboxwidth=1](G)
12  (B){CCD}
13  \pentaprism[beam, labeloffset=1](A)(G)(B){PP}
14 \end{pspicture}

```

Listing 1: Code example on how to use the conn parameter in experimental setups.

This method works unless objects without internal beams (e.g. an optical diode, Sec. 3.8) or with internal reflections (e.g. a Dove prism, Sec. 3.9) are used in the straight light paths of the setup. One 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 (Sec. 2.6) to another named object or a node.

```

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

```

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

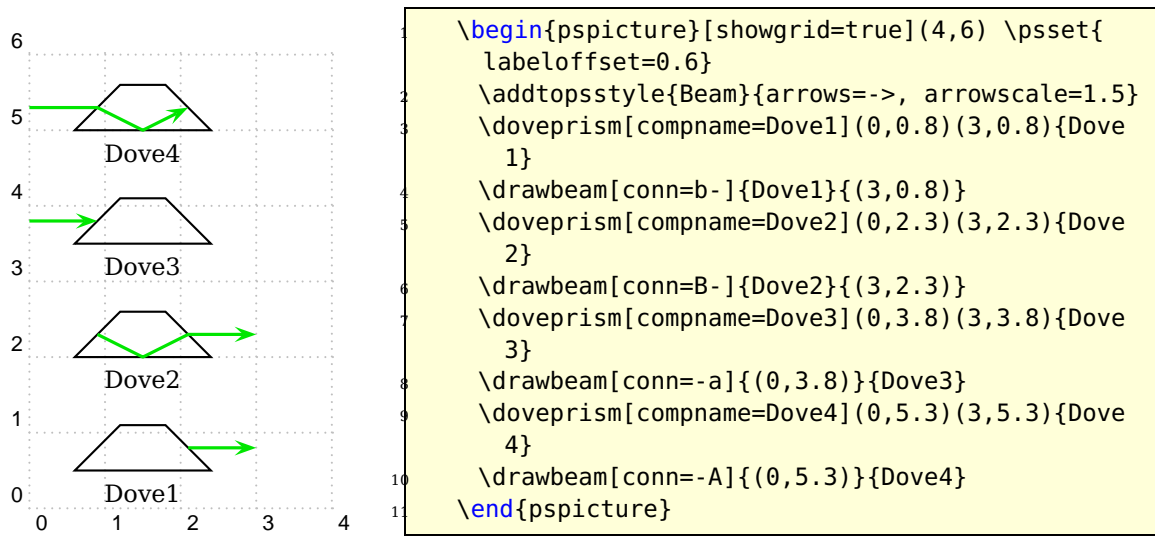
```

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

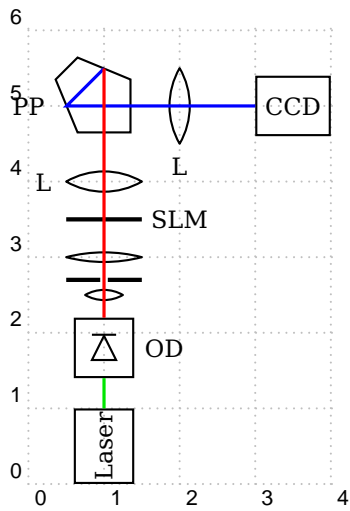
```

connects the named object *Obj* to the node.

The type of beam connection which `\drawbeam` draws 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 (*left outer node*) and node B (*right outer node*). Consequently, the two letters of parameter `conn` can take the values a, A, b or B when used together with `\drawbeam`. The detailed descriptions of the individual possibilities are listed in Tab. 3. The letter before the dash in `conn` refers to the <from> object, the other one to the <to> object. Again, the next examples should clarify how to apply the `\drawbeam` macro together with the different `conn` settings.



With the help of the `\drawbeam` macro we can adapt Listing 1 to use an optical diode before the beam clearing and connect it to the other components. In order to illustrate the beams that are drawn by the different mechanisms, they are colorcoded in the resulting Listing 2: *green* is the direct connecting of the optical diode, *red* is the `\drawbeam` connection and *blue* the direct connecting of the penta prism.



```

1 \begin{pspicture}[showgrid=true](4,6)
2   \node(1,1){A}\node(1,5){G}\node(3,5){B}
3   \optbox[endbox, labelref=relative, labeloffset
4     =0, optboxwidth=1](G)(A){Laser}
5   \lens[lens=0.5 0.5 0.5, abspos=1.5](A)(G){}
6   \pinhole[abspos=1.7](A)(G){}
7   \lens[lens=2, abspos=2](A)(G){}
8   \lens[abspos=3, labelangle=180](A)(G){L}
9   \optplate[abspos=2.5, labeloffset=1](A)(G){SLM}
10  \lens[abspos=1](G)(B){L}
11  \optbox[endbox, labeloffset=0, optboxwidth=1](G)
12    (B){CCD}
13  \optdiode[abspos=0.8, conn=o-, compname=OD](A)(G)
14    {OD}
15  \addtopsstyle{Beam}{linecolor=blue}
16  \pentaprism[conn=-i, labeloffset=1, compname=PP]
17    (A)(G)(B){PP}
18  \addtopsstyle{Beam}{linecolor=red}
19  \drawbeam[conn=b-a]{OD}{PP}
20 \end{pspicture}

```

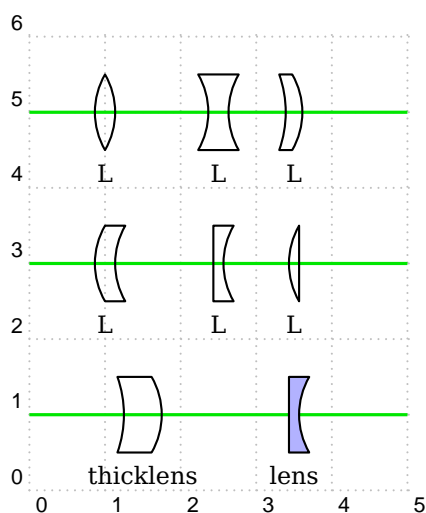
Listing 2: More sophisticated code example which employs both connecting methods. The beam connections are color-coded: *green* is the direct connecting of the optical diode, *red* is the `\drawbeam` connection and *blue* the direct connecting of the pentaprism.

3 Free-Ray Objects

The general appearance of all objects can be customized using the standard PSTricks parameter like `linewidth` or `fillstyle`. Some components allow changing a special part (e.g. for a piezo mirror) for which they use certain `psstyles`. For the automatic beam connections the `Beam` style is used.

3.1 Lens

`lensheight`: <num> (default: 1)
`lenswidth`: <num> (default: 0.2)
`lensradius`: <num> [<num>] (default: {})
`lensradiusleft`: <num> (default: 1)
`lensradiusright`: <num> (default: 1)
`lens`: <num> [<num> [<num> [<num>]]] (default: {})
`thicklens`: <boolean> (default: false)



