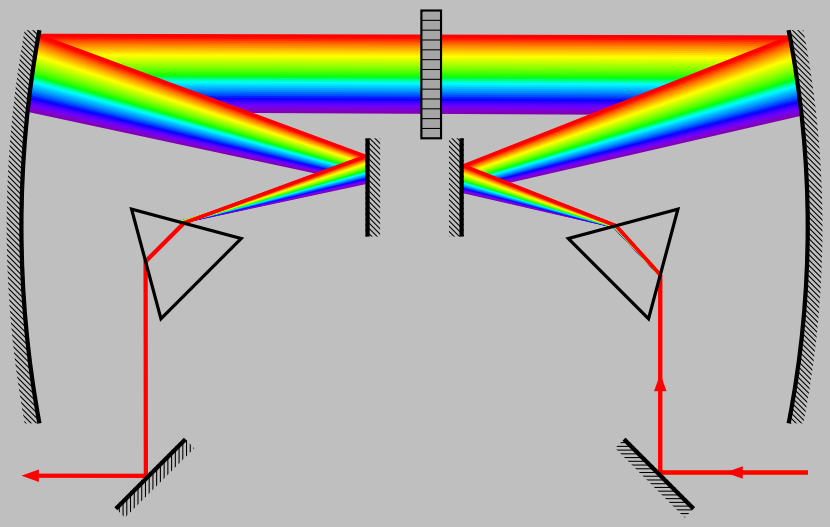
PSTricks

pst-optexp
Optische Versuchsaufbauten

v3.0

2012/07/09



Paketautor:
Christoph Bersch

Inhaltsverzeichnis

1. Einführung

1.1. Über das Paket

pst-optexp ist ein PSTricks-Paket zum Skizzieren optischer Versuchsaufbauten. Dafür werden viele unterschiedliche Freistrahl- und Faserkomponenten bereitgestellt, deren Ausrichtung, Positionierung und Beschriftung einfach und flexibel eingestellt werden kann. Die Komponenten können dann mit Fasern oder Lichtstrahlen verbunden werden, wobei auch realistische Strahlengänge mit Raytracing möglich sind.

1.2. Anforderungen

pst-optexp Version 3.0 benötigt LaTeX und aktuelle Versionen der Pakete pst-node, pstricks-add, multido, pst-eucl und environ.

Alle PSTricks-Pakete machen regen Gebrauch von der Postscript-Sprache, so dass der typische Arbeitsfluss latex, dvips und ggf. ps2pdf umfasst. Es gibt viele alternative Methoden um die Dokumente zu kompilieren.[1]

1.3. Verbreitung und Installation

Dieses Paket ist auf CTAN[2] erhältlich und in TeXLive and MiKTeX enthalten.

Das pst-optexp-Paket umfasst die zwei Hauptdateien pst-optexp.ins und pst-optexp.dtx. Durch Aufrufen von tex pst-optexp.ins werden die beiden folgenden Dateien erzeugt:

- pst-optexp.pro: die Postscript Prologdatei

[1] http://tug.org/PSTricks/main.cgi?file=pdf/pdfoutput
[2] http://mirror.ctan.org/help/Catalogue/entries/pst-optexp.html

- `pst-optexp.sty`: die LaTeX Stildatei

Speichern Sie diese Dateien in einem Verzeichnis der Teil Ihres lokalen TeX-Baums ist.

Vergessen Sie nicht `texhash` aufzurufen um den Baum zu aktualisieren. MiKTeX-Benutzer müssen die Dateinamen-Datenbank (FNDB) aktualisieren.

Detailliertere Information finden Sie in der Dokumentation Ihrer LaTeX-Distribution über die Installation in den lokalen TeX-Baum.

1.4. Lizenz

Es wird die Erlaubnis gewährt, dieses Dokument zu kopieren, zu verteilen und/oder zu modifizieren, unter den Bestimmungen der LaTeX Project Public License, Version 1.3c.[3]. Dieses Paket wird vom Autor betreut (author-maintained).

1.5. Abwärtskompatibilität

In Version 3.0 wurde dem Paket erheblich fortgeschrittenere Funktionalität hinzugefügt, die es zu schwierig machte zur Vorgängerversion 2.1 vollständig abwärtskompatibel zu bleiben. Insbesondere war die Funktionsweise des `\drawbeam` Makros grundlegend falsch konzipiert, so dass ein Wechsel zur neuen Funktionalität so schnell wie möglich erfolgen sollte.

Eine genauere Zusammenstellung der nicht-kompatiblen Änderungen und Hinweise zur Migration von älteren Dokumenten finden Sie in Kap. 11.3.

1.6. Notation

Die meisten Parametertypen, die im Laufe des Dokuments verwendet werden, sind selbsterklärend, andere bedürfen einer genaueren Erläuterung (z.B. der Unterschied zwischen ⟨*num*⟩ und ⟨*psnum*⟩). Tab. 1.1 erklärt die geläufigsten Typen.

[3] `http://www.latex-project.org/lppl.txt`

Name	Beschreibung
num	Gleitkommazahl
psnum	Postscript-Kode der zu einer Zahl auswertet
int	Ganzzahl
dimen	Länge
psstyle	Benutzerdefinierte Parameterkonfiguration definiert mit `\newpsstyle`
refpoint	Analog zum Referenzpunkt von `\rput`, kann jede Kombination von `c` (mittig), `t` (oben), `b` (unten), `l` (links) und `r` (rechts) sein.

Tabelle 1.1.: Parametertypen die in den Parameterdefinitionen verwendet werden.

1.7. Danksagung

Ich danke allen Aktiven auf der PSTricks-Mailingliste für ihre Hilfe, insbesondere Herbert Voß. Mein Dank gilt ebenfalls unterschiedlichen Paketautoren, von denen ich Code kopiert und viel gelernt habe: Florent Chervet, Rolf Niepraschk und Heiko Oberdiek. Christine Römer hat mich mit ihrem Artikel in der DTK[4] überzeugt, ebenfalls eine deutsche Dokumentation bereitzustellen. Der Dokumentationsstil ist eine Mischung aus der `pst-doc` Klasse (Herbert Voß) und dem `ltxdockit` Paket für die `biblatex` Dokumentation (Philipp Lehmann).

[4] Pakete in Deutsch dokumentieren. in Die TeXnische Komödie. Heft 2/2011, S. 28-35.

2. Grundlegende Ideen

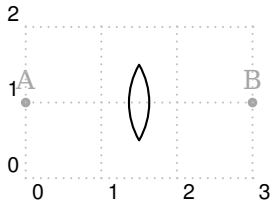
Dieses Kapitel zeigt die grundlegenden Ideen und Konzepte, die in diesem Paket stecken. Anhand von elementaren Beispielen werden die Grundfunktionen wie Ausrichtung, Positionierung (2.1) und Beschriftung (2.2) von Komponenten beschrieben. Anschließen wird deren Verbindung mit Strahlen (2.3) oder Fasern (2.4) vorgestellt.

In Kap. 2.5 wird Schritt für Schritt die angedachte Vorgehensweise für das Erstellen von umfassenden Experimentskizzen erläutert. Eine vollständige Referenz aller Makros und deren Parameter finden Sie in Kap. 3–9.

Ein Versuchsaufbau besteht aus Komponenten, die anhand ihrer Referenzknoten positioniert, verschoben und gedreht werden können und optional eine Beschriftung erhalten. Anschließen werden die Komponenten mit Lichtstrahlen oder Fasern verbunden.

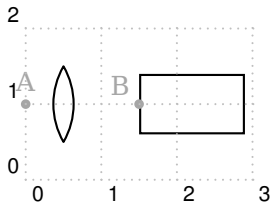
2.1. Die Komponenten

Eine Komponente wird anhand der Knoten, die bei ihrere Definition angegeben werden – den Referenzknoten – platziert. Im folgenden Beispiel wird eine Linse mittig auf die Verbindungslinie zwischen den Knoten $\langle A \rangle$ und $\langle B \rangle$ gesetzt:




```
\begin{pspicture}[showgrid=true](0,-0.3)(3,2.3)
  \pnode(0,1){A}\pnode(3,1){B}
  \lens(A)(B)
\end{pspicture}
```

Diese Positionierung kann anhand von unterschiedlichen Parametern, wie z.B. `abspos` beinflusst werden (siehe dazu Kap. 3.2 und Kap. 3.3). Die Linse wird nun `0.5` Einheiten entfernt vom Knoten $\langle A \rangle$ platziert, die Box sitzt am Ende der Referenzlinie $\overline{AB}$.

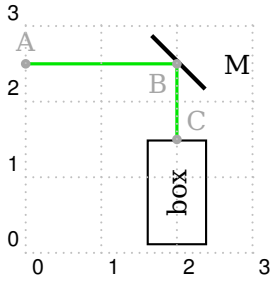



```
\begin{pspicture}[showgrid=true](0,-0.3)(3,2.3)
  \pnode(0,1){A}\pnode(1.5,1){B}
  \lens[abspos=0.5](A)(B)
  \optbox[position=end](A)(B)
\end{pspicture}
```

Die Komponenten lassen sich allgemein in zwei Kategorien unterteilen: Freistrahl- und Faserkomponenten. Diese unterscheiden sich in erster Linie bezüglich ihrer Verbindungsmöglichkeiten: Freistrahlkomponenten können sowohl mit Lichtstrahlen als auch mit Fasern verbunden werden, wohingegen Faserkomponenten nur Faserverbindungen erlauben.

2.2. Beschriftungen

Jede Komponente kann optional eine Beschriftung erhalten, die relativ zur Komponente platziert wird.



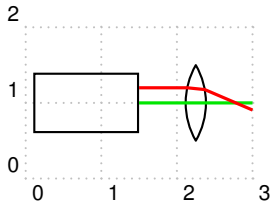

```
\begin{pspicture}[showgrid=true](0,-0.3)(3,3.3)
  \pnode(0,2.5){A}\pnode(2,2.5){B}\pnode(2,1.5){C}
  \mirror[labelangle=-45](A)(B)(C){M}
  \optbox[position=start, labeloffset=0,
      labelref=relative](C)(B){box}
  \drawbeam{(A)}{1}{2}
\end{pspicture}
```

2.3. Freistrahl-Verbindungen

Freistrahlkomponenten können mit Lichtstrahlen verbunden werden, deren Weg mit Raytracing berechnet werden kann. Da es sich um ein Paket zum *Skizzieren* experimenteller Aufbauten handelt, werde hier trotz der realistischen Berechnungen viele Möglichkeiten offen gehalten den Strahlengang zu manipulieren.

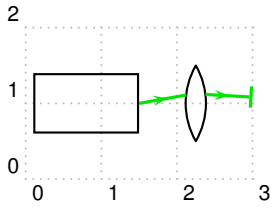
Im einfachsten Fall wird eine einzelne Linie gezeichnet, deren optischer Weg über den Brechungsindex `n` und das Snelliussche Brechungsgesetz bestimmt wird.

Das Beispiel zeigt zwei Strahlen, die mit unterschiedlichen Anfangsbedingungen auf die Linse treffen, der rote Strahl startet mit einem Abstand von `0.2` von der optischen Achse (`beampos`) und wird entsprechend von der Linse abgelenkt.



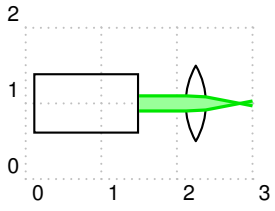

```
\begin{pspicture}[showgrid](0,-0.3)(3,2.3)
  \pnode(1.5,1){A}\pnode(3,1){B}
  \optbox[position=start](A)(B)
  \lens[n=2](A)(B)
  \drawbeam{1}{2}{(B)}
  \drawbeam[beampos=0.2, linecolor=red]{1}{2}{(B)}
\end{pspicture}
```

Die Strahlen können mit beliebigen Strichlierungen und Pfeilen dekoriert werden.



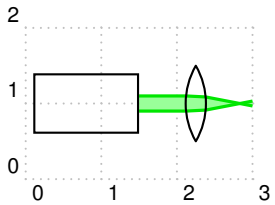

```
\begin{pspicture}[showgrid](0,-0.3)(3,2.3)
  \pnode(1.5,1){A}\pnode(3,1){B}
  \optbox[position=start](A)(B)
  \lens[n=2](A)(B)
  \drawbeam[beamangle=10, ArrowInside=->, arrows=-|,
      beaminside=false]{1}{2}{(B)}
\end{pspicture}
```

Ebenfalls können ausgedehnte Strahlen gezeichnet werden deren Randstrahlen mit `linestyle` etc. und deren Füllung über `fillstyle` geändert werden können.




```
\begin{pspicture}[showgrid](0,-0.3)(3,2.3)
  \pnode(1.5,1){A}\pnode(3,1){B}
  \optbox[position=start](A)(B)
  \lens[n=2](A)(B)
  \drawwidebeam[beamwidth=0.2, fillstyle=solid,
      fillcolor=green!40]{1}{2}{(B)}
\end{pspicture}
```

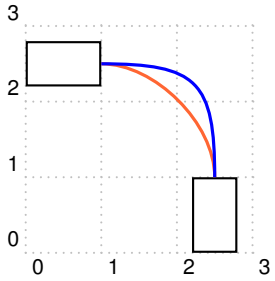
Die Strahlen können auch hinter die Komponenten gelegt werden. Dafür müssen alle beteiligten Komponenten und Strahlen in eine `optexp`-Umgebung gepackt werden (vergleiche das folgende Beispiel mit dem vorangegangenen).




```
\begin{pspicture}[showgrid](0,-0.3)(3,2.3)
  \pnode(1.5,1){A}\pnode(3,1){B}
  \begin{optexp}
    \optbox[position=start](A)(B)
    \lens[n=2](A)(B)
    \drawwidebeam[beamwidth=0.2, fillstyle=solid,
        fillcolor=green!40]{1}{2}{(B)}
  \end{optexp}
\end{pspicture}
```

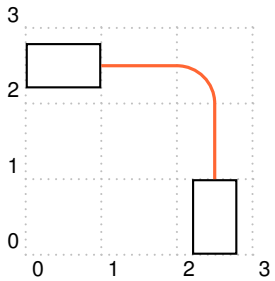
2.4. Faserverbindungen

Eine andere Möglichkeit, Komponenten zu verbinden, sind Fasern (\drawfiber). Hiermit bestehen weitreichende Möglichkeiten die Fasern automatisch an die Komponentenausrichtung anzupassen.




```
\begin{pspicture}[showgrid](0,-0.3)(3,3.3)
  \pnode(1,2.5){A}\pnode(2.5,1){B}
  \psset{optboxsize=1 0.6}
  \optbox[position=start](A)(B|A)
  \optbox[position=end](B|A)(B)
  \drawfiber{1}{2}
  \drawfiber[linecolor=blue, ncurv=1.2]{1}{2}
\end{pspicture}
```

Üblicherweise werden Fasern mit einer \nccurve-Verbindung gezeichnet, jede andere Knotenverbindung ist aber ebenfalls möglich (fiberstyle).




```
\begin{pspicture}[showgrid](0,-0.3)(3,3.3)
  \pnode(1,2.5){A}\pnode(2.5,1){B}
  \psset{optboxsize=1 0.6}
  \optbox[position=start](A)(B|A)
  \optbox[position=end](B|A)(B)
  \drawfiber[fiberstyle=angle, linearc=0.5]{1}{2}
\end{pspicture}
```

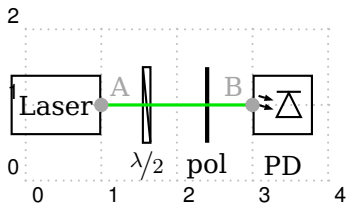
2.5. Schritt-für-Schritt-Beispiele

2.5.1. Freistrahlaufbau

In diesem ersten Beispiel werden alle Komponenten auf eine Verbindungslinie verteilt und mit einem Strahl verbunden.

Zuerst werden Anfangsknoten $\langle A \rangle$ und Endknoten $\langle B \rangle$ definiert. Die erste `\optbox` wird an den Anfang platziert, die Wellenplatte und der Polarisator werden innerhalb der Verbindungslinie positioniert. Der Parameter `position` erwartet Werte zwischen `0` und `1`, die relativ zur Länge der Verbindungslinie sind, oder `start`/`end`. Der Detektor wird immer an das Ende der Linie gesetzt.

Zuletzt werden alle Komponenten mit einem einfachen Lichtstrahl verbunden.



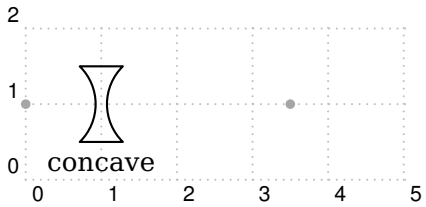

```
\begin{pspicture}[showgrid](4,2)
  \pnode(1,1){A}\pnode(3,1){B}
  \optbox[position=start, optboxwidth=1.2,
      labeloffset=0](A)(B){Laser}
  \optretplate[position=0.3](A)(B){$\nicefrac{\lambda}{2}$}
  \optplate[position=0.7](A)(B){pol}
  \optdetector[dettype=diode](A)(B){PD}
  \drawbeam{1}{2}{3}{4}
\end{pspicture}
```

2.5.2. Galilei-Fernrohr

Im Folgenden wird ein sehr einfacher Aufbau mit einer konkaven und einer konvexen Linse schrittweise erläutert.

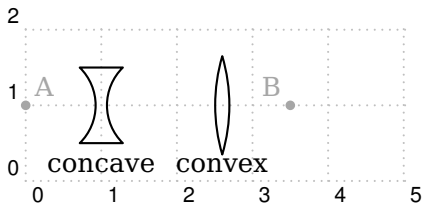
Zuerst werden die Referenzknoten $\langle A \rangle$ und $\langle B \rangle$ definiert, die für die Ausrichtung der Komponenten verwendet werden. Zur besseren Übersicht werden beide Knoten gekennzeichnet, desweiteren dient ein Koordinatengitter der Orientierung.

Die erste Linse soll konkav gekrümmte Grenzflächen haben. Mit `lensradius` werden die Radien der linken und rechten Fläche auf negative Werte gesetzt, was eine konkave Krümmung ergibt. Die Linse wird mit `abspos` eine Einheit vom $\langle A \rangle$-Knoten entfernt positioniert.



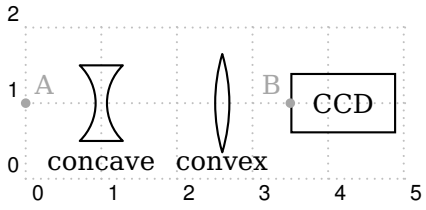

```
\begin{pspicture}[showgrid](0,-0.3)(5,2.3)
  \pnode(0,1){A}\pnode(3.5,1){B}
  \lens[lensradius=-0.7,
        abspos=1](A)(B){concave}
\end{pspicture}
```

Die zweite Linse bekommt eine konvexe Krümmung (`lensradius` ist positiv) und wird `2.6` Einheiten von ⟨*A*⟩ positioniert. Die Linsenhöhe wird mit `lensheight` eingestellt.



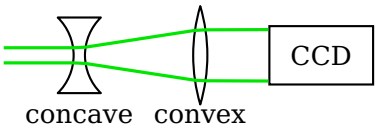

```
\begin{pspicture}[showgrid](0,-0.3)(5,2.3)
  \pnode(0,1){A}\pnode(3.5,1){B}
  \lens[lensradius=-0.7,
        abspos=1](A)(B){concave}
  \lens[lensradius=2.3, lensheight=1.3,
        abspos=2.6](A)(B){convex}
\end{pspicture}
```

Die Lichtstrahlen sollen auf einer CCD-Kamera enden, die mit einer `\optbox` gezeichnet wird. Mit `position`=end wird die Box ans Ende der Referenzlinie von ⟨*A*⟩ nach ⟨*B*⟩ gesetzt, die Beschriftung wird mit `labeloffset` in die Mitte der Komponente gelegt.



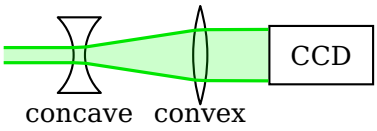

```
\begin{pspicture}[showgrid](0,-0.3)(5,2.3)
  \pnode(0,1){A}\pnode(3.5,1){B}
  \lens[lensradius=-0.7,
        abspos=1](A)(B){concave}
  \lens[lensradius=2.3, lensheight=1.3,
        abspos=2.6](A)(B){convex}
  \optbox[position=end, labeloffset=0](A)(B){CCD}
\end{pspicture}
```

Zum Schluß wird ein aufgeweiteter Strahl mit einer Strahlbreite von `0.2` und ohne anfängliche Divergenz durch die Komponenten geschickt. Der Strahlengang wird mittels Raytracing berechnet. Die Linsenparameter in diesem Beispiel wurden so gewählt das bei kollimiertem Eingangsstrahl der Ausgangsstrahl ebenfalls kollimiert ist.




```
\begin{pspicture}(5,2)
  \pnode(0,1){A}\pnode(3.5,1){B}
  \lens[lensradius=-0.7,
        abspos=1](A)(B){concave}
  \lens[lensradius=2.3, lensheight=1.3,
        abspos=2.6](A)(B){convex}
  \optbox[position=end, labeloffset=0](A)(B){CCD}
  \drawwidebeam[beamwidth=0.2]{(A)}{1}{2}{3}
\end{pspicture}
```

Das Aussehen des Lichtstrahls wird über den Beam-Stil eingestellt, und kann z.B. mit einer halbtransparenten Füllung versehen werden.



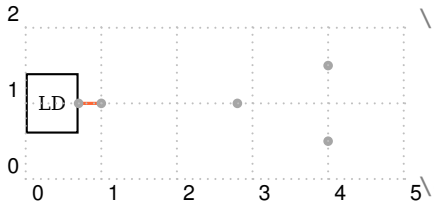

```
\begin{pspicture}(5,2)
  \pnode(0,1){A}\pnode(3.5,1){B}
  \lens[lensradius=-0.7,
        abspos=1](A)(B){concave}
  \lens[lensradius=2.3, lensheight=1.3,
        abspos=2.6](A)(B){convex}
  \optbox[position=end, labeloffset=0](A)(B){CCD}
  \addtopsstyle{Beam}{fillstyle=solid,
    fillcolor=green, opacity=0.2}
  \drawwidebeam[beamwidth=0.2]{(A)}{1}{2}{3}
\end{pspicture}
```

2.5.3. Faseraufbau

Dieses Beispiel zeigt, wie ein faseroptischer Aufbau mit automatisch gezeichneten Faserverbindungen konstruiert werden kann.

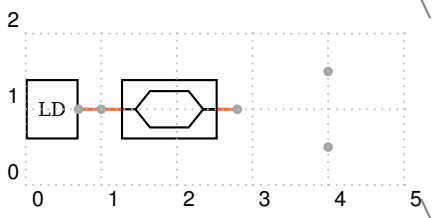
Auch hier werden die Referenzknoten zur besseren Orientierung eingezeichnet. Im Gegensatz zum vorhergehenden Beispiel benötigen wir mehr Referenzknoten, da die Faserkomponenten direkt mit diesen verbunden werden.

Der Laser wird wieder an den Anfang gesetzt. Der Abstand der Beschriftung wird mit dem Parameter `label` eingestellt, mit dem alle Beschriftungsparameter gesammelt eingestellt werden können. Da eine `\optbox` sowohl als Freistrahl- als auch als Faserkomponente verwendet werden kann, wird in der Standardeinstellung keine Faser gezeichnet, was mit `fiber` geändert werden kann. Die Breite dieser und aller folgenden `\optbox` wird auf `0.7` geändert.



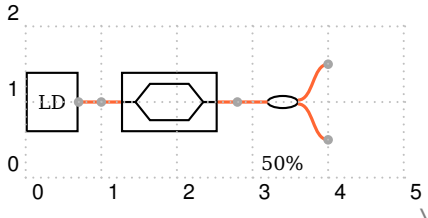

```
\begin{pspicture}[showgrid=true](4.7,2)
  \pnode(0.7,1){A}\pnode(1,1){B}\pnode(2.8,1){C}
  \pnode(4,1.5){D}\pnode(4,0.5){E}
  \psset{optboxwidth=0.7, labelstyle=\scriptsize}
  \optbox[position=start, label=0, fiber](A)(B){LD}
\end{pspicture}
```

Als nächstes wird ein Mach-Zehnder-Modulator ohne Beschriftung gezeichnet, der automatisch mit Fasern mit seinen Referenzknoten verbunden wird.




```
\begin{pspicture}[showgrid=true](4.7,2)
  \pnode(0.7,1){A}\pnode(1,1){B}\pnode(2.8,1){C}
  \pnode(4,1.5){D}\pnode(4,0.5){E}
  \psset{optboxwidth=0.7, labelstyle=\scriptsize}
  \optbox[position=start, label=0, fiber](A)(B){LD}
  \optmzm(B)(C)
\end{pspicture}
```

Danach wird ein Faserkoppler mit einem Eingang und zwei Ausgängen gezeichnet. Der Koppler wird zwischen dem Eingangsknoten und dem Mittelpunkt der beiden Ausgangsknoten ausgerichtet.

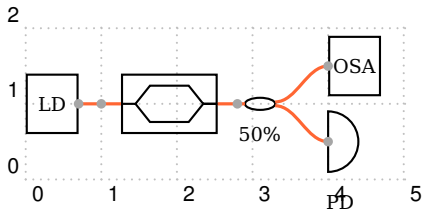



```
\begin{pspicture}[showgrid=true](4.7,2)
  \pnode(0.7,1){A}\pnode(1,1){B}\pnode(2.8,1){C}
  \pnode(4,1.5){D}\pnode(4,0.5){E}
  \psset{optboxwidth=0.7, labelstyle=\scriptsize}
  \optbox[position=start, label=0, fiber](A)(B){LD}
  \optmzm(B)(C)
  \wdmsplitter(C)(D)(E){50\%}
\end{pspicture}
```

An die Enden des Kopplers wird ein optischer Spektrumanalysator (OSA) und eine Leistungsmessgerät (PD) gesetzt. Der `\optdetector` wird immer ans Ende der Referenzlinie gesetzt (`position`=end). Diese beiden Komponenten werden ohne weitere Einstellungen ohne Faserverbindungen gezeichnet. Das passt, da ihre jeweiligen Endpunkte mit den Ausgängen des Kopplers zusammenfallen.

Die Schreibweise `(A|E)` nimmt die x-Koordinate von $\langle A \rangle$ und die y-Koordinate von $\langle E \rangle$.

Bsp. 2.1




```
\begin{pspicture}[showgrid](4.7,2)
  \pnode(0.7,1){A}\pnode(1,1){B}\pnode(2.8,1){C}
  \pnode(4,1.5){D}\pnode(4,0.5){E}
  \psset{optboxwidth=0.7, labelstyle=\scriptsize}
  \optbox[position=start, label=0, fiber](A)(B){LD}
  \optmzm(B)(C)
  \wdmsplitter[abspos=0.3, label=0.4](C)(D)(E){50\%}
  \optbox[position=end, label=0](A|D)(D){OSA}
  \optdetector(A|E)(E){PD}
\end{pspicture}
```

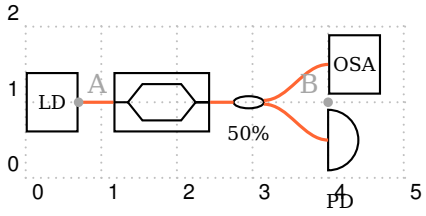
2.5.4. Faseraufbau (Alternative)

Dieses Beispiel ist vom Ergebnis her identisch mit dem vorangegangenen (Bsp. 2.5.3), wird aber anders angegangen.

Hier verwenden wir nicht die automatischen Faserverbindungen sondern zeichnen, wie bei Freistrahlaufbauten, zuerst die Komponenten und verbinden diese zum Schluß explizit mit Fasern (`\drawfiber`). Daher benötigen wir weniger Referenzknoten, die Faserverbindungen werden mit `fiber`=none global unterdrückt.

Welche Herangehensweise verwendet wird ist meistens Geschmackssache, bei Ringstrukturen wie z.B. Faserlasern kann das nachträgliche Verbinden mit `\drawfiber` von Vorteil sein.

Bsp. 2.2

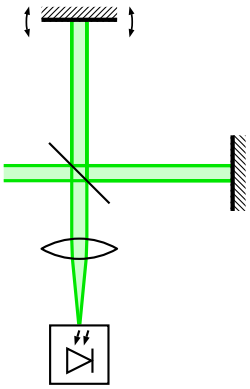



```
\begin{pspicture}[showgrid](4.9,2)
  \pnode(0.7,1){A}\pnode(4,1){B}
  \psset{optboxwidth=0.7, labelstyle=\scriptsize,
      fiber=none}
  \optbox[position=start, labeloffset=0](A)(B){LD}
  \optmzm[abspos=1.1](A)(B)
  \wdmsplitter[abspos=2.25, label=0.4](A)(B)(B){50\%}
  \optbox[position=end,label=0,
      compshift=0.5](A)(B){OSA}
  \optdetector[compshift=-0.5](A)(B){PD}
  \drawfiber{1}{2}{3}{4}
  \drawfiber{3}{5}
\end{pspicture}
```

2.5.5. Michelson-Interferometer

In diesem Beispiel werden neben Linsen auch reflektierende Elemente (Spiegel) eingesetzt sowie Elemente, die sowohl transmittieren als auch reflektieren (Strahlteiler).

Bsp. 2.3



```
1  \begin{pspicture}(3.2,5)
2    \pnode(0,3){A}\pnode(1,3){BS}\pnode(3,3){M1}
3    \pnode(1,5){M2}\pnode(1,1){PD}
4    \psset{mirrortype=extended, mirrordepth=0.2}
5    \begin{optexp}
6      \beamsplitter[bsstyle=plate, compname=BS](A)(BS)(PD)
7      \mirror[compname=M1](BS)(M1)(BS)
8      \mirror[compname=M2, variable](BS)(M2)(BS)
9      \lens[compname=L](BS)(PD)
10     \optdetector[compname=Det, dettype=diode](BS)(PD)
11     \addtopsstyle{Beam}{beamwidth=0.2, fillstyle=solid,
           fillcolor=green!20!white}
12     \drawwidebeam{(A)}{BS}{M1}{BS}{M2}{BS}{L}{Det}
13   \end{optexp}
14 \end{pspicture}
```

Wir beginnen wieder, indem wir die Referenzknoten definieren (Zeile 2–3) und ein paar grundlegende Einstellungen setzen (Zeile 4).

Dann werden der Reihe nach alle Komponenten platziert (Zeile 6–10). Die reflektierenden benötigen drei Referenzknoten: den Eingangsknoten, den Knoten an dem die reflektierende Grenzfläche ist und den Ausgangsknoten. Anschließend wird noch das Aussehen des Lichtstrahls festgelegt (Zeile 11) und der Strahl gezeichnet (Zeile 12).

Diese Mal haben wir den Komponenten Namen gegeben (`compname`) die beim Raytracing anstelle der IDs – der automatisch vergebenen Nummern – verwendet werden können. Dies kann die Nachverfolgung des Strahlengangs erleichtern.

Alle Komponenten und Strahlen wurden zusätzlich in eine `optexp`-Umgebung gekapselt. Innerhalb dieser werden alle Lichtstrahlen hinter die Komponenten gelegt, sodass diese nicht verdeckt werden.

