

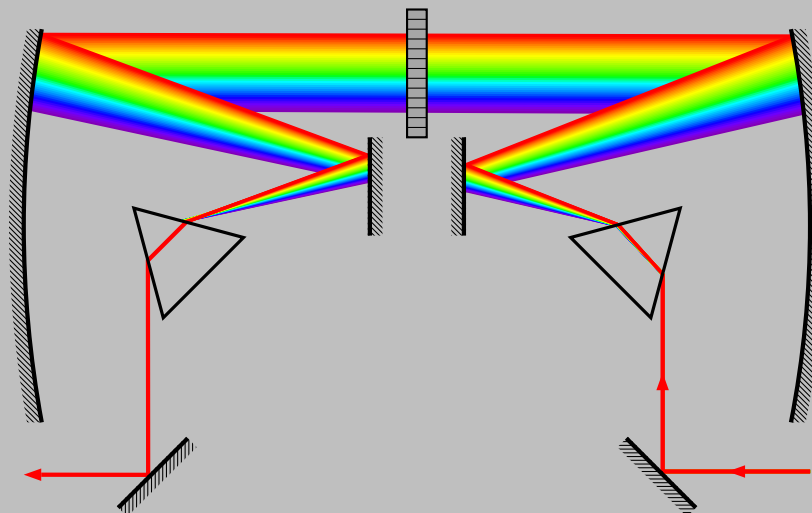
# PSTricks

pst-optexp

## Optische Versuchsaufbauten

v5.1

2014/11/19



Paketautor:  
**Christoph Bersch**

# Inhaltsverzeichnis

<b>1. Einführung</b>	<b>6</b>
1.1. Über das Paket . . . . .	6
1.2. Anforderungen . . . . .	6
1.3. Verbreitung und Installation . . . . .	6
1.4. Lizenz . . . . .	7
1.5. Abwärtskompatibilität . . . . .	7
1.6. Notation . . . . .	7
1.7. Danksagung . . . . .	8
<b>2. Grundlegende Ideen</b>	<b>9</b>
2.1. Die Komponenten . . . . .	9
2.2. Beschriftungen . . . . .	10
2.3. Freistrahl-Verbindungen . . . . .	10
2.4. Faserverbindungen . . . . .	12
2.5. Schritt-für-Schritt-Beispiele . . . . .	13
2.5.1. Freistrahlaufbau . . . . .	13
2.5.2. Galilei-Fernrohr . . . . .	13
2.5.3. Faseraufbau . . . . .	15
2.5.4. Faseraufbau (Alternative) . . . . .	17
2.5.5. Michelson-Interferometer . . . . .	18
<b>3. Allgemeine Komponentenparameter</b>	<b>19</b>
3.1. Beschriftungen . . . . .	19
3.2. Positionierung . . . . .	22
3.3. Drehen und Verschieben . . . . .	23
3.4. Verwendung von Stilen . . . . .	25
3.5. Aussehen der Komponenten . . . . .	25
<b>4. Freistrahlkompontenten</b>	<b>27</b>
4.1. Linse . . . . .	27
4.2. Optisches Plättchen . . . . .	29
4.3. Verzögerungsplättchen . . . . .	29
4.4. Lochblende . . . . .	30

4.5. Box . . . . .	31
4.6. Pfeilkomponente . . . . .	31
4.7. Balkenkomponente . . . . .	32
4.8. Optische Quelle . . . . .	33
4.9. Kristall . . . . .	34
4.10. Optische Diode . . . . .	36
4.11. Doveprisma . . . . .	36
4.12. Glan-Thompson-Prisma . . . . .	37
4.13. Polarisation . . . . .	38
4.14. Spiegel . . . . .	39
4.15. Parabolspiegel . . . . .	42
4.16. Off-axis-Parabolspiegel . . . . .	43
4.17. Strahlteiler . . . . .	44
4.18. Optisches Gitter . . . . .	45
4.19. Transmissionsgitter . . . . .	47
4.20. Akustooptischer Modulator . . . . .	47
4.21. Prisma . . . . .	51
4.22. Umkehrprisma . . . . .	52
4.23. Pentaprisma . . . . .	53
<b>5. Faserkomponenten</b>	<b>54</b>
5.1. Optische Faser . . . . .	54
5.2. Optischer Verstärker . . . . .	55
5.3. Mach-Zehnder-Modulator . . . . .	56
5.4. Polarisationssteller . . . . .	56
5.5. Isolator . . . . .	56
5.6. Optischer Schalter . . . . .	57
5.7. Faserverzögerungstrecke . . . . .	58
5.8. Polarisator . . . . .	58
5.9. Optischer Zirkulator . . . . .	58
5.10. Faserkoppler . . . . .	60
5.10.1. Eingangs- und Ausgangsknoten . . . . .	62
5.10.2. Referenzknoten . . . . .	62
5.11. Faserbox . . . . .	63
<b>6. Elektrische Komponenten</b>	<b>65</b>
6.1. Koppler . . . . .	65
6.2. Synthesizer . . . . .	66
6.3. Mixer . . . . .	67