```

1 \begin{pspicture}[showgrid=true](5,6)
2   % concave lenses
3   \pnode(0,5){A}\pnode(5,5){B}
4   \psline[style=Beam](A)(B)
5   \lens[position=0.2](A)(B){L}
6   \lens[lensradius=-1,position=0.5](A)(B){L}
7   \lens[lens=-1.5 1,position=0.7](A)(B){L}
8   % convex lenses
9   \pnode(0,3){A}\pnode(5,3){B}
10  \psline[style=Beam](A)(B)
11  \lens[position=0.2,lens=1 -1](A)(B){L}
12  \lens[lens=0 -1](A)(B){L}
13  \lens[lens=1 0,position=0.7](A)(B){L}
14  % thick lenses
15  \pnode(0,1){A}\pnode(5,1){B}
16  \psline[style=Beam](A)(B)
17  \lens[position=0.3, lens=-1.5 1 1 0.5,
18    thicklens](A)(B){thicklens}
19  \lens[lens=0 -1, position=0.7, fillstyle=
20    solid, fillcolor=blue!30!white](A)(B){lens}
21 \end{pspicture}
  
```

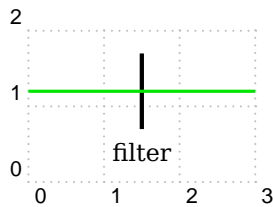
The shape of a lens is defined by its two surface radii. A negative radius gives a concave, a positive radius a convex and a radius of 0 a plain surface. The parameters `lensradiusleft` and `lensradiusright` allow to define independent values for both surfaces. `lensradius` sets both curvatures to the 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` which is use only if `thicklens` is set to `true`.

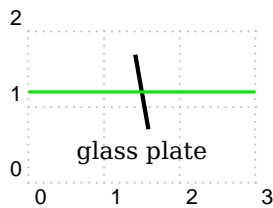
Compatibility: The whole implementation of the lens was changed in version 1.2. It allows a much more flexible definition of different lens types. However, I could not get full compatibility with the older way to define lens using only `lensheight` and `lenswidth`. To use this old behaviour, you have to set the `lens` type 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)
`angle`: <degree> (*default*: 0)



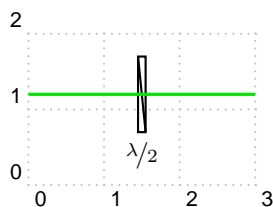
```
\begin{pspicture}[showgrid=true](3,2)
2 \optplate[beam](0,1.2)(3,1.2){filter}
3 \end{pspicture}
```



```
\begin{pspicture}[showgrid=true](3,2)
2 \optplate[angle=10, beam](0,1.2)(3,1.2){glass plate}
3 \end{pspicture}
```

3.3 Retardation Plate

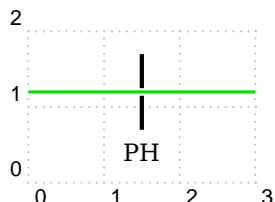
`plateheight`: <num> (*default*: 1)
`platewidth`: <num> (*default*: 0.1)



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

3.4 Pinhole

outerheight: <num> (*default: 1*)
 innerheight: <num> (*default: 0.1*)
 phlinewidth: <num> (*default: 2\pslinewidth*)

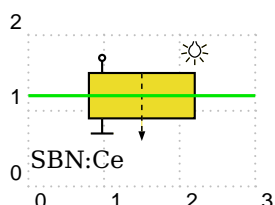


```
\begin{pspicture}[showgrid=true](3,2)
2 \node(0,1.2){A}
3 \node(3,1.2){B}
4 \pinhole[beam](A)(B){PH}
5 \end{pspicture}
```

3.5 Crystal

crystalwidth: <num> (*default: 1.4*)
 crystalheight: <num> (*default: 0.6*)
 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.

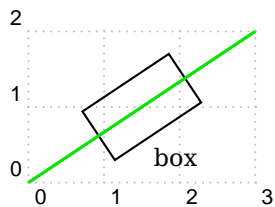


```
\begin{pspicture}[showgrid=true](3,2)
2 \node(0,1.2){A}
3 \node(3,1.2){B}
4 \crystal[fillstyle=solid, fillcolor=yellow!90!black,
5 labelangle=-45, labeloffset=1.2, voltage, lamp, beam
6 ](A)(B){SBN:Ce}
7 \end{pspicture}
```

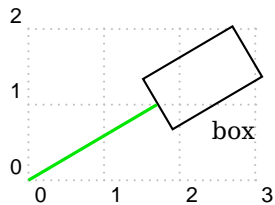
3.6 Box

optboxheight: <num> (*default: 0.8*)
 optboxwidth: <num> (*default: 1.4*)
 endbox: <boolean> (*default: false*)
 angle: <degree> (*default: 0*)
 rotateref: <ref string>³ (*default: c*)
 refractiveindex: <num> (*default: {}*)

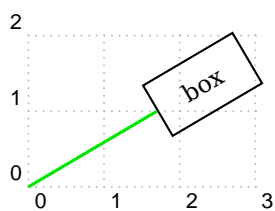
³ A <ref string> is any combination of c (center), t (top), b (bottom), l (left), r (right)



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

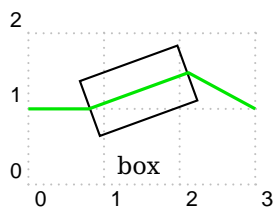


```
\begin{pspicture}[showgrid=true](3,2)
2 \optbox[beam, endbox](0,0)(1.7,1){box}
3 \end{pspicture}
```



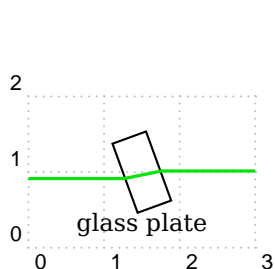
```
\begin{pspicture}[showgrid=true](3,2)
2 \node(0,0){A}
3 \node(1.7,1){B}
4 \optbox[beam, endbox, labelref=relative, labeloffset
=0](A)(B){box}
5 \end{pspicture}
```

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. 2.7). Note, that all connection-related nodes are also rotate, while the label is not affected.



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

Together with the parameter `refractiveindex` this can be exploited to sketch the refraction through a tilted homogeneous medium (e.g. a glass plate). Then, however, the reference nodes for the beam connection must be corrected which is rather easy using the outer nodes of the object as coordinate references and the `|` node operator.



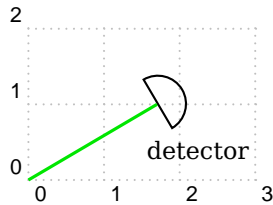
```
\begin{pspicture}[showgrid=true](3,2)
2 \node(0,1){A}
3 \node(3,1){B}
4 \optbox[labeloffset=0.7, optboxwidth=0.5, optboxheight
=1, angle=20, refractiveindex=2.3, compname=Box](A)(
B){glass plate}
5 \drawbeam[conn=-a]{(A|BoxIntern1)}{Box}
6 \drawbeam[conn=B-]{Box}{(B|BoxInternN)}
7 \end{pspicture}
```

3.7 Detector

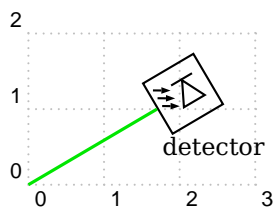
detsize: <num> (default: 0.8)

detttype: round|diode (default: round)

With pst-optexp version 2.0 the name for the detector was changed to `\optdetector` as the package `pst-circ` also provides a `\detector` macro. For compatibility reasons the old `\detector` macro is available when `pst-circ` is not loaded before `pst-optexp`.



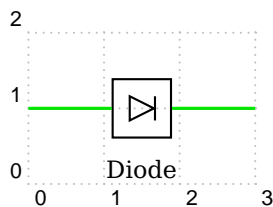
```
\begin{pspicture}[showgrid=true](3,2)
2 \nnode(0,0){A}
3 \nnode(1.7,1){B}
4 \optdetector[beam](A)(B){detector}
5 \end{pspicture}
```



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

3.8 Optical Diode

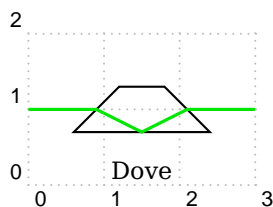
optdiodesize: <num> (default: 0.8)



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

3.9 Dove Prism

doveprismsize: <num> (default: 0.6)



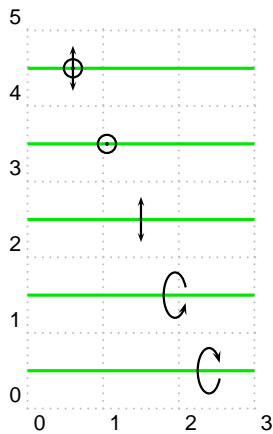
```
\begin{pspicture}[showgrid=true](3,2)
2 \doveprism[beam](0,1)(3,1){Dove}
3 \end{pspicture}
```

3.10 Polarization

poltype: parallel|perp|misc|lccirc|rccirc (default: parallel)

polsize: <num> (default: 0.6)

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