3. Allgemeine Komponentenparameter

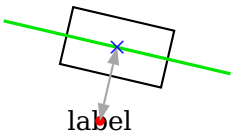
Dieses Kapitel enthält alle Informationen zu allgemeinen Parametern und Konzepten. Das beinhaltet das Setzen von Beschriftungen (3.1), Positionierung (3.2), Drehen und Verschieben von Komponenten (3.3) und Anpassen der Aussehens (3.4 und 3.5).

3.1. Beschriftungen

Alle Komponenten können mit einer Beschriftung versehen werden, deren Position und Ausrichtung genau angepasst werden kann. Die Beschriftung ist für jede Komponente optional. Sollten die vielfältigen Parameter nicht ausreichend sein, können Sie den Beschriftungsknoten – in einigen Beispielen mit einem roten Punkt gekennzeichnet – auch direkt verwenden (siehe auch Kap. 7.4).

`labeloffset`=⟨*num*⟩ default: `0.8`

Der Abstand des Referenzknotens für die Beschriftung (roter Punkt) zur Komponentenmitte (blaues Kreuz).



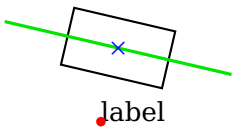

```
\begin{pspicture}(3,1.8)
  \optbox[beam, labeloffset=1](0,1.5)(3,0.8){label}
\end{pspicture}
```

`labelstyle`=⟨*macros*⟩

Der Schriftstil der Beschriftungen.

`labelalign`=⟨*refpoint*⟩ default: `c`

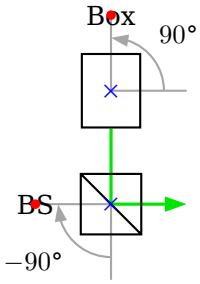
Definiert die Ausrichtung der Beschriftung bezüglich des Beschriftungsknotens.




```
\begin{pspicture}(3,1.8)
  \psset{beam, labeloffset=1}
  \optbox[labelalign=bl](0,1.5)(3,0.8){label}
\end{pspicture}
```

`labelangle`=⟨*num*⟩ default: `0`

Der Rotationswinkel des Beschriftungsknotens um die Mitte. Der Nullpunkt hängt sowohl von der jeweiligen Komponenten und dem gewählten Referenzsystem ab (siehe `labelref`).

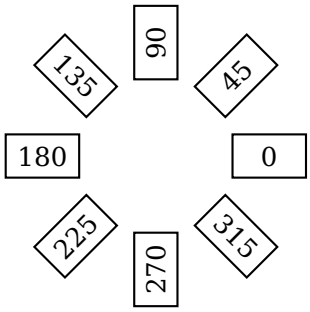



```
\begin{pspicture}(-0.3,0)(2,3.7)
  \psset{labeloffset=1, optboxwidth=1, arrowscale=1.5, arrowinset=0}
  \optbox[position=end, labelangle=90](1,1)(1,2){Box}
  \beamsplitter[labelangle=-90](1,2)(1,1)(2,1){BS}
  \drawbeam[arrows=->]{1}{2}{(2,1)}
\end{pspicture}
```

`labelref`=`relative`, `relgrav`, `global` default: `relgrav`

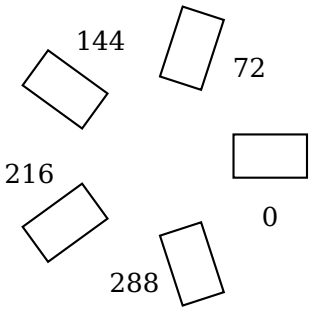
Wählt das Bezugssystem für `labelangle` und die Orientierung der Beschriftung.

`relative` Die Beschriftung ist parallel zu der Referenzlinie der Komponente und der Winkel wird immer auf den Bereich zwischen $-90°$ und $+90°$ zurückgerechnet.



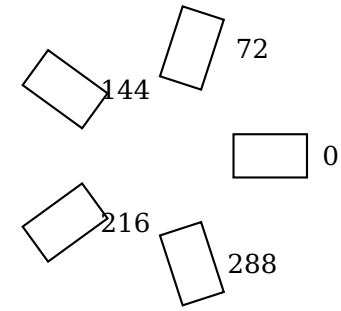

```
\begin{pspicture}(0,-2.2)(2,2.2)
  \psset{position=end, optboxsize=1 0.6, labeloffset=0,
      labelref=relative}
  \multido{\i=0+45}{8}{%
    \optbox(0,0)(1;\i){\i}
  }
\end{pspicture}
```

`relgrav` Die Beschriftung ist immer horizontal, lediglich der Beschriftungsknoten wird zusammen mit der Komponente gedreht.




```
\begin{pspicture}(0,-2.2)(2,2.2)
  \psset{position=end, optboxsize=1 0.6, labelref=relgrav}
  \multido{\i=0+72}{5}{%
    \optbox(0,0)(1;\i){\i}
  }
\end{pspicture}
```

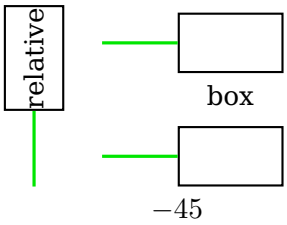
`global` Die Beschriftung ist unabhängig von der Komponentenausrichtung.




```
\begin{pspicture}(0,-2.2)(2,2.2)
  \psset{position=end, optboxsize=1 0.6, labelref=global}
  \multido{\i=0+72}{5}{%
    \optbox(0,0)(1;\i){\i}
  }
\end{pspicture}
```

`label`=⟨*offset*⟩[⟨*angle*⟩[⟨*refpoint*⟩[⟨*labelref*⟩]]]

Diese Option erlaubt die kompakte Definition mehrerer Beschriftungsparameter. Es können bis zu vier Leerzeichen-getrennte Argumente übergeben werden (`labeloffset`, `labelangle`, `labelalign` und `labelref`). Mit einem Punkt kann eine Option übersprungen werden.




```
\begin{pspicture}(0,0)(4,3)
  \psset{endbox, beam}
  \optbox[label=0.7](1.5,2.5)(2.5,2.5){box}
  \optbox[label=1 -45](1.5,1)(2.5,1){$-45$}
  \optbox[label=0 . . relative]%
        (0.6,0.6)(0.6,1.6){relative}
\end{pspicture}
```

`innerlabel`=`true`

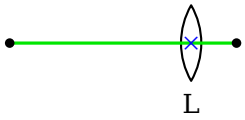
Das ist ein Alias für `label=0 . . relative`

3.2. Positionierung

Alle Komponenten außer den Freistrahl-Dreipolen können zwischen ihren beiden Referenzknoten \oenodeRefA und \oenodeRefB positioniert werden.

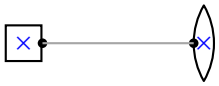
position=⟨*num*⟩, start, end

Ist der Wert eine Zahl, so wird damit die relative Position der Mitte eines Objektes (blaues Kreuz) bezüglich seiner beiden Referenzknoten (schwarze Punkte) gesetzt. Das ist äquivalent zum npos Parameter von \ncput, d.h. eine Zahl im Intervall $[0, 1]$ (siehe pst-node Dokumentation[1]).



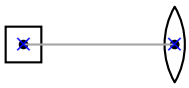

```
\begin{pspicture}(3,1.5)
  \lens[beam, position=0.8](0,1)(3,1){L}
\end{pspicture}
```

Der Wert start setzt ein Objekt an den Anfang, end ans Ende der Referenzlinie, so dass die entsprechende Grenzfläche auf dem Referenzknoten (schwarzer Punkt) liegt, und nicht die Mitte des Objektes (blaues Kreuz), wie es mit position=1 bzw. position=0 der Fall wäre. Unter Verwendung von Zahlen wäre dieses Verhalten deutlich schwieriger und weniger flexibel zu erreichen, da die Größe und Form des Objektes berücksichtigt werden muss.



```
\begin{pspicture}(3,2)
  \pnode(0.5,1){A}\pnode(2.5,1){B}\psdot(A)\psdot(B)
  \optbox[optboxsize=0.5 0.5, position=start](A)(B)
  \lens[position=end](A)(B)
\end{pspicture}
```

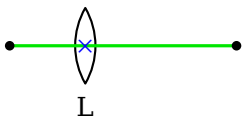
Hier ist im Vergleich dazu die Positionierung mit position=0/1.



```
\begin{pspicture}(3,2)
  \pnode(0.5,1){A}\pnode(2.5,1){B}\psdot(A)\psdot(B)
  \optbox[optboxsize=0.5 0.5, position=0](A)(B)
  \lens[position=1](A)(B)
\end{pspicture}
```

abspos=⟨*num*⟩, start, end

Eine Zahl setzt die absolute Position eines Objektes auf der Referenzlinie. Die Werte start und end werden identisch wie bei position behandelt.




```
\begin{pspicture}(3,1.5)
  \lens[beam, abspos=1](0,1)(3,1){L}
\end{pspicture}
```

[1] http://mirror.ctan.org/help/Catalogue/entries/pst-node.html

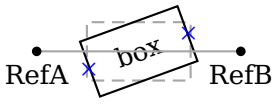
3.3. Drehen und Verschieben

Alle Komponenten können versetzt und um verschiedene Bezugspunkte gedreht werden.

Beachten Sie, dass mit den Komponenten auch die Grenzflächenknoten (Kap. 7.6, in den folgenden Beispielen mit blauen Kreuzen markiert) transformiert werden, wodurch Verbindungen beinflußt werden. Die transformierten Referenzknoten sind als eigene Knoten verfügbar (Kap. 7.2).

`angle`=⟨*num*⟩ default: `0`

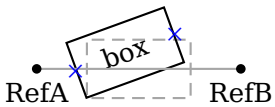
Dreht eine Komponente um den Winkel ⟨*num*⟩ (in Grad). Die Ausgangsposition ist grau gestrichelt eingezeichnet.




```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(2.7,1){B}
  \optbox[angle=20, innerlabel](A)(B){box}
\end{pspicture}
```

`rotateref`=⟨*refpoint*⟩ default: `c`

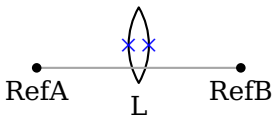
Setzt den Bezugspunkt für die Drehung der Komponente. Bitte lesen Sie Kap. 7.7 für eine genauere Beschreibung und entnehmen Sie Kap. 11.2.1 die möglichen Bezugspunkte für alle Komponenten. Die Ausgangsposition ist grau gestrichelt eingezeichnet.



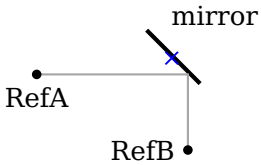

```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(2.7,1){B}
  \optbox[angle=20, rotateref=bl, innerlabel](A)(B){box}
\end{pspicture}
```

`compshift`=⟨*num*⟩ default: `0`

Verschiebt eine Komponente vertikal bezüglich ihrer Referenzlinie (siehe Kap. 7.2). Reflektive Komponenten werden entlang der reflektiven Grenzfläche verschoben.




```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(2.7,1){B}
  \lens[compshift=0.3](A)(B){L}
\end{pspicture}
```



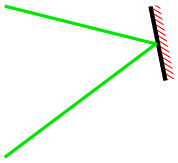

```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(2,0){B}
  \mirror[compshift=0.3](A)(2,1)(B){mirror}
\end{pspicture}
```

3.4. Verwendung von Stilen

Stile sind benutzerdefinierte Grafik-Konfigurationen für PSTricks-Objekte. Das `pst-optexp`-Paket macht umfangreichen Gebrauch von Stilen um einzelne, logische Bereiche von Komponenten flexibel gestalten zu können, was über das optionale Argument meistens nicht möglich ist.

Vorhandene Stile können mit `\addtopsstyle` erweitert und mit `\newpsstyle` überschrieben werden.

Als Beispiel nehmen wir einen `extended` Spiegel und möchten nur den „extended“-Teil verändern. Hierfür wird der Stil `ExtendedMirror` bereitgestellt:



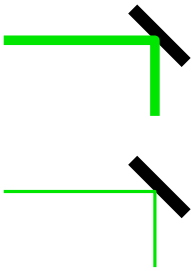
```
\begin{pspicture}(3,2)
  \addtopsstyle{ExtendedMirror}{hatchcolor=red}
  \mirror[mirrortype=extended, beam](0,0)(2,1.5)(0,2)
\end{pspicture}
```

Die veränderbaren Unterbereiche einer Komponente sind in der jeweiligen Referenz zu finden.

3.5. Aussehen der Komponenten

`OptComp` ⟨*psstyle*⟩

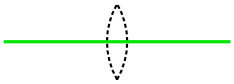
Bestimmt das grundlegende Aussehen aller optischen Komponenten. Die üblichen Parameter, wie z.B. `linestyle`, würden auch die Verbindungen beeinflussen, die mit der Komponente zusammen gezeichnet werden (`beam`, `fiber` usw.). Mit `newOptComp` und `addtoOptComp` kann der Stil für einzelne Komponenten oder zusammen mit `\newpsobject` im optionalen Argument verwendet werden.



```
\begin{pspicture}(2.5,4)
  \psset{beam}
  % falsch, die Strahlbreite wird auch geaendert
  \mirror[linewidth=3\pslinewidth](0,3)(2,3)(2,2)
  % korrektes Ergebnis
  \addtopsstyle{OptComp}{linewidth=3\pslinewidth}
  \mirror(0,1)(2,1)(2,0)
\end{pspicture}
```

OptionalStyle ⟨*psstyle*⟩ default: `linestyle=dashed,dash=1.5pt 1pt`

Der Stil wird zusätzlich zu OptComp auf Komponenten angewendet, die als optional markiert sind.



```
\begin{pspicture}(3,1.5)
\lens[optional, beam](0,1)(3,1)
\end{pspicture}
```

OptionalStyle wird nach OptComp angewendet, so dass einzelne Einstellungen überschrieben werden können.

```
\begin{pspicture}(3,1.5)
\addtopsstyle{OptComp}{linewidth=3\pslinewidth, linecolor=red}
\newpsstyle{OptionalStyle}{linecolor=blue}
\lens(0,1)(2,1)
% Liniefarbe wird ueberschrieben, die Linienbreite nicht.
\lens[optional](0,1)(3,1)
\end{pspicture}
```

VariableStyle ⟨*psstyle*⟩ default: `linewidth=0.8\pslinewidth, arrowinset=0, arrowscale=0.8, arrows=<->`

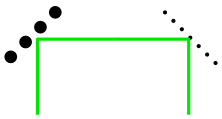
Der Stil der Pfeile für verstellbare Komponenten (\mirror und \optgrating).

addtoOptComp=⟨*list*⟩

Der vorhandene OptComp-Stil wird lokal um die Parameter in ⟨*list*⟩ erweitert, ⟨*list*⟩ muss mit geschweiften Klammern gekapselt werden.

newOptComp=⟨*list*⟩

Ähnlich wie addtoOptComp, nur wird der OptComp-Stil mit den neuen Parametern überschrieben.



```
\begin{pspicture}(3,1.5)
  \psset{beam}
  \newpsstyle{OptComp}{linewidth=3\pslinewidth}
  \mirror[addtoOptComp={linestyle=dotted}](0.5,0)(0.5,1)(1.5,1)
  % Ueberschreibt linewidth-Einstellungen
  \mirror[newOptComp={linestyle=dotted}](1.5,1)(2.5,1)(2.5,0)
\end{pspicture}
```

optional=true, false default: false

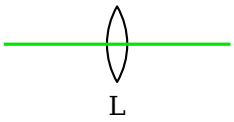
Markiert eine Komponente als optional indem der Stil OptionalStyle angewendet wird.

4. Freistrahlkomponenten

In diesem Kapitel werden alle Freistrahlkomponenten und deren Parameter beschrieben. Freistrahlkomponenten können, im Gegensatz zu Faserkomponenten, transmittive und reflektive Grenzflächen haben, die für Raytracing von Lichtstrahlen verwendet werden können. Es gibt zwei unterschiedliche Komponententypen: manche benötigen nur zwei Knoten um positioniert zu werden (4.1–4.10), wohingegen z.B. Spiegel drei Knoten benötigen um korrekt ausgerichtet zu werden (4.11–4.16).

4.1. Linse

`\lens[`⟨*options*⟩`](`⟨*in*⟩`)(`⟨*out*⟩`){`⟨*label*⟩`}`

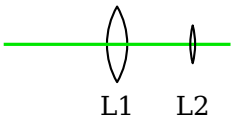



```
\begin{pspicture}(3,1.5)
  \lens[beam](0,1)(3,1){L}
\end{pspicture}
```

Eine Linse wird durch ihre Höhe und die beiden Radien der Eingangs- und Ausgangsflächen definiert.

`lensheight=`⟨*num*⟩ default: 1

Setzt die Höhe der Linse.



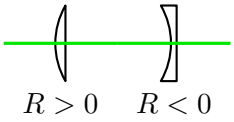

```
\begin{pspicture}(3,1.5)
  \lens[beam](0,1)(3,1){L1}
  \lens[lensheight=0.5](2,1)(3,1){L2}
\end{pspicture}
```

`lensradiusleft=`⟨*num*⟩ default: 1

Setzt den Radius der linken Linsengrenzfläche. Ein positiver Wert ⟨*num*⟩ ist für eine konvexe, ein negativer für eine konkave Krümmung. Null ergibt eine ebene Fläche.

lensradiusright=⟨*num*⟩ default: 1

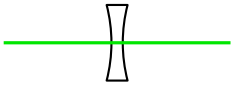
Das Gleiche wie lensradiusleft, nur für die rechte Grenzfläche.




```
\begin{pspicture}(3,1.5)
  \psset{lensradiusright=0, beam}
  \lens[lensradiusleft=1](0,1)(1.5,1){$R > 0$}
  \lens[lensradiusleft=-1](1.5,1)(3,1){$R < 0$}
\end{pspicture}
```

lensradius=⟨*left*⟩[⟨*right*⟩]

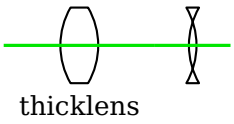
Setzt die linke und rechte Linsenkrümmung. Wird nur ein Wert angegeben, so wird dieser für beide Krümmungen verwendet.



```
\begin{pspicture}(3,1.5)
  \lens[lensradius=-2, beam](0,1)(3,1)
\end{pspicture}
```

lenswidth=⟨*num*⟩ default: 0

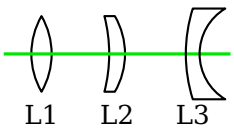
Üblicherweise werden nur die Höhe und die beiden Radien für die Konstruktion der Linse verwendet, die Mittenbreite wird automatisch berechnet. Wenn lenswidth auf einen Wert größer Null gesetzt wird, wird diese Breite verwendet. Das ist nur sinnvoll, wenn dicke Linsen gezeichnet werden sollen, da es andernfalls für zu kleine Werte zu unschönen Ergebnissen führt.




```
\begin{pspicture}(3,2)
  \lens[lenswidth=0.5, beam](0,1)(2,1){thicklens}
  \lens[lenswidth=0.1, beam](2,1)(3,1)
\end{pspicture}
```

lens=⟨*radiusleft*⟩[⟨*radiusright*⟩[⟨*height*⟩[⟨*width*⟩]]]

Eine Option zum gleichzeitigen Setzen mehrerer Linsenparameter. Werte mittendrin können mit einem Punkt übersprungen und Werte am Ende können weggelassen werden (z.B. `lens=. 1 1`).



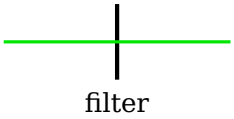

```
\begin{pspicture}(3,1.5)
  \lens[lens=1](0,1)(1,1){L1}
  \lens[lensradius=-2 1, beam](0,1)(3,1){L2}
  \lens[lens=2 -0.7 1.2](2,1)(3,1){L3}
\end{pspicture}
```

`thicklens`=`true`, `false` default: `false`

Diese Option was bis Version 2.1 notwendig um dicke Linsen der Breite `lenswidth` zu zeichnen, anstatt die berechnete Breite zu verwenden. Seit Version 3.0 ist diese Option überflüssig, da dicke Linsen gezeichnet werden, sobald `lenswidth` auf einen positiven Wert gesetzt wird.

4.2. Optisches Plättchen

`\optplate`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}




```
\begin{pspicture}(3,1.5)
  \optplate[beam](0,1)(3,1){filter}
\end{pspicture}
```

`plateheight`=⟨*num*⟩ default: `1`

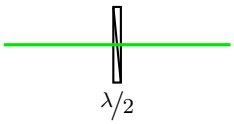
Die Höhe des Plättchens.

`platelinewidth`=⟨*num*⟩ or ⟨*dimen*⟩ default: `2\pslinewidth`

Die Linienbreite des Plättchens. Diese könnte ebenfalls über den `linewidth` Parameter gesetzt werden. Mit diesem Parameter kann aber die Linienbreite aller Plättchen global eingestellt werden.

4.3. Verzögerungsplättchen

`\optretplate`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}




```
\begin{pspicture}(3,1.5)
  \optretplate[beam](0,1)(3,1){$\nicefrac{\lambda}{2}$}
\end{pspicture}
```

`plateheight`=⟨*num*⟩ default: `1`

Die Höhe des Plättchens.

`platewidth`=⟨*num*⟩ default: `0.1`

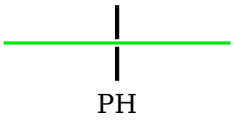
Die Breite des Plättchens.

`platesize`=⟨*width*⟩ ⟨*height*⟩

Die Breite und Höhe des Plättchens, äquivalent zum Aufruf von `platewidth` und `plateheight`.

4.4. Lochblende

`\pinhole`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}




```
\begin{pspicture}(3,1.5)
  \pinhole[beam](0,1)(3,1){PH}
\end{pspicture}
```

`outerheight`=⟨*num*⟩ default: `1`

Die Gesamthöhe der Lochblende.

`innerheight`=⟨*num*⟩ default: `0.1`

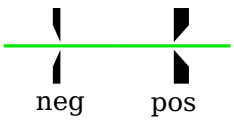
Die Höhe des Lochs.

`phlinewidth`=⟨*num*⟩ or ⟨*dimen*⟩ default: `2\pslinewidth`

Die Linienbreite der Lochblende. Diese könnte ebenfalls über den `linewidth` Parameter gesetzt werden. Mit diesem Parameter kann aber die Linienbreite aller Lochblenden global eingestellt werden.

`phwidth`=⟨*num*⟩ default: `0`

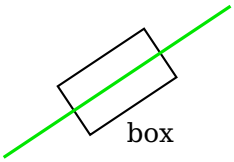
Die Lochblende wird plastischer gezeichnet, falls `phwidth` ungleich Null ist. In diesem Fall wird `phlinewidth` ignoriert. Bei einem negativen Wert wird die Form gespiegelt.




```
\begin{pspicture}(3,1.5)
  \pinhole[beam, phwidth=-0.1](0,1)(1.5,1){neg}
  \pinhole[beam, phwidth=0.2](1.5,1)(3,1){pos}
\end{pspicture}
```

4.5. Box

`\optbox`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}




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

`optboxwidth`=⟨*num*⟩ default: `1.4`

Die Breite der Box.

`optboxheight`=⟨*num*⟩ default: `0.8`

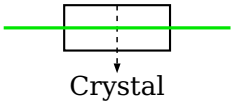
Die Höhe der Box.

`optboxsize`=⟨*width*⟩ ⟨*height*⟩

Die Breite und Höhe der Box, äquivalent zum Aufruf von `optboxwidth` und `optboxheight`.

4.6. Kristall

`\crystal`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}




```
\begin{pspicture}(3,1.3)
  \crystal[beam](0,1)(3,1){Crystal}
\end{pspicture}
```

`crystalwidth`=⟨*num*⟩ default: `1.4`

Die Breite des Kristalls.

`crystalheight`=⟨*num*⟩ default: `0.6`

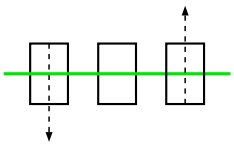
Die Höhe des Kristalls.

`crystalsize`=⟨*width*⟩ ⟨*height*⟩

Die Breite und Höhe des Kristalls, äquivalent zum Aufruf von `crystalwidth` und `crystalheight`.

caxislength=⟨*num*⟩ default: 0.3

Die zusätzliche Länge des Pfeils für die c-Achse, wird weggelassen falls die Länge 0 ist und gespiegelt falls der Wert negativ ist. Die Gesamtlänge ist caxislength plus crystalheight.



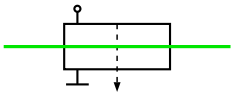
```
\begin{pspicture}(0,0.4)(3,1.5)
  \psset{crystalsize=0.5 0.8, label=-45 . l}
  \crystal[position=0.2, caxislength=0.5](0,1)(3,1){pos}
  \crystal[beam, caxislength=0](0,1)(3,1){0}
  \crystal[position=0.8, caxislength=-0.5](0,1)(3,1){neg}
\end{pspicture}
```

caxisinv=true, false default: false

Invertiert die Richtung der c-Achse, ist äquivalent zu einem Vorzeichenwechsel von caxislength.

voltage=true, false default: false

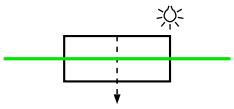
Zeichnet einen Spannungsanschluss und ein Erdungszeichen.



```
\begin{pspicture}(0,0.4)(3,1.5)
  \crystal[voltage, beam](0,1)(3,1)
\end{pspicture}
```

lamp=true, false default: false

Zeichnet eine Lampe neben den Kristall.



```
\begin{pspicture}(0,0.4)(3,1.7)
  \crystal[lamp, beam](0,1)(3,1)
\end{pspicture}
```

lampscale=⟨*num*⟩

Skalierung der Lampe. Dieser Parameter ist seit Version 3.0 veraltet, verwenden Sie stattdessen den Stil CrystalLamp.

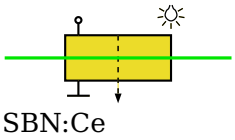
CrystalCaxis ⟨*psstyle*⟩ default: linewidth=0.7\pslinewidth, linestyle=dashed, dash=2pt 2pt, arrowinset=0, arrows=->

Der Stil für die c-Achse inklusive der Pfeilart.

```
\begin{pspicture}(0,0.5)(3,1.6)
  \newpsstyle{CrystalCaxis}{linecolor=red, arrows=->>}
  \crystal[beam](0,1)(3,1)
\end{pspicture}
```

`CrystalLamp` ⟨*psstyle*⟩ default: `linewidth=0.6\pslinewidth`

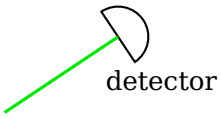
Der Stil für die Hintergrundlampe.




```
\begin{pspicture}(3,1.7)
  \crystal[fillstyle=solid, fillcolor=yellow!90!black,
           label=1.2 -45, voltage,
           lamp, beam](0,1)(3,1){SBN:Ce}
\end{pspicture}
```

4.7. Detektor

`\optdetector`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}




```
\begin{pspicture}(3,1.2)
  \optdetector[beam](0,0)(1.5,1){detector}
\end{pspicture}
```

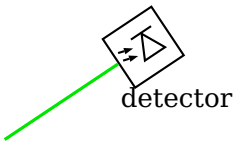
Der Detektor wird immer ans Ende der Referenzlinie platziert (`position=end`). Diese Einstellung kann nicht verändert werden.

`detsize`=⟨*num*⟩ or ⟨*width*⟩ ⟨*height*⟩ default: `0.8`

Wenn eine einzelne Zahl angegeben ist, dann ist das die Seitenlänge (`diode`) bzw. der Durchmesser (`round`) des Detektors. Werden zwei Zahlen angegeben, so sind das Breite und Höhe des Detektors. Beachten Sie, dass `detsize=1` und `detsize=1 1` für `dettype=round` nicht dasselbe ist.

`dettype`=`round`, `diode` default: `round`

Dieser Parameter bestimmt den Typ und damit das Aussehen des Detektors.




```
\begin{pspicture}(3,1.5)
  \optdetector[beam, dettype=diode](0,0)(1.5,1){detector}
\end{pspicture}
```

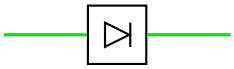
`DetectorStyle` ⟨*psstyle*⟩

Der Stil der Diode für `dettype=diode`.

```
\begin{pspicture}(3,1.5)
  \newpsstyle{OptComp}{linewidth=2\pslinewidth}
  \newpsstyle{DetectorStyle}{linewidth=0.5\pslinewidth}
  \optdetector[dettype=diode](0,1)(2,1)
\end{pspicture}
```

4.8. Optische Diode

\optdiode[⟨options⟩](⟨in⟩)(⟨out⟩){⟨label⟩}


Diode

```
\begin{pspicture}(3,2)
  \optdiode[beam](0,1)(3,1){Diode}
\end{pspicture}
```

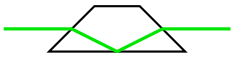
Die optische Diode hat die Voreinstellung allowbeaminside=false.

optdiodesize=⟨num⟩ default: 0.8

Die Seitenlänge der optischen Diode.

4.9. Doveprisma

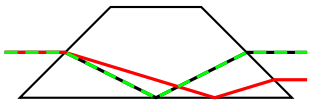
\doveprism[⟨options⟩](⟨in⟩)(⟨out⟩){⟨label⟩}


Dove

```
\begin{pspicture}(3,1.5)
  \doveprism[beam](0,1)(3,1){Dove}
\end{pspicture}
```

doveprismsize=⟨num⟩ or ⟨width⟩ ⟨height⟩ default: 0.6

Wenn eine Zahl angegeben ist, dann ist das die Höhe des Prismas, die Gesamtbreite beträgt das dreifache. Zwei Zahlen geben Breite und Höhe des Doveprisma an. Die Winkel an der Eingangs- und Ausgangsfläche betragen immer 45°.



```
\begin{pspicture}(4,1.5)
  \optplane(0,1)
  \doveprism[doveprismsize=1.2](0,1)(4,1)
  \optplane(4,1)
  \drawbeam[raytrace=false, linecolor=black]{-}
  \drawbeam[n=1.5, linecolor=red]{-}
  \drawbeam[n=*sqrt(5), linecolor=green, linestyle=dashed]{-}
\end{pspicture}
```

4.10. Polarisation

\polarization[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}

```
\begin{pspicture}(3,1)
  \polarization[beam](0,0.5)(3,0.5)
\end{pspicture}
```

polsize=⟨*num*⟩ default: 0.6

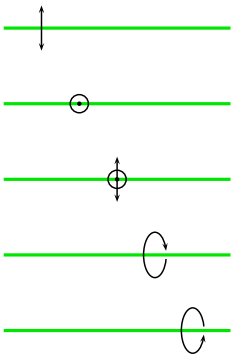
Die Größe des Polarisationszeichens, der Kreis für perp und polmisc ist halb so groß.

pollinewidth=⟨*num*⟩ or ⟨*dimen*⟩ default: 0.7\pslinewidth

Die Linienbreite des Polarisationszeichens. Diese könnte ebenfalls über den linewidth Parameter gesetzt werden. Mit diesem Parameter kann aber die Linienbreite aller Polarisationszeichen global eingestellt werden. Dieser Parameter ist seit Version 3.0 veraltet, verwenden Sie stattdessen den Stil Polarization.

poltype=parallel, perp, misc, lcirc, rcirc default: parallel

Dieser Parameter wählt die Polarizationsart aus.