<b>7. Hybridkomponenten</b>	<b>69</b>
7.1. Optischer Filter . . . . .	69
7.2. Faserkollimator . . . . .	70
7.3. Detektor . . . . .	71
<b>8. Spezielle Knoten</b>	<b>73</b>
8.1. Komponenten-Bezeichner . . . . .	74
8.2. Referenzknoten . . . . .	75
8.3. Mittelpunkt-knoten . . . . .	76
8.4. Beschriftungsknoten . . . . .	76
8.5. Externe Knoten . . . . .	77
8.6. Grenzflächenknoten . . . . .	79
8.7. Referenzknoten für die Rotation . . . . .	80
8.8. Strahlknoten . . . . .	80
8.9. Strahlvektor . . . . .	83
8.10. Knotenübersicht . . . . .	85
<b>9. Verbinden von Komponenten</b>	<b>86</b>
9.1. Zugriff auf Komponenten . . . . .	86
9.2. Strahlen zeichnen . . . . .	88
9.2.1. Raytracing . . . . .	88
9.2.2. Brechungsindex . . . . .	90
9.2.3. Anfangsbedingungen . . . . .	93
9.2.4. Interner Strahlengang . . . . .	96
9.2.5. Verbinden mit Knoten . . . . .	99
9.2.6. Automatische Verbindungen . . . . .	101
9.2.7. Strahlaussehen . . . . .	101
9.3. Aufgeweitete Strahlen . . . . .	104
9.3.1. Strahlaussehen . . . . .	104
9.4. Fehlerbehandlung . . . . .	105
9.5. Angepasste Strahlen . . . . .	107
9.6. Faserverbindungen . . . . .	110
9.6.1. Faserwinkel . . . . .	111
9.6.2. Faseraussehen . . . . .	115
9.7. Elektrische Verbindungen . . . . .	116
9.8. Automatische Faser- und Drahtverbindungen . . . . .	118
9.8.1. Aussehen der automatischen Verbindungen . . . . .	121
9.8.2. Bekannte Einschränkungen . . . . .	122
9.9. Zeichenebenen . . . . .	122

<b>10. Benutzerdefinierte Komponenten</b>	<b>124</b>
10.1. Benutzerdefinierte Version existierender Komponenten . . . . .	126
10.2. Neue Objekte definieren . . . . .	127
10.2.1. Die Zeichnung der Komponente . . . . .	128
10.2.2. Unterstützung von Raytracing . . . . .	129
10.2.3. Unterstützung von externen Knoten . . . . .	130
10.2.4. Beispiele für benutzerdefinierte Komponenten . . . . .	131
<b>11. Beispiele</b>	<b>133</b>
<b>12. Zusatzinformationen</b>	<b>159</b>
12.1. Interne Struktur der Komponenten . . . . .	159
12.2. Übersicht der Spezialknoten . . . . .	161
12.2.1. Externe und Rotationsreferenzknoten . . . . .	161
12.2.2. Grenzflächenknoten . . . . .	163
12.3. Abwärtskompatibilität . . . . .	165
12.3.1. Version 4.3a . . . . .	165
12.3.2. Version 4.3 . . . . .	165
12.3.3. Version 4.2 . . . . .	166
12.3.4. Version 4.0 . . . . .	166
12.3.5. Version 3.3 . . . . .	166
12.3.6. Version 3.0 . . . . .	167
<b>Dokumentationsindex</b>	<b>170</b>
<b>A. Versionsgeschichte</b>	<b>192</b>