```

\begin{pspicture}[showgrid=true](3,5)
  \nnode(0,0.5){A1}\nnode(3,0.5){B1}\nnode(0,1.5){A2}
  \nnode(3,1.5){B2}\nnode(0,2.5){A3}\nnode(3,2.5){B3}
  \nnode(0,3.5){A4}\nnode(3,3.5){B4}\nnode(0,4.5){A5}
  \nnode(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=rccirc,position=0.65](A2)(B2)
  \polarization[poltype=lccirc,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)

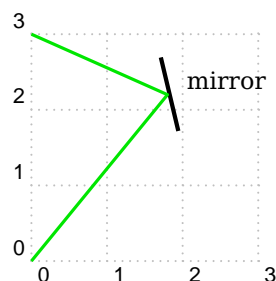
mirrordepth: <num> (default: 0.1)

variable: <num> (default: false)

ExtendedMirror: <psstyle>

PiezoMirror: <psstyle>

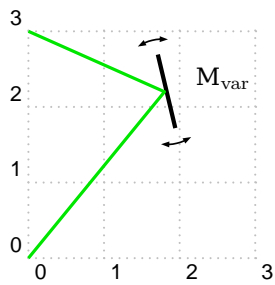
The parameter `mirrorradius` defines the curvature of the mirror. A value of 0 is for a plain mirror, a negative radius is for a concave mirror and a positive radius gives you a convex mirror. The style of the extended mirror is defined as a `psstyle` `ExtendedMirror` and can be changed using `\newpsstyle`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



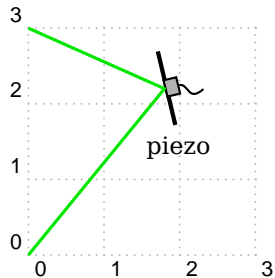
```

\begin{pspicture}[showgrid=true](3,3)
  \nnode(0,0){A}
  \nnode(1.8,2.2){G}
  \nnode(0,3){B}
  \mirror[beam](A)(G)(B){mirror}
\end{pspicture}

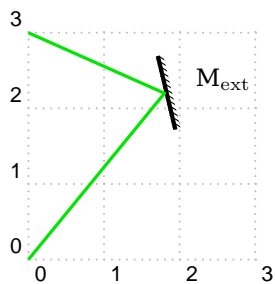
```



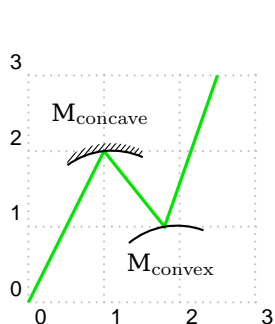
```
\begin{pspicture}[showgrid=true](3,3)
2 \node(0,0){A}
3 \node(1.8,2.2){G}
4 \node(0,3){B}
5 \mirror[beam, variable](A)(G)(B){M$_{\mathrm{var}}$}
6 \end{pspicture}
```



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



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

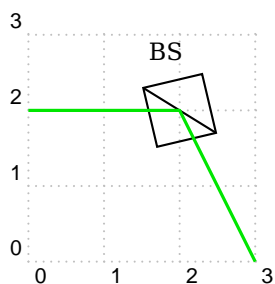


```
\begin{pspicture}[showgrid=true](3,3)
2 \node(0,0){A}\node(1,2){G1}
3 \node(1.8,1){G2}\node(2.5,3){B}
4 \psset{labeloffset=0.5}
5 \psline[style=Beam](A)(G1)(G2)(B)
6 \mirror[mirrortype=extended, mirrorradius=1](A)(G1)(G
7 2){M$_{\mathrm{concave}}$}
8 \mirror[mirrorradius=-1](G1)(G2)(B){M$_{\mathrm{convex}}$}
9 \end{pspicture}
```

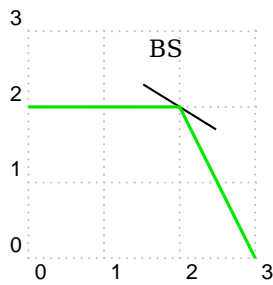
3.12 Beamsplitter

bssize: <num> (default: 0.8)

bsstyle: cube|plate (default: cube)



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



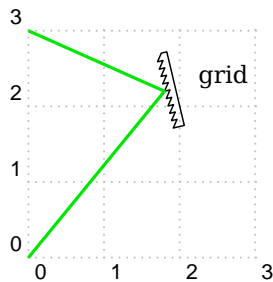
```

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

```

3.13 Optical Grid

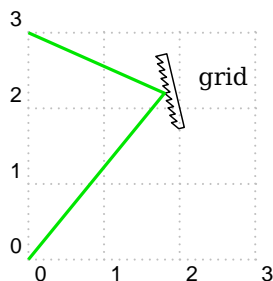
optgridcount: <integer> (default: 10)
 optgridwidth: <num> (default: 1)
 optgridheight: <num> (default: 0.1)
 optgriddepth: <num> (default: 0.05)
 optgridtype: blazed|binary (default: blazed)
 optgridlinewidth: <num> (default: 0.7\pslinewidth)
 reverse: <boolean> (default: false)



```

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

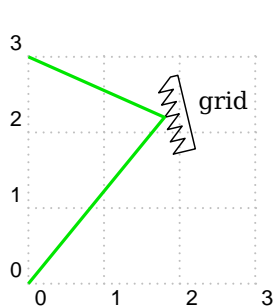
```



```

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

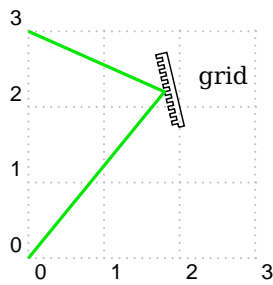
```



```

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

```



```

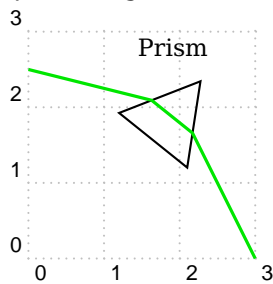
\begin{pspicture}[showgrid=true](3,3)
2 \node(0,3){A}
3 \node(1.8,2.2){G}
4 \node(0,0){B}
5 \optgrid[beam, optgridtype=binary](A)(G)(B){grid}
6 \end{pspicture}

```

3.14 Prism

prismsize: <num> (default: 1)
 prismangle: <num> (default: 60)

The prism has always a symmetric refraction independent of the beams and the prismangle.



```

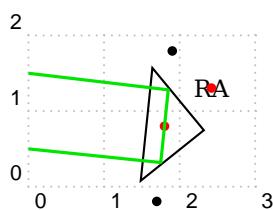
\begin{pspicture}[showgrid=true](3,3)
2 \node(0,2.5){A}
3 \node(2,2){G}
4 \node(3,0){B}
5 \optprism[beam](A)(G)(B){Prism}
6 \end{pspicture}

```

3.15 Right-Angle Prism

raprismsize: <num> (default: 1.5)

The right-angle prism is constructed such that the two incoming beams are parallel and the middle reference node is vertically centered in the prism.