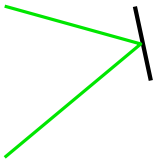
```
\begin{pspicture}(0,-0.3)(3,4.3)
  \psset[optexp]{beam}
  \begin{optexp}
  \polarization[poltype=parallel, abspos=0.5](0,4)(3,4)
  \polarization[poltype=perp, abspos=1](0,3)(3,3)
  \polarization[poltype=misc, abspos=1.5](0,2)(3,2)
  \polarization[poltype=lcirc, abspos=2](0,1)(3,1)
  \polarization[poltype=rcirc, abspos=2.5](0,0)(3,0)
  \end{optexp}
\end{pspicture}
```

Polarization ⟨*psstyle*⟩ default: arrowscale=0.8, linewidth=\pslinewidth, dotsize=3\pslinewidth

Bestimmt das Aussehen der Polarisationssymbole.

4.11. Spiegel

`\mirror`[⟨*options*⟩](⟨*in*⟩)(⟨*center*⟩)(⟨*out*⟩){⟨*label*⟩}




```
\begin{pspicture}(3,2)
  \mirror[beam](0,0)(1.8,1.5)(0,2){M}
\end{pspicture}
```

`mirrorwidth`=⟨*num*⟩ default: `1`

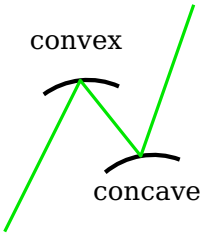
Die Breite des Spiegels.

`mirrorlinewidth`=⟨*num*⟩ or ⟨*dimen*⟩ default: `2\pslinewidth`

Die Linienbreite des Spiegels. Diese könnte ebenfalls über den `linewidth` Parameter gesetzt werden. Mit diesem Parameter kann aber die Linienbreite aller Spiegel global eingestellt werden.

`mirrorradius`=⟨*num*⟩ default: `0`

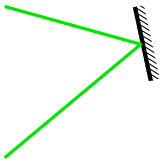
Diese Parameter setzt die Krümmung des Spiegels. Null ergibt einen ebenen, ein negativer Radius einen konvexen und ein positiver Radius einen konkaven Spiegel.




```
\begin{pspicture}(3,3)
  \psset{labeloffset=0.5}
  \mirror[mirrorradius=1](0,0)(1,2)(1.8,1){convex}
  \mirror[mirrorradius=-1](1,2)(1.8,1)(2.5,3){concave}
  \drawbeam{(0,0)}{1}{2}{(2.5,3)}
\end{pspicture}
```

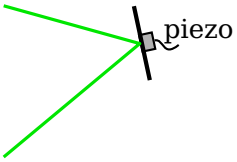
`mirrortype`=`plain`, `piezo`, `extended`, `semitrans` default: `plain`

Diese Parameter wählt die Spiegelart aus. Das Aussehen wird mit `PiezoMirror`, `ExtendedMirror` und `SemitransMirror` gesteuert.



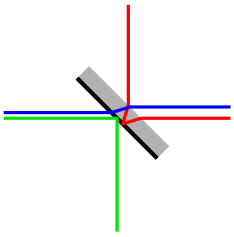

```
\begin{pspicture}(3,2)
  \mirror[mirrortype=extended,
          beam](0,0)(1.8,1.5)(0,2){ext}
\end{pspicture}
```

Beachten Sie, dass der Anschlussdraht für den Piezospiegel nicht gezeichnet wird, falls `extnode` verwendet wird.

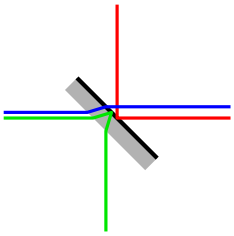



```
\begin{pspicture}(3,2)
\mirror[mirrortype=piezo,
        beam](0,0)(1.8,1.5)(0,2){piezo}
\end{pspicture}
```

Ein halbdurchlässiger Spiegel (`semitrans`) hat, im Gegensatz zu den anderen Spiegeltypen, zwei Grenzflächen, deren Abstand und Position vom Wert und Vorzeichen von `mirrordepth` abhängt. Beachten Sie, dass keine gekrümmten Spiegel dieses Typs möglich sind.



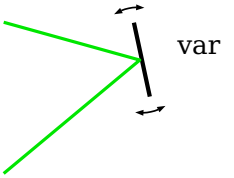
```
\psset{unit=1.5}
\begin{pspicture}(2,2)
  \mirror[mirrortype=semitrans](0,1)(1,1)(1,0)
  \drawbeam{(0,1)}{}{(1,0)}
  \drawbeam[linecolor=red]{(2,1)}{}{(1,2)}
  \drawbeam[beampos=0.05, linecolor=blue]{(0,1)}{}{(2,1)}
\end{pspicture}
```



```
\psset{unit=1.5}
\begin{pspicture}(2,2)
  \mirror[mirrortype=semitrans, mirrordepth=-0.15](0,1)(1,1)(1,0)
  \drawbeam{(0,1)}{}{(1,0)}
  \drawbeam[linecolor=red]{(2,1)}{}{(1,2)}
  \drawbeam[beampos=0.05, linecolor=blue]{(0,1)}{}{(2,1)}
\end{pspicture}
```

`variable`=`true`, `false` default: `false`

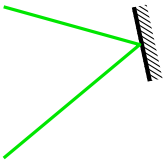
Zeichnet einen verstellbaren Spiegel, der mit zwei zusätzlichen gekrümmten Pfeilen an beiden Seiten angedeutet wird. Das Aussehen, inklusive der Pfeile, wird von `VariableStyle` vorgegeben.




```
\begin{pspicture}(3,2)
  \mirror[beam, variable](0,0)(1.8,1.5)(0,2){var}
\end{pspicture}
```

`mirrordepth`=⟨*num*⟩ default: `0.1`

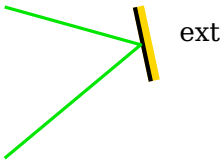
Die Gesamttiefe eines breiten Spiegels.



```
\begin{pspicture}(3,2)
  \mirror[mirrortype=extended, mirrordepth=0.2,
          beam](0,0)(1.8,1.5)(0,2)
\end{pspicture}
```

ExtendedMirror ⟨*psstyle*⟩ default: `linestyle=none, hatchwidth=0.5\pslinewidth, hatchsep=1.4\pslinewidth, fillstyle=hlines`

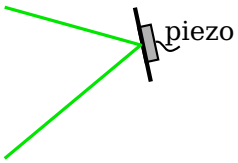
Der Stil für den `extended` Spiegel.




```
\begin{pspicture}(3,2)
  \newpsstyle{ExtendedMirror}{fillstyle=solid,
                              fillcolor=Gold,
                              linestyle=none}
  \mirror[mirrortype=extended,
          beam](0,0)(1.8,1.5)(0,2){ext}
\end{pspicture}
```

PiezoMirror ⟨*psstyle*⟩ default: `fillstyle=solid, fillcolor=black!30`

Der Stil für den `piezo` Spiegel. Dieser kann auch dazu verwendet werden, die Größe des Piezos zu verändern, wie in dem folgenden Beispiel zu sehen ist.



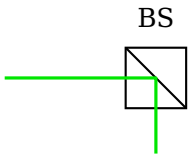

```
\begin{pspicture}(3,2)
  \addtopsstyle{PiezoMirror}{xunit=2}
  \mirror[mirrortype=piezo,
          beam](0,0)(1.8,1.5)(0,2){piezo}
\end{pspicture}
```

SemitransMirror ⟨*psstyle*⟩ default: `linestyle=none, fillstyle=solid, fillcolor=black!30`

Der Stil für den `semitrans` Spiegel.

4.12. Strahlteiler

`\beamsplitter`[⟨*options*⟩](⟨*in*⟩)(⟨*center*⟩)(⟨*out*⟩){⟨*label*⟩}



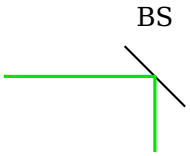

```
\begin{pspicture}(3,2)
  \beamsplitter[beam](0,1)(2,1)(2,0){BS}
\end{pspicture}
```

bssize=⟨*num*⟩ default: `0.8`

Die Größe des Strahlteilers.

bsstyle=`cube`, `plate` default: `cube`

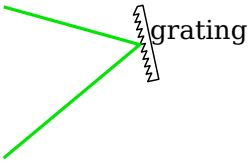
Wählt zwischen zwei Strahlteilertypen: einem Strahlteilerwürfel (`cube`) und einem halbdurchlässigen Spiegel (`plate`). Dieser halbdurchlässige Spiegel ist äquivalent zu `\mirror` mit `mirrortype=semitrans` für `mirrordepth=0`.




```
\begin{pspicture}(3,2)
  \beamsplitter[bsstyle=plate, beam](0,1)(2,1)(2,0){BS}
\end{pspicture}
```

4.13. Optisches Gitter

\optgrating[⟨*options*⟩](⟨*in*⟩)(⟨*center*⟩)(⟨*out*⟩){⟨*label*⟩}




```
\begin{pspicture}(3,2)
  \optgrating[beam](0,2)(1.8,1.5)(0,0){grating}
\end{pspicture}
```

Vor Paketversion 3.0 hieß diese Komponente \optgrid. Seitdem ist diese Benennung veraltet und wird in zukünftigen Versionen entfernt werden.

gratingwidth=⟨*num*⟩ default: 1

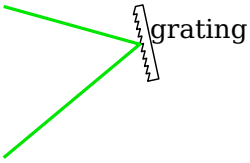
Die Breite des Gitters.

gratingheight=⟨*num*⟩ default: 0.15

Die Gesamthöhe des Gitters.

gratingdepth=⟨*num*⟩ default: 0.075

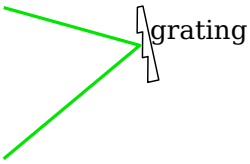
Die absolute Modulationstiefe der Gitterstruktur. Die Gesamthöhe wird an diesen Wert angepasst falls sie kleiner als die Modulationtiefe ist, andernfalls wird sie nicht beinflusst.




```
\begin{pspicture}(3,2)
  \optgrating[gratingdepth=0.05,
              beam](0,2)(1.8,1.5)(0,0){grating}
\end{pspicture}
```

gratingcount=⟨*int*⟩ default: 10

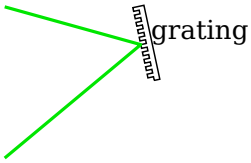
Die Anzahl der Gitterlinien.




```
\begin{pspicture}(3,2)
  \optgrating[gratingcount=3,
              beam](0,2)(1.8,1.5)(0,0){grating}
\end{pspicture}
```

`gratingtype`=`blazed`, `binary` default: `blazed`

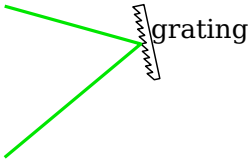
Wählt zwischen einem binären Gitter (`binary`) und einem Blazegitter (`blazed`).




```
\begin{pspicture}(3,2)
  \optgrating[gratingtype=binary,
              beam](0,2)(1.8,1.5)(0,0){grating}
\end{pspicture}
```

`reverse`=`true`, `false` default: `false`

Invertiert die Steigung des Blazegitters.




```
\begin{pspicture}(3,2)
  \optgrating[reverse, beam](0,2)(1.8,1.5)(0,0){grating}
\end{pspicture}
```

`gratinglinewidth`=⟨*num*⟩ or ⟨*dimen*⟩ default: `0.7\pslinewidth`

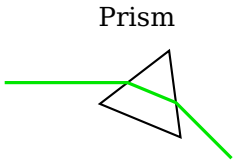
Die Linienbreite des Gitters. Diese könnte ebenfalls über den `linewidth` Parameter gesetzt werden. Mit diesem Parameter kann aber die Linienbreite aller Gitter global eingestellt werden.

`optgridwidth`
`optgridheight`
`optgriddepth`
`optgridcount`
`optgridtype`
`optgridlinewidth`

Diese Parameter wurden in die entsprechenden `grating*` Parameter umbenannt und sind seit Version 3.0 veraltet.

4.14. Prisma

`\optprism`[⟨*options*⟩](⟨*in*⟩)(⟨*center*⟩)(⟨*out*⟩){⟨*label*⟩}




```
\begin{pspicture}(3,2)
  \optprism[beam](0,1)(2,1)(3,0){Prism}
\end{pspicture}
```

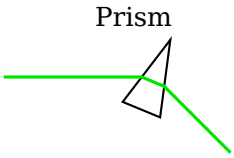
Das Prisma wird immer symmetrisch bezüglich der ⟨*in*⟩ und ⟨*out*⟩ Knoten ausgerichtet. Für asymmetrischen Strahlengang siehe Kap. 8.

`prismsize`=⟨*num*⟩ default: 1

Die Höhe des Prismas.

`prismangle`=⟨*num*⟩ default: 60

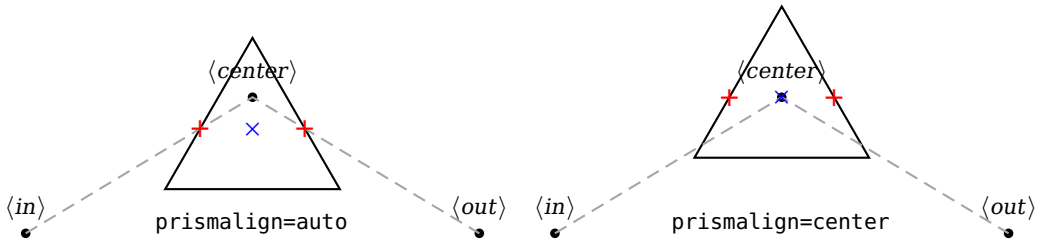
Der obere Winkel des Prismas.




```
\begin{pspicture}(3,2)
  \optprism[prismangle=30, beam](0,1)(2,1)(3,0){Prism}
\end{pspicture}
```

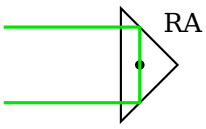
`prismalign`=`auto`, `center` default: `auto`

Wählt die vertikale Ausrichtung des Prisma bezüglich des „Reflektionsknoten“ ⟨*center*⟩. Ist der Wert `auto`, so wird das Prisma so verschoben, dass die Grenzflächenknoten auf der Verbindungslinie zwischen dem entsprechenden Referenzknoten und dem „Reflektionsknoten“ liegen. Für `center` fällt der „Reflektionsknoten“ mit der Komponentenmitte zusammen. Siehe die folgende Beispiele zur weiteren Erklärung.



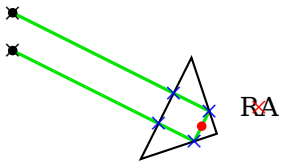
4.15. Umkehrprisma

`\rightangleprism`[⟨*options*⟩](⟨*in*⟩)(⟨*center*⟩)(⟨*out*⟩){⟨*label*⟩}




```
\begin{pspicture}(3,2)
  \psdot(1.8,1)
  \rightangleprism[beam](0,1.5)(1.8,1)(0,0.5){RA}
\end{pspicture}
```

Das Umkehrprisma wird so ausgerichtet, dass der einfallende und der reflektierte Strahl parallel sind und der ⟨*center*⟩ Knoten vertikal zentriert in dem Prisma liegt.



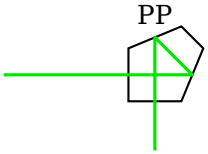

```
\begin{pspicture}(3.5,2)
  \pnode(0,2){A}\pnode(2.5,0.5){G}\pnode(0,1.5){B}
  \begin{optexp}
  \rightangleprism[beam, showifcnodes, showoptdots](A)(G)(B){RA}
  \end{optexp}
\end{pspicture}
```

raprismsize=⟨*num*⟩ default: 1.5

Die Länge der Eingangsfläche.

4.16. Pentaprisma

\pentaprism[⟨*options*⟩](⟨*in*⟩)(⟨*center*⟩)(⟨*out*⟩){⟨*label*⟩}




```
\begin{pspicture}(3,2)
  \pentaprism[beam](0,1)(2,1)(2,0){PP}
\end{pspicture}
```

pentaprismsize=⟨*num*⟩ default: 0.7

Die Länge der Eingangs- und Ausgangsfläche.

5. Faserkomponenten

In diesem Kapitel werden alle Faserkomponenten und deren Parameter beschrieben. Diese werden automatisch über Fasern mit ihren Referenzknoten verbunden und unterstützen kein Raytracing. Die meisten Komponenten benötigen nur zwei Referenzknoten und werden wie die Freistrahlzweipole behandelt. Manche spezielle Komponenten wie Koppler (5.10) und Zirkulatoren (5.9) werden anders ausgerichtet.

`usefiberstyle`=`true`, `false`

Bei manchen Komponenten (z.B. `\optfilter` oder `\optmzm`) kann es vorteilhaft sein interne Faserteile hervorzuheben. Mit diesem Parameter werden z.B. die durchlässigen Teile eines Filter angezeigt und mit dem `Fiber`-Stil gezeichnet. In der Dokumentation ist dieses Verhalten angeschaltet um die betroffenen Bereiche hervorzuheben.

5.1. Optische Faser

`\optfiber`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}


```
\begin{pspicture}(3,1.5)
  \optfiber[label=0.3](0,0.5)(3,0.5){SSMF}
\end{pspicture}
```

`fiberloops`=⟨*int*⟩ default: `3`

Anzahl der Faserschleifen.


```
\begin{pspicture}(3,1.5)
  \optfiber[fiberloops=2, label=0.3](0,0.5)(3,0.5){SSMF}
\end{pspicture}
```

`fiberloopradius`=⟨*num*⟩ default: `0.4`

Radius der Faserschleifen.


```
\begin{pspicture}(3,1.5)
  \optfiber[fiberloopradius=0.2, label=0.3](0,0.5)(3,0.5){SSMF}
\end{pspicture}
```

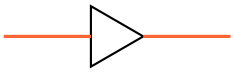
`fiberloopsep`=⟨*num*⟩ default: `0.3`

Abstand zwischen zwei aufeinanderfolgenden Faserschleifen.


```
\begin{pspicture}(3,1.5)
  \optfiber[fiberloopsep=0.6, label=0.3](0,0.5)(3,0.5){SSMF}
\end{pspicture}
```

5.2. Optischer Verstärker

`\optamp`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}



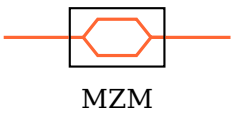

```
\begin{pspicture}(3,1.5)
  \optamp(0,1)(3,1){EDFA}
\end{pspicture}
```

`optampsize`=⟨*num*⟩ or ⟨*width*⟩ ⟨*height*⟩ default: `0.8`

Eine einzelne Zahl gibt die Seitenlänge des Verstärkers an, zwei Zahlen die Breite und Höhe. Beachten Sie, dass `optampsize=1` und `optampsize=1 1` nicht dasselbe Ergebnis liefern. Alternativ kann das Verhältnis zwischen Höhe und Breite mit `xunit` und `yunit` geändert werden.

5.3. Mach-Zehnder-Modulator

`\optmzm`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}



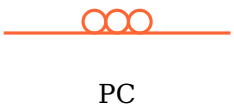

```
\begin{pspicture}(3,1.5)
  \optmzm(0,1)(3,1){MZM}
\end{pspicture}
```

`optmzmsize`=⟨*num*⟩ or ⟨*width*⟩ ⟨*height*⟩ default: `0.8`

Eine einzelne Zahl gibt die Höhe des Modulators an, die Breite ist `1.6` mal die Höhe. Zwei Zahlen geben Breite und Höhe des Modulators direkt an.

5.4. Polarisationssteller

`\polcontrol`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}



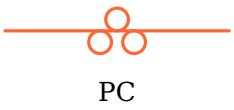

```
\begin{pspicture}(3,1.5)
  \polcontrol(0,1)(3,1){PC}
\end{pspicture}
```

`polcontrolsize`=⟨*num*⟩ default: `0.15`

Der Radius der Polarisationssteller-Kreise.

`polcontroltype`=`linear`, `triangle` default: `linear`

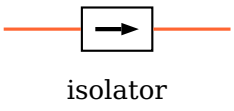
Der Typ des Polarisationsstellers.




```
\begin{pspicture}(3,1.5)
  \polcontrol[polcontroltype=triangle](0,1)(3,1){PC}
\end{pspicture}
```

5.5. Isolator

`\optisolator`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}



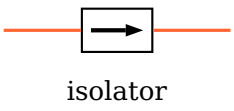

```
\begin{pspicture}(3,1.5)
  \optisolator(0,1)(3,1){isolator}
\end{pspicture}
```

`isolatorsize`=⟨*num*⟩ or ⟨*width*⟩ ⟨*height*⟩ default: `0.6`

Eine einzelne Zahl gibt die Höhe des Isolators an, die Breite ist `1.6` mal die Höhe. Zwei Zahlen geben Breite und Höhe des Isolators direkt an.

`IsolatorArrow` ⟨*psstyle*⟩ default: `linewidth=2\pslinewidth, arrowinset=0`

Der Stil für den Isolatorpfeil. Das kann insbesondere nützlich sein um die Länge des Pfeils zu verändern.




```
\begin{pspicture}(3,1.5)
  \addtopsstyle{IsolatorArrow}{xunit=1.2}
  \optisolator(0,1)(3,1){isolator}
\end{pspicture}
```

5.6. Optischer Schalter

`\optswitch`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}

opened switch

```
\begin{pspicture}(3,1.5)
  \optswitch(0,1)(3,1){opened switch}
\end{pspicture}
```

`switchsize`=⟨*num*⟩ default: `0.8`

Die Seitenlänge des Schalters.

`switchstyle`=`opened`, `closed` default: `opened`

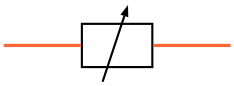
Der Zustand des Schalters kann geschlossen (`closed`) oder geöffnet (`opened`) sein.

closed switch

```
\begin{pspicture}(3,1.5)
  \optswitch[switchstyle=closed](0,1)(3,1){closed switch}
\end{pspicture}
```

5.7. Faserverzögerungstrecke

`\fiberdelayline`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}



delay line

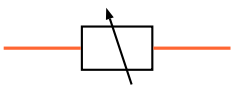
```
\begin{pspicture}(3,1.5)
  \fiberdelayline(0,1)(3,1){delay line}
\end{pspicture}
```

`fdlsize`=⟨*num*⟩ or ⟨*width*⟩ ⟨*height*⟩ default: `0.6`

Eine einzelne Zahl gibt die Höhe der Komponente an, die Breite ist `1.6` mal die Höhe. Zwei Zahlen geben Breite und Höhe der Komponente direkt an.

`FdlArrow` ⟨*psstyle*⟩ default: `arrowinset=0, arrows=->`

Der Stil für den Pfeil. Das kann insbesonders nützlich sein um die Länge des Pfeils zu verändern, die bei geänderter Größe nicht mehr passend sein kann, oder um den Pfeil zu spiegeln.

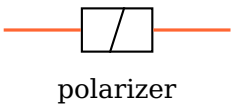


delay line

```
\begin{pspicture}(3,1.5)
  \addtopsstyle{FdlArrow}{xunit=-1}
  \fiberdelayline(0,1)(3,1){delay line}
\end{pspicture}
```

5.8. Polarisator

\optfiberpolarizer[⟨options⟩](⟨in⟩)(⟨out⟩){⟨label⟩}



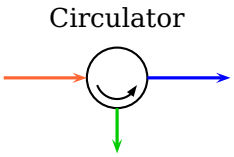

```
\begin{pspicture}(3,1.5)
  \optfiberpolarizer(0,1)(3,1){polarizer}
\end{pspicture}
```

fiberpolsize=⟨num⟩ or ⟨width⟩ ⟨height⟩ default: 0.6

Eine einzelne Zahl gibt die Höhe der Komponente an, die Breite ist 1.6 mal die Höhe. Zwei Zahlen geben Breite und Höhe der Komponente direkt an.

5.9. Optischer Zirkulator

\optcirculator(⟨left⟩)(⟨right⟩)(⟨bottom⟩){⟨label⟩}




```
\begin{pspicture}(3,2)
  \addtopsstyle{Fiber}{arrows=->}
  \addtopsstyle{FiberOut1}{linecolor=blue}
  \addtopsstyle{FiberOut2}{linecolor=green!80!black}
  \optcirculator(0,1)(3,1)(1.5,0){Circulator}
\end{pspicture}
```

optcircsize=⟨num⟩ default: 0.8

Der Durchmesser des Zirkulators.

optcircangleA=⟨num⟩ default: -160

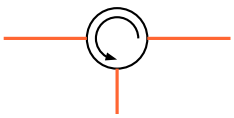
Der Startwinkel des internen Pfeils.

optcircangleB=⟨num⟩ default: -20

Der Endwinkel des internen Pfeils.

optcircangle=⟨num⟩ ⟨num⟩

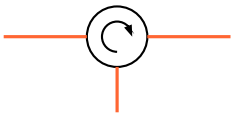
Kurzschreibweise um beide Winkel gleichzeitig zu setzen.



```
\begin{pspicture}(3,1.5)
  \optcirculator[optcircangle=0 -90](0,1)(3,1)(1.5,0)
\end{pspicture}
```

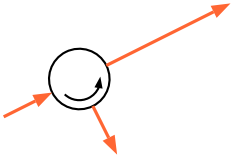
`OptCircArrow` ⟨*psstyle*⟩ default: `unit=0.7, arrows=->, arrowinset=0`

Der Stil für den internen Pfeil. Damit wird sowohl die Richtung des Pfeils als auch die Größe des Bogens bestimmt.



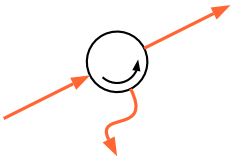
```
\begin{pspicture}(3,1.5)
  \newpsstyle{OptCircArrow}{unit=0.5, arrows=<-, arrowinset=0}
  \optcirculator[optcircangle=0 -90](0,1)(3,1)(1.5,0)
\end{pspicture}
```

Der Zirkulator wird so positioniert, dass Eingangs- und Ausgangsverbindung senkrecht zueinander stehen, wie das folgende Beispiel verdeutlicht:



```
\begin{pspicture}(3,2)
  \addtopsstyle{Fiber}{arrowscale=1.3, arrows=->, arrowinset=0}
  \optcirculator(0,0.5)(3,2)(1.5,0)
\end{pspicture}
```

Die Positionierungsparameter (Kap. 3.2) beziehen sich wie üblich auf die Referenzknoten ⟨*left*⟩ und ⟨*right*⟩, und nicht auf die automatisch bestimmte Position. Das bedeutet auch, dass die automatische Position i.a. nicht mit `position=0.5` übereinstimmt.

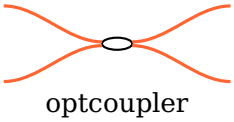


```
\begin{pspicture}(3,2)
  \addtopsstyle{Fiber}{arrowscale=1.3, arrows=->, arrowinset=0,
      ncurv=1.5}
  \optcirculator[position=0.5](0,0.5)(3,2)(1.5,0)
\end{pspicture}
```

5.10. Faserkoppler

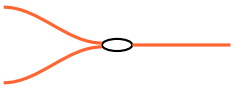
Es stehen drei Faserkoppler zur Verfügung, die alle die gleichen Formen und Parameter verwenden.

`\optcoupler`(⟨*tl*⟩)(⟨*bl*⟩)(⟨*tr*⟩)(⟨*br*⟩){⟨*label*⟩}



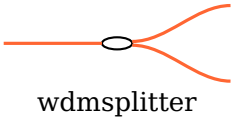

```
\begin{pspicture}(0,-0.4)(3,1)
  \optcoupler(0,1)(0,0)(3,1)(3,0){optcoupler}
\end{pspicture}
```

`\wdmcoupler`(⟨*tl*⟩)(⟨*bl*⟩)(⟨*r*⟩){⟨*label*⟩}




```
\begin{pspicture}(0,-0.4)(3,1)
  \wdmcoupler(0,1)(0,0)(3,0.5){wdmcoupler}
\end{pspicture}
```

`\wdmsplitter`(⟨*l*⟩)(⟨*tr*⟩)(⟨*br*⟩){⟨*label*⟩}



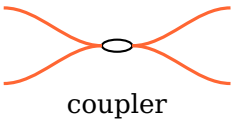

```
\begin{pspicture}(0,-0.4)(3,1)
  \wdmsplitter(0,0.5)(3,1)(3,0){wdmsplitter}
\end{pspicture}
```

`couplersize`=⟨*num*⟩ or ⟨*width*⟩ ⟨*height*⟩ default: `0.2`

Wird eine Zahl angegeben, so ist die Breite des Kopplers doppelt so groß, die Höhe entspricht `0.8` mal dem Wert. Mit zwei Zahlen werden Breite und Höhe direkt angegeben.

`couplersep`=⟨*num*⟩ default: `0.1`

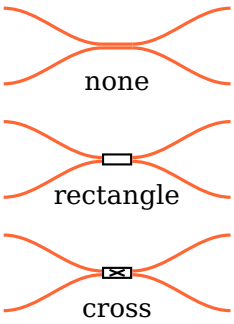
Der vertikale Abstand zwischen zwei Faserports.




```
\begin{pspicture}(0,-0.5)(3,1)
  \optcoupler[couplersep=0](0,1)(0,0)(3,1)(3,0){coupler}
\end{pspicture}
```

`couplertype`=`none`, `ellipse`, `rectangle`, `cross` default: `ellipse`

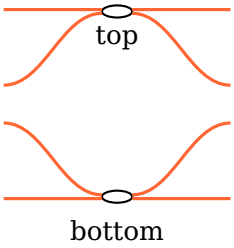
Wählt zwischen unterschiedlichen Kopplertypen.




```
\begin{pspicture}(3,4.5)
  \psset{labeloffset=0.5}
  \optcoupler[couplertype=none]%
      (0,4.5)(0,3.5)(3,4.5)(3,3.5){none}
  \optcoupler[couplertype=rectangle]%
      (0,3)(0,2)(3,3)(3,2){rectangle}
  \optcoupler[couplertype=cross]%
      (0,1.5)(0,0.5)(3,1.5)(3,0.5){cross}
\end{pspicture}
```

`coupleralign`=`t`, `top`, `b`, `bottom`, `c`, `center` default: `center`

Die Ausrichtung des Kopplers bezüglich der Referenzknoten.

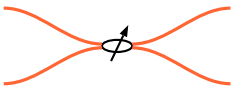



```
\begin{pspicture}(3,3)
  \psset{labeloffset=0.4}
  \optcoupler[coupleralign=top]%
             (0,3)(0,2)(3,3)(3,2){top}
  \optcoupler[coupleralign=bottom]%
             (0,1.5)(0,0.5)(3,1.5)(3,0.5){bottom}
\end{pspicture}
```

`align`=`top`, `bottom`, `center` default: `center`

Dieser Parameter wurde in Version 3.0 in `coupleralign` umbenannt und ist seitdem veraltet.

`variable`=`true`, `false` default: `true`



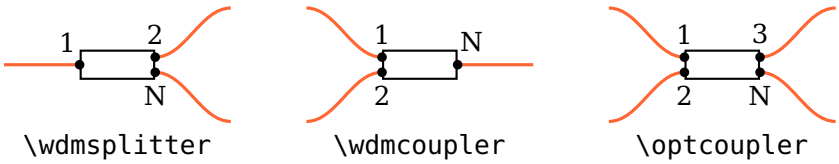
```
\begin{pspicture}(3,1)
  \optcoupler[variable](0,1)(0,0)(3,1)(3,0)
\end{pspicture}
```

`VariableCoupler`
⟨*psstyle*⟩ default: `arrowinset=0, arrows=->`

Der Stil des Pfeiles des verstellbaren Kopplers.