# 1. Einführung

## 1.1. Über das Paket

`pst-optexp` ist ein PSTricks-Paket zum Skizzieren optischer Versuchsaufbauten. Dafür werden viele unterschiedliche Freistrah- 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` benötigt  $\text{\LaTeX}$  und aktuelle Versionen der Pakete `pstricks` ( $\geq 2.53$ ), `pst-plot` ( $\geq 1.6.1$ ), `pst-node`, `pstricks-add`, `multido`, `pst-eucl`, `pst-intersect` ( $\geq 0.4$ ) 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.<sup>1</sup>

## 1.3. Verbreitung und Installation

Dieses Paket ist auf CTAN<sup>2</sup> erhältlich und in  $\text{\TeX}$ Live and  $\text{\MiKTeX}$  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

---

<sup>1</sup><http://tug.org/PSTricks/main.cgi?file=pdf/pdfoutput>

<sup>2</sup><http://mirror.ctan.org/help/Catalogue/entries/pst-optexp.html>

- `pst-optexp.sty`: die L<sup>A</sup>T<sub>E</sub>X Stildatei

Speichern Sie diese Dateien in einem Verzeichnis der Teil Ihres lokalen T<sub>E</sub>X-Baums ist.

Vergessen Sie nicht `texhash` aufzurufen um den Baum zu aktualisieren. MiK<sub>T</sub>E<sub>X</sub>-Benutzer müssen die Dateinamen-Datenbank (FNDB) aktualisieren.

Detailliertere Information finden Sie in der Dokumentation Ihrer L<sup>A</sup>T<sub>E</sub>X-Distribution über die Installation in den lokalen T<sub>E</sub>X-Baum.

## 1.4. Lizenz

Es wird die Erlaubnis gewährt, dieses Dokument zu kopieren, zu verteilen und/oder zu modifizieren, unter den Bestimmungen der L<sup>A</sup>T<sub>E</sub>X Project Public License, Version 1.3c.<sup>3</sup>. 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. 12.3.

## 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  $\langle num \rangle$  und  $\langle pnum \rangle$ ). Tab. 1.1 erklärt die geläufigsten Typen.

---

<sup>3</sup><http://www.latex-project.org/lppl.txt>

<i>Name</i>	Beschreibung
<i>num</i>	Gleitkommazahl
<i>psnum</i>	Postscript-Kode der zu einer Zahl ausgewertet
<i>int</i>	Ganzzahl
<i>dimen</i>	Länge
<i>psstyle</i>	Benutzerdefinierte Parameterkonfiguration definiert mit <code>\newpsstyle</code>
<i>refpoint</i>	Analog zum Referenzpunkt von <code>\rput</code> , 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: David Carlisle, Florent Chervet, Enrico Gregorio, Rolf Niepraschk und Heiko Oberdiek. Christine Römer hat mich mit ihrem Artikel in der DTK<sup>4</sup> 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).

<sup>4</sup>Pakete in Deutsch dokumentieren. in Die TeXnische Komödie. Heft 2/2011, S. 28-35.



## 2. Grundlegende Ideen

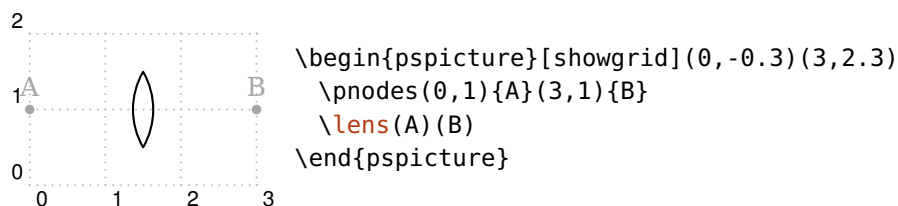
Dieses Kapitel zeigt die grundlegenden Ideen und Konzepte, die in diesem Paket stecken. Anhand von elementaren Beispielen werden die Grundfunktionen wie Ausrichtung, Positionierung (2.1) und Beschriftung (2.2) von Komponenten beschrieben. Anschließend wird deren Verbindung mit Strahlen (2.3) oder Fasern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

In Kap. 2.5 wird Schritt für Schritt die angedachte Vorgehensweise für das Erstellen von umfassenden Experimentsskizzen erläutert. Eine vollständige Referenz aller Makros und deren Parameter finden Sie in Kap. 3–10.