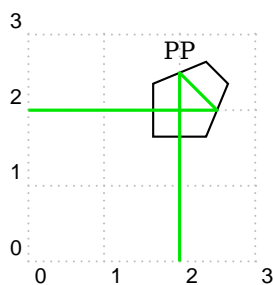
```

\begin{pspicture}[showgrid=true](3,2)
2 \node(0,1.5){A}
3 \node(1.8,0.8){G}
4 \node(0,0.5){B}
5 \rightangleprism[beam, showoptdots](A)(G)(B){RA}
6 \end{pspicture}

```

3.16 Penta Prism

pentaprismsize: <num> (default: 0.7)



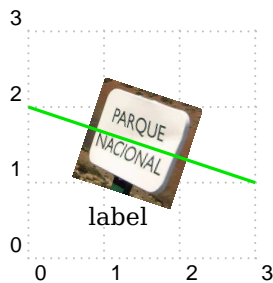
```

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

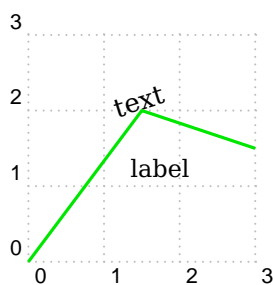
```

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 on how to define your own components see Sec. 5.2.



```
\begin{pspicture}[showgrid=true](3,3)
  \pnode(0,2){A}
  \pnode(3,1){B}
  \optdipole[labeloffset=1, beam](A)(B){%
    \rput(0,0){%
      \includegraphics[scale=0.25]{parque-nacional}
    }
  }{label}
\end{pspicture}
```



```
\begin{pspicture}[showgrid=true](3,3)
  \pnode(0,0){A}
  \pnode(1.5,2){G}
  \pnode(3,1.5){B}
  \opttripole[beam](B)(G)(A){\rput[b](0,0){text}}{label}
\end{pspicture}
```

4 Fiber-Optical Objects

`usefiberstyle`: <boolean> (default: false)

Fiber-optical objects are automatically connected to the reference nodes. The style of all fiber connections can be configured independently (see Sec. 4.12).

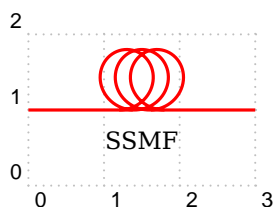
For some components it might be nice to highlight some internals. If `usefiberstyle` is enabled, for examples the passing parts of the optical filter are drawn with the Fiber style. In the documentation this parameter is enabled to show the parts which would be highlighted.

4.1 Fiber

`fiberloops`: <integer> (default: 3)

`fiberloopradius`: <num> (default: 0.4)

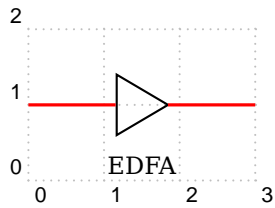
`fiberloopsep`: <num> (default: 0.3)



```
\begin{pspicture}[showgrid=true](3,2)
  \optfiber[labeloffset=0.4](0,1)(3,1){SSMF}
\end{pspicture}
```

4.2 Amplifier

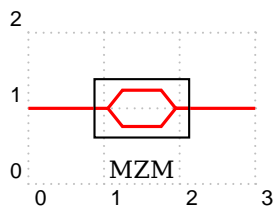
optampsize: <num> (default: 0.8)



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

4.3 Mach-Zehnder Modulator

optmzsize: <num> (default: 0.8)

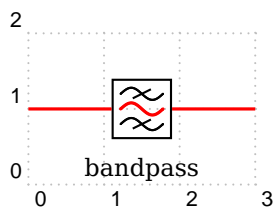


```
\begin{pspicture}[showgrid=true](3,2)
2 \optmzm(0,1)(3,1){MZM}
3 \end{pspicture}
```

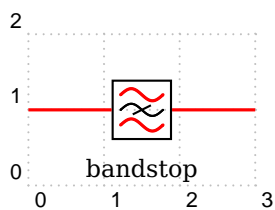
4.4 Filter

filtersize: <num> (default: 0.8)

filtertype: bandpass|bandstop (default: bandpass)



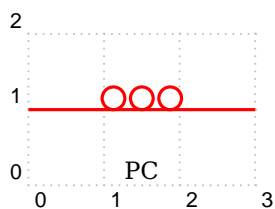
```
\begin{pspicture}[showgrid=true](3,2)
2 \optfilter(0,1)(3,1){bandpass}
3 \end{pspicture}
```



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

4.5 Polarization Controller

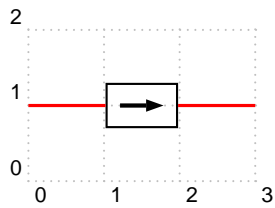
polcontrolsize: <num> (default: 0.15)



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

4.6 Isolator

isolatorsize: <num> (default: 0.6)

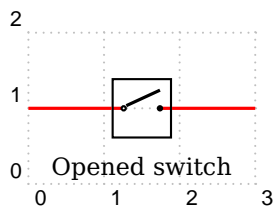


```
\begin{pspicture}[showgrid=true](3,2)
2 \optisolator(0,1)(3,1){}
3 \end{pspicture}
```

4.7 Optical Switch

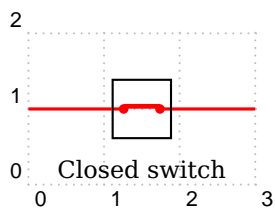
switchsize: <num> (default: 0.8)

switchstyle: opened|closed (default: opened)



Opened switch

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

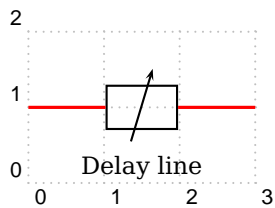


Closed switch

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

4.8 Fiber Delay Line

fdlsize: <num> (default: 0.6)

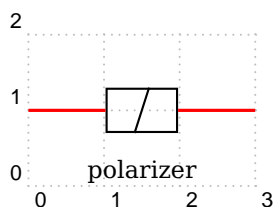


Delay line

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

4.9 Fiber Polarizer

fiberpolsize: <num> (default: 0.6)



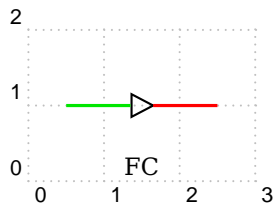
polarizer

```
\begin{pspicture}[showgrid=true](3,2)
2 \optfiberpolarizer(0,1)(3,1){polarizer}
3 \end{pspicture}
```

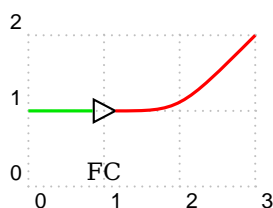
4.10 Fiber Collimator

fibersize: <num> (default: 0.3)

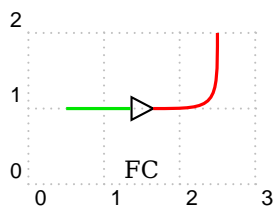
The connection type for the fiber collimator is fixed to `conn=o-f`. The component can be used with two, three or four nodes. With more than two points, the fiber is drawn as a `\psbezier` curve. In the case of three nodes, the middle one is used twice. Positioning parameters can still be used to shift the component between the first two nodes.