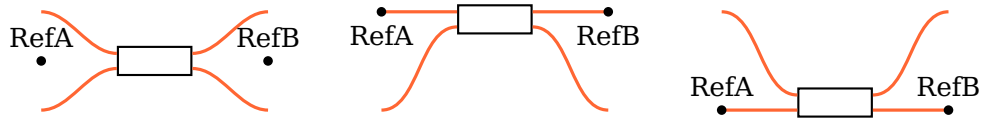
5.10.1. Eingangs- und Ausgangsknoten

Die Definition der Eingangs- und Ausgangsknoten aus Kap. 7.6 ist für Koppler nicht anwendbar. Hier werden die Knoten einfach von `1` (links oben) bis `N` (rechts unten) durchnummeriert.



5.10.2. Referenzknoten

Die Definition der Referenzknoten aus Kap. 7.2 ist für Koppler nicht anwendbar und hängt von `coupleralign` ab.



`coupleralign=none` `coupleralign=top` `coupleralign=bottom`

6. Hybridkomponenten

Dieses Kapitel beschreibt die Komponenten, die sowohl für Freistrahl- als auch Faseroptik zu verwenden sind, sich von diesen aber etwas abgrenzen.

Der optische Filter (\optfilter) wird mit den Voreinstellungen aber als Faserkomponente gehandhabt, kann jedoch im Gegensatz zu den reinen Faserkomponenten (Kap. 5) auch als Freistrahlkomponente verwendet werden. Der Faserkollimator (\fibercollimator) hat nur eine Faser, die andere Verbindung ist Freistrahl.

6.1. Optischer Filter

\optfilter[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}

bandpass

```
\begin{pspicture}(3,1.5)
  \optfilter(0,1)(3,1){bandpass}
\end{pspicture}
```

filtersize=⟨*num*⟩ default: 0.8

Die Größe des Filters.

filtertype=bandpass, bandstop, lowpass, highpass default: bandpass

Wähle zwischen unterschiedlichen Filtertypen.

bandstop

```
\begin{pspicture}(3,1.5)
  \optfilter[filtertype=bandstop](0,1)(3,1){bandstop}
\end{pspicture}
```

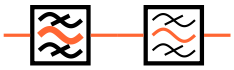
lowpass

```
\begin{pspicture}(3,1.5)
  \optfilter[filtertype=lowpass](0,1)(3,1){lowpass}
\end{pspicture}
```


```
\begin{pspicture}(3,1.5)
  \optfilter[filtertype=highpass](0,1)(3,1){highpass}
\end{pspicture}
```

FilterStyle ⟨*psstyle*⟩

Beinflusst das Aussehen der internen Filterlinien.



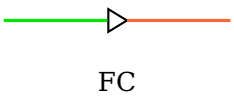
```
\begin{pspicture}(3,1.5)
  \newpsstyle{OptComp}{linewidth=2\pslinewidth}
  \optfilter(0,1)(1.5,1)
  \newpsstyle{FilterStyle}{linewidth=0.5\pslinewidth}
  \optfilter(1.5,1)(3,1)
\end{pspicture}
```

Verwendung als Freistrahlkomponenten, allowbeaminside ist auf false voreingestellt:

```
\begin{pspicture}(3,1.5)
  \optfilter[fiber=none](0,1)(3,1)
  \addtopsstyle{Beam}{beamwidth=0.3,
     fillstyle=solid, fillcolor=green, opacity=0.2}
  \drawwidebeam{(0,1)}{}{(3,1)}
\end{pspicture}
```

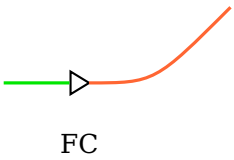
6.2. Faserkollimator

\fibercollimator(⟨*in*⟩)(⟨*A*⟩)(⟨*B*⟩)(⟨*out*⟩){⟨*label*⟩}



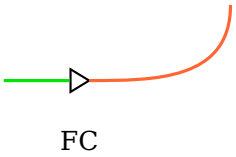

```
\begin{pspicture}(3,1.5)
  \fibercollimator[beam](0,1)(3,1){FC}
\end{pspicture}
```

Der Faserkollimator kann mit zwei, drei oder vier Punkten verwendet werden. Bei zwei Punkten wird der Kollimator wie jeder andere Zweipol zwischen ⟨*in*⟩ und ⟨*out*⟩ Knoten platziert. Bei drei Punkten wird eine \psbezier Kurve gezeichnet, wobei der mittlere Punkt doppelt verwendet wird. Die Positionierungsparameter (siehe Kap. 3.2) können verwendet werden um den Kollimator zwischen dem ⟨*in*⟩ und ⟨*A*⟩ Knoten auszurichten.




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

Bei vier Knoten wird eine `\psbezier` Kurve mit allen vier Knoten gezeichnet. Die Positionierungsparameter (siehe Kap. 3.2) können verwendet werden um den Kollimator zwischen den ersten beiden Knoten ($\langle in \rangle$ und $\langle A \rangle$) zu verschieben.




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

`fibercolsize`=$\langle num \rangle$ or $\langle width \rangle$ $\langle height \rangle$ default: `0.3`

Eine einzelne Zahl gibt die Seitenlänge des Kollimators an, zwei Zahlen die Breite und Höhe. Beachten Sie, dass `fibercolsize=1` und `fibercolsize=1 1` nicht dasselbe Ergebnis liefern. Alternativ kann das Verhältnis zwischen Höhe und Breite mit `xunit` und `yunit` geändert werden.

7. Spezielle Knoten

Jedes pst-optexp-Objekt stellt mehrere spezielle Knoten zur Verfügung, die mit dessen Geometrie und Positionierung zusammenhängen. Diese Knoten stehen für weitere Verwendung zur Verfügung.

Sie sollten immer die dafür vorgesehenen Makros verwenden um auf die Knotennamen zuzugreifen.

\oenode{⟨*node*⟩}{⟨*comp*⟩}

Das ist das grundlegende Makro mit dem auf die Knotennamen einer Komponente zugegriffen werden kann. Das erste Argument ⟨*node*⟩ ist der Bezeichner des angeforderten Knotens. Das zweite Argument ⟨*comp*⟩ ist der Name der Komponente (gemäß Kap. 7.1). Ist dieses leer so wird das zuletzt definierte Objekt verwendet.

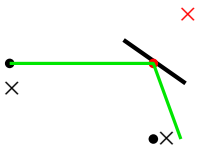
Für viele der Knoten wird ein eigenes Makro bereitgestellt, das Sie dann auch verwenden sollten, da sich die Namenskonventionen ändern könnten. Die Makros stellen sicher, dass Sie immer die passenden Knotennamen erhalten. Daher sind die verfügbaren Bezeichner auch nicht aufgelistet.

namingscheme=old, new default: new

Diese Option ist nur aus Kompatibilitätsgründen vorhanden. In Version 2.1 mussten spezielle Komponentenknoten über ihren expliziten Namen angesprochen werden. Sie sollten diese Option nur dann verwenden und auf `old` setzen, wenn Sie in älteren Dokumenten direkt auf die Knoten zugegriffen haben. Seit Version 3.0 werden Makros für den Zugriff auf die Komponentenknoten bereitgestellt, so dass das eigentliche Namensschema unerheblich ist.

showoptdots=true, false default: false

Markiert einige der speziellen Komponentenknoten: die schwarzen Punkte sind die normalen und die schwarzen Kreuze die transformierten Referenzknoten (Kap. 7.2), der rote Punkt ist der Mittelpunktknoten (Kap. 7.3) und das rote Kreuz der Beschriftungsknoten (Kap. 7.4).



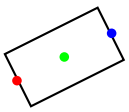
```
\begin{pspicture}(3,2)
\mirror[showoptdots, angle=10, beam](0,1)(1.9,1)(1.9,0)
\end{pspicture}
```

7.1. Komponenten-Bezeichner

Alle Komponenten einer Aufbauskizze werden in Reihenfolge ihrer Definition im Code aufsteigend nummeriert, angefangen bei 1. Die Komponenten und deren Spezialknoten können immer über diese Nummer (ID) referenziert werden.

`compname`=⟨*string*⟩

Weist einer Komponente einen Bezeichner zu. Die Komponente kann nun sowohl über diesen Bezeichner als auch über die ID referenziert werden. Dieser Parameter kann nur im optionalen Argument einer `pst-optexp` Komponente verwendet werden. Der Bezeichner sollte innerhalb einer `pspicture`-Umgebung eindeutig sein.



```
\begin{pspicture}(2,2)
  \optbox[compname=MyBox](0,1)(2,2)
  \psdot[linecolor=red](\oenodeIn{MyBox}) % Verwende den Namen
  \psdot[linecolor=blue](\oenodeOut{1}) % Verwende die ID
  \psdot[linecolor=green](\oenodeCenter{})% Nimm die letzte Komponente
\end{pspicture}
```

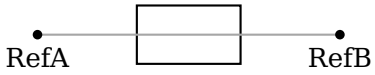
7.2. Referenzknoten

`\oenodeRefA`{⟨*comp*⟩}

`\oenodeRefB`{⟨*comp*⟩}

Die Eingangs- und Ausgangs-Referenzknoten.

Das sind die ursprünglichen Knoten, die für die Positionierung der Komponente benutzt wurden, `\oenodeRefA` ist der erste und `\oenodeRefB` der letzte Knoten. Diese Zuordnung ist nicht gültig für Faserkoppler (5.10.2) und Zirkulatoren (5.9).



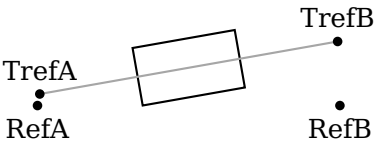

```
\begin{pspicture}(5,1)
  \pnode(0.5,0.5){A}\pnode(4.5,0.5){B}
  \optbox(A)(B)
  \psline[style=Refline](\oenodeRefA{})(\oenodeRefB{})
  \psdot(\oenodeRefA{})\uput[-90](\oenodeRefA{}){RefA}
  \psdot(\oenodeRefB{})\uput[-90](\oenodeRefB{}){RefB}
\end{pspicture}
```

\oenodeTrefA{⟨*comp*⟩}

\oenodeTrefB{⟨*comp*⟩}

Die transformierten Eingangs- und Ausgangs-Referenzknoten.

Das sind die Referenzknoten nachdem sie zusammen mit der Komponente gemäß der compshift und angle Parameter transformiert wurden.

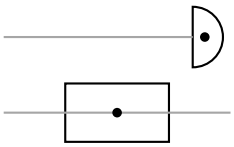



```
\begin{pspicture}(5,1.8)
  \pnode(0.5,0.5){A}\pnode(4.5,0.5){B}
  \optbox[compshift=0.5, angle=10](A)(B)
  \psdot(\oenodeRefA{})\uput[-90](\oenodeRefA{}){RefA}
  \psdot(\oenodeRefB{})\uput[-90](\oenodeRefB{}){RefB}
  \psdot(\oenodeTrefA{})\uput[90](\oenodeTrefA{}){TrefA}
  \psdot(\oenodeTrefB{})\uput[90](\oenodeTrefB{}){TrefB}
\end{pspicture}
```

7.3. Mittelpunktknoten

\oenodeCenter{⟨*comp*⟩}

Dieser Knoten liegt, bis auf wenige Ausnahmen (z.B. \optdetector), im Mittelpunkt der Komponente.

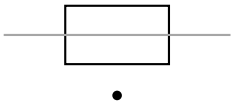


```
\begin{pspicture}(3,2)
  \optbox(0,0.5)(3,0.5)
  \optdetector(0,1.5)(2.5,1.5)
\end{pspicture}
```

7.4. Beschriftungsknoten

\oenodeLabel{⟨*comp*⟩}

Auf diesen Knoten wird die Beschriftung platziert. Der Knoten ist auch verfügbar wenn keine Beschriftung angegeben wurde. Für labeloffset=0 ist dieser Knoten identisch mit dem Mittelpunktsknoten (Kap. 7.3).



```
\begin{pspicture}(3,1.5)
  \optbox(0,1)(3,1)
\end{pspicture}
```

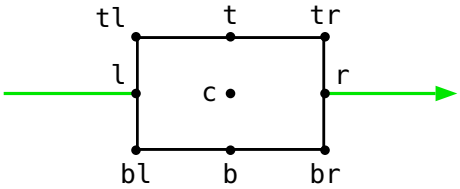
7.5. Externe Knoten

\oenodeExt{⟨*comp*⟩}

Ein externer Knoten kann an unterschiedliche Positionen entlang des Komponentenrandes gesetzt werden. Dieser wird nur für den externen Zugriff definiert, und beeinflusst in keiner Weise die Komponente.

extnode=⟨*refpoint*⟩

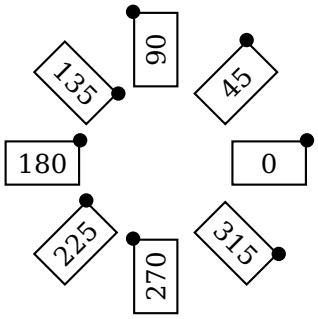
Bestimmt die Position des externen Knotens. Analog zum Referenzpunkt von \rput, kann jede Kombination von c (mittig), t (oben), b (unten), l (links) und r (rechts) sein. Nicht jede Komponente unterstützt alle möglichen Kombinationen, die möglichen Positionen jeder Komponente sind in Kap. 11.2.1 zusammengefasst.



extnodealign=rel, relative, abs, absolute default: abs

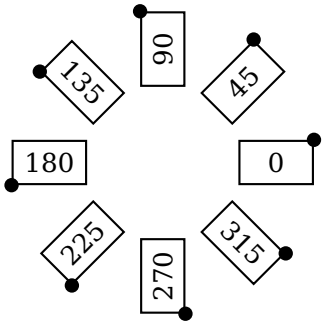
Die Bezeichnung von „oben“ (t) und den anderen Positionierungsparametern von extnode können absolut oder relativ zur Komponente betrachtet werden.

In dem folgenden Beispiel ist der externe Knoten immer „oben rechts“ (tr) platziert, unabhängig von der Reihenfolge der Referenzknoten (extnodealign=abs). Das Verhalten ist identisch zu labelref=relative für die Ausrichtung der Beschriftung.




```
\begin{pspicture}(-2,-2)(2,2)
  \psset{endbox, optboxsize=1 0.6, dotscale=1.5}
  \psset{extnodealign=abs, extnode=tr}
  \multido{\i=0+45}{8}{%
    \optbox[innerlabel](0,0)(1;\i){\i}
    \psdot(\oenodeExt{})
  }
\end{pspicture}
```

In dem darauffolgenden Beispiel bezieht sich die Positionierung relativ zur Verbindung zwischen Eingangs- und Ausgangsreferenzknoten (`extnodealign=rel`).




```
\begin{pspicture}(-2,-2)(2,2)
  \psset{endbox, optboxsize=1 0.6, dotscale=1.5}
  \psset{extnodealign=rel, extnode=tr}
  \multido{\i=0+45}{8}{%
    \optbox[innerlabel](0,0)(1;\i){\i}
    \psdot(\oenodeExt{})
  }%
\end{pspicture}
```

7.6. Grenzflächenknoten

`\oenodeIfc{`⟨*num*⟩`}{`⟨*comp*⟩`}`

Der Grenzflächenknoten ⟨*num*⟩, wobei ⟨*num*⟩ eine positive Ganzzahl oder `N` ist. Der letzte Knoten ist immer `N`.

`\oenodeIn{`⟨*comp*⟩`}`

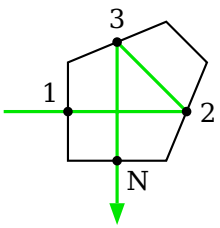
Der Knoten ⟨*1*⟩ ist der Eingangsknoten und sollte immer über `\oenodeIn` angesprochen werden.

`\oenodeOut{`⟨*comp*⟩`}`

Der Knoten `N` ist der Ausgangsknoten und sollte immer über `\oenodeOut` angesprochen werden.

„Eingang“ und „Ausgang“ können nur anhand einer relativen Orientierung definiert werden. Per Definition ist der Eingangsknoten derjenige zu dem ein Lichtstrahl ausgehend vom Referenzknoten `\oenodeRefA` verläuft. Analog dazu ist die Definition ds Ausgangsknotens. Diese Bezeichnung kann nur für Faserkoppler (siehe Kap. 5.10.1) und Strahlteiler (4.12) nicht angewendet werden.

\oenodeIn ist äquivalent zu \oenodeIfc{1} und \oenodeOut ist äquivalent zu \oenodeIfc{N}. Die Eingangs- und Ausgangsknoten sollten über die explizit bereitgestellten Makros angesprochen werden, \oenodeIfc sollte nur für die weiteren Grenzflächenknoten, sofern vorhanden, verwendet werden.




```
\begin{pspicture}(2.8,3)
  \pentaprism[pentaprismsize=1.3](0,1.5)(1.5,1.5)(1.5,0)
  \drawbeam[arrows=->, arrowinset=0,
      arrowscale=1.5]{(0,1.5)}{}{(1.5,0)}
  \psdot(\oenodeIn{})\uput[135](\oenodeIn{}){1}
  \psdot(\oenodeIfc{2}{})\uput[0](\oenodeIfc{2}{}){2}
  \psdot(\oenodeIfc{3}{})\uput[90](\oenodeIfc{3}{}){3}
  \psdot(\oenodeOut{})\uput[-45](\oenodeOut{}){N}
\end{pspicture}
```

In Kap. 11.2.2 finden Sie eine vollständige Liste aller Komponenten mit ihren Grenzflächenknoten.

7.7. Referenzknoten für die Rotation

\oenodeRotref{⟨*comp*⟩}

Der Referenzknoten um den eine Komponente mit angle gedreht wird. Die Position des Knotens wird mit dem rotateref Parameter definiert und kann dieselben Werte annehmen wie extnode. Eine vollständige Liste der möglichen Positionen finden Sie in Kap. 11.2.1.

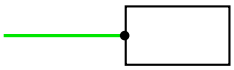

```
\begin{pspicture}(0,0.1)(12,1.8)
  \optbox[angle=20](0.5,1)(4.5,1)
  \psdot(\oenodeRotref{})\uput[-90](\oenodeRotref{}){Rotref}
  \optbox[angle=20, rotateref=bl](7.5,1)(11.5,1)
  \psdot(\oenodeRotref{})\uput[-90](\oenodeRotref{}){Rotref}
\end{pspicture}
```

7.8. Strahlknoten

Die Endknoten eines \drawbeam oder \drawwidebeam Kommandos können wieder verwendet werden, wenn savebeampoints gesetzt ist.

\oenodeBeam{⟨num⟩}

Damit wird der Endknoten eines vorangegangenen \drawbeam Kommandos angesprochen. Wird ⟨num⟩ leer gelassen so wird 1 genommen, was in der Regel dem letzten Strahlengang entspricht. Auf welchen Strahl genau Bezug genommen wird, hängt i.a. von den gewählten Einstellungen von savebeampoints und entsprechender Wahl von ⟨num⟩ ab.

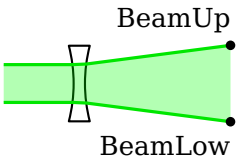


```
\begin{pspicture}(0,0)(3,2)
  \pnode(0,1){A}\pnode(1.6,1){B}
  \optbox[position=end](A)(B)
  \drawbeam[beaminsidelast=false]{(A)}{1}
  \psdot(\oenodeBeam{})
\end{pspicture}
```

\oenodeBeamUp{⟨num⟩}

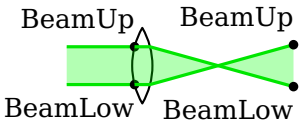
\oenodeBeamLow{⟨num⟩}

Damit wird der obere („upper“) oder untere („lower“) Endknoten eines vorangegangenen \drawwidebeam Kommandos angesprochen. Wird ⟨num⟩ leer gelassen so wird 1 genommen, womit man sich in der Regel auf den letzten Strahlengang bezieht. Auf welchen Strahl genau Bezug genommen wird, hängt i.a. von den gewählten Einstellungen von savebeampoints und entsprechender Wahl von ⟨num⟩ ab.




```
\begin{pspicture}(0,0)(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \lens[abspos=1, lensradius=-2](A)(B)
  \addtopsstyle{Beam}{%
    fillstyle=solid, fillcolor=green, opacity=0.3}
  \drawwidebeam[beamwidth=0.5]{(A)}{}{(B)}
\end{pspicture}
```

Beachten Sie, dass „oben“ („upper“) und „unten“ („lower“) bezüglich der Ausbreitungsrichtung des Strahls definiert ist. Ein Randstrahl der als „upper“ startet can zu „lower“ werden, wenn er den anderen Randstrahl kreuzt. Das wird in dem folgenden Beispiel erläutert:

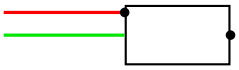



```
\begin{pspicture}(0,0)(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \lens[abspos=1, n=2](A)(B)
  \addtopsstyle{Beam}{%
    fillstyle=solid, fillcolor=green, opacity=0.3}
  \drawwidebeam[beamwidth=0.5]{(A)}{}
  \drawwidebeam[loadbeam, beaminsidefirst]{}{(B)}
\end{pspicture}
```

Die Strahlknoten können nur innerhalb der pspicture-Umgebung verwendet werden, in der sie definiert wurden. Um auf bestimmte Strahlknoten von außerhalb zugreifen zu können müssen diese vorher explizit umdefiniert werden:

```
\pnode(\oenodeBeam{}){MyBeamNode}
```

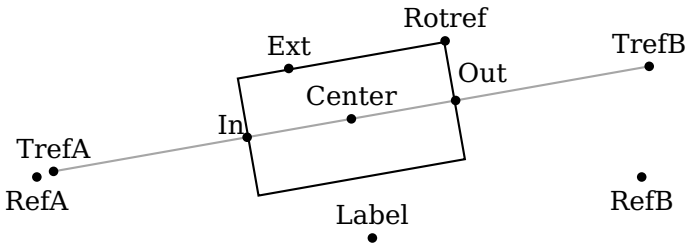
Desweiteren werden die Endknoten durch stopinside beeinflusst. Siehe Kap. 8.5 für weitere Details.



```
\begin{pspicture}(0,0)(3,2)
  \pnode(0,1){A}\pnode(1.6,1){B}
  \optbox[position=end](A)(B)
  \psset{beaminsidelast=false}
  \drawbeam[linecolor=red, beampos=0.3]{(A)}{1}
  \psdot(\oenodeBeam{})
  \drawbeam[stopinside]{(A)}{1}
  \psdot(\oenodeBeam{})
\end{pspicture}
```

7.9. Knotenübersicht

Dies ist eine Übersicht über alle verfügbaren Komponentenknoten.




```
\begin{pspicture}(0.3,0.1)(4.8,1.6)
  \pnode(0.5,0.5){A}\pnode(4.5,0.5){B}
  \optbox[compshift=0.5, angle=10, rotateref=tr, extnode={-0.5,1}](A)(B)
\end{pspicture}
```

8. Verbinden von Komponenten

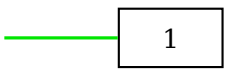
Das `pst-optexp`-Paket stellt unterschiedlichen Methoden bereit, Komponenten vollautomatisch oder manuell, mit Fasern oder mit Lichtstrahlen zu verbinden.

- In Kap. 8.1 wird beschrieben wie Komponenten und Knoten für Verbindungen angesprochen werden.
- Strahlverbindungen im Allgemeinen und das Verhalten von Einzelstrahlen wir in Kap. 8.2 beschrieben.
- Kap. 8.3 erweitert diese Beschreibung von Strahlverbindungen auf ausgedehnte Strahlen.
- In Kap. 8.4 wird gezeigt, wie fehlerhafte Strahlverbindungen (verfehlen von Grenzflächen) gehandhabt wird).
- Kap. 8.5 erläutert, wie stückweise definierte Strahlengänge einfach realisiert werden können.
- Manuelle und automatische Faserverbindungen werden in Kap. 8.6 behandelt.
- Kap. 8.7 erklärt das Konzept der „front“ und „back“-Ebene.

8.1. Zugriff auf Komponenten

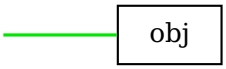
Alle Makros für Verbindungen akzeptieren entweder eine ID, ein `compname` Bezeichner (siehe Kap. 7.1) oder einen PSTricks-Knoten als Argument, die als $\langle obj_1 \rangle$, $\langle obj_2 \rangle$, ... bezeichnet werden. Um zwischen Knoten und Komponenten unterscheiden zu können, müssen Knoten in runde Klammern innerhalb der geschweiften Klammern eingeschlossen werden.

1. Auf alle Komponenten kann immer unter Verwendung der ID zugegriffen werden.



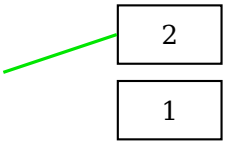

```
\begin{pspicture}(3,1.5)
  \pnode(0,1){A}\pnode(1.5,1){B}
  \optbox[position=end, labeloffset=0](A)(B){1}
  \drawbeam{(A)}{1}
\end{pspicture}
```

2. Wenn eine Komponente mit `compname` einen zusätzlichen Namen erhalten hat, so kann dieser anstelle der ID verwendet werden.




```
\begin{pspicture}(3,2)
  \psset{position=end, labeloffset=0}
  \optbox[compname=obj](0,1)(1.5,1){obj}
  \drawbeam{(0,1)}{obj}
\end{pspicture}
```

3. Wird das Argument leer gelassen, dann greift man auf die zuletzt definierte Komponente zu.




```
\begin{pspicture}(3,2)
  \psset{position=end, labeloffset=0}
  \optbox(0,0.5)(1.5,0.5){1}
  \optbox(0,1.5)(1.5,1.5){2}
  \drawbeam{(0,1)}{}
\end{pspicture}
```

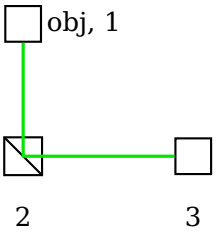
4. Bei `\drawbeam` und `\drawwidebeam` können auch Zahlintervalle verwendet werden. Gültige Intervallangaben sind

   x-y Von x bis y, ist x größer als y, so wird heruntergezählt.

   x- Von x bis zur letzten Komponente.

   -y Von der ersten Komponente bis y.

   - Verbinde alle Komponenten.




```
\begin{pspicture}(3,3)
  \psset{position=end, optboxsize=0.5 0.5, bssize=0.5}
  \optbox[compname=obj](0.5,0.5)(0.5,2.5){obj, 1}
  \beamsplitter(0.5,2.5)(0.5,1)(2.5,1){2}
  \optbox(0.5,1)(2.5,1){3}
  \drawbeam{1-3}
\end{pspicture}
```

8.2. Strahlen zeichnen

Es gibt mehrere Möglichkeiten Komponenten und Knoten mit Strahlen zu verbinden: einfache Linien oder ausgedehnte Strahlen. Die meisten Parameter haben für beide Typen dieselbe Bedeutung, daher beschränken sich die entsprechenden Beispiele auf die einfachen Strahlen. Die Parameter die nur ausgedehnte Strahlen betreffen, werden in Kap. 8.3 beschrieben.

`\drawbeam`[⟨*options*⟩]{⟨obj_1⟩}{⟨obj_2⟩}...

`\drawwidebeam`[⟨*options*⟩]{⟨obj_1⟩}{⟨obj_2⟩}...

Diese beiden Makros sind für die Strahlverbindungen zuständig, sie akzeptieren eine variable Anzahl an Argumenten (mindestens zwei), die entweder eine ID, ein `compname`, oder ein PSTricks-Knoten sein können. Um zwischen Knoten und Komponenten unterscheiden zu können, müssen Knoten in runde Klammern innerhalb der geschweiften Klammern eingeschlossen werden.

8.2.1. Raytracing

Der Strahlengang durch die Komponenten kann mit zwei unterschiedlichen Methoden ermittelt werden: über den Brechungsindex und das Snelliussche Brechungsgesetz, oder indem die Komponenten unabhängig von ihren optischen Eigenschaften einfach verbunden werden.

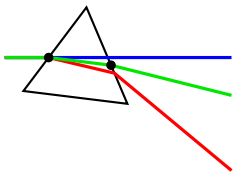
Behalten Sie immer im Hinterkopf, dass dieses Paket für *Skizzen* optischer Aufbauten gedacht ist und daher keine vollständige Umgebung zur Strahlverfolgung bereitstellt. Abweichungen vom physikalisch richtigen Weg können in vielen Fällen erwünscht sein: Winkel und Divergenzen werden häufig extremer dargestellt um bestimmte Effekte hervorzuheben. Daher sollten Sie den Brechungsindex (siehe Kap. 8.2.2) lediglich as Mittel zur Optimierung des gewünschten Lichtweges sehen und nicht zu sehr an den physikalisch richtigen Werten festhalten.

`raytrace`=`true`, `false` default: `true`

Wählt die Methode der Strahlverfolgung, mit `true` wird der Brechungsindex („raytracing“) verwendet, `false` verbindet die Komponente lediglich („connect“ Variante).

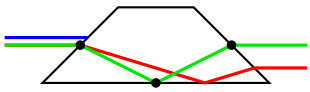
In vielen Fällen ist „connect“ identisch mit „raytracing“ für `n=1`, außer für `\optprism` oder `\doveprism` bei denen die Grenzflächenknoten (Kap. 7.6) dann einfach nur verbunden werden. Die folgenden beiden Beispiele zeigen die Unterschiede der beiden Methoden für diese Komponenten. Die Grenzflächenknoten sind zum besseren Verständnis hervorgehoben.

Der „raytracing“ Strahl mit `n=1` (blau) wird durch die Komponente transmittiert ohne die Richtung zu verändern, für `n=1.5` (rot) wird die Brechung an den Grenzflächen berechnet, und der „connect“ Strahl (grün) durchläuft genau die Grenzflächenknoten.



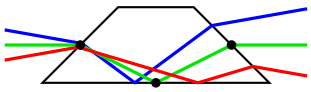
```
\begin{pspicture}(3,2.2)
  \pnode(0,1.5){A}\pnode(1,1.5){B}\pnode(3,1){C}
  \optprism[prismsize=1.2](A)(B)(C)
  \optplane(C)
  \drawbeam[n=1, linecolor=blue]{(A)}{1}{2}
  \drawbeam[n=1.5, linecolor=red]{(A)}{1}{2}
  \drawbeam[raytrace=false, linecolor=DGreen]{(A)}{1}{2}
\end{pspicture}
```

Das \doveprism wird im Prinzip genauso behandelt wie das \optprism, der „raytracing“ Strahl mit n=1 (blau) trifft wegen fehlender Brechung die zweite Grenzfläche gar nicht, und endet daher schon an der ersten Grenzflächen (siehe Kap. 8.4). Der grüne „connect“ Strahl verläuft wieder durch die Grenzflächenknoten.

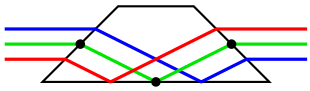


```
\begin{pspicture}(4,2)
  \pnode(0,1){A}\pnode(4,1){B}
  \doveprism[doveprismsize=1](A)(B)
  \drawbeam[n=1, linecolor=blue, beampos=0.1]{(A)}{1}{(B)}
  \drawbeam[n=1.5, linecolor=red]{(A)}{1}{(B)}
  \drawbeam[raytrace=false, linecolor=DGreen]{(A)}{1}{(B)}
\end{pspicture}
```

Für „connect“ können auch unterschiedliche Anfangsbedingungen (8.2.3) verwendet werden.