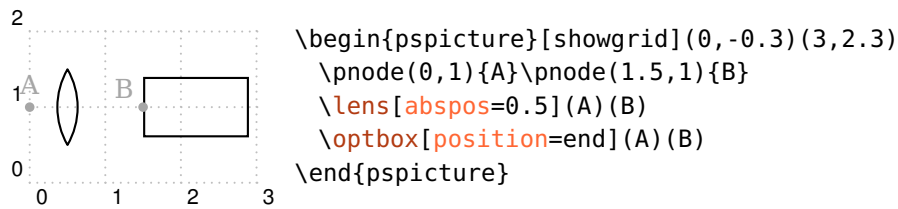
Ein Versuchsaufbau besteht aus Komponenten, die anhand ihrer Referenzknoten positioniert, verschoben und gedreht werden können und optional eine Beschriftung erhalten. Anschließend werden die Komponenten mit Lichtstrahlen oder Fasern verbunden.

### 2.1. Die Komponenten

Eine Komponente wird anhand der Knoten, die bei ihrer 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:



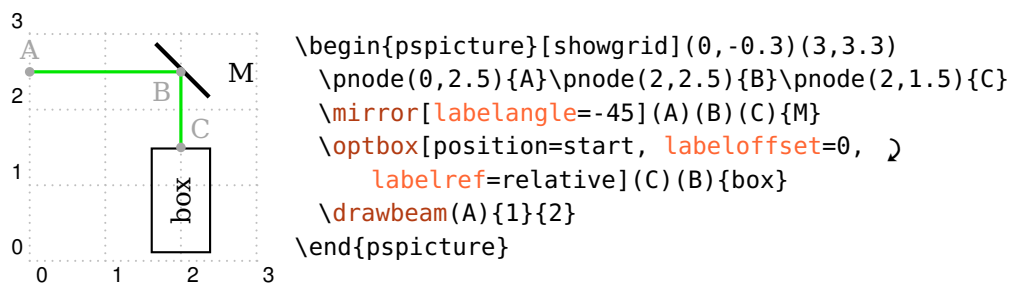
Diese Positionierung kann anhand von unterschiedlichen Parametern, wie z.B. **abspos** beeinflusst werden (siehe dazu Kap. 3.2 und Kap. 3.3). Im folgenden Beispiel wird die Linse 0.5 Einheiten vom Knoten  $\langle A \rangle$  entfernt platziert, die Box sitzt am Ende der Referenzlinie  $\overline{AB}$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



Die Komponenten lassen sich allgemein in zwei Kategorien unterteilen: Faser- und Freistrahalkomponenten. Diese unterscheiden sich in erster Linie bezüglich ihrer Verbindungsmöglichkeiten: Freistrahalkomponenten 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.

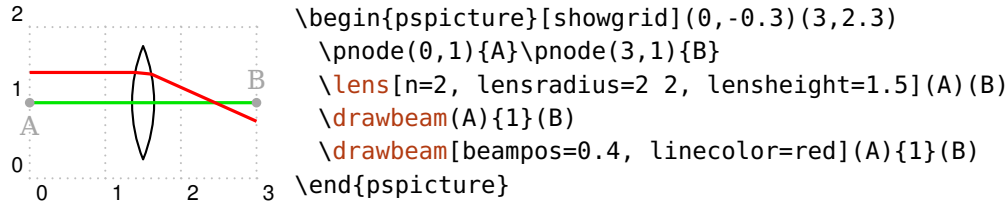


## 2.3. Freistrahl-Verbindungen

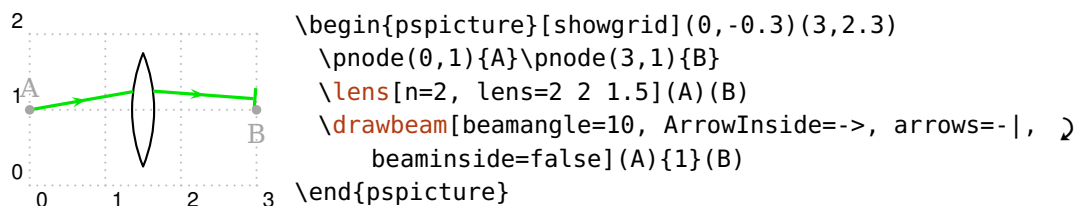
Freistrahalkomponenten können mit Lichtstrahlen verbunden werden, deren Weg mit Raytracing berechnet werden kann. Da es sich um ein Paket zum Skizzieren experimenteller Aufbauten handelt, werden hier trotz der realistischen Berechnungen viele Möglichkeiten offen gehalten den Strahlengang zu manipulieren.