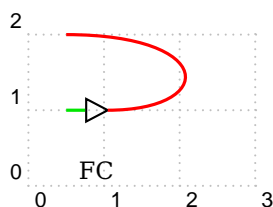
```
\begin{pspicture}[showgrid=true](3,2)
2 \fibercollimator(0.5,1)(2.5,1){FC}
3 \end{pspicture}
```



```
\begin{pspicture}[showgrid=true](3,2)
2 \fibercollimator(0,1)(2,1)(3,2){FC}
3 \end{pspicture}
```



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



```
\begin{pspicture}[showgrid=true](3,2)
2 \fibercollimator[position=0.2](0.5,1)(2.5,1)(2.5,2)
3 (0.5,2){FC}
\end{pspicture}
```

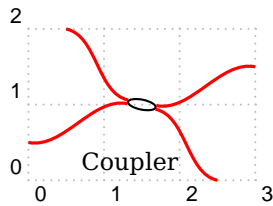
4.11 Coupler

couplersize: <num> (default: 0.2)

couplersep: <num> (default: 0.1)

couplertype: none|elliptic (default: elliptic)

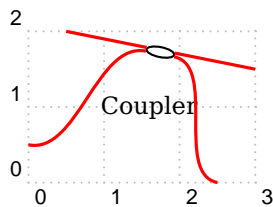
align: top|bottom|center (default: center)

2 × 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}

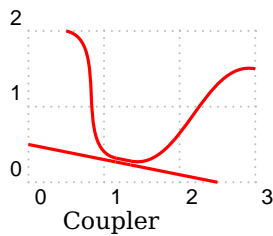
```



```

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

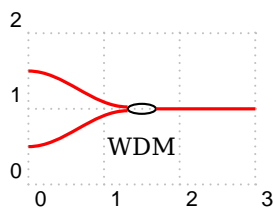
```



```

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

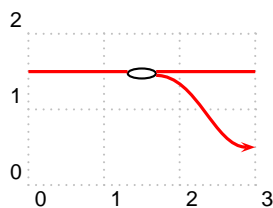
```

WDM Coupler

```

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

```

WDM Splitter

```

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

```

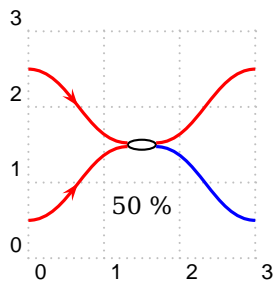
4.12 Fiber Styles

Fiber: <psstyle>
 FiberIn: <psstyle>
 FiberIn1: <psstyle>
 FiberIn2: <psstyle>
 FiberOut: <psstyle>
 FiberOut1: <psstyle>
 FiberOut2: <psstyle>

All these psstyles control the appearance of the fiber parts before and after each object. 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 connections on the left of the object and to the output the ones on the right side.

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 psstyles are used only by the various 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. All fiber connections are drawn as `\pccurve` which means that also the curvature and the input and output angles of each connection can be changed as you will see in a following code example.



```

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 `newFiber...` and `addtoFiber...` parameter keys for each of them.

```

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

```

is equivalent to

```

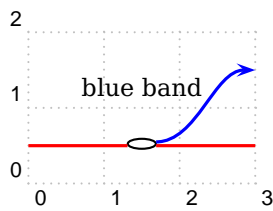
1 \addtopsstyle{FiberIn}{arrows=->, arrowscale=1.3}

```

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

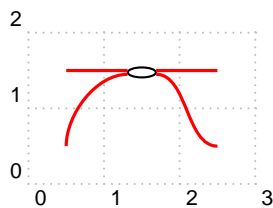
At first glance these keys make no sense. The reason why I introduced them was to be able to define special couplers with `\newpsobject`. This is only possible if all modifications can be expressed as parameter keys. Consider for example a WDM

splitter which only couples out a certain spectral range of the input and you want to mark the output with an arrow:



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

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

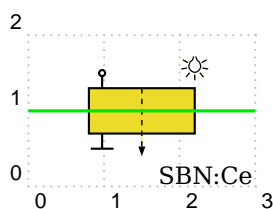


```
\begin{pspicture}[showgrid=true](3,2)
2 \newpsobject{mycoupler}{optcoupler}{addtoFiberIn2={
angleA=90}, align=top}
3 \mycoupler(0.5,1.5)(0.5,0.5)(2.5,1.5)(2.5,0.5){}
4 \end{pspicture}
```

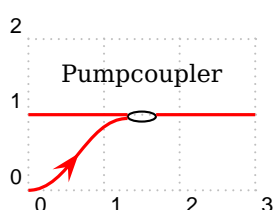
5 Defining New Objects

5.1 Customized Versions of Existing Macros

The easiest way to define your own components is to use the `\newpsobject` 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 when calling the macro. The following examples define a new object `\sbn` for the special crystal used in Sec. 3.5.

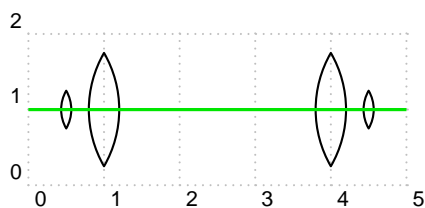


```
\newpsobject{sbn}{crystal}{voltage, lamp, labelangle=45,
labeloffset=1.2, fillstyle=solid, fillcolor=yellow
!90!black}
2 \begin{pspicture}[showgrid=true](3,2)
3 \sbn(0,1)(3,1){SBN:Ce}
4 \psline[style=Beam](0,1)(3,1)
5 \end{pspicture}
```



```
\newpsobject{pumpcoupler}{wdmcoupler}{align=top,
labelangle=180, labeloffset=0.5, addtoFiberIn2={
ArrowInside=->, arrowscale=2}}
2 \begin{pspicture}[showgrid=true](3,2)
3 \pumpcoupler(0,1)(0,0)(3,1){Pumpcoupler}
4 \end{pspicture}
```