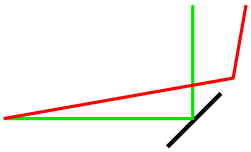
```
\begin{pspicture}(3.7,2)
  \pnode(0,1){A}\pnode(4,1){B}
  \doveprism[doveprismsize=1](A)(B)
  \psset[optexp]{raytrace=false}
  \drawbeam[linecolor=blue, beampos=0.2,
     beamangle=-10]{(A)}{1}{(B)}
  \drawbeam[raytrace=false, linecolor=DGreen]{(A)}{1}{(B)}
  \drawbeam[linecolor=red, beampos=-0.2,
     beamangle=10]{(A)}{1}{(B)}
\end{pspicture}
```



```
\begin{pspicture}(3.7,2)
  \pnode(0,1){A}\pnode(4,1){B}
  \doveprism[doveprismsize=1](A)(B)
  \psset[optexp]{raytrace=false}
  \drawbeam[linecolor=blue, beampos=0.2]{(A)}{1}{(B)}
  \drawbeam[linecolor=DGreen]{(A)}{1}{(B)}
  \drawbeam[linecolor=red, beampos=-0.2]{(A)}{1}{(B)}
\end{pspicture}
```

useNA=true, false default: true

Ein Strahlengang wird abgebrochen, falls eine Komponente nicht getroffen wird. Für manche Zwecke, z.B. zum Ermitteln des richtigen Brechungsindex, kann es sinnvoll sein, wenn die numerische Apertur eine Komponente nicht berücksichtigt wird. In dem folgenden Beispiel wird der rote Strahl gezeichnet, obwohl er außerhalb der gezeichneten Spiegelfläche liegt.



```
\begin{pspicture}(3,2)
  \pnode(0,0.5){A}\pnode(2.5,0.5){B}\pnode(2.5,2){C}
  \mirror(A)(B)(C)
  \psset{useNA=false}
  \drawbeam{(A)}{}{(C)}
  \drawbeam[linecolor=red, beamangle=10]{(A)}{}{(C)}
\end{pspicture}
```

8.2.2. Brechungsindex

Das Raytracing wird durch den Brechungsindex bestimmt, der für jede Komponente einzeln gesetzt werden kann. Der Wert ist relativ zum Hintergrund, für den ein konstanter Wert für die gesamte Skizze angenommen wird.

n=⟨*code*⟩ default: 1.5

Setzt den Brechungsindex relativ zum Hintergrund. Diese Option hat zwei Anwendungsbereiche: zum einen kann der Index einer Komponente gesetzt werden, zum anderen der Brechungsindex für einen Strahlengang:

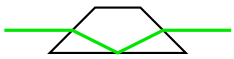
1. Wird n als globaler Parameter oder als Komponentenparameter verwendet, so wird der Index für alle betroffenen Komponenten gesetzt. Ohne weitere Änderungen wird jeder Lichtstrahl diese Werte verwenden.
2. Als Parameter zu \drawbeam oder \drawwidebeam kann der Brechungsindex der Komponenten für einzelne Strahlengänge überschrieben oder geändert werden. Das kann sehr nützlich sein, um z.B. chromatische Dispersion zu illustrieren. Soll die Änderung global für alle oder mehrere Lichtstrahlen wirksam sein, so muss der gewünschte Wert dem Beam-Stil hinzugefügt werden, da globale Definitionen von n nur die Komponenten betreffen.

Der ⟨*code*⟩ kann jeglicher Postscript-Kode sein, der zu einer Zahl ausgewertet werden kann, also z.B. 1.5 oder 5 sqrt. Beginnt ⟨*code*⟩ mit einem ∗, so wird der

Code als algebraischer Ausdruck[1] ausgewertet, d.h. `*sqrt(5)` ist äquivalent zu `5 sqrt`. Innerhalb von ⟨*code*⟩ kann mit `n` auf den vordefinierten Brechungsindex der Komponenten zugegriffen werden, es kann also eine Änderung relativ zum ursprünglichen Wert erzielt werden. Mit `n=n 1.01 mul` kann `n` z.B. um ein Prozent geändert werden.

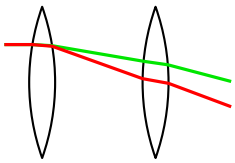
In den folgenden Beispielen werden diese unterschiedlichen Möglichkeiten von `n` gezeigt.

1. Setze den Brechungsindex einer Komponente. Interessanterweise ist für die Standardform des `\doveprism` ein Brechungsindex von $\sqrt{5}$ der ideale Wert so dass der Ausgangsstrahl parallel zum Eingangsstrahl verläuft.



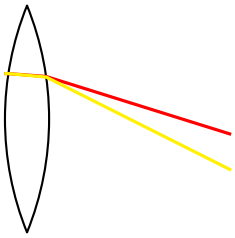
```
\begin{pspicture}(3,1.5)
  \pnode(0,1){A}\pnode(3,1){B}
  \doveprism[n=*sqrt(5)](A)(B)
  \drawbeam{(A)}{}{(B)}
\end{pspicture}
```

2. Ein Strahl (grün) verwendet die voreingestellten Werte der Komponenten, der rote überschreibt diese mit 2.



```
\begin{pspicture}(3,3)
  \pnode(0,1){A}\pnode(3,1){B}
  \optplane(A)
  \psset{lens=3 3 2, n=1.5}
  \lens[abspos=0.5](A)(B)
  \lens[abspos=2](A)(B)
  \optplane(B)
  \addtopsstyle{Beam}{beampos=0.5}
  \drawbeam{-}
  \drawbeam[linecolor=red, n=2]{-}
\end{pspicture}
```

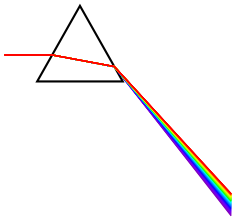
3. Der rote Strahl verwendet die voreingestellten Werte, der grüne ändert diese um `0.5`.



```
\begin{pspicture}(3,3)
  \pnode(0,1.5){A}\pnode(3,1.5){B}
  \lens[lens=4 4 3, n=2, abspos=0.3](A)(B)
  \optplane[compname=Plane](B)
  \psset{beampos=0.6}
  \drawbeam[linecolor=red]{(A)}{1}{Plane}
  \drawbeam[linecolor=yellow, n=*(n+0.5)]{(A)}{1}{Plane}
\end{pspicture}
```

[1] `http://mirror.ctan.org/graphics/pstricks/base/doc/pst-news08.pdf`

4. Mit dynamischer Anpassung von n kann chromatische Dispersion sehr gut emuliert werden. Hier wird die Dispersion eines Prisma gezeigt, der Brechungsindex wird mit der Sellmaier Gleichung[2] für die unterschiedlichen Wellenlängen berechnet.



```
\begin{pspicture}(3,3)
  \pnode(-1,0){A}\pnode(1,2.2){B}\pnode(3,0){C}
  \optplane(0,2.15)
  \optprism[prismalign=center, prismangle=59](A)(B)(C)
  \optplane(C)
  \definecolor[ps]{bl}{rgb}{%
    tx@addDict begin Red Green Blue end}%
  \addtopsstyle{Beam}{linecolor=bl,
      linewidth=0.4\pslinewidth, beamalign=abs}
  \multido{\i=0+1}{60}{%
    \pstVerb{%
      \i\space 650 400 sub 59 div mul 400 add
      tx@addDict begin wavelengthToRGB end }%
    \drawbeam[n=\i\space 650 400 sub 59 div mul 400 add
        Sellmaier]{-}%
  }%
\end{pspicture}
```

5. Die numerische Apertur einer Lochblende als spektraler Filter.

```
\begin{pspicture}(0.3,0)(3.3,5.2)
  \pnode(0.3,1){A}\pnode(2,1){B}\pnode(3,2){C}\pnode(2.5,5){D}
  \optplane(A)
  \optgrating[reverse](A)(B)(C)
  \pinhole[phwidth=-0.1, innerheight=0.03, abspos=0.5](D)(B)
  \optplate[position=end, plateheight=1.5](B)(D)
  \definecolor[ps]{bl}{rgb}{%
    tx@addDict begin Red Green Blue end}%
  \drawbeam[linecolor=red]{1-2}
  \addtopsstyle{Beam}{linecolor=bl,
      linewidth=0.4\pslinewidth}
  \multido{\i=0+1}{60}{%
    \pstVerb{%
      \i\space 650 400 sub 59 div mul 400 add
      tx@addDict begin wavelengthToRGB end }%
    \drawbeam[beamangle=\i\space 0.1 mul 3 sub]{2-}%
  }%
\end{pspicture}
```

[2] http://en.wikipedia.org/wiki/Sellmeier_equation

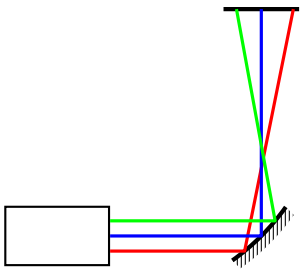
refractiveindex
=⟨*code*⟩

Dieser Parameter ist seit Version 3.0 veraltet, verwenden Sie stattdessen n.

8.2.3. Anfangsbedingungen

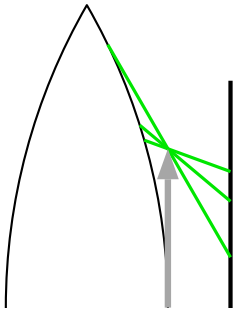
beampos=[⟨*x*⟩]⟨*y*⟩ default: 0

Die Startposition (⟨*x*⟩, ⟨*y*⟩) des Strahls an der ersten Grenzfläche. Beide Zahlen sind vom Typ ⟨*psnum*⟩. Wird eine einzelne Zahl angegeben, wird die x-Koordinate auf 0 gesetzt.



```
\begin{pspicture}(4,3.8)
  \pnode(1.5,0.5){A}\pnode(3.5,0.5){B}\pnode(3.5,3.5){C}
  \optbox[position=end](B)(A)
  \mirror[mirrortype=extended, mirrorradius=3](A)(B)(C)
  \optplate[position=end](B)(C)
  \drawbeam[linecolor=red, beampos=-0.2]{1-3}
  \drawbeam[linecolor=blue, beampos=0]{1-3}
  \drawbeam[linecolor=green, beampos=0.2]{1-3}
\end{pspicture}
```

Die Startposition ist relativ zum entsprechenden Grenzflächenknoten, wie im folgenden Beispiel gezeigt wird. Die Startposition ist fest, abhängig z.B. von beamangle kann der eigentliche Schnittpunkt mit der ersten Grenzfläche jedoch anders sein.

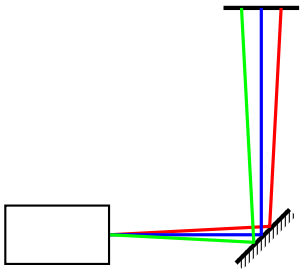


```
\begin{pspicture*}(0,2)(3.05,6)
  \lens[lens=8 8 8, abspos=0.6](0.5,2)(3,2)
  \optplate[position=1, plateheight=6](0.5,2)(3,2)
  \psset{beampos=2.1}
  \drawbeam[beamangle=-60]{1}{2}
  \drawbeam[beamangle=-40]{1}{2}
  \drawbeam[beamangle=-20]{1}{2}
\end{pspicture*}
```

An welcher Grenzfläche der ersten Komponente gestartet wird hängt von beaminsidefirst und startinside ab.

`beamangle`=⟨*psnum*⟩ default: `0`

Der Anfangswinkel des Strahls. Dieser ist in der Regel relativ zu der Verbindung von der ersten zur zweiten Komponente, mit `beamalign` kann dieses Verhalten verändert werden.

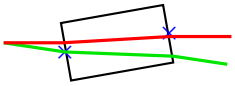


```
\begin{pspicture}(4,4)
  \pnode(1.5,0.5){A}\pnode(3.5,0.5){B}\pnode(3.5,3.5){C}
  \optbox[position=end](B)(A)
  \mirror[mirrortype=extended](A)(B)(C)
  \optplate[position=end](B)(C)
  \drawbeam[linecolor=red, beamangle=3]{-3}
  \drawbeam[linecolor=blue, beamangle=0]{1-}
  \drawbeam[linecolor=green, beamangle=-3]{-}
\end{pspicture}
```

`beamalign`=`rel`, `relative`, `abs`, `absolute` default: `relative`

Gibt vor, ob der Winkel `beamangle` relativ zu der Verbindung zwischen den ersten beiden Komponenten ist, oder ob es sich um einen absoluten Winkel handelt.

Ein absoluter Winkel kann sehr hilfreich sein, falls eine der beiden ersten Komponenten gedreht oder verschoben wurde. Dabei werden die Grenzflächenknoten, auf die sich ein relativer Winkel bezieht, ebenfalls transformiert (siehe Kap. 3.3). In dem folgenden Beispiel verläuft daher der grüne Strahl mit der relativen Ausrichtung nicht horizontal.



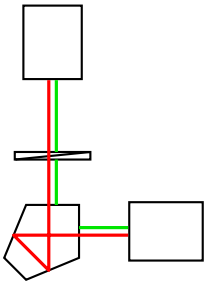
```
\begin{pspicture}(3,1.5)
  \pnode(0,0.75){A}\pnode(3,0.75){B}
  \optbox[angle=10, showifcnodes](A)(B)
  \drawbeam{(A)}{}{(B)}
  \drawbeam[linecolor=red, beamalign=abs]{(A)}{}{(B)}
\end{pspicture}
```

8.2.4. Interner Strahlengang

Ein Strahlengang setzt sich aus Strahlen zwischen den Komponenten und innerhalb der Komponenten zusammen. Ob diese internen Strahlen gezeichnet werden, kann mit mehreren Parametern festgelegt werden. Die erste, letzte und die übrigen Komponenten werden separat gehandhabt.

`beaminside`=`true`, `false` default: `true`

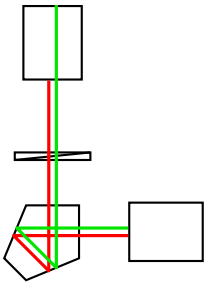
Zeichnet den Strahlengang innerhalb aller Komponenten mit Ausnahme der ersten und letzten, die von dieser Option nicht beeinflusst.



```
\begin{pspicture}(0.3,0.3)(3,4)
  \psset{optboxwidth=1}
  \optbox[position=end](1,1)(1,3)
  \optretplate(1,1)(1,3)
  \pentaprism(1,3)(1,1)(2,1)
  \optbox[position=end](1,1)(2,1)
  \drawbeam[beampos=-0.05, linecolor=red]{-}
  \drawbeam[beampos=0.05, beaminside=false]{-}
\end{pspicture}
```

beaminsidefirst=true, false default: false

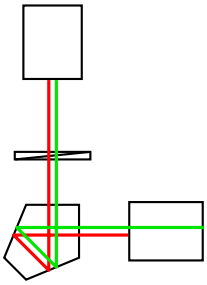
Zeichne den Strahlengang innerhalb der ersten Komponenten.



```
\begin{pspicture}(0.3,0.3)(3,4)
  \psset{optboxwidth=1}
  \optbox[position=end](1,1)(1,3)
  \optretplate(1,1)(1,3)
  \pentaprism(1,3)(1,1)(2,1)
  \optbox[position=end](1,1)(2,1)
  \drawbeam[beampos=-0.05, linecolor=red]{-}
  \drawbeam[beampos=0.05, beaminsidefirst]{-}
\end{pspicture}
```

beaminsidelast=true, false default: false

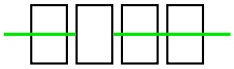
Zeichne den Strahlengang innerhalb der letzten Komponenten.



```
\begin{pspicture}(0.3,0.3)(3,4)
  \psset{optboxwidth=1}
  \optbox[position=end](1,1)(1,3)
  \optretplate(1,1)(1,3)
  \pentaprism(1,3)(1,1)(2,1)
  \optbox[position=end](1,1)(2,1)
  \drawbeam[beampos=-0.05, linecolor=red]{-}
  \drawbeam[beampos=0.05, beaminsidelast]{-}
\end{pspicture}
```

allowbeaminside=true, false default: true

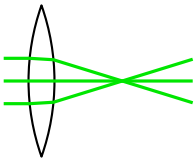
Damit kann bestimmt werden, ob der Strahlengang innerhalb einer Komponente gezeichnet werden darf. Dieser Parameter wirkt nur auf Komponenten und überschreibt, falls auf false gesetzt, jegliche Strahleinstellungen mit beaminside, beaminsidefirst und beaminsidelast. Damit kann der Strahlengang in einzelnen Komponenten unterdrückt werden, für die es keinen Sinn macht (z.B. \optfilter und \optdiode, was sonst nicht möglich ist.



```
\begin{pspicture}(3,2)
  \psset{optboxwidth=0.5}
  \pnode(0,1){A}\pnode(3,1){B}
  \optbox[position=0.2](A)(B)
  \optbox[position=0.4, allowbeaminside=false](A)(B)
  \optbox[position=0.6](A)(B)
  \optbox[position=0.8](A)(B)
  \drawbeam{(A)}{-}{(B)}
\end{pspicture}
```

8.2.5. Verbinden mit Knoten

Wie vorher schon einmal erwähnt können Komponenten auch mit Knoten verbunden werden. Für Raytracing wird in der Regel eine Ebene benötigt, was ein Problem hinsichtlich der Knoten darstellt. Das wird so gehandhabt, das eine temporäre Ebene definiert wird, die durch den Knoten verläuft und vertikal zur Verbindung zwischen dem Knoten und der vorangegangenen Komponente steht. Fängt ein Strahlengang mit einem Knoten an, so wird die Ausrichtung durch den Knoten und die folgende Komponente bestimmt.

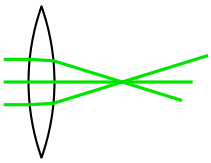


```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(2.5,1){B}
  \lens[lens=3 3 2, n=2.5, abspos=0.5](A)(B)
  \multido{\r=-0.3+0.3}{3}{%
    \drawbeam[beampos=\r]{(A)}{}{(B)}}
\end{pspicture}
```

Das ist in vielen Fällen sehr praktisch, kann aber wiederum unerwünschte Ergebnisse bei gedrehten und verschobenen Komponenten ergeben und die Ebenenausrichtung kann nicht beliebig gesetzt werden. Das kann mit einer unsichtbaren Ebene erreicht werden, die mit `\optplane` definiert wird.

`\optplane`(⟨*center*⟩)

Die Ebene wird durch lediglich einen Knoten definiert, die Drehung darum kann mit `angle` angegeben werden. Der Winkel ist relativ zur ursprünglichen, vertikalen Ausrichtung.



```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(2.5,1){B}
  \lens[lens=3 3 2, n=2.5, abspos=0.5](A)(B)
  \optplane[angle=-30](B)
  \multido{\r=-0.3+0.3}{3}{%
    \drawbeam[beampos=\r]{(A)}{-}}
\end{pspicture}
```

8.2.6. Automatische Verbindungen

Lichtstrahlen können auch zusammen mit der Komponentendefinition gezeichnet werden. Das kann für einfache Strahlengänge sehr praktisch sein, ist aber sehr unflexibel.

`beam=true, false` default: `false`

Kann als Option für jede Komponente verwendet werden, ist äquivalent zu

```
\drawbeam[raytrace=false]{(\oenodeRefA{})}{}{(\oenodeRefB{})}
```

womit ein Strahl vom ersten Referenzknoten über die Komponente zum zweiten Referenzknoten gezeichnet wird.

`conn=`⟨*string*⟩

Diese Option wird nur aus Kompatibilitätsgründen bereitgestellt und wird aus zukünftigen Versionen entfernt. Siehe Kap. 11.3.1 für weitere Informationen.

8.2.7. Strahlaussehen

`Beam` ⟨*psstyle*⟩ default: `linecolor=green!90!black, linejoin=1`

Das Aussehen der Lichtstrahlen kann mit diesem Stil gesteuert werden. Das optionale Argument von `\draw*beam` kann ebenfalls dafür verwendet werden und kann die Einstellungen des `Beam`-Stils überschreiben.

`addtoBeam=`⟨*list*⟩

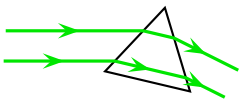
Der vorhandene `Beam`-Stil wird lokal um die Parameter in ⟨*list*⟩ erweitert, ⟨*list*⟩ muss mit geschweiften Klammern gekapselt werden.

newBeam=⟨*list*⟩

Ähnlich wie addtoBeam, nur wird der Beam-Stil mit den neuen Parametern überschrieben.

ArrowInsideMinLength=⟨*psnum*⟩
default: 0.2

\draw*beam verwendet auch die Option ArrowInside aus pstricks-add. In dem Fall wird für alle Strahlsegmente ein Pfeil gezeichnet, was häufig für die internen, kurzen Strahlengänge unerwünscht ist. Mit ArrowInsideMinLength kann angegeben werden ab welcher Mindestlänge eines Strahlsegmentes die Pfeil gezeichnet werden. In dem folgenden Beispiel ist der Wert so gewählt, dass nur der untere interne Strahl einen Pfeil hat (siehe auch Bsp. 10.16).



```
\begin{pspicture}(3,2)
  \optprism(0,1)(2,1)(3,0.5)
  \drawwidebeam[beamwidth=0.4, ArrowInside=->,
     ArrowInsideMinLength=0.5, arrowscale=1.5,
     raytrace=false]{(0,1)}{}{(3,0.5)}
\end{pspicture}
```

ArrowInsideMaxLength=⟨*psnum*⟩
default: -1

Gibt analog zu ArrowInsideMinLength die maximale Länge eines Segments an, für das Pfeile gezeichnet werden. Ist der Wert negativ, dann gibt es keine obere Grenze.

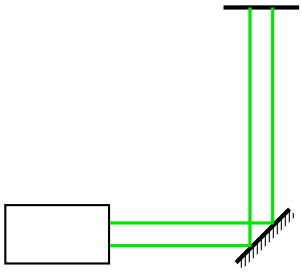
8.3. Aufgeweitete Strahlen

Alle Informationen über einfache Strahlen (siehe Kap. 8.2) betreffen auch aufgeweitete Strahlen. Hier werden nur zusätzliche Einstellmöglichkeiten aufgeweiteter Strahlen behandelt.

\drawwidebeam[⟨*options*⟩]{⟨obj_1⟩}{⟨obj_2⟩}...

beamwidth=⟨*psnum*⟩ default: 0

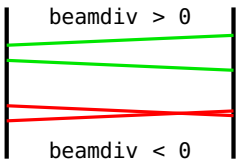
Die Anfangsstrahlbreite an der mit beampos angegebenen Position, gemessen vertikal zur Verbindungslinie zur nächsten Komponente.



```
\begin{pspicture}(4,4)
  \pnode(1.5,0.5){A}\pnode(3.5,0.5){B}\pnode(3.5,3.5){C}
  \optbox[position=end](B)(A)
  \mirror[mirrortype=extended](A)(B)(C)
  \optplate[position=end](B)(C)
  \drawwidebeam[beamwidth=0.3]{-}
\end{pspicture}
```

beamdiv=⟨*psnum*⟩ default: 0

Die Strahldivergenz an der Startposition, angegeben in Grad. Positive Werte ergebenen einen expandierenden Strahl, negative einen kontrahierenden.



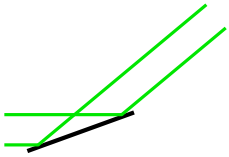

```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \psset{plateheight=2}
  \optplate[position=start](A)(B)
  \optplate[position=end](A)(B)
  \addtopsstyle{Beam}{beamwidth=0.2}
  \drawwidebeam[beampos=0.4, beamdiv=5]{-}
  \drawwidebeam[beampos=-0.4, beamdiv=-5, linecolor=red]{-}
\end{pspicture}
```

8.3.1. Strahlaussehen

Das Aussehen aufgeweiteter Strahlen kann genauso wie bei einfachen Strahlen über Beam oder das optionale Argument vorgegeben werden (siehe Kap. 8.2.7).

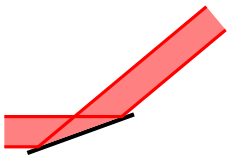
Breite Strahlen bestehen aus zwei Randstrahlen, die gemäß der Linieneinstellungen gezeichnet werden, und können zusätzlich gefüllt werden.

1. Nur die Randstrahlen.



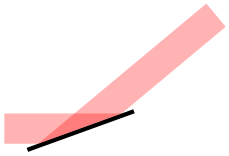
```
\begin{pspicture}(3,2)
  \pnode(0,0.5){A}\pnode(1,0.5){B}\pnode(2.8,2){C}
  \mirror[mirrorwidth=1.5](A)(B)(C)
  \drawwidebeam[beamwidth=0.4]{(A)}{}{(C)}
\end{pspicture}
```

2. Randstrahlen, und Füllstil.



```
\begin{pspicture}(3,2)
  \pnode(0,0.5){A}\pnode(1,0.5){B}\pnode(2.8,2){C}
  \mirror[mirrorwidth=1.5](A)(B)(C)
  \addtopsstyle{Beam}{fillstyle=solid,
      fillcolor=red!50!white, linecolor=red}
  \drawwidebeam[beamwidth=0.4]{(A)}{}{(C)}
\end{pspicture}
```

3. Nur Füllstil.

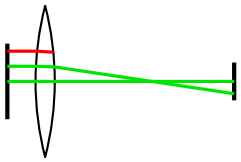


```
\begin{pspicture}(3,2)
  \pnode(0,0.5){A}\pnode(1,0.5){B}\pnode(2.8,2){C}
  \mirror[mirrorwidth=1.5](A)(B)(C)
  \addtopsstyle{Beam}{fillstyle=solid, fillcolor=red,
      opacity=0.3, linestyle=none}
  \drawwidebeam[beamwidth=0.4]{(A)}{}{(C)}
\end{pspicture}
```

8.4. Fehlerbehandlung

Da alle Strahlen direkt mit Postscript berechnet und gezeichnet werden, ist eine umfassende Fehlerbehandlung zur Zeit der Kompilation nicht möglich. Um jedoch trotzdem fehlerfreie Bilder zu bekommen, wird der Strahlengang abgebrochen, sobald eine Situation eintritt, die nicht unterstützt wird. Das kann eintreten, falls der Strahl eine Grenzfläche nicht trifft, oder Totalreflektion auftritt.

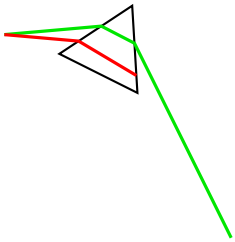
1. Ein Strahl trifft eine Grenzfläche nicht. Der rote Strahl in dem folgenden Beispiel verpasst die letzte Grenzfläche und endet daher an der vorhergehenden.



```
\begin{pspicture}(3,2.7)
  \pnode(0,1){A}\pnode(3,1){B}
  \optplate[position=start](A)(B)
  \lens[lens=4 4 2, abspos=0.5, n=2.5](A)(B)
  \optplate[position=end, plateheight=0.5](A)(B)
  \drawbeam{-}
  \drawbeam[beampos=0.2]{-}
  \drawbeam[beampos=0.4, linecolor=red]{-}
\end{pspicture}
```

2. Totale interne Reflektion tritt an einer Grenzfläche auf, die nicht dafür vorgesehen ist. Der rote Strahl im folgenden Beispiel würde reflektiert

werden, aber das wird für diese Komponente nicht unterstützt und der Strahl endet an der total reflektierenden Grenzfläche.



```
\begin{pspicture}(3,2.5)
  \pnode(0,2){A}\pnode(3,0){B}
  \optplane(A)
  \optprism[n=1.8](A)([Xnodesep=1.5]A)(B)
  \optplane(B)
  \drawbeam[beamangle=5]{-}
  \drawbeam[beamangle=-5, linecolor=red]{-}
\end{pspicture}
```

Dasselbe gilt ebenfalls für aufgeweitete Strahlen (siehe Kap. 8.3). Bei diesen wird der Strahlengang abgebrochen, falls einer der beiden Randstrahlen einen Fehler aufweist.

`pswarning`=`true`, `false` default: `false`

Um eine Fehlersuche zu vereinfachen kann ein Hinweis ausgegeben werden sobald ein Strahlengang vorzeitig beendet wird weil entweder eine Grenzfläche verpasst wird, oder totale interne Reflektion auftritt.

8.5. Angepasste Strahlen

Im Prinzip können alle Strahlengänge mit passender Wahl des Brechungsindex und der Komponentenparameter (z.B. der Linsenkrümmung) realisiert werden, was jedoch sehr aufwendig werden kann. Um das zu vereinfachen, besteht die Möglichkeit die Endpunkte eines Strahls als Anfangspunkte für folgende Strahlen zu nutzen, so dass man abschnittsweise die Divergenz und die Ausbreitungsrichtung definieren kann.

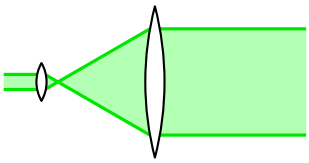
`savebeampoints`=`true`, `false`, ⟨*int*⟩ default: `true`

Speichert die Endpunkte eines `\draw*beam` Makros, vorangegangene Punkte werden überschrieben. `\drawbeam` und `\drawwidebeam` werden getrennt behandelt. Wird hier eine Zahl $\langle int \rangle > 0$ angegeben, so können beliebig viele Endpunkte separat gespeichert werden. In der Regel reicht an- und abschalten vollkommen aus.

`loadbeampoints`=`true`, `false`, ⟨*int*⟩ default: `false`

Verwende die Endpunkte eines vorangegangenen Strahls als Startpunkte. Wird keine Zahl ⟨*int*⟩ angegeben, dann wird 1 verwendet.

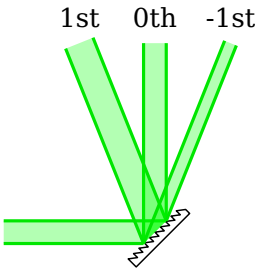
Als Beispiel nehmen wir ein Teleskop, die Strahldivergenz am Eingang und am Ausgang ist Null. Anstatt die Linsenparameter und den Brechungsindex genau einzustellen, so dass die Ausgangsdivergenz Null ist, zeichnen wir einfach die drei Strahlenteile mit den gewünschten Divergenzen, und verwenden die Endpunkte des jeweils vorangegangenen Strahls als neue Ausgangspunkte.



```
\begin{pspicture}(4,2)
  \pnode(0,1){A}\pnode(4,1){B}
  \begin{optexp}
    \lens[lens=0.5 0.5 0.5, abspos=0.5](A)(B)
    \lens[lens=4 4 2, abspos=2](A)(B)
    \addtopsstyle{Beam}{%
      fillstyle=solid, fillcolor=green, opacity=0.3}
    \psset{loadbeampoints}
    \drawwidebeam[beamwidth=0.2, stopinside]{(A)}{1}
    \drawwidebeam[beamdiv=-60]{1}{2}
    \drawwidebeam{2}{(B)}
  \end{optexp}
\end{pspicture}
```

Als weiteres Beispiel betrachten wir die Beugung an einem Gittern in die nullte und erste Beugungsordnung. Zuerst wird der Strahl von dem Eingang bis zum Gitter gezeichnet, die Endpunkte dieses Strahls werden nun für die Strahlen in beide Beugungsordnungen verwendet.