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

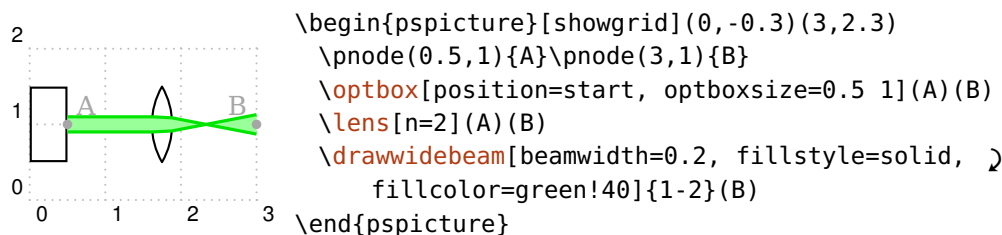
Das Beispiel zeigt zwei Strahlen, die mit unterschiedlichen Anfangsbedingungen auf die Linse treffen, der rote Strahl startet mit einem Abstand von 0.2 von der optischen Achse (`beampos`) und wird entsprechend von der Linse abgelenkt.



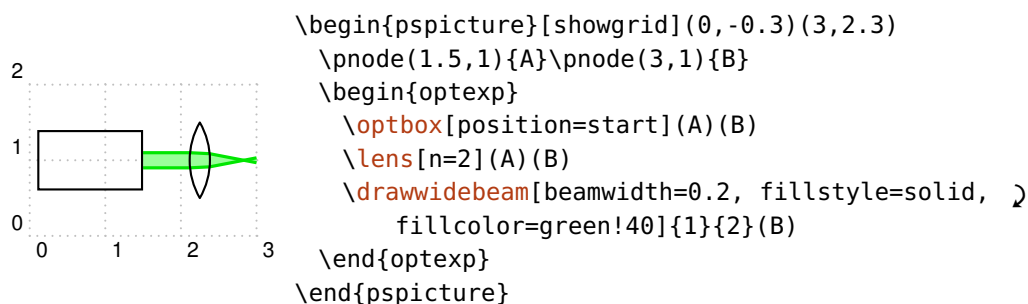
Die Strahlen können mit beliebigen Strichlierungen und Pfeilen dekoriert werden.



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

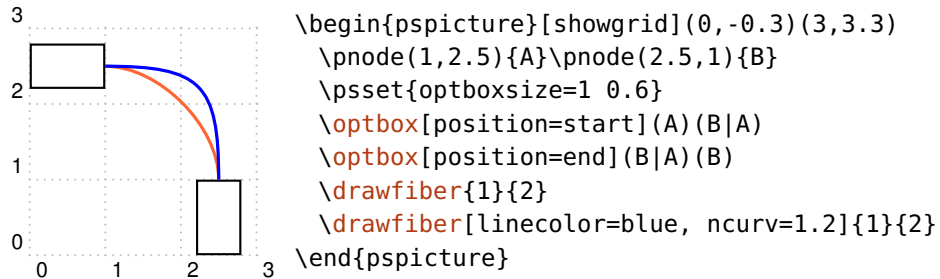


Die Strahlen können auch hinter die Komponenten gelegt werden. Dafür müssen alle beteiligten Komponenten und Strahlen in eine `optexp`-Umgebung gepackt werden (vergleiche das folgende Beispiel mit dem vorangegangenen).