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



```

1 \newpsobject{MOLensIn}{lens}{lens=0.5 0.5
2 0.5}
3 \newpsobject{MOLensOut}{lens}{lens=1.5 1.5
4 1.5}
5 \begin{pspicture}[showgrid=true](5,2)
6 \pnode(0,1){A}\pnode(5,1){B}
7 \MOLensIn[abspos=0.5](A)(B){}
8 \MOLensOut[abspos=1](A)(B){}
9 \MOLensOut[abspos=4](A)(B){}
10 \MOLensIn[abspos=4.5](A)(B){}
11 \psline[style=Beam](A)(B)
12 \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 without 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 its reference 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

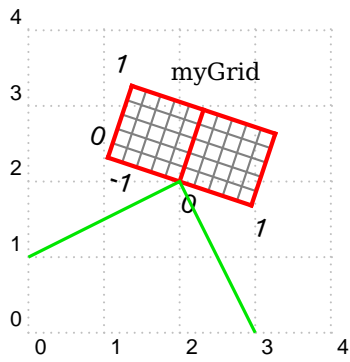
```

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 within the `\mycomponent@iii` macro.



```

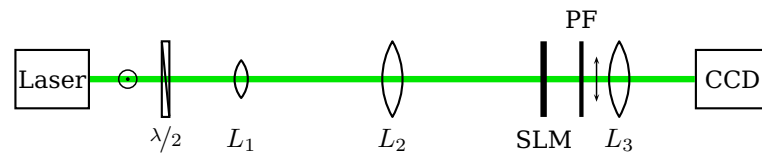
1 \newOptexpTripole{mygrid}{subgriddiv=5, griddots=0,
   subgridwidth=\pslinewidth, gridwidth=2\
   pslinewidth}
2 \makeatletter
3 \def\mygrid@iii{% put here all PSTricks drawing
   code
4 \psgrid(-1,0)(1,1)
5 }%
6 \makeatother
7 \begin{pspicture}[showgrid=true](4,4)
8 \pnode(0,1){A}\pnode(2,2){G}\pnode(3,0){B}
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.

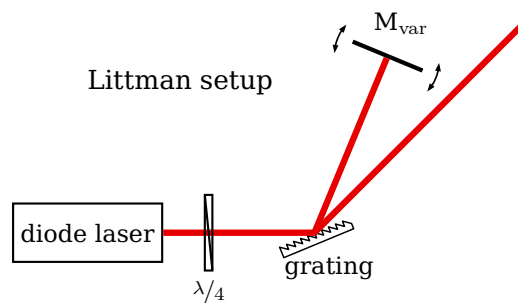
6 Examples



```

1 \begin{pspicture}(10,2)
2 \psset{optboxwidth=1}\addtopsstyle{Beam}{linewidth=2\pslinewidth}
3 \node(1,1){Start}\node(9,1){CCD}\optbox[endbox, labeloffset=0](CCD)(Start)
4 {Laser}
5 \optbox[endbox, labeloffset=0, beam](Start)(CCD){CCD}
6 \polarization[poltype=perp, abspos=0.5](Start)(CCD)
7 \optretplate[abspos=1](Start)(CCD){$\nicefrac{\lambda}{2}$}
8 \lens[lens=0.4 0.4 0.5, abspos=2](Start)(CCD){$L_1$}\lens[abspos=4](Start)(
9 CCD){$L_2$}
10 \optplate[abspos=6, platelinewidth=3\pslinewidth](Start)(CCD){SLM}
11 \optplate[abspos=6.5, labelangle=180](Start)(CCD){PF}
12 \polarization[abspos=6.7](Start)(CCD)\lens[abspos=7](Start)(CCD){$L_3$}
13 \end{pspicture}

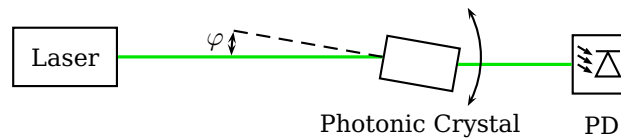
```



```

1 \begin{pspicture}(-4,-1)(3,3)
2 \addtopsstyle{Beam}{linewidth=2\pslinewidth, linecolor=red!90!black}
3 \psset{labeloffset=0.5}
4 \node(-2,0){LaserOut}\node(0,0){Grat}
5 \node(4;45){Out}\node(2.5;67.5){Mvar}
6 \optbox[optboxwidth=2, labeloffset=0, endbox](Grat)(LaserOut){diode laser}
7 \mirror[variable, conn=o-](Grid)(Mvar)(Grid){M$\mathrm{var}$}
8 \optgrid[beam](LaserOut)(Grat)(Out){grating}
9 \optretplate[position=0.3, labeloffset=0.8]%
10 (LaserOut)(Grat){$\nicefrac{\lambda}{4}$}
11 \rput[l](-3,2){Littman setup}
12 \end{pspicture}

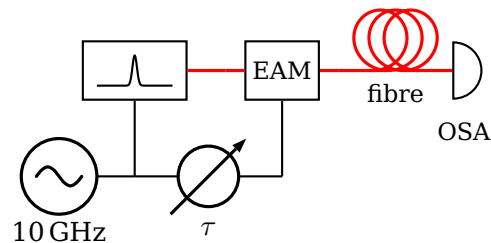
```



```

1 \begin{pspicture}(8.5,1.6)
2   \addtopsstyle{Beam}{linecolor=green!90!black}
3   \nnode(1.6,1){Laser}\nnode(7.6,1){Diode}
4   \optbox[endbox,labeloffset=0](Diode)(Laser){Laser}%
5   \optbox[abspos=4, optboxwidth=1, optboxheight=0.6, labeloffset=1,
6     compname=PC, conn=o-, angle=-10, rotateref=l, refractiveindex=2.3](
7     Laser)(Diode){Photonic Crystal}
8   \optdetector[dettype=diode, conn=o-](PCInternN)(Diode|PCInternN){PD}
9   \defShiftedNode(PCIntern1)(2;170){Angle1}
10  \psline[linestyle=dashed](PCIntern1)(Angle1)
11  \psarc{<->}(PCIntern1){1.3}{330}{30}
12  \psarc[arcsep=1pt]{<->}(PCIntern1){2}{170}{180}
13  \uput{2.1}[175](PCIntern1){\small $\varphi$}
14 \end{pspicture}

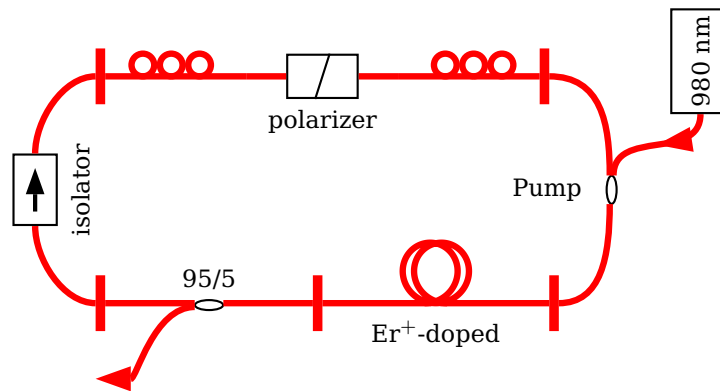
```



```

1 \begin{pspicture}(6.4,3.2)
2   \addtopsstyle{Fiber}{linecolor=red}
3   \nnode(2.3,2.3){Lin}\nnode([Xnodesep=0.5]Lin){Lout}
4   \nnode([Xnodesep=1.5]Lout){EAMout}
5   \nnode([Xnodesep=1.5]EAMout){Det}
6   \optbox[fiber, labeloffset=-0.2, endbox, compname=L, extnode=b](Lout)(Lin){%
7     \psGauss[yunit=0.03, sigma=0.03]{-0.5}{0.5}}
8   \optbox[fiber, labeloffset=0, optboxwidth=1, compname=EAM, extnode=b](Lout)(
9     EAMout){EAM}
10  \optfiber[labeloffset=0.3](EAMout)(Det){fibre}
11  \optdetector(EAMout)(Det){OSA}
12  \nnode([Xnodesep=-1, offset=-1]LExtNode){Osc}
13  \nnode(LExtNode|Osc){PSin}\nnode(EAMExtNode|Osc){PSout}
14  \oscillator[output=right](Osc){10\,GHz}{}
15  \phaseshifter[labeloffset=-0.7](PSin)(PSout){$\tau$}
16  \wire(LExtNode)(PSin)\wire(EAMExtNode)(PSout)
17 \end{pspicture}