Bsp. 8.1




```
\begin{pspicture}(3.5,3.5)
  \pnode(0,0.5){A}\pnode(2,0.5){B}\pnode(2,3){C}
  \nodexn{(C)-(1,0)}{C'}
  \nodexn{(C)(1,0)}{C''}
  \optgrating(A)(B)(C)
  \addtopsstyle{Beam}{%
    fillstyle=solid, fillcolor=green, opacity=0.3}
  \drawwidebeam[beamwidth=0.3]{(A)}{1}
  \psset{savebeampoints=false, loadbeampoints}
  \drawwidebeam{1}{(C)}
  \drawwidebeam{1}{(C')}
  \drawwidebeam{1}{(C'')}
\end{pspicture}
```

`savebeam`=`true`, `false` default: `true`

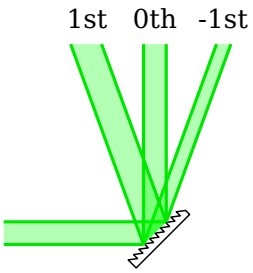
Speichert die Endzustände, d.h. Endpunkte und Ausgangsvektoren, eines `\draw*beam` Makros, vorangegangene Werte werden überschrieben. `\drawbeam` und `\drawwidebeam`

werden getrennt behandelt. Wird hier eine Zahl $\langle int \rangle > 0$ angegeben, so können beliebig viele Strahlen separat gespeichert werden. In der Regel reicht an- und abschalten vollkommen aus.

`loadbeam`=`true`, `false` default: `false`

Verwende den Endzustand (Punkte und Richtung) eines vorangegangenen Strahls als Anfangszustand. Der Parameter `beamangle` kann verwendet werden um die Richtung relativ zu der geladenen zu verändern (siehe folgendes Beispiel).

Bsp. 8.2

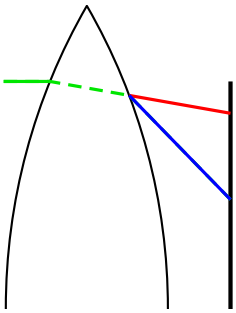



```
\begin{pspicture}(3.5,3.5)
  \pnode(0,0.5){A}\pnode(2,0.5){B}\pnode(2,3){C}
  \optgrating(A)(B)(C)
  \addtopsstyle{Beam}{%
    fillstyle=solid, fillcolor=green, opacity=0.3}
  \drawwidebeam[savebeam, beamwidth=0.3]{(A)}{1}
  \psset{savebeam=false, loadbeam}
  \drawwidebeam[beamangle=-20]{1}{(C)}
  \drawwidebeam{1}{(C)}
  \drawwidebeam[beamangle=20]{1}{(C)}
\end{pspicture}
```

`startinside`=`true`, `false` default: `false`

Mit dem Parameter `beaminsidefirst` kann eingestellt werden, ob der interne Strahlengang in der erste Komponente gezeichnet wird. Für angepasste Strahlengänge kann es aber notwendig sein dem internen Strahlengang in der ersten Komponente zu folgen ohne diesen zu zeichnen, z.B. wenn der vorangegangene Strahl an der ersten Grenzfläche schon endet.

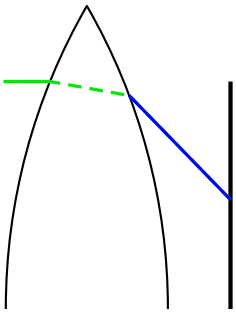
Im folgenden Beispiel wird der einfallende grüne Strahl falsch durch den roten Strahl fortgesetzt, da der grüne an der linken Grenzfläche endet, der rote aber fälschlicherweise annimmt, dass der Endzustand sich auf die rechte Grenzfläche bezieht. Dem blauen Strahl wird nun mit `startinside` mitgeteilt, dass der Endzustand sich auf die linke Grenzfläche bezieht.



```
\begin{pspicture*}(0,2)(3.05,6)
  \optplane(0,2)
  \lens[lens=8 8 8, abspos=0.6, n=1.8](0.5,2)(3,2)
  \optplate[position=1, plateheight=6](0.5,2)(3,2)
  \psset{savebeam=false}
  \drawbeam[beampos=3, linestyle=dashed]{-}
  \drawbeam[beampos=3, savebeam]{1-2}
  \drawbeam[loadbeam, linecolor=red]{2-3}
  \drawbeam[loadbeam, startinside, linecolor=blue]{2-3}
\end{pspicture*}
```

stopinside=true, false default: false

Äquivalent zu startinside, bezieht sich aber auf die letzte Komponenten.



```
\begin{pspicture*}(0,2)(3.05,6)
  \optplane(0,2)
  \lens[lens=8 8 8, abspos=0.6, n=1.8](0.5,2)(3,2)
  \optplate[position=1, plateheight=6](0.5,2)(3,2)
  \psset{savebeam=false}
  \drawbeam[beampos=3, linestyle=dashed]{-}
  \drawbeam[beampos=3, savebeam, stopinside]{1-2}
  \drawbeam[loadbeam, linecolor=blue]{2-3}
\end{pspicture*}
```

8.6. Faserverbindungen

Alle Komponenten können mit Fasern verbunden werden, die Faserkomponenten werden automatisch verbunden.

\drawfiber[⟨*options*⟩]{⟨obj_1⟩}{⟨obj_2⟩}...

Das \drawfiber Makro kann ID, compname oder Knoten als ⟨obj_N⟩ Argumente verarbeiten (siehe Kap. 8.1), aber ID-Intervalle wie bei \draw*beam sind nicht möglich. Werden mehr als zwei Argumente mitgegeben, werden die Verbindungen $1 \to 2$, $2 \to 3$, usw. gezeichnet.

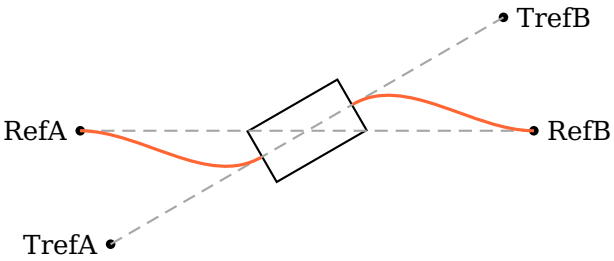
Die automatischen Faserverbindungen werden intern als \drawfiber von den Referenzknoten zur Komponente gezeichnet.

8.6.1. Faserwinkel

Üblicherweise werden Fasern als \nccurve-Verbindungen gezeichnet, die Winkel an beiden Kurvenenden werden automatisch anhand der Komponentenausrichtung bestimmt und hängen davon ab, ob eine Komponente mit einer anderen Komponente oder einem Knoten verbunden wird.

Komponente mit Knoten

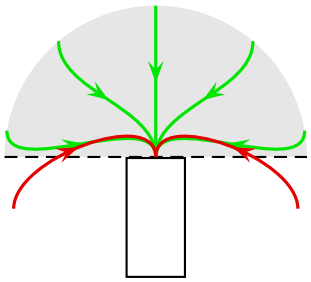
Eine Verbindung zwischen einer Komponente und einem Knoten ist das, was bei automatischen Verbindungen verwendet wird (siehe Kap. 8.6.3). Folgendes Beispiel erläutert die Ausrichtung der Kurve:




```
\begin{pspicture}(-1,-0.7)(7,2.7)
  \pnode(0,1){A}\pnode(6,1){B}
  \optbox[angle=30](A)(B)
  \drawfiber{(A)}{}{(B)}
\end{pspicture}
```

Der Winkel an der Komponente entspricht dem der transformierten Referenzlinie. Der Winkel an dem Knoten entspricht dem der ursprünglichen Referenzlinie der Komponente zu der die Verbindung geht. Die Faser ist in Richtung des entsprechenden Komponentenknotens orientiert.

Das führt dazu, dass im folgenden Beispiel die Anfangswinkel der roten Fasern um 180° gegenüber den grünen Fasern gedreht sind, die Winkel an der Komponente bleiben unverändert.

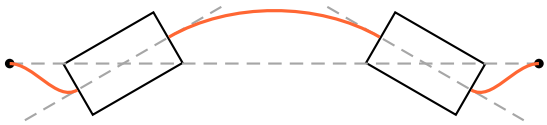


```
\begin{pspicture}(-2,-1.6)(2,2)
  \optbox[position=end](0,1)(0,0)
  \addtopsstyle{Fiber}{linecolor=green!90!black}
  \multido{\i=10+40}{5}{%
    \drawfiber{(2;\i)}{}
  }
  \addtopsstyle{Fiber}{linecolor=red!90!black}
  \drawfiber{(2;-20)}{}
  \drawfiber{(2;200)}{}
\end{pspicture}
```

Komponente mit Komponente

Diese Variante ist sehr nützlich um glattere Verbindungen zwischen Komponenten zu erreichen, die nicht auf derselben Referenzlinie positioniert sind, sondern z.B. in einer Schleife angeordnet werden, siehe Bsp. 8.3 und Bsp. 8.4.

In diesem Fall werden die Kurvenwinkel von der Komponente bestimmt, mit der das jeweilige Ende verbunden ist. Die Fasern zeigen immer von der Komponentenmitte weg.



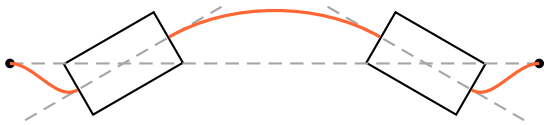
```
\begin{pspicture}(0,-0.8)(7,0.8)
  \pnode(0,0){A}\pnode(3,0){B}
  \pnode(4,0){C}\pnode(7,0){D}
  \optbox[angle=30](A)(B)
  \optbox[angle=-30](C)(D)
  \drawfiber{(A)}{1}{2}{(D)}
\end{pspicture}
```

Diese automatisch berechneten Winkel können vielfältig manipuliert werden, die möglichen Parameter sind im Folgenden aufgelistet.

`fiberalign`=`rel`, `relative`, `center`, `abs`, `absolute` default: `relative`

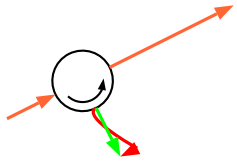
Wählt die Referenz für die Faserwinkel.

`relative` Die Faserwinkel sind relativ zur Komponente. Wird ein Knoten mit der Komponente verbunden, so entspricht der Winkel am Knoten dem der Referenzlinie der Komponente. Der Winkel an der Komponente entspricht dem der transformierten Referenzlinie.



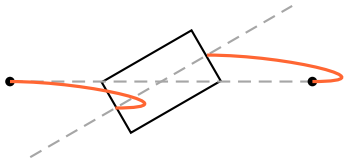
```
\begin{pspicture}(1,-1.7)(7,1.7)
  \pnode(0,0){A}\pnode(3,0){B}
  \pnode(4,0){C}\pnode(7,0){D}
  \optbox[angle=30](A)(B)
  \optbox[angle=-30](C)(D)
  \drawfiber{(A)}{1}{2}{(D)}
\end{pspicture}
```

`center` Die Winkel sowohl am Knoten als auch an der Komponente sind relativ zu der Verbindung zwischen dem betroffenen Komponentenknoten und der Komponentenmitte. Meistens ist das äquivalent zu `relative`, wird aber z.B. für `\optcirculator` benötigt. In dem Beispiel hat die grüne Faser die korrekte Ausrichtung, während die rote `relative` ausgerichtet wird.



```
\begin{pspicture}(1,0)(3,2)
  \addtopsstyle{Fiber}{arrowscale=1.3, arrows=->,
      arrowinset=0}
  \optcirculator[fiber=t](0,0.5)(3,2)(1.5,0)
  \drawfiber[linecolor=red]{}{(1.5,0)}
  \drawfiber[fiberalign=center, linecolor=green]{}{(1.5,0)}
\end{pspicture}
```

absolute Die Faserwinkel werden als absolute Werte angegeben, die automatischen Winkel werden nicht berücksichtigt.



```
\begin{pspicture}(1,-1)(4.5,1)
  \pnode(0,0){A}\pnode(4,0){B}
  \optbox[angle=30](A)(B)
  \drawfiber[fiberalign=abs, ncurv=1.5]{(A)}{}{(B)}
\end{pspicture}
```

fiberangleA=⟨*num*⟩ default: 0

Ein additiver Wert für den Anfangswinkel, der gesamte Winkel hängt von fiberalign ab. Ist dieser auf absolute eingestellt, so ist fiberangleA der absolute Winkel, andernfalls wird der Wert zu dem automatisch berechneten Winkel hinzuaddiert.

fiberangleB=⟨*num*⟩ default: 0

Wie fiberangleB, aber für den Endwinkel.

startnode=auto, N, 1, 2, ... default: auto

Für die Verbindungen mit \drawfiber werden die Knoten genommen, die sich am nächsten sind. Das passt häufig, aber nicht immer. Hiermit kann der Startknoten explizit gewählt werden.

stopnode=auto, N, 1, 2, ... default: auto

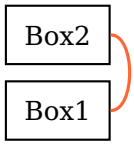
Genau wie startnode, aber für den Stopknoten.

Die folgenden Beispiele zeigen Anwendungsmöglichkeiten für startnode und stopnode.

1. In diesem Fall ist der nächstgelegene Knoten auch der richtige.

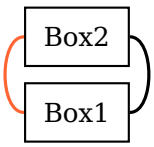

```
\begin{pspicture}(3,2)
  \pnode(-0.5,1){A}\pnode(1.5,1){B}\pnode(3.5,1){C}
  \psset{label=0, optboxwidth=1}
  \optbox(A)(B){Box1}\optbox(B)(C){Box2}
  \drawfiber{1}{2}
\end{pspicture}
```

2. Hier ist der kürzeste Abstand nicht eindeutig definiert, da es zwei Verbindungen mit der gleichen Länge gibt.




```
\begin{pspicture}(3,2)
  \pnode(0,0.5){A}\pnode(3,0.5){B}
  \pnode([offset=1]B){C}\pnode(A|C){D}
  \psset{label=0}
  \optbox(A)(B){Box1}\optbox(C)(D){Box2}
  \drawfiber{1}{2}
\end{pspicture}
```

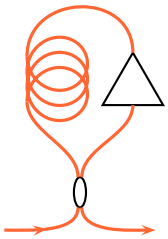
3. Um die gewünschte Verbindung zu wählen ist es meistens ausreichend entweder `startnode` oder `stopnode` zu setzen. Ist ein Knoten fest vorgegeben, wird der anderen wieder als der mit dem kürzesten Abstand gewählt. Nur in Ausnahmefällen sollte es notwendig sein beide Knoten explizit anzugegeben.




```
\begin{pspicture}(2,2)
  \pnode(0,0.5){A}\pnode(2,0.5){B}
  \pnode([offset=1]B){C}\pnode(A|C){D}
  \psset{label=0}
  \optbox(A)(B){Box1}\optbox(C)(D){Box2}
  \drawfiber[startnode=1]{1}{2}
  \drawfiber[startnode=N, linecolor=black]{1}{2}
\end{pspicture}
```

4. Ein weiteres Beispiel, bei dem `startnode` vorgegeben werden muss.

Bsp. 8.3



```
\begin{pspicture}(2.1,3)
  \pnode(0,0){A}\pnode(2,0){B}
  \pnode(0.3,3){C}\pnode(1.7,3){D}
  \psset{fiber=none}
  \optcoupler[fiber=l,
              addtoFiberIn1={angleA=0, ArrowInside=->},
              addtoFiberIn2={angleA=180, arrows=<-},
              abspos=0.5, compname=Cpl](A)(B)(C)(D)
  \optfiber[compname=Hnlf, abspos=1](C)(C|A)
  \optamp[abspos=2, compname=Amp](D|B)(D)
  \drawfiber{Cpl}{Hnlf}
  \drawfiber[startnode=1, ncurv=1.2]{Hnlf}{Amp}
  \drawfiber{Cpl}{Amp}
\end{pspicture}
```

8.6.2. Faseraussehen

`Fiber`⟨*psstyle*⟩

Das Aussehen der Fasern wird durch diesen Stil vorgegeben. Das optionale Argument von `\drawfiber` kann ebenfalls verwendet werden, die entsprechenden Parameter können die Einstellungen von `Fiber` überschreiben.

`addtoFiber`=⟨*list*⟩

Der vorhandene `Fiber`-Stil wird lokal um die Parameter in ⟨*list*⟩ erweitert, ⟨*list*⟩ muss mit geschweiften Klammern gekapselt werden.

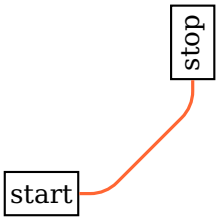
`newFiber`=⟨*list*⟩

Ähnlich wie `addtoFiber`, nur wird der `Fiber`-Stil mit den neuen Parametern überschrieben.

`fiberstyle`=⟨*string*⟩ default: `curve`

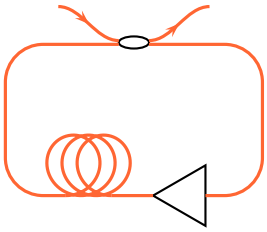
Die Fasern werden in der Regel als `\nccurve` gezeichnet, aber eine beliebige andere Knotenverbindung `\nc`⟨*string*⟩ des `pst-node` Paketes kann ausgewählt werden, lesen Sie die entsprechende Dokumentation[3] für mögliche Werte. In den folgenden Beispielen werden Anwendungsmöglichkeiten gezeigt, eine weitere ist in Bsp. 10.13 zu sehen.

[3] `http://mirror.ctan.org/help/Catalogue/entries/pst-node.html`




```
\begin{pspicture}(3,3)
  \pnode(1, 0.5){A}\pnode(2.5,0.5){B}\pnode(2.5,2){C}
  \psset{optboxsize=1 0.6, innerlabel}
  \optbox[position=start](A)(B){start}
  \optbox[position=end](B)(C){stop}
  \drawfiber[fiberstyle=diag, linearc=0.5]{1}{2}
\end{pspicture}
```

Bsp. 8.4



```
\begin{pspicture}(4,2.5)
  \pnode(1,2.5){In}\pnode(3,2.5){Out}
  \pnode(0,2){A}\pnode(4,2){B}\pnode(4,0){C}\pnode(0,0){D}
  \optcoupler[fiber=t, addtoFiber={ArrowInside=->},
      coupleralign=b](In)(A)(Out)(B)
  \psset{fiber=none}
  \optamp[position=0.35](C)(D)
  \optfiber[position=0.35](D)(C)
  \drawfiber{2}{3}
  \addtopsstyle{Fiber}{fiberstyle=bar, linearc=0.5, armA=1.5}
  \drawfiber[stopnode=1]{1}{2}
  \drawfiber[stopnode=1]{1}{3}
\end{pspicture}
```

8.6.3. Automatische Faserverbindungen

Alle Faserkomponenten, die in Kap. 5 beschrieben werden, werden automatisch mit `\drawfiber` mit ihren Referenzknoten verbunden, das Verhalten ist in Kap. 8.6.1 beschrieben. Die automatischen Faserverbindungen können umfangreich kontrolliert werden, ohne explizit `\drawfiber` zu benutzen. Das Aussehen jeder Verbindung kann ebenfalls separat konfiguriert werden, siehe Kap. 8.6.4.

`fiber`=`[*+]none`, `all`, `i`, `o`, ⟨*refpoint*⟩ default: `all`

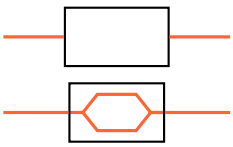
Dieser Parameter gibt an, welche Faserverbindungen gezeichnet werden. Der Parameter kann nur die Faserkomponenten betreffen (falls ein `*` vorangestellt ist), nur die Freistrahlkomponenten (`+` wird vorangestellt), oder alle Komponenten. Typische Freistrahlkomponenten, die häufig als Faserkomponenten verwendet werden, sind `\optbox` und `\optdetector`.

Es können alle Verbindungen gezeichnet werden (`all`), keine (`none`), oder es können bestimmte ausgewählt werden. Die Werte `i` (`l` ist äquivalent) zeichnet nur die Eingangsfasern, `o` (bzw. `r`) alle ausgehenden Fasern. Diese Angaben sind ausschließend, d.h. mit `io` werden gar keine Verbindungen gezeichnet.

Komponenten, die über mehr als einen Eingang oder Ausgang verfügen (Koppler und `\optcirculator`) erlauben weitergehende Auswahl mit `t` und `b`, möglicherweise in Verbindung mit `i`, `o`, `l` und `r`.

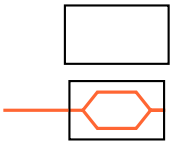
Die folgenden Beispiele zeigen einige Möglichkeiten.

1. Zeichne alle Fasern für alle Komponenten.



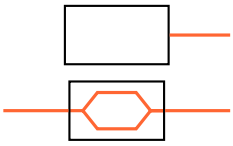
```
\begin{pspicture}(3,2)
  \psset{fiber}
  \optbox(0,1.5)(3,1.5)
  \optmzm(0,0.5)(3,0.5)
\end{pspicture}
```

2. Nur die Eingangsverbindung einer Faserkomponente (`\optmzm`) wird gezeichnet, die Freistrahlkomponente (`\optbox`) bleibt unbeeinflusst.



```
\begin{pspicture}(3,2)
  \psset{fiber=*i}
  \optbox(0,1.5)(3,1.5)
  \optmzm(0,0.5)(3,0.5)
\end{pspicture}
```

3. Von der Freistahlkomponente wird nur die Ausgangsverbindung gezeichnet, die Faserkomponente erhält jedoch alle Verbindungen.



```
\begin{pspicture}(3,2)
  \psset{fiber=+o}
  \optbox(0,1.5)(3,1.5)
  \optmzm(0,0.5)(3,0.5)
\end{pspicture}
```

4. Die Werte `t` und `b` betreffen nur die Komponenten mit mehr als zwei Verbindungen, also `\optcirculator` und die Koppler (Kap. 5.10).

```
\begin{pspicture}(3,1)
  \optcoupler[fiber=t](0,1)(0,0)(3,1)(3,0)
\end{pspicture}
```

```
\begin{pspicture}(3,1)
  \wdmsplitter[fiber=br](0,0.5)(3,1)(3,0)
\end{pspicture}
```

```
\begin{pspicture}(3,1)
  \optcirculator[fiber=b](0,0.5)(3,1)(1,0)
\end{pspicture}
```

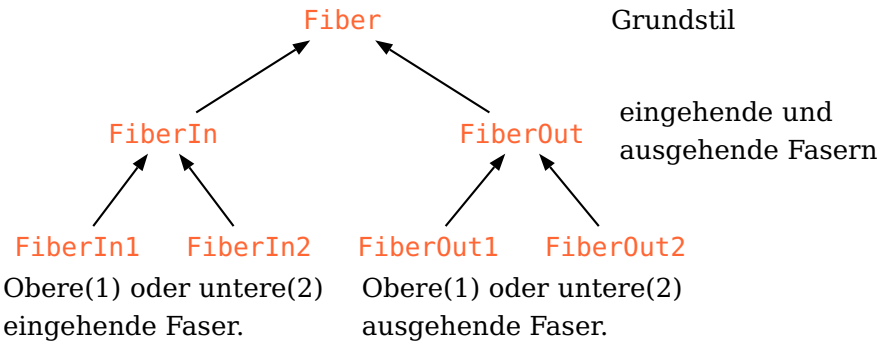


Abbildung 8.1.: Vererbungsdiagramm der PS-Stile für die automatischen Faserverbindungen. Sie sollten diese Stile mit `\addtopsstyle` oder den entsprechenden `addto`⟨*style*⟩ Parametern ändern um die Vererbungslinie beizubehalten.

8.6.4. Aussehen der automatischen Faserverbindungen

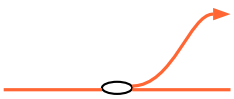
Die Stile aller automatischen Faserverbindungen können separat konfiguriert werden, der zugrundeliegende Stil ist immer `Fiber`, der in Kap. 8.6.2 beschrieben ist.

`FiberIn` `FiberIn1` `FiberIn2` `FiberOut` `FiberOut1` `FiberOut2`

Der Stil `FiberIn` bezieht sich auf alle Eingangsfasern und erbt alle Eigenschaften von `Fiber`. Hat eine Komponente zwei davon, so können diese mit `FiberIn1` und `FiberIn2` angesprochen werden, die alles von `FiberIn` übernehmen. Entsprechend bezieht sich `FiberOut` auf alle Ausgangsfasern, falls es davon zwei gibt dann können diese mit `FiberOut1` und `FiberOut2` geändert werden. Siehe Abb. 8.1 für ein Vererbungsdiagramm der Faserstile. Beachten Sie, dass diese Vererbung nur erhalten bleibt, falls die Stile nur mit `\addtopsstyle` oder den entsprechenden `addto`⟨*style*⟩ Parametern verändert werden.

`new`⟨*style*⟩ `addto`⟨*style*⟩

Für jeden Faserstil werden zwei Parameter `new`⟨*style*⟩ und `addto`⟨*style*⟩ bereitgestellt, mit denen die Stile für einzelne Komponenten verändert werden können. Damit können neuen Komponenten mit anderen voreingestellten Faserverbindungen definiert werden (Kap. 9.1), oder explizite Gruppierung kann vermieden werden.



99/1

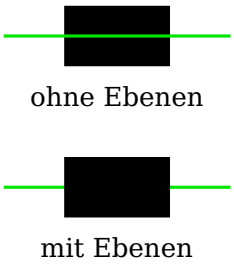
```
\begin{pspicture}(3,2)
  \newpsobject{tapcoupler}{wdmsplitter}{%
    coupleralign=b,
    addtoFiberOut1={arrows=->, arrowscale=1.2, arrowinset=0}}
  \tapcoupler(0,0.5)(3,1.5)(3,0.5){99/1}
\end{pspicture}
```

8.7. Zeichenebenen

Die Reihenfolge der Konstruktion einer Skizze ist die, dass zuerst die Komponenten definiert werden müssen, bevor diese verbunden werden können. Das beinhaltet aber, dass alle Verbindungen über die Komponenten gezeichnet werden, was nicht erwünscht sein muss.

Um das zu vermeiden, wird die optexp-Umgebung bereitgestellt, innerhalb derer alle Verbindungen hinter die Komponenten gelegt werden.

```
\begin{optexp}
  \end{optexp}
```




```
\begin{pspicture}(3,4)
  \psset{beam, fillstyle=solid, fillcolor=black}
  \optbox(0,3.5)(3,3.5){ohne Ebenen}
  \begin{optexp}
    \optbox(0,1.5)(3,1.5){mit Ebenen}
  \end{optexp}
\end{pspicture}
```

Beachten Sie, dass das erreicht wird, indem der gesamte Code innerhalb der Umgebung doppelt ausgeführt wird. Das funktioniert mit allen `pst-optexp`-Makros, da diese sich intern darum kümmern nichts doppelt sondern nur in geänderter Reihenfolge zu zeichnen. Alle anderen Zeichnungen und Texte werden zweimal gezeichnet, was zu unschönen Ergebnissen aufgrund von geändertem Antialiasing in dem Betrachter führen kann.

Um zu verhindern, dass anderer Code doppelt ausgeführt wird, muss dieser entweder außerhalb der optexp-Umgebung definiert werden, oder durch eins der beiden folgenden Makros auf eine Ebene beschränkt werden.

`\backlayer{`⟨*code*⟩`}`

Führe ⟨*code*⟩ nur beim ersten Durchgang aus.

`\frontlayer{`⟨*code*⟩`}`

Führe ⟨*code*⟩ nur beim zweiten Durchgang aus.

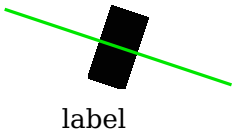
In diesem Beispiel wird der Text über dem Rechteck angezeigt, obwohl der Text zuerst definiert wird:


```
\begin{pspicture}(3,1)
\begin{optexp}
  \frontlayer{\rput(1.5,0.5){front}}
  \backlayer{\psframe*[linecolor=DOrange!50](1,0)(2,1)}
\end{optexp}
\end{pspicture}
```

9. Benutzerdefinierte Komponenten

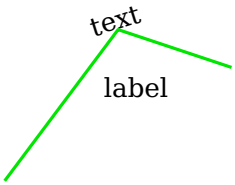
Das `pst-optexp`-Paket stellt zwei Makros zur Verfügung um eigene Komponenten zu zeichnen. Diese können z.B. aus Bildern oder eigenen Zeichnungen bestehen.

`\optdipole`[⟨*options*⟩](⟨*in*⟩)(⟨*out*⟩){⟨*label*⟩}




```
\begin{pspicture}(3,3)
  \optdipole[labeloffset=1, beam](0,2)(3,1){%
    \rput(0,0){\rule{0.5cm}{1cm}}%
  }{label}
\end{pspicture}
```

`\opttripole`[⟨*options*⟩](⟨*in*⟩)(⟨*center*⟩)(⟨*out*⟩){⟨*label*⟩}

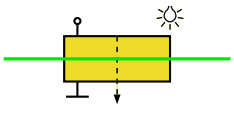



```
\begin{pspicture}(3,3)
  \opttripole[beam](3,1.5)(1.5,2)(0,0){%
    \rput[b](0,0){text}%
  }{label}
\end{pspicture}
```

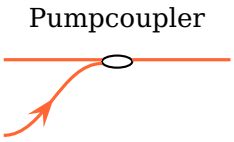
Diese Komponenten können sowohl mit Fasern als auch mit Strahlen verbunden werden, können aber nicht deren volles Potential ausschöpfen. Sie haben nur eine einzelne Grenzfläche, die die Geometrie der benutzerdefinierten Zeichnung natürlich nicht berücksichtigen kann.

9.1. Benutzerdefinierte Version existierender Komponenten

Sie können eine benutzerdefinierte Version einer existierenden Komponente mit `\newpsobject` definieren, die z.B. andere Voreinstellungen oder und ein anderes Aussehen hat.

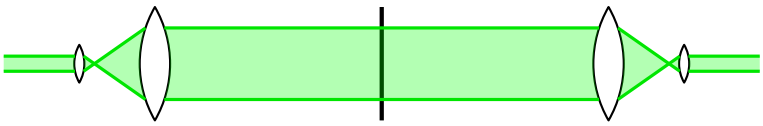



```
\begin{pspicture}(3,1.7)
\newpsobject{sbn}{crystal}{%
  voltage, lamp, %
  fillstyle=solid, fillcolor=yellow!90!black}
\sbn[label=1.2 45, beam](0,1)(3,1){SBN:Ce}
\end{pspicture}
```




```
\begin{pspicture}(3,1.7)
\newpsobject{pumpcoupler}{wdmcoupler}{%
  coupleralign=top, addtoFiberIn2={ArrowInside=->, arrowscale=2}}
\pumpcoupler[label=0.5 180](0,1)(0,0)(3,1){Pumpcoupler}
\end{pspicture}
```

Bsp. 9.1



```
\begin{pspicture}(10,2)
  \newpsobject{MOLensIn}{lens}{lens=0.5 0.5 0.5}
  \newpsobject{MOLensOut}{lens}{lens=1.5 1.5 1.5}
  \pnode(0,1){A}\pnode(10,1){B}
  \MOLensIn[abspos=1](A)(B)\MOLensOut[abspos=2](A)(B)
  \optplate[plateheight=1.5](A)(B)
  \MOLensOut[abspos=8](A)(B)\MOLensIn[abspos=9](A)(B)
  \addtopsstyle{Beam}{n=1, fillstyle=solid, fillcolor=green, opacity=0.3}
  \psset{loadbeampoints}
  \drawwidebeam[beamwidth=0.2, stopinside]{(A)}{1}
  \drawwidebeam[beamdiv=-70]{1}{2}
  \drawwidebeam[stopinside]{2-4}
  \drawwidebeam[beamdiv=-70]{4}{5}
  \drawwidebeam{5}{(B)}
\end{pspicture}
```

9.2. Neue Objekte definieren

`pst-optexp` stellt einige Benutzermakros zur Verfügung, mit denen sehr bequem neue, eigene Komponenten definiert werden können.