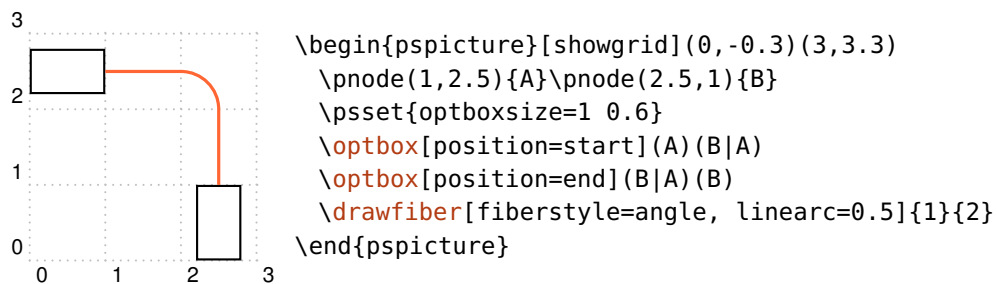


## 2.4. Faserverbindungen

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



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



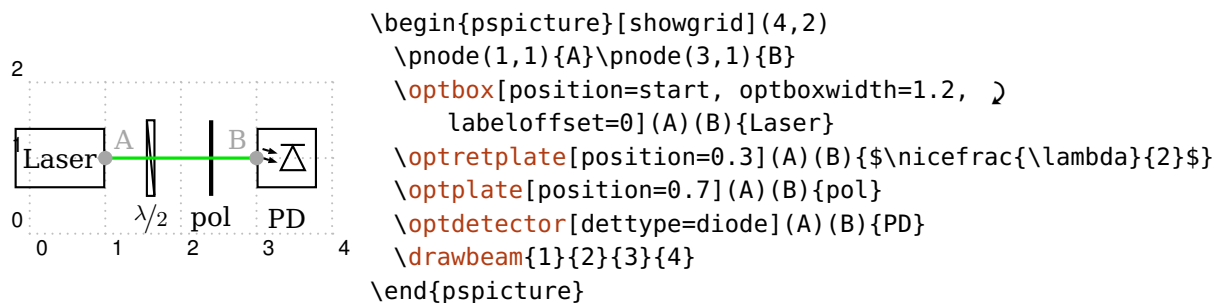
## 2.5. Schritt-für-Schritt-Beispiele

### 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.

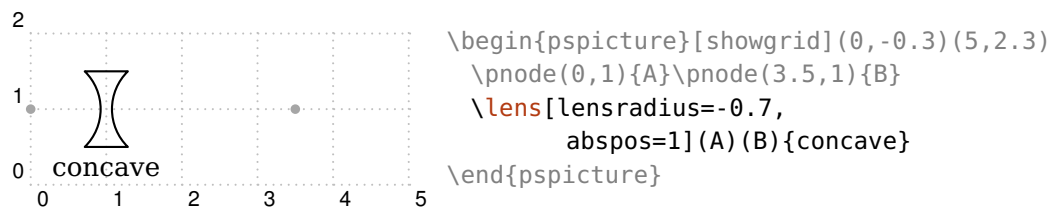


### 2.5.2. Galilei-Fernrohr

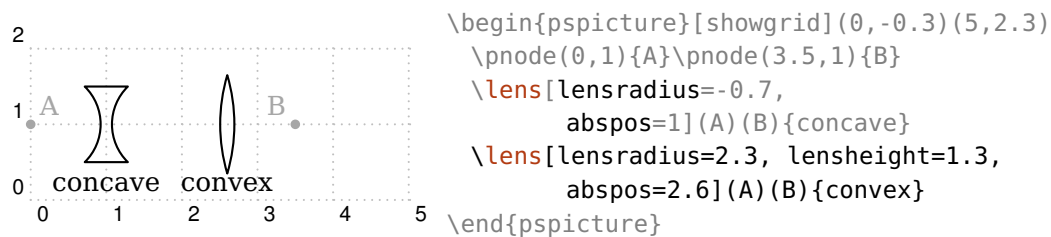
Im Folgenden wird ein sehr einfacher Aufbau mit einer konkaven und einer konvexen Linse schrittweise erläutert.

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.

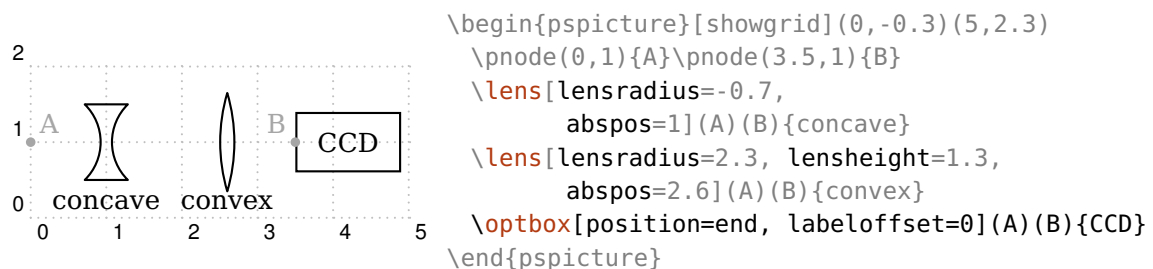
Die erste Linse soll konkav gekrümmte Grenzflächen haben. Mit `lensradius` werden die Radien der linken und rechten Fläche auf negative Werte gesetzt, was eine konkave Krümmung ergibt. Die Linse wird mit `abspos` eine Einheit vom  $\langle A \rangle$ -Knoten entfernt positioniert.