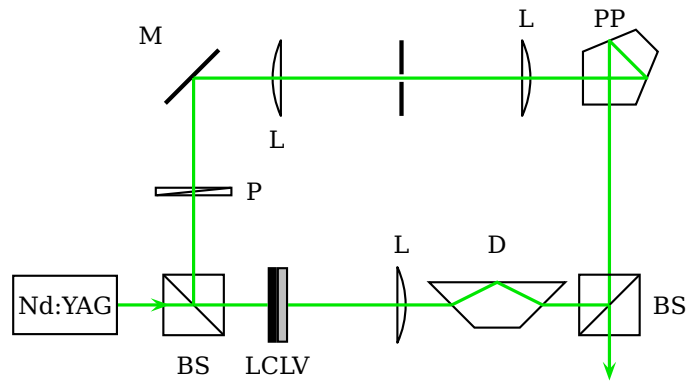
```



```

1 \begin{pspicture}(0.9,0.9)(10.4,5.9)
2   \psset{arrowscale=1.5, arrowinset=0}
3   \addtopsstyle{Fiber}{linewidth=2\pslinewidth}
4   \nnode(2,5){PC1in}\nnode(4,5){PC1out}\nnode(6,5){PC2in}
5   \nnode(8,5){PC2out}\nnode(2,2){CplSig}\nnode(5,2){CplIn}
6   \nnode(2,1){CplOut}\nnode(10,4.5){Pump}\nnode(8,2){PumpSig}
7   \optisolator[compshift=0.8, addtoFiberIn={angleA=180}, addtoFiberOut={
8     angleB=180}, labelref=relative, labeloffset=0.6](CplSig)(PC1in){isolator}
9   \polcontrol[addtoFiberIn={arrows=|-}](PC1in)(PC1out){}
10  \optfiberpolarizer[labeloffset=0.6](PC1out)(PC2in){polarizer}
11  \polcontrol[addtoFiberOut={arrows=-|}](PC2in)(PC2out){}
12  \wdmsplitter[labeloffset=0.3, align=bottom, addtoFiberIn={arrows=|-},
13    addtoFiberOut1={arrows=->}, addtoFiberOut2={arrows=-|}](CplIn)(CplOut)(
14    CplSig){95/5}
15  \wdmcoupler[addtoFiberIn1={ArrowInside=->}, addtoFiberIn2={angleA=0},
16    addtoFiberOut={angleB=0,arrows=-|}, ncurv=0.9, align=bottom, compshift
17    =0.8](Pump)(PC2out)(PumpSig){Pump}
18  \optbox[endbox,labeloffset=0,labelref=relative]([offset=-0.1]Pump)(Pump)
19    {980~nm}
20  \optfiber[fiberloops=2, labeloffset=0.4](CplIn)(PumpSig){Er$^+$-doped}
21 \end{pspicture}

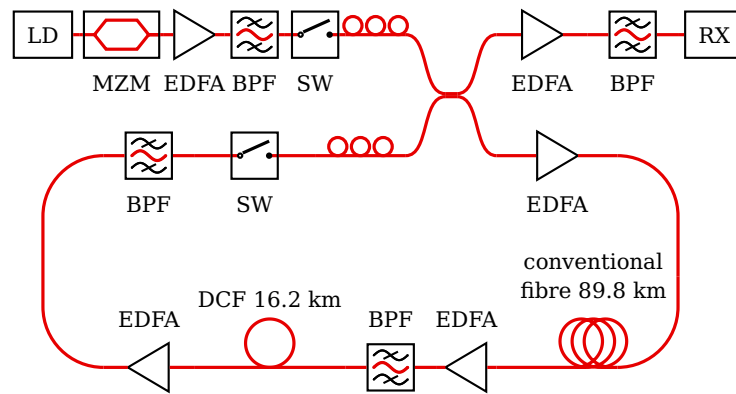
```



```

1 \makeatletter
2 \def\LCLV@iii{%
3   \psframe[fillstyle=solid,fillcolor=black,dimen=outer](-0.12,-0.5)(0,0.5)
4   \psframe[fillstyle=solid,fillcolor=gray!50,dimen=outer](0,-0.5)(0.15,0.5)
5   \pnode(-0.12,0){\optexp@nodeA}\pnode(0.15,0){\optexp@nodeB}}
6 \makeatother
7 \begin{pspicture}(9,5)
8   \newOptexpDipole{LCLV}{}\psset{lens=1.2 0 1}
9   \pnode(2.4,1){BS1}\pnode([offset=3]BS1){M1}\pnode([Xnodesep=5.5]M1){PP}\
   \pnode(PP|BS1){BS2}
10  \LCLV[position=0.2, comname=LCLV](BS1)(BS2){LCLV}\beamsplitter[comname=BS
   ](BS2)(BS1)(M1){BS}
11  \optretplate(BS1)(M1){P}\mirror[conn=i-](BS1)(M1)(PP){M}\lens[position=0.2](
   M1)(PP){L}
12  \pinhole(M1)(PP){}\lens[position=0.2](PP)(M1){L}\pentaprism[beam](M1)(PP)(BS
   2){PP}
13  \beamsplitter(PP)(BS2)(BS1){BS}\lens(BS2)(BS1){L}
14  \doveprism[comname=Dove,conn=i-,position=0.27](BS2)(BS1){D}
15  \drawbeam[conn=b-b]{Dove}{LCLV}\drawbeam[conn=b-a]{BS}{LCLV}
16  \psline[arrowscale=1.3, style=Beam]{->}(BS2)([offset=-1]BS2)
17  \addtopsstyle{Beam}{arrowscale=1.3, ArrowInside=-<}
18  \optbox[labeloffset=0, endbox, conn=o-](BS1)([Xnodesep=-1]BS1){Nd:YAG}
19 \end{pspicture}

```