Beachten Sie, dass dieser Teil der Dokumentation nicht vollständig ist. Die notwendigen Schritte zur Definition neuer Komponenten in Version 3.0 werden

grob beschrieben, können sich aber noch ändern und sind nicht zwangsläufig abwärtskompatibel.

The alte Syntax mit `\mycomp@iii` wird noch unterstützt, gilt aber als veraltet und wird in zukünftigen Versionen entfernt werden. Bitte passen Sie ihre vorhandenen Makros entsprechen an.

`\newOptexpDipole`[⟨*fixopt*⟩]{⟨*name*⟩}{⟨*dftopt*⟩}

`\newOptexpTripole`[⟨*fixopt*⟩]{⟨*name*⟩}{⟨*dftopt*⟩}

`\newOptexpFiberDipole`[⟨*fixopt*⟩]{⟨*name*⟩}{⟨*dftopt*⟩}

Diese Makros erzeugen den Code für die Positionierung, Drehung, Beschriftung und die Ebenenverwaltung. Im einfachsten Fall müssen Sie lediglich die eigentliche Zeichnung der Komponente erstellen.

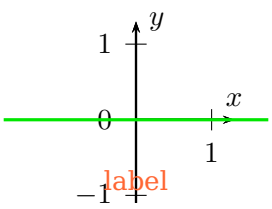
Seit Version 3.0 ist das Makro `\newOptexpDipoleNolabel` überflüssig, da seitdem die Beschriftung für alle Komponenten optional ist.

Die Definition der neuen Komponenten kann in unterschiedliche Schritte eingeteilt werden, je nach Komplexität und Anforderung des Benutzers:

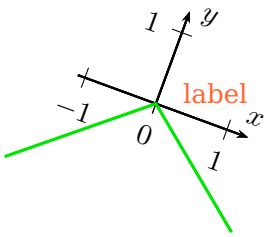
1. Die Zeichnung (siehe Kap. 9.2.1), das ist der einzige erforderliche Schritt.
2. Unterstützung von Raytracing und Ebenen (optional).
3. Unterstützung für `rotateref`, `extnode` und `position=start|end`.

9.2.1. Die Zeichnung der Komponente

Sie müssen ein Makro `\...@comp` definieren, welches die Zeichnung enthält. In den folgenden Beispielen sehen Sie die Ausrichtung und Position des Koordinatensystems für einen Zweipol und einen Dreipol.



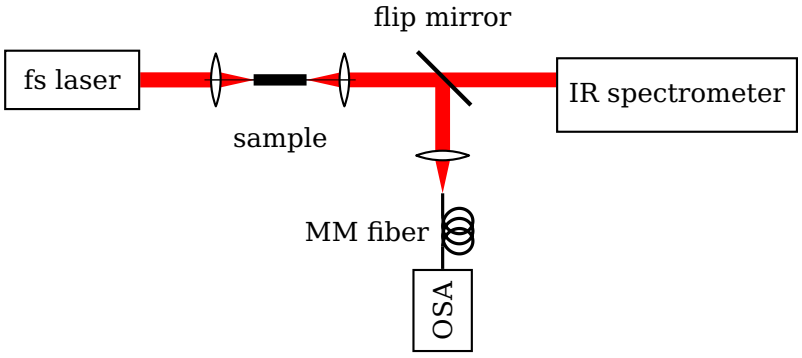

```
\begin{pspicture}(3.5,2.6)
\newOptexpDipole{mydipole}{}
\makeatletter
\def\mydipole@comp{%
  \psaxes[arrows=->](0,0)(0,-1.1)(1.3,1.3)[$x$,90][$y$,0]
}%
\makeatother
\mydipole(0,1)(3.5,1){\color{spot}label}
\drawbeam{(0,1)}{}{(3.5,1)}
\end{pspicture}
```




```
\begin{pspicture}(3.5,3.7)
\newOptexpTripole{mytripole}{labelangle=-60}
\makeatletter
\def\mytripole@comp{%
  \psaxes[arrows=->](0,0)(-1.1,0)(1.3,1.3)[$x$,90][$y$,0]
}%
\makeatother
\mytripole(0,1)(2,1.7)(3,0){\color{spot}label}
\drawbeam{(0,1)}{}{(3,0)}
\end{pspicture}
```

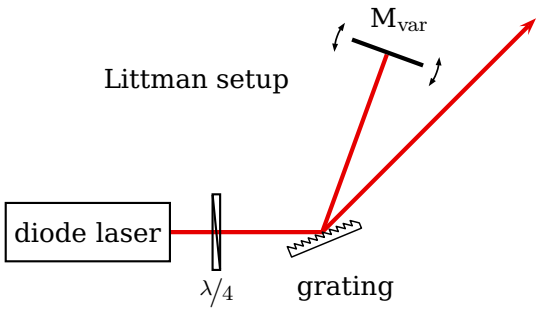
10. Beispiele

Bsp. 10.1



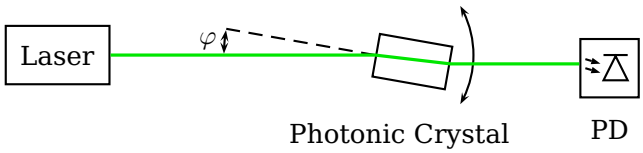

```
\begin{pspicture}(0.1,0.4)(10,4.5)
  \psset[optexp]{lens=1 1 0.7, loadbeampoints}
  \pnode(2,4){L}\pnode([Xnodesep=4]L){M}
  \pnode([Xnodesep=1.5]M){IRSpec}\pnode([offset=-2.5]M){OSA}
  \begin{optexp}
  \optbox[position=start, innerlabel, optboxwidth=1.8](L)(M){fs laser}
  \lens[abspos=1](L)(M)
  \optbox[abspos=1.85, optboxsize=0.7 0.15, fillstyle=solid,
      fillcolor=black](L)(M){sample}
  \lens[abspos=2.7](L)(M)
  \mirror[labelangle=45](L)(M)(OSA){flip mirror}
  \optbox[position=end, optboxsize=3.2 1, innerlabel,
      compshift=-0.2](M)(IRSpec){IR spectrometer}
  \lens[abspos=1](M)(OSA)
  \optplane([offset=-1.5]M)
  \optfiber[fiberloopradius=0.2, fiberloopsep=0.2, label=1,
      newFiber={linewidth=1.5\pslinewidth}]([offset=-1.5]M)(OSA){MM fiber}
  \optbox[position=end, innerlabel, optboxwidth=1.1](M)(OSA){OSA}
  \newpsstyle{Beam}{linestyle=none, fillcolor=red, fillstyle=solid, raytrace=false}
  \drawwidebeam[beamwidth=0.2, stopinside]{1-2}
  \drawwidebeam[beamdiv=-25]{2-3}
  \drawwidebeam[loadbeampoints=false, beamdiv=25]{3-4}
  \drawwidebeam[startinside]{4-5}
  \drawwidebeam[savebeampoints=false, beamalign=abs]{5-6}
  \drawwidebeam[stopinside]{5}{7}
  \drawwidebeam[beamdiv=-25]{7-8}
\end{optexp}
\end{pspicture}
```

Bsp. 10.2



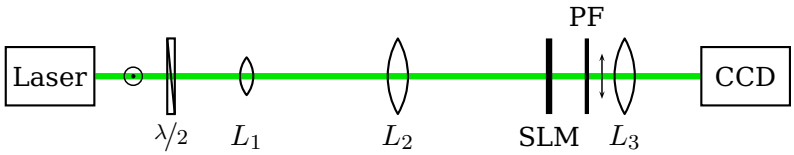

```
\begin{pspicture}(-4.2,-1)(3,3)
\pnode(-2,0){LOut}\pnode(0,0){Grat}\pnode(4;45){Out}\pnode(2.5;70){Mvar}
\newpsstyle{Beam}{linewidth=2\pslinewidth, linecolor=red!90!black}
\begin{optexp}
  \optbox[optboxwidth=2.2, labeloffset=0, position=start](LOut)(Grat){diode laser}
  \mirror[variable, label=0.5](Grat)(Mvar)(Grat){M$_\mathrm{var}$}
  \optretplate[position=0.3](LOut)(Grat){$\nicefrac{\lambda}{4}$}
  \optgrating(LOut)(Grat)(Out){grating}
  \drawbeam[arrows=->]{1}{3}{4}{(Out)}
  \drawbeam{2}{4}
\end{optexp}
\rput[l](-3,2){Littman setup}
\end{pspicture}
```

Bsp. 10.3



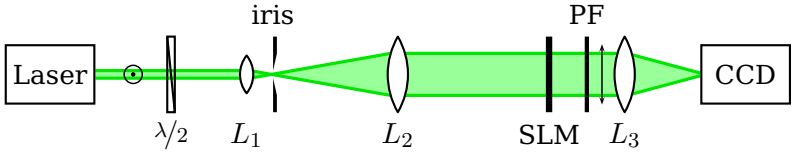

```
\begin{pspicture}(8.5,1.9)
\pnode(1.4,1.3){Laser}\pnode(7.6,1.3){Diode}
\optbox[position=start, labeloffset=0](Laser)(Diode){Laser}%
\optbox[abspos=4, optboxsize=1 0.6, labeloffset=1, n=3,
       compname=PC, angle=-10, rotateref=l](Laser)(Diode){Photonic Crystal}
\optdetector[dettype=diode](\oenodeOut{PC})(Diode|\oenodeOut{PC}){PD}
\nodexn{(\oenodeIn{PC}) + (2;170)}{Angle1}
\psline[linestyle=dashed](\oenodeIn{PC})(Angle1)
\psarc{<->}(\oenodeIn{PC}){1.3}{330}{30}
\psarc[arcsep=1pt]{<->}(\oenodeIn{PC}){2}{170}{180}
\uput{2.1}[175](\oenodeIn{PC}){\small $\varphi$}
\drawbeam{-}
\end{pspicture}
```

Bsp. 10.4



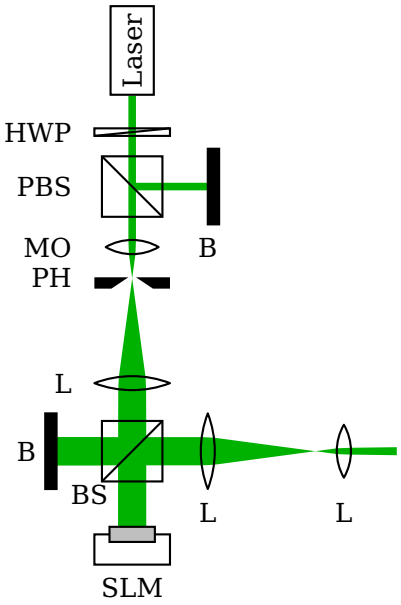

```
\begin{pspicture}(10.4,1.7)
\pnode(1.2,1){Start}\pnode(9.2,1){CCD}
\begin{optexp}
\optbox[position=end, label=0, optboxwidth=1.2](CCD)(Start){Laser}
\optbox[position=end, label=0, optboxwidth=1.2](Start)(CCD){CCD}
\polarization[poltype=perp,abspos=0.5](Start)(CCD)
\optretplate[abspos=1](Start)(CCD){$\nicefrac{\lambda}{2}$}
\lens[lens=0.4 0.4 0.5,abspos=2](Start)(CCD){$L_1$}
\lens[abspos=4](Start)(CCD){$L_2$}
\optplate[abspos=6, platelinewidth=3\pslinewidth](Start)(CCD){SLM}
\optplate[abspos=6.5, labelangle=180](Start)(CCD){PF}
\polarization[abspos=6.7](Start)(CCD)
\lens[abspos=7](Start)(CCD){$L_3$}
\drawbeam[linewidth=2\pslinewidth]{1}{2}
\end{optexp}
\end{pspicture}
```

Bsp. 10.5



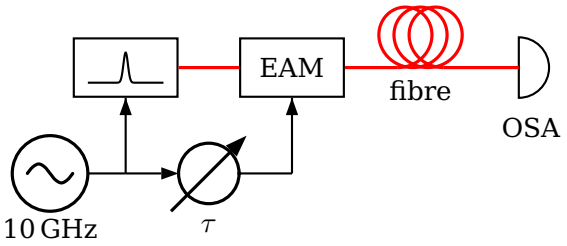

```
\begin{pspicture}(10.4,1.7)
\pnode(1.2,1){Start}\pnode(9.2,1){CCD}
\begin{optexp}
\optbox[position=start, label=0, optboxwidth=1.2](Start)(CCD){Laser}
\polarization[poltype=perp,abspos=0.5](Start)(CCD)
\optretplate[abspos=1](Start)(CCD){$\nicefrac{\lambda}{2}$}
\lens[lens=0.4 0.4 0.5,abspos=2](Start)(CCD){$L_1$}
\pinhole[phwidth=0.05, abspos=2.35, labelangle=180](Start)(CCD){iris}
\lens[abspos=4](Start)(CCD){$L_2$}
\optplate[abspos=6,platelinewidth=3\pslinewidth](Start)(CCD){SLM}
\optplate[abspos=6.5,labelangle=180](Start)(CCD){PF}
\lens[abspos=7](Start)(CCD){$L_3$}
\optbox[position=end, label=0, optboxwidth=1.2](Start)(CCD){CCD}
\polarization[abspos=6.7, polsize=0.8](Start)(CCD)
\addtopsstyle{Beam}{fillstyle=solid, fillcolor=green!40!white, loadbeampoints}
\drawwidebeam[beamwidth=0.1, stopinside]{1}{4}
\drawwidebeam[beamdiv=-20]{4}{6}
\drawwidebeam[stopinside]{6-9}
\drawwidebeam[beamdiv=-30]{9-10}
\end{optexp}
\end{pspicture}
```

Bsp. 10.6



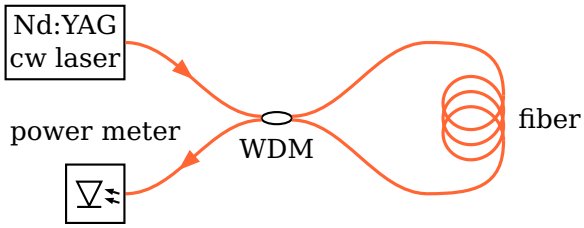

```
\begin{pspicture}(0.3,0.6)(5.5,7.8)
\newOptexpTripole{reflslm}{}
\makeatletter
\def\reflslm@comp{%
  \psframe[fillstyle=solid,fillcolor=gray!50](-0.3,0)(0.3,0.2)
  \psline(-0.3,0.1)(-0.5,0.1)(-0.5,0.5)(0.5,0.5)(0.5,0.1)(0.3,0.1)}%
\makeatother
\pnode(2,7){L}\pnode(2,5.8){Pbs}\pnode([Xnodesep=1]Pbs){Blk1}\pnode(2,2.3){Bs}
\pnode([Xnodesep=-1]Bs){Blk2}\pnode([Xnodesep=3.5]Bs){Out}\pnode(2,1.3){Slm}
\begin{optexp}
  \optbox[position=end, innerlabel, optboxsize=1.2 0.6](Pbs)(L){Laser}
  \psset{labelalign=r, mirrortype=extended}
  \newpsstyle{ExtendedMirror}{fillstyle=solid, fillcolor=black}
  \optretplate[position=0.4](L)(Pbs){HWP}
  \beamsplitter[labelangle=-90](L)(Pbs)(Blk1){PBS}
  \mirror[label=. -90 c](Pbs)(Blk1)(Pbs){B}
  \lens[abspos=0.8, lens=0.7 0.7 0.7, n=2](Pbs)(Bs){MO}
  \pinhole[phwidth=0.15, abspos=1.2](Pbs)(Bs){PH}
  \lens[abspos=2.6, lensradius=1.4](Pbs)(Bs){L}
  \psset{labelalign=c}
  \beamsplitter[labelangle=-135](Pbs)(Bs)(Blk2){BS}
  \mirror[labeloffset=0.4](Bs)(Blk2)(Bs){B}
  \lens[abspos=1, lensradius=1.4](Bs)(Out){L}
  \lens[abspos=2.8, lens=0.7 0.7 0.7, n=2](Bs)(Out){L}
  \reflslm(Bs)(Slm)(Bs){SLM}
  \addtopsstyle{Beam}{linestyle=none, fillstyle=solid, fillcolor=green!70!black}
  \drawwidebeam[beamwidth=0.1]{1-4}
  \drawwidebeam[beamwidth=0.1, beaminsidefirst, beaminsidelast]{3}{5-8}
  \psset{savebeampoints=false, loadbeampoints}
  \drawwidebeam{8-11}{(Out)}\drawwidebeam[beaminsidefirst]{8}{12}
\end{optexp}
\end{pspicture}
```

Bsp. 10.7



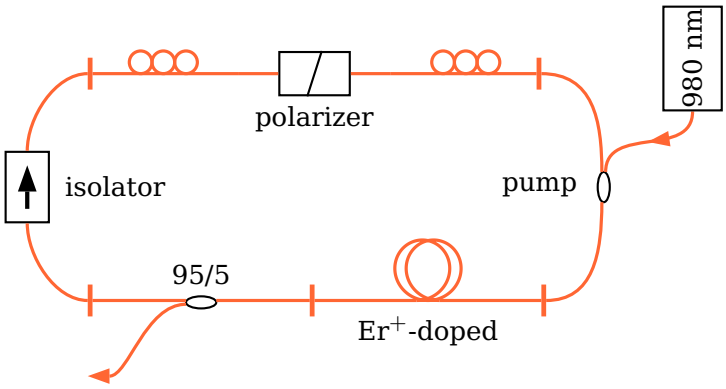

```
\begin{pspicture}(6.4,3.2)
\addtopsstyle{Fiber}{linecolor=red}\psset{fiber=none}
\pnode(2.3,2.3){Lin}\pnode([Xnodesep=4.5]Lin){Det}
\optbox[label=0.2, position=start, compname=L, extnode=b](Lin)(Det){%
  \psGauss[yunit=0.03,sigma=0.03]{-0.5}{0.5}}
\optbox[label=0, compname=EAM, extnode=b, abspos=1.5](Lin)(Det){EAM}
\optfiber[labeloffset=0.3, abspos=3.2](Lin)(Det){fibre}
\optdetector(Lin)(Det){OSA}
\pnode([Xnodesep=-1,offset=-1]\oenodeExt{L}){Osc}
\pnode(\oenodeExt{L}|Osc){PSin}\pnode(\oenodeExt{EAM}|Osc){PSout}
\oscillator[output=right](Osc){10\,GHz}{}
\phaseshifter[arrowscale=1.5, inputarrow, labeloffset=-0.7](PSin)(PSout){$\tau$}
\drawfiber{1}{2}{3}{4}
\psset{arrows=->, arrowinset=0, arrowscale=1.5}
\wire(PSin)(\oenodeExt{L})\wire(PSout)(\oenodeExt{EAM})
\end{pspicture}
```

Bsp. 10.8



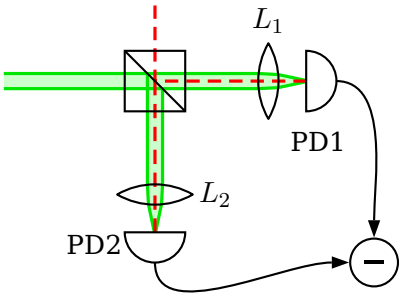

```
\begin{pspicture}(0.4,0.6)(8.2,3.5)
\pnode(2,3){Laser}\pnode(2,1){PwMeter}\pnode(6,3){CplTop}\pnode(6,1){CplBot}
\psset{arrowscale=1.5, arrowinset=0}
\optbox[position=start, optboxsize=1.6 1,
    labeloffset=0](Laser)([Xnodesep=0.1]Laser){%
  \begin{tabular}{@{}c@{}}Nd:YAG\\[-0.4ex]cw laser\end{tabular}}
\optcoupler[addtoFiberIn1={ArrowInside=->}, addtoFiberIn2={ArrowInside=-<},
          labeloffset=0.4](Laser)(PwMeter)(CplTop)(CplBot){WDM}
\optfiber[addtoFiberOut={ncurv=1, angleB=0}, addtoFiberIn={ncurv=1, angleA=0},
        compshift=-1, label=0.2 . l](CplBot)(CplTop){fiber}
\optdetector[dettype=diode]([Xnodesep=0.1]PwMeter)(PwMeter){power meter}
\end{pspicture}
```

Bsp. 10.9



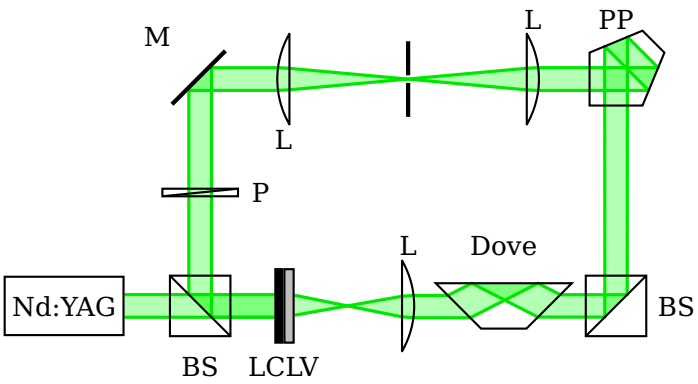

```
\begin{pspicture}(0.9,0.9)(10.4,5.9)
\psset{arrowscale=1.5, arrowinset=0}
\pnode(2,5){PC1in}\pnode(4,5){PC1out}\pnode(6,5){PC2in}
\pnode(8,5){PC2out}\pnode(2,2){CplSig}\pnode(5,2){CplIn}
\pnode(2,1){CplOut}\pnode(10,4.5){Pump}\pnode(8,2){PumpSig}
\optisolator[compshift=0.8, addtoFiberIn={angleA=180},
           addtoFiberOut={angleB=180}, label=0.5 . l]%
           (CplSig)(PC1in){isolator}
\polcontrol[addtoFiberIn={arrows=|-}](PC1in)(PC1out)
\optfiberpolarizer[labeloffset=0.6](PC1out)(PC2in){polarizer}
\polcontrol[addtoFiberOut={arrows=-|}](PC2in)(PC2out)
\wdmsplitter[labeloffset=0.3, coupleralign=bottom, addtoFiberIn={arrows=|-},
           addtoFiberOut1={arrows=->}, addtoFiberOut2={arrows=-|}]%
           (CplIn)(CplOut)(CplSig){95/5}
\wdmcoupler[addtoFiberIn1={ArrowInside=->}, addtoFiberIn2={angleA=0},
          addtoFiberOut={angleB=0,arrows=-|}, ncurv=0.9,
          coupleralign=bottom, compshift=0.8](Pump)(PC2out)(PumpSig){pump}
\optbox[position=start, innerlabel](Pump)([offset=-0.1]Pump){980~nm}
\optfiber[fiberloops=2, labeloffset=0.4](CplIn)(PumpSig){Er$^+$-doped}
\end{pspicture}
```

Bsp. 10.10



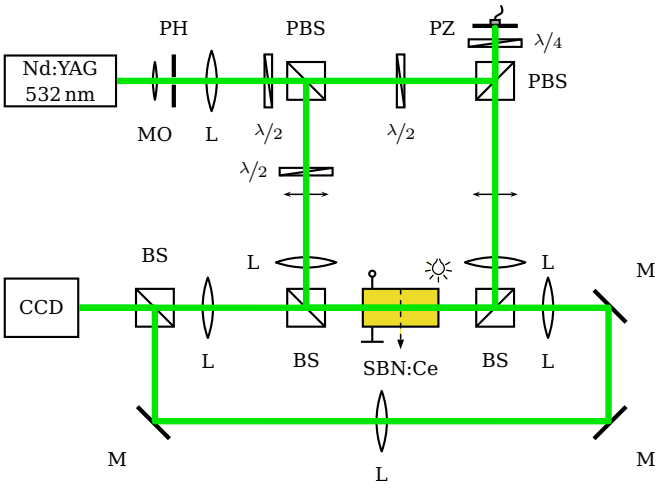

```
\begin{pspicture}(1,1.2)(6.2,5)
\pnode(1,4){SigIn}\pnode(3,4){BS}\pnode(3,5){LO}
\pnode(5,4){Det1}\pnode(3,2){Det2}
\begin{optexp}
  \optplane[compname=LO, angle=90](LO)
  \beamsplitter[compname=BS](SigIn)(BS)(Det2)
  \lens[abspos=0.5, compname=L2, n=2.1](Det2)(BS){$L_2$}
  \lens[abspos=0.5, compname=L1, n=2.1](Det1)(BS){$L_1$}
  \psset[optexp]{extnodealign=rel, extnode=r}
  \optdetector[compname=Det1](BS)(Det1){PD1}
  \optdetector[compname=Det2](BS)(Det2){PD2}
  \addtopsstyle{Beam}{beamwidth=0.2, fillstyle=solid,
                      fillcolor=green, opacity=0.2}
  \drawwidebeam{(SigIn)}{BS}{L2}{Det2}
  \drawwidebeam[beaminsidefirst]{BS}{L1}{Det1}
  \newpsstyle{Beam}{linecolor=red, linestyle=dashed, linewidth=1.5\pslinewidth}
  \drawbeam{LO}{BS}{L1}{Det1}
  \drawbeam[beaminsidefirst]{BS}{L2}{Det2}
\end{optexp}
\cnodeput[framesep=5pt]([Xnodesep=0.5,
    offset=-0.5]\oenodeExt{Det1}|\oenodeExt{Det2}){M}{\rule{2.5mm}{1pt}}
\psset{arrows=<-, arrowscale=1.5, arrowinset=0}
\nccurve[angleA=90]{M}{\oenodeExt{Det1}}
\nccurve[angleA=180, angleB=-90]{M}{\oenodeExt{Det2}}
\end{pspicture}
```

Bsp. 10.11



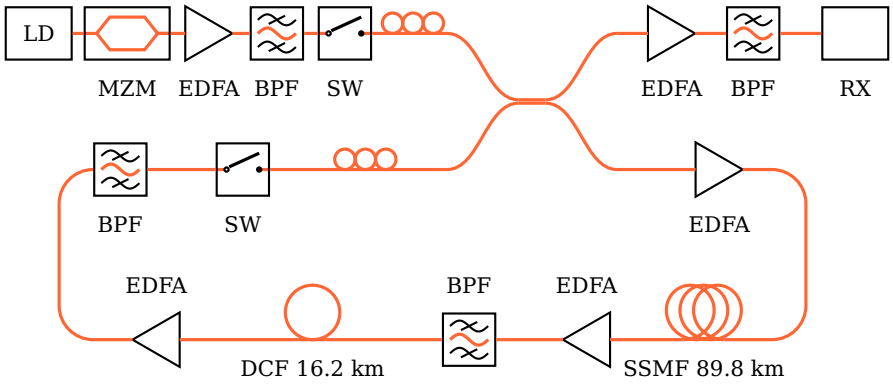

```
\begin{pspicture}(-0.2,0)(9,5)
\makeatletter
\newOptexpDipole{LCLV}{}
\def\LCLV@nodes{%
  \newOptexpComp{ {-0.12 0} {0 1} -0.5 0.5 trans {PlainIfc}
                  {0.15 0} {0 1} -0.5 0.5 trans {PlainIfc} \POE@key@n}%
}%
\def\LCLV@comp{%
  \psframe[fillstyle=solid,fillcolor=black,dimen=outer](-0.12,-0.5)(0,0.5)
  \psframe[fillstyle=solid,fillcolor=gray!50,dimen=outer](0,-0.5)(0.15,0.5)
}%
\makeatother
\psset[optexp]{lens=1.2 0 1.2, n=1.72}
\addtopsstyle{Beam}{fillstyle=solid, fillcolor=green, opacity=0.3}
\pnode(2.4,1){BS1}\pnode([offset=3]BS1){M1}
\pnode([Xnodesep=5.5]M1){PP}\pnode(PP|BS1){BS2}
\begin{optexp}
  \optbox[label=0, position=start, optboxwidth=1.6]([Xnodesep=-1]BS1)(BS1){Nd:YAG}
  \beamsplitter[compname=BS](BS2)(BS1)(M1){BS}
  \LCLV[position=0.2, compname=LCLV](BS1)(BS2){LCLV}
  \optretplate(BS1)(M1){P}
  \mirror(BS1)(M1)(PP){M}
  \lens[position=0.2](M1)(PP){L}
  \pinhole(M1)(PP)
  \lens[position=0.2](PP)(M1){L}
  \pentaprism(M1)(PP)(BS2){PP}
  \beamsplitter(PP)(BS2)(BS1){BS}
  \doveprism[compname=Dove, position=0.27, n=2.3](BS2)(BS1){Dove}
  \lens[n=2.5](BS2)(BS1){L}
  \drawwidebeam[beamwidth=0.3]{1-3}
  \drawwidebeam[loadbeampoints]{3}{2}{4-}{3}
\end{optexp}
\end{pspicture}
```

Bsp. 10.12



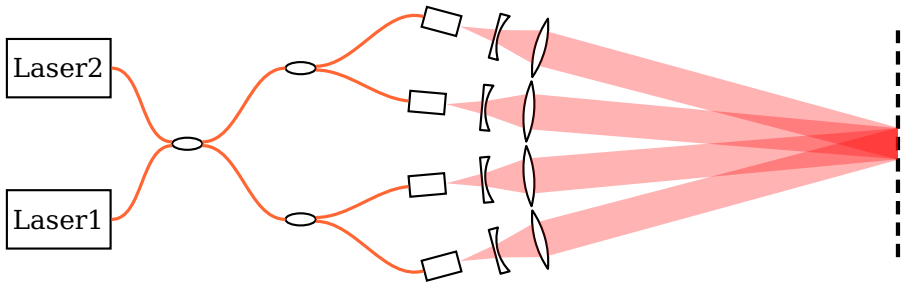

```
\begin{pspicture}(0,-0.2)(8.6,5.6)
\pnode(1.5,5){Laser}\pnode(4,5){PBS}\pnode(6.5,5){PBS2}\pnode(6.5,5.7){piezo}
\pnode(4,2){BSFwd}\pnode(6.5,2){BSBwd}\pnode(2,2){BS4f}\pnode(2,0.5){M4f3}
\pnode(8,2){M4f1}\pnode(8,0.5){M4f2}\pnode(1,2){CCD}
\psset{mirrorwidth=0.6, plateheight=0.7, outerheight=0.7, labeloffset=0.7,
    labelstyle=\scriptsize, lens=1.2 1.2 0.8, bssize=0.5}
\optbox[position=start, optboxsize=1.5 0.7, innerlabel]%
  (Laser)(PBS){\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}
\crystal[crystalsize=1 0.5, voltage, lamp, fillstyle=solid,
    fillcolor=yellow!90!black, labeloffset=0.8](BSFwd)(BSBwd){SBN:Ce}
\beamsplitter(PBS)(BSFwd)(BSBwd){BS}
\beamsplitter[labelangle=-90](PBS2)(BSBwd)(BSFwd){BS}
\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[position=end, label=0, optboxwidth=1](BS4f)(CCD){CCD}
\lens[abspos=0.7](BS4f)(BSFwd){L}\lens[abspos=0.7](BSBwd)(M4f1){L}
\addtopsstyle{Beam}{linewidth=2\pslinewidth}
\drawbeam{1-6}{11-14}{13}{12}{15}{18}{16}{17}{25}{23}{24}
\drawbeam[beaminsidefirst]{6}{7}{10}{17}{16}{18}{26}{19-24}
\end{pspicture}
```