```

1 \begin{pspicture}(0.5,4)(13.2,10.5)
2 \addtopsstyle{Fiber}{linecolor=red!90!black}\psset{usefiberstyle,
3   optboxwidth=1}
4 \pnode(2,10){LD}\pnode([Xnodesep=5.5]LD){CPLin1}
5 \pnode([offset=-2]CPLin1){CPLin2}\pnode([Xnodesep=1.5]CPLin1){CPLout1}
6 \pnode([Xnodesep=1.5]CPLin2){CPLout2}
7 \optbox[endbox, labeloffset=0, fiber]([Xnodesep=0.1]LD)(LD){LD}
8 \optmzm([Xnodesep=0.1]LD)([Xnodesep=1.5]LD){MZM}
9 \optamp([Xnodesep=1.5]LD)([Xnodesep=2.5]LD){EDFA}
10 \optfilter([Xnodesep=2.5]LD)([Xnodesep=3.5]LD){BPF}
11 \optswitch([Xnodesep=3.5]LD)([Xnodesep=4.5]LD){SW}
12 \polcontrol([Xnodesep=4.5]LD)(CPLin1){}
13 \optcoupler[couplertype=none](CPLin1)(CPLin2)(CPLout1)(CPLout2){}
14 \optamp(CPLout1)([Xnodesep=1.5]CPLout1){EDFA}
15 \optfilter([Xnodesep=1.5]CPLout1)([Xnodesep=3]CPLout1){BPF}
16 \optbox[endbox, labeloffset=0, conn=f-f]([Xnodesep=3]CPLout1)([Xnodesep=
17   =3.1]CPLout1){RX}
18 \pnode([Xnodesep=2]CPLout2){LoopRU}\pnode([offset=-3.5]LoopRU){LoopRL}
19 \pnode([Xnodesep=-5]CPLin2){LoopLU}\pnode([offset=-3.5]LoopLU){LoopLL}
20 \optamp(CPLout2)(LoopRU){EDFA}
21 \psline[linearc=1,style=Fiber](LoopRU)([Xnodesep=1]LoopRU)([Xnodesep=1,
22   offset=-2]LoopRU)
23 \psline[linearc=1,style=Fiber]([Xnodesep=1,offset=1.5]LoopRL)%
24   ([Xnodesep=1]LoopRL)(LoopRL)
25 \optfiber[labelalign=b, labeloffset=-1, position=0.8]([Xnodesep=-2]LoopRL)(
26   LoopRL){\begin{tabular}{c}conventional\\fibre 89.8-km\end{tabular}}
27 \optamp([Xnodesep=-2]LoopRL)([Xnodesep=-3]LoopRL){EDFA}
28 \optfilter([Xnodesep=-3]LoopRL)([Xnodesep=-4.5]LoopRL){BPF}
29 \optfiber[fiberloops=1, labeloffset=-1, labelalign=b]([Xnodesep=-7]LoopRL)
30   ([Xnodesep=-4.5]LoopRL){DCF 16.2-km}
31 \optamp([Xnodesep=1.5]LoopLL)(LoopLL){EDFA}
32 \psline[style=Fiber, linearc=1](LoopLL)([Xnodesep=-1]LoopLL)%
33   ([Xnodesep=-1,offset=3.5]LoopLL)(LoopLU)
34 \optfilter(LoopLU)([Xnodesep=1.5]LoopLU){BPF}
35 \optswitch([Xnodesep=1.5]LoopLU)([Xnodesep=3.5]LoopLU){SW}
36 \polcontrol([Xnodesep=3.5]LoopLU)(CPLin2){}
37 \end{pspicture}

```

7 Complete List of Parameters

parameter	allowed values	default
abspos	<num>	{}
addtoBeam	<psstyle>	
addtoFiber*	<psstyle>	
addtoOptComp	<psstyle>	
align	top bottom center	center
angle	<degree>	0
beam	alias for conn=o-i	
bssize	<num>	0.8
bsstyle	cube plate	cube
caxisinv	<boolean>	false
caxislength	<num>	0.6
compname	<string>	{}
compshift	<num>	0
conn	<conn definition>	-
connjoin	<int>	1
couplersep	<num>	0.1
couplersize	<num>	0.2
couplertype	none elliptic	elliptic
crystalheight	<num>	0.6
crystalwidth	<num>	1.4
detsize	<num>	0.8
detype	round diode	round
doveprismsize	<num>	0.6
endbox	<boolean>	false
extnode	<ref string>	{}
fdlsize	<num>	0.6
fiber	alias for conn=f-f	
fibercolsize	<num>	0.3
fiberloopradius	<num>	0.4
fiberloops	<integer>	3
fiberloopsep	<num>	0.3
fiberpolsize	<num>	0.6
filtersize	<num>	0.8
filtertype	bandpass bandstop	bandpass
innerheight	<num>	0.1
isolatorsize	<num>	0.6
label	<offset> <angle> <ref> <labelref>	
labelalign	<ref string>	c
labelangle	<num>	0
labeloffset	<num>	0.8
labelref	relative relgrav global	relgrav
labelstyle	<macro>	\small

parameter	allowed values	default
lamp	<boolean>	false
lampscale	<num>	0.3
lens	<num> [<num> [<num> [<num>]]]	{}
lensheight	<num>	1
lensradius	<num> [<num>]	{}
lensradiusleft	<num>	1
lensradiusright	<num>	1
lenswidth	<num>	0.2
mirrordepth	<num>	0.1
mirrorlinewidth	<num>	2\pslinewidth
mirrorradius	<num>	0
mirrortype	normal piezo extended	normal
mirrorwidth	<num>	1
newBeam	<psstyle>	
newFiber*	<psstyle>	
newOptComp	<psstyle>	
optampsize	<num>	0.8
optboxheight	<num>	0.8
optboxwidth	<num>	1.4
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
optmzsize	<num>	0.8
outerheight	<num>	1
pentaprismsize	<num>	0.7
phlinewidth	<num>	2\pslinewidth
plateheight	<num>	1
platelinewidth	<num>	2\pslinewidth
platewidth	<num>	0.1
polcontrolsize	<num>	0.15
polllinewidth	<num>	0.7\pslinewidth
polsize	<num>	0.6
poltype	parallel perp misc lcirc rcirc	parallel
position	<num>	{}
prismangle	<num>	60
prismsize	<num>	1
raprismsize	<num>	1.5
refractiveindex	<num>	{}
reverse	<boolean>	false
rotateref	<ref string>	c

parameter	allowed values	default
showoptdots	<boolean>	false
switchsize	<num>	0.8
switchstyle	opened closed	opened
thicklens	<boolean>	false
usefiberstyle	<boolean>	false
variable	<num>	false
voltage	<boolean>	false

8 Complete List of Styles

style	description
Beam	All automatic free-ray connections are drawn using this style
ExtendedMirror	Affects the additional part for <code>mirrortype=extended</code>
Fiber	Parent style for all fiber connections. For a detailed discussion see Sec. 4.12
FiberIn	Left fiber style if only one connection to be drawn (inherits from Fiber)
FiberIn1	Upper left connection (used for couplers only, inherits from FiberIn)
FiberIn2	Lower left connection (used for couplers only, inherits from FiberIn)
FiberOut	Right fiber style if only one connection to be drawn (inherits from Fiber)
FiberOut1	Upper right connection (used for couplers only, inherits from FiberOut)
FiberOut2	Lower right connection (used for couplers only, inherits from FiberOut)
OptComp	Affects only the appearance of optical components without changing connections that may be drawn inside the component
OptionalStyle	Used for objects with parameter <code>optional</code> set to true
PiezoMirror	Affects the additional part for <code>mirrortype=piezo</code>

9 Requirements

`pst-optexp` version 2.1 requires at least version 2.87 of `pstricks-add` and \LaTeX . It does not work with plain \TeX .

10 Todo

- Automatic sizing of optboxes (like a `\psframebox`)
- Even more sophisticated beam drawing (draw beam before object contours)
- 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.

11 Acknowledgements

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