Bsp. 10.13



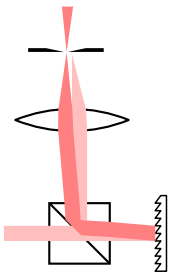

```
\begin{pspicture}(13,5.5)
\psset{usefiberstyle, optboxwidth=1, labelstyle=\footnotesize, fiber=none}
\pnode(1,5){LD}\pnode([Xnodesep=5.5]LD){CPLin1}
\pnode([offset=-2]CPLin1){CPLin2}\pnode([Xnodesep=2.5]CPLin1){CPLout1}
\pnode([Xnodesep=2.5]CPLin2){CPLout2}\pnode([Xnodesep=3]CPLout1){RX}
\optbox[position=start, label=0](LD)(CPLin1){LD}
\optmzm[abspos=0.8](LD)(CPLin1){MZM}
\optamp[abspos=2](LD)(CPLin1){EDFA}
\optfilter[abspos=3](LD)(CPLin1){BPF}
\optswitch[abspos=4](LD)(CPLin1){SW}
\polcontrol[abspos=5, fiber=out](LD)(CPLin1)
\drawfiber{1}{2}{3}{4}{5}{6}
\optcoupler[couplertype=none, fiber](CPLin1)(CPLin2)(CPLout1)(CPLout2)
\optamp[abspos=0.8, fiber=in](CPLout1)(RX){EDFA}
\optfilter[abspos=2](CPLout1)(RX){BPF}
\optbox[position=end](CPLout1)(RX){RX}
\drawfiber{8}{9}{10}
\pnode([Xnodesep=-0.5]LD|CPLin2){TL}\pnode(RX|TL){TR}
\pnode([offset=-2.5]TR){BR}\pnode(TL|BR){BL}
\optamp[fiber=i](CPLout2)(TR){EDFA}
\optfiber[label=0.3 . t, position=0.85](BL)(BR){SSMF 89.8~km}
\drawfiber[fiberstyle=angle, arm=1.2, linearc=0.5, startnode=N]{11}{12}
\optamp[position=0.3](BR)(BL){EDFA}
\optfilter[position=0.55, labelangle=180](BL)(BR){BPF}
\optfiber[fiberloops=1, label=0.3 . t, position=0.35](BL)(BR){DCF 16.2~km}
\optamp[position=0.85](BR)(BL){EDFA}
\optfilter[position=0.2](TL)(CPLin2){BPF}
\optswitch(TL)(CPLin2){SW}
\polcontrol[position=0.8, fiber=out](TL)(CPLin2)
\drawfiber{12}{13}{14}{15}{16}\drawfiber{17}{18}{19}
\drawfiber[fiberstyle=angle, arm=0.5, linearc=0.5, stopnode=1]{16}{17}
\end{pspicture}
```

Bsp. 10.14



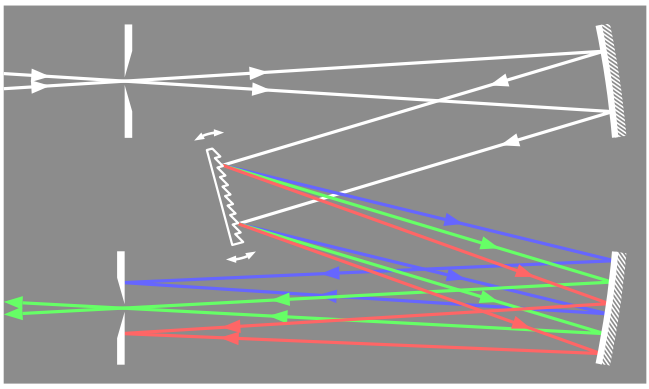

```
\begin{pspicture}(12, 4)
  \psset{labeloffset=0.3, fiber=none}
  \pnode(1.6,2){In}\pnode(12, 2){Ref}
  \optplate[linestyle=dashed, plateheight=3, position=1, compname=RefPlane](In)(Ref)
  \addtopsstyle{Beam}{linestyle=none, beamdiv=20, beaminside=false,
      fillstyle=solid, fillcolor=red, opacity=0.3}
  \multido{\i=1+1, \ii=165+10}{4}{%
    \rput(Ref){\pnode(6;\ii){A\i}}
    \optbox[optboxsize=0.5 0.3, position=end, compname=Box\i, extnode=l](Ref)(A\i)
    \lens[lens=0 -0.6 0.6, abspos=0.5, compname=L\i](A\i)(Ref)
    \lens[lens=1.2 1.2 0.8, abspos=1.1, n=1.65, compname=LC\i](A\i)(Ref)
    \drawwidebeam{Box\i}{L\i}{LC\i}{RefPlane}
  }%
  \optcoupler[abspos=1](In)(In)(Ref)(Ref)
  \wdmsplitter[abspos=2.5, compshift=1](In)(Ref)(Ref)
  \wdmsplitter[abspos=2.5, compshift=-1](In)(Ref)(Ref)
  \optbox[compshift=-1, position=start, label=0](In)(Ref){Laser1}
  \optbox[compshift=1, position=start, label=0](In)(Ref){Laser2}
  \drawfiber{17}{14}{15}{Box1}\drawfiber{15}{Box2}
  \drawfiber{18}{14}{16}{Box3}\drawfiber{16}{Box4}
\end{pspicture}
```

Bsp. 10.15



```
\begin{pspicture}(0.5,0)(2.2,4)
  \pnode(0,1){A}\pnode(1,1){BS}\pnode(2,1){G}\pnode(1,4){B}
  \beamsplitter(G)(BS)(B)
  \optgrating(BS)(G)(BS)
  \lens[lens=2 2 1.5, compshift=0.1, n=2.25](BS)(B)
  \pinhole[phwidth=0.05, innerheight=0.05, position=0.8,
      compshift=0.18](BS)(B)
  \optplane[angle=90](B)
  \addtopsstyle{Beam}{linestyle=none, beamwidth=0.2, fillstyle=solid}
  \drawwidebeam[fillcolor=red!25!white]{(A)}{1-2}{1}{3-5}
  \drawwidebeam[fillcolor=red!50!white, beamangle=-5]{2}{1}{3-5}
\end{pspicture}
```

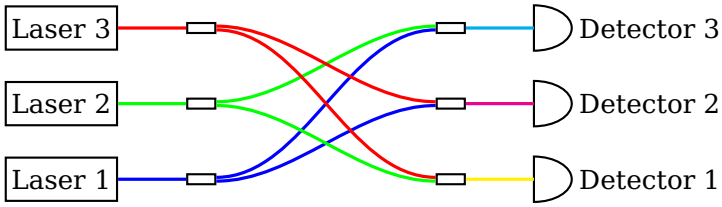
Bsp. 10.16



Angepasst von http://de.wikipedia.org/w/index.php?title=Datei:Czerny-turner.png

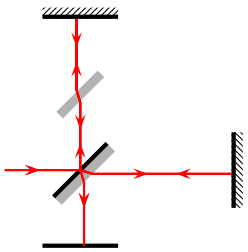
```
\begin{pspicture}(0,-1.5)(8,4)
\psframe[fillstyle=solid,fillcolor=gray!90,linestyle=none](0,-1)(8.5,4)
\addtopsstyle{OptComp}{linecolor=white}
\newpsstyle{Beam}{ArrowInside=->, linejoin=2, arrowscale=1.3, arrowinset=0}
\addtopsstyle{ExtendedMirror}{hatchcolor=white, hatchsep=0.5\pslinewidth}
\pnode(0,3){A}\pnode(8,3){B}\pnode(3,1.5){C}\pnode(8,0){D}\pnode(0,0){E}%
\psset{linewidth=1.5\pslinewidth, mirrorradius=13, mirrorwidth=1.5,
    gratingwidth=1.3, mirrortype=extended, phwidth=0.1, outerheight=1.5}%
\begin{optexp}
  \pinhole[position=0.2](A)(B)%
  \mirror(A)(B)(C)
  \optgrating[reverse, angle=15, variable](B)(C)(D)
  \mirror(C)(D)(E)
  \pinhole[position=0.8](D)(E)
  \drawwidebeam[ArrowInsidePos=0.3, beamwidth=0.2, linecolor=white,
      beamdiv=-7.15]{(A)}{1-3}
  \psset{loadbeampoints, savebeampoints=false}%
  \drawwidebeam[ArrowInsidePos=0.6, beamangle=3, linecolor=blue!60!white]{3-6}
  \drawwidebeam[ArrowInsidePos=0.7, linecolor=green!60!white, arrows=->,
      ArrowInsideMinLength=2]{3-5}{(E)}
  \drawwidebeam[ArrowInsidePos=0.8, beamangle=-3, linecolor=red!60!white]{3-6}
\end{optexp}
\end{pspicture}
```

Bsp. 10.17




```
\begin{pspicture}(0,0.7)(9.4,3.3)
  \psset[optexp]{couplertype=rectangle, fiber=none, detsize=0.5 0.6}
  \multido{\i=1+1}{3}{%
    \optbox[optboxsize=1.5 0.6, innerlabel,position=start](1.5,\i)(7,\i){Laser \i}
    \wdmsplitter[position=0.2](1.5,\i)(7,\i)(7,\i)
    \wdmcoupler[position=0.8](1.5,\i)(1.5,\i)(7,\i)
    \optdetector[label=0.4 90 l](1.5,\i)(7,\i){Detector \i}
  }%
  \drawfiber[linecolor=blue, startnode=N]{1}{2}{7}
  \drawfiber[linecolor=blue]{2}{11}
  \drawfiber[linecolor=green, stopnode=1]{5}{6}{11}
  \drawfiber[linecolor=green, stopnode=2]{6}{3}
  \drawfiber[linecolor=red]{9}{10}{3}
  \drawfiber[linecolor=red, startnode=2]{10}{7}
  \drawfiber[linecolor=cyan]{11}{12}
  \drawfiber[linecolor=magenta]{7}{8}
  \drawfiber[linecolor=yellow]{3}{4}
\end{pspicture}
```

Bsp. 10.18



```
\begin{pspicture}(0,1)(3.2,4.2)
  \pnode(0,2){In}\pnode(1,2){G}\pnode(1,4){M1}
  \pnode(3,2){M2}\pnode(1,1){S}
  \mirror[mirrortype=semitrans](In)(G)(M1)
  \optbox[optboxwidth=0.15, angle=45,
      style=SemitransMirror](G)(M1)
  \mirror[mirrortype=extended](G)(M1)(G)
  \mirror[mirrortype=extended](G)(M2)(G)
  \optplate[position=end](G)(S)
  \newpsstyle{Beam}{linecolor=red, ArrowInside=->,
      ArrowInsidePos=0.4, arrowscale=1.3}
  \drawbeam{(In)}{1}{4}{1}{5}
  \drawbeam{(In)}{1-3}{2}{1}{5}
\end{pspicture}
```

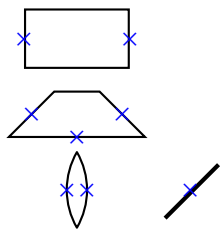
11. Zusatzinformationen

11.1. Interne Struktur der Komponenten

showifcnodes=true, false default: false

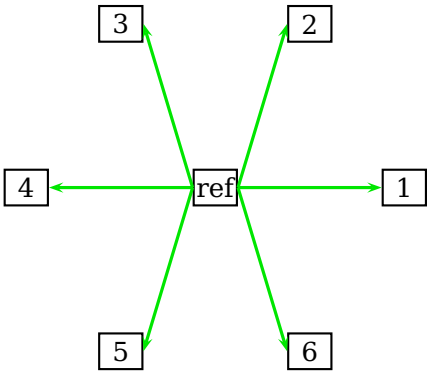
IfcNodeStyle⟨*psstyle*⟩ default: dotstyle=x, dotscale=1.5, linecolor=blue

Jede Komponente besteht aus Grenzflächen, die über einen Knoten auf der optischen Achse, einen Ebenevektor bzw. einen Krümmungsradius, und ihre optische Eigenschaft (reflektiv oder transmittiv, oder beides für z.B. \beamsplitter) definiert ist. Die Grenzflächenknoten können mit showifcnodes angezeigt werden.



```
\begin{pspicture}(4,3)
  \psset{showifcnodes}
  \optbox(0,2.5)(4,2.5)
  \doveprism(0,1.5)(4,1.5)
  \lens(0,0.5)(4,0.5)
  \mirror(0,0.5)(3.5,0.5)(3.5,3)
\end{pspicture}
```

\draw*beam berechnet den Abstand zwischen den Grenzflächenknoten zweier Komponenten und verbindet die beiden nächstgelegenen.




```
\begin{pspicture}(-3,-3)(3,3)
  \psset{optboxsize=0.6 0.5, labeloffset=0}
  \multido{\i=0+60,\ii=1+1}{6}{%
    \pnode(2.5;\i){A}
    \optbox([Xnodesep=-1]A)([Xnodesep=1]A){\ii}
  }
  \optbox[compname=ref](-1,0)(1,0){ref}
  \addtopsstyle{Beam}{arrows=->}
  \multido{\i=1+1}{6}{\drawbeam{ref}{\i}}
\end{pspicture}
```

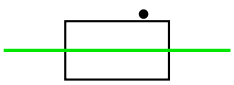
11.2. Übersicht der Spezialknoten

In diesem Abschnitt finden sie Übersichten für alle Spezialknoten, die von allen Komponenten bereitgestellt werden: die möglichen Positionen für externe und Rotationsknoten, und die Grenzflächenknoten.

11.2.1. Externe und Rotationsreferenzknoten

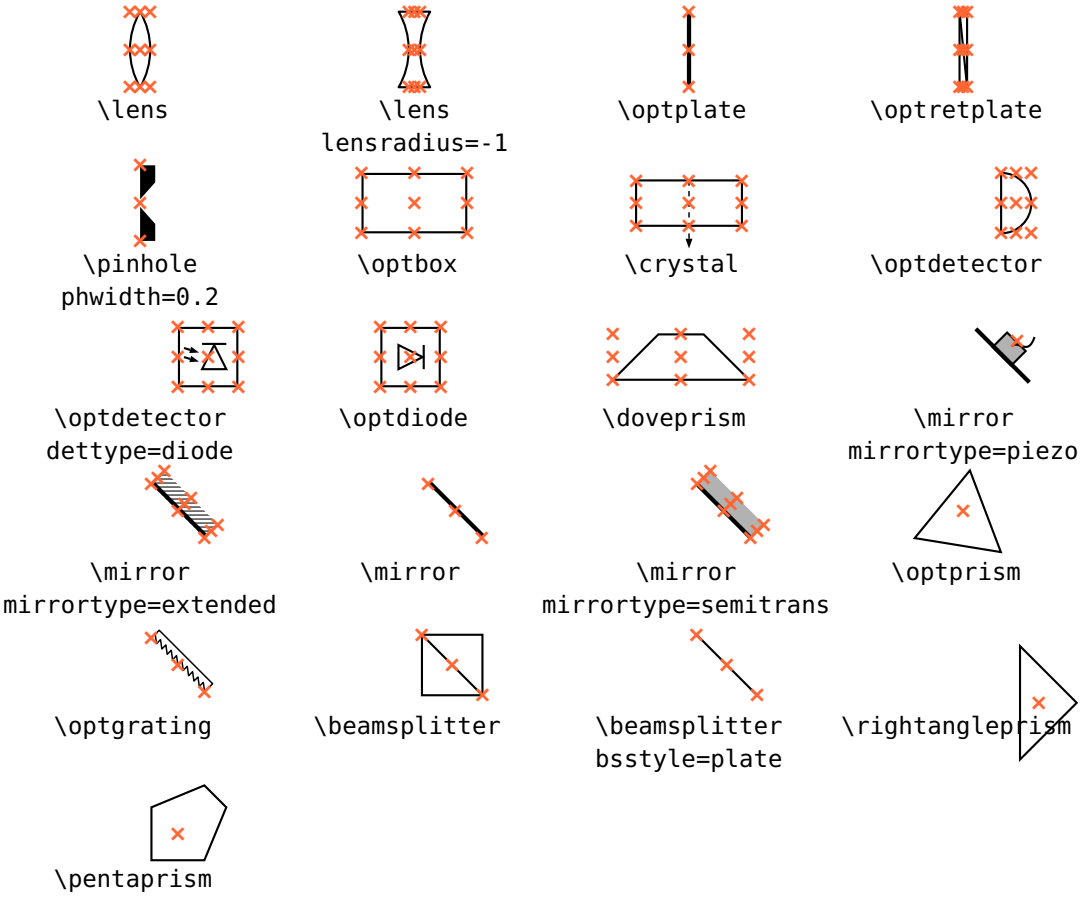
Hier finden Sie eine Übersicht über alle möglichen externen Knoten (Kap. 7.5) und Rotationsknoten (Kap. 7.7) aller Komponenten. Angezeigt werden die Positionen für alle möglichen Kombinationen von `t`, `c` oder `b` mit `l`, `c` oder `r`. Es sind nicht immer alle Möglichkeiten vorhanden, so können `t` und `b` ignoriert werden (z.B. bei `\mirror`), oder auch `r` und `l` (z.B. bei `\optplate`).

Anstelle dieser Werte können auch Bruchteile davon verwendet werden: `t` entspricht $+1 \cdot \Delta y$, `b` entspricht $-1 \cdot \Delta y$, `l` entspricht $-1 \cdot \Delta x$ und `r` entspricht $+1 \cdot \Delta x$. Im folgenden Beispiel wird der Punkt $(0.5\Delta x, 1.2\Delta y)$ als externer Knoten bereitgestellt.

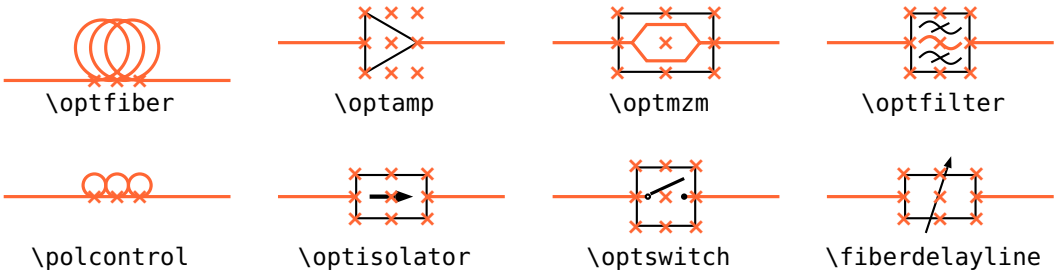


```
\begin{pspicture}(3,1)
  \optbox[extnode={0.5,1.2}, beam](0,0.5)(3,0.5)
  \psdot(\oenodeExt{})
\end{pspicture}
```

Freistrahlkomponenten



Faserkomponenten

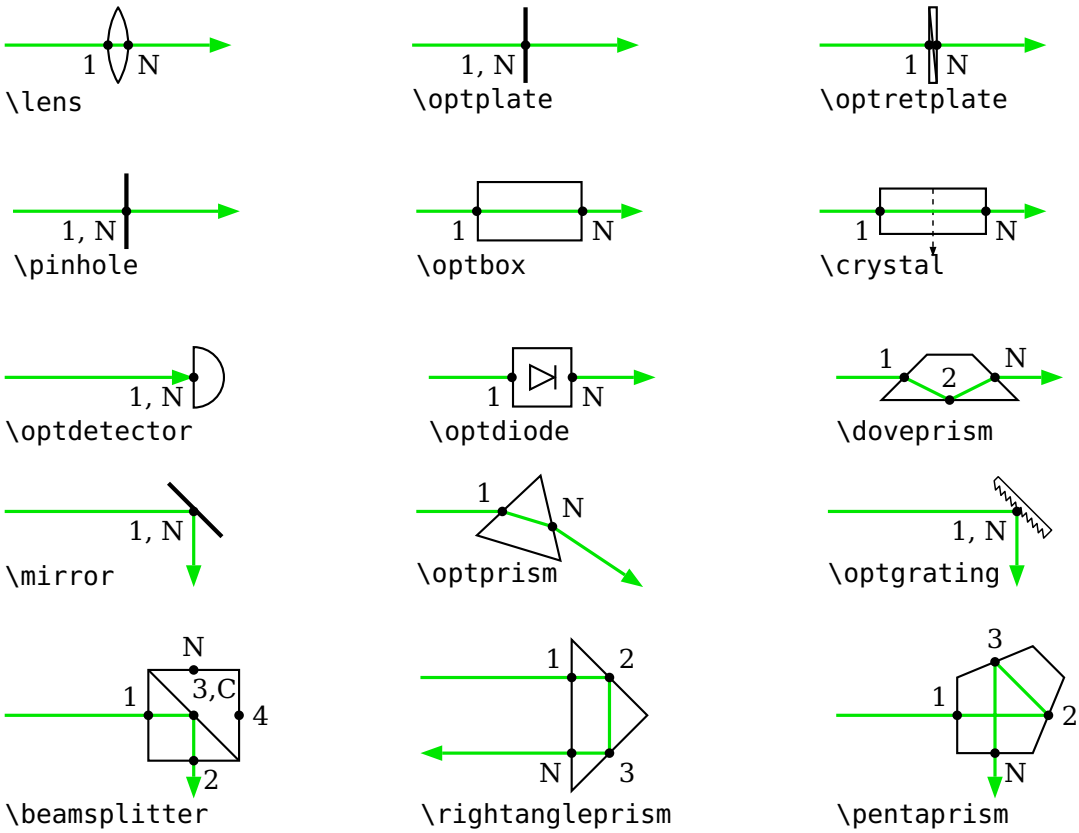


\optfiberpolarizer \fibercollimator \optcirculator \optcoupler couplersize=0.5 0.3

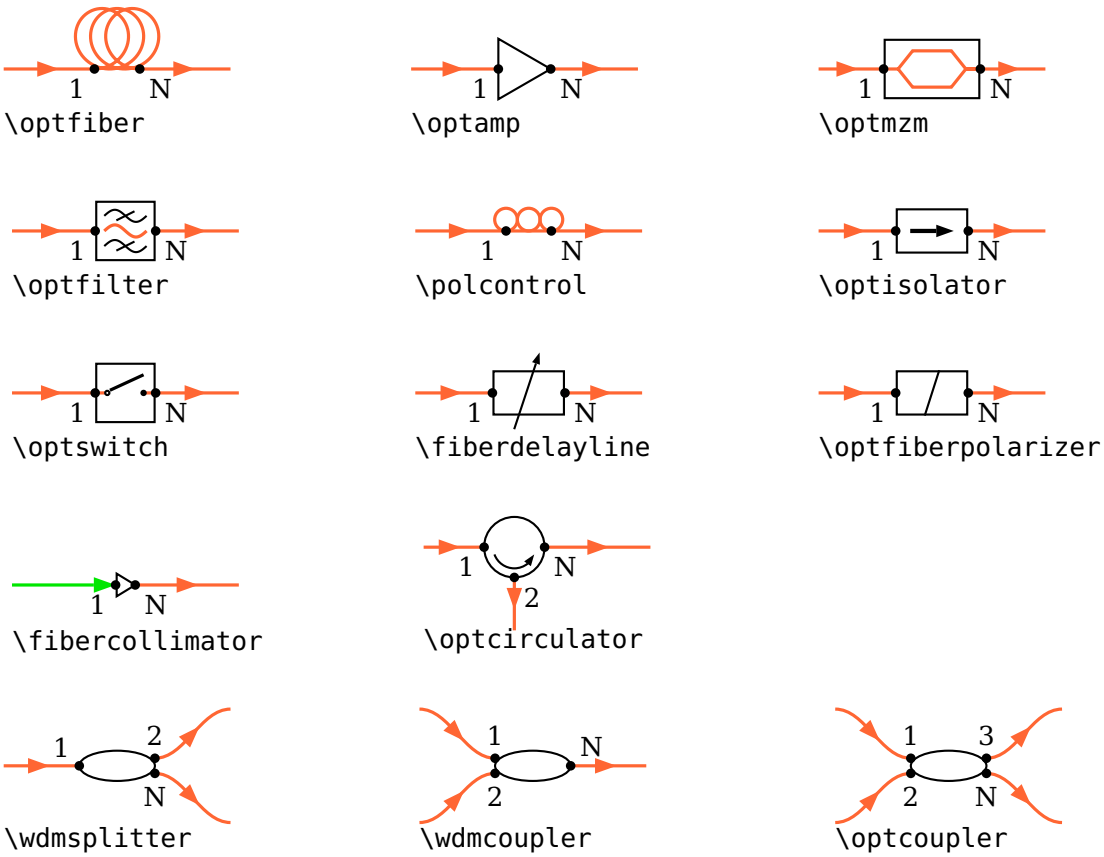
11.2.2. Grenzflächenknoten

Hier sind alle Komponenten und deren Grenzflächenknoten aufgelistet. Ist ein Knoten mit „1, N“ beschriftet, dann fallen beide Knoten zusammen.

Freistrahlkomponenten



Faserkomponenten



11.3. Abwärtskompatibilität

In diesem Abschnitt finden Sie eine umfangreiche Auflistung aller Änderungen in einer Version bezüglich der Vorgängerversion, die nicht abwärts kompatibel sind. Sie finden ebenfalls Informationen was Sie bei einem Wechsel zur neuen Version beachten müssen, und was sie bei älteren Dokumenten gegebenenfalls ändern müssen um mit der neuen Version korrekt zu funktionieren.

11.3.1. Version 3.0

Die Änderungen in Version 3.0 gegenüber Version 2.1, die nicht abwärtskompatibel sind.

Allgemeine Parameter

`namingscheme`=old, new default: `new`

In Version 2.1 mussten spezielle Komponentenknoten über ihren expliziten Namen angesprochen werden. Wenn Sie das alte Namensschema benötigen, da Sie direkt auf die Knoten zugegriffen haben, so müssen Sie `namingscheme=old` verwenden. Seit Version 3.0 werden Makros für den Zugriff auf die Komponentenknoten bereitgestellt, so dass das eigentlichen Namensschema unerheblich ist, siehe Kap. 7

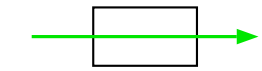
`endbox` `angle` `rotateref` Diese Parameter wirken sich jetzt auf alle Komponenten aus.

`extnode` Einigen Komponenten stellen nun mehr mögliche Positionen für `extnode` zur Verfügung. So muss war z.B. für `\optdetector` nur ein Knoten möglich, der für jeden Wert von `extnode` zugänglich war, jetzt aber nur mit `extnode=r`. Alle anderen Werte setzen den Knoten an andere Stellen.

Strahlverbindungen

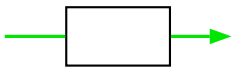
In Version 2.1 konnten Strahlverbindungen intern sein, d.h. die Verbindung wurde zusammen mit der Komponenten definiert (z.B. mit `beam`, oder `conn`) oder extern über einen separaten Aufruf von `\drawbeam`. Der Hauptgrund für die internen Verbindungen war der ästhetische Aspekt, zuerst die Verbindung und dann die Komponente darüber zu zeichnen, aber Bequemlichkeit spielte auch eine Rolle. Seit Version 3.0 kann die Reihenfolge von Verbindungen und Komponenten über Ebenen geregelt werden (Kap. 8.7).

Manche Varianten für interne Strahlverbindungen können auch in 3.0 verwendet werden:



`conn=o-i, conn=i-o`

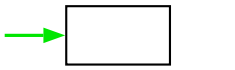
```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \optbox[beam](A)(B)
\end{pspicture}
```



conn=o-o

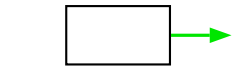
```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \optbox[beam, beaminside=false](A)(B)
\end{pspicture}
```

Andere Verbindungen können nur über einen zusätzlichen \drawbeam Aufruf erreicht werden, falls die Verbindung dann hinter der Komponente sein soll, musst der beteiligte Code in eine optexp-Umgebung geschachtelt werden. Außer diesem Aspekt sind die separaten Verbindungen äquivalent:



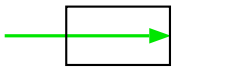
conn=o-, conn=a-

```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \optbox(A)(B)
  \drawbeam{(A)}{}
\end{pspicture}
```



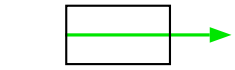
conn=-o, conn=-b

```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \optbox(A)(B)
  \drawbeam{}{(B)}
\end{pspicture}
```



conn=i-, A-

```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \begin{optexp}
    \optbox(A)(B)
    \drawbeam[beaminsidelast]{(A)}{}
  \end{optexp}
\end{pspicture}
```



conn=-i, conn=-B

```
\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \begin{optexp}
    \optbox(A)(B)
    \drawbeam[beaminsidefirst]{}{(B)}
  \end{optexp}
\end{pspicture}
```

Dokumentationsindex

A. Versionsgeschichte

Diese Versionsgeschichte ist eine Liste von Änderungen, die für den Nutzer des Pakets von Bedeutung sind. Änderungen, die eher technischer Natur sind und für den Nutzer des Pakets nicht relevant sind und das Verhalten des Pakets nicht ändern, werden nicht aufgeführt. Wenn ein Eintrag der Versionsgeschichte ein Feature als *improved* oder *extended* bekannt gibt, so bedeutet dies, dass eine Modifikation die Syntax und das Verhalten des Pakets nicht beeinflusst, oder das es für ältere Versionen kompatibel ist. Einträge, die als *deprecated*, *modified*, *renamed*, oder *removed* deklariert sind, verlangen besondere Aufmerksamkeit. Diese bedeuten, dass eine Modifikation Änderungen in existierenden Dokumenten mit sich ziehen kann. Die Zahlen an der rechten Seite stehen für die relevante Stelle dieser Dokumentation, zusätzliche Informationen zur Migration von älteren Dokumenten sind in Kap. 11.3 zu finden.

3.0 2012-07-09

Modified beam connections with `\drawbeam` to support raytracing 8.2
Added wide beams with `\drawwidebeam` 8.3
Added `\drawfiber` for fiber connections 8.6
Added `optexp` environment for layering of components and connections 8.7
Added german documentation
Modified naming of component nodes 7
Modified `extnode` to work with more components 7.5
Modified `angle` and `rotateref` to affect all components 3.3
Modified `endbox` to affect all components 3.2
Extended `position` by values `start` and `end` 3.2
Extended `abspos` by values `start` and `end` 3.2
Removed deprecated lens code which used `lenswidth` and `lensheight` for construction 4.1
Added option `platesize` 4.3
Added option `phwidth` 4.4
Modified option `caxislength` 4.6
Deprecated option `lampscale` 4.6

2.1 2009-11-05

2.0 2008-07-27

1.2 2008-06-17

1.1 2007-09-06

1.0 2007-07-18

First CTAN version