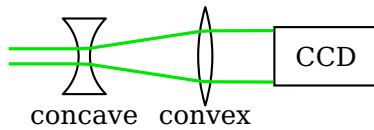
Die zweite Linse bekommt eine konvexe Krümmung (`lensradius` ist positiv) und wird 2.6 Einheiten von  $\langle A \rangle$  positioniert. Die Linsenhöhe wird mit `lensheight` eingestellt.



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  $\langle A \rangle$  nach  $\langle B \rangle$  gesetzt, die Beschriftung wird mit `labeloffset` in die Mitte der Komponente gelegt.



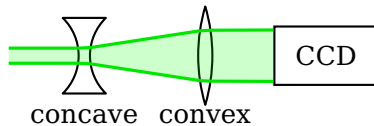
Zum Schluß wird ein aufgeweiteter Strahl mit einer Strahlbreite von 0.2 und ohne anfängliche Divergenz durch die Komponenten geschickt. Der Strahlengang wird mittels Raytracing berechnet. Die Linsenparameter in diesem Beispiel wurden so gewählt das bei kollimiertem Eingangsstrahl der Ausgangsstrahl ebenfalls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

#### Bsp. 2.1



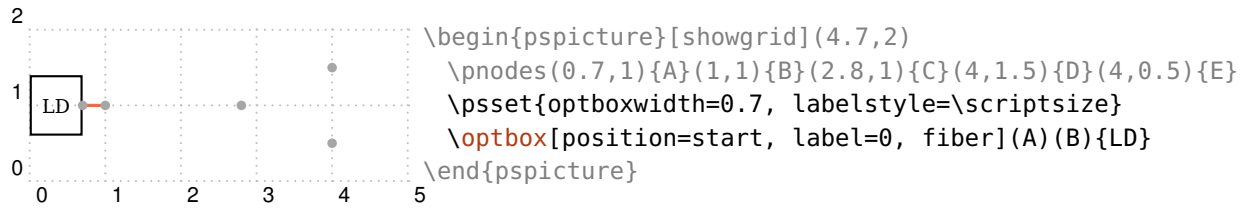
```
\begin{pspicture}(5,2)
  \pnode(0,1){A}\pnode(3.5,1){B}
  \lens[lensradius=-0.7,
    abspos=1](A)(B){concave}
  \lens[lensradius=2.3, lensheight=1.3,
    abspos=2.6](A)(B){convex}
  \optbox[position=end, labeloffset=0](A)(B){CCD}
  \addtopsstyle{Beam}{fillstyle=solid,
    fillcolor=green, opacity=0.2}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

### 2.5.3. Faseraufbau

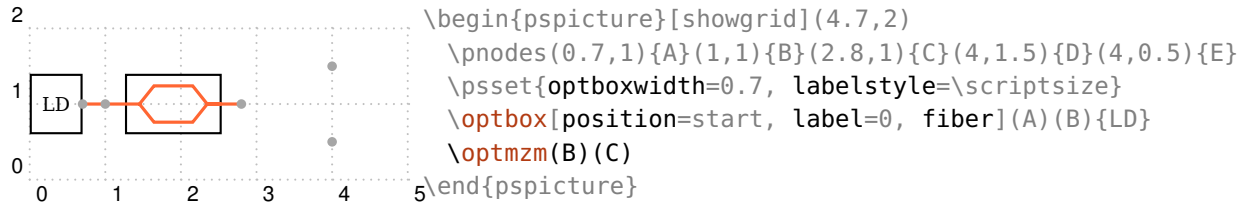
Dieses Beispiel zeigt, wie ein faseroptischer Aufbau mit automatisch gezeichneten Faserverbindungen konstruiert werden kann.

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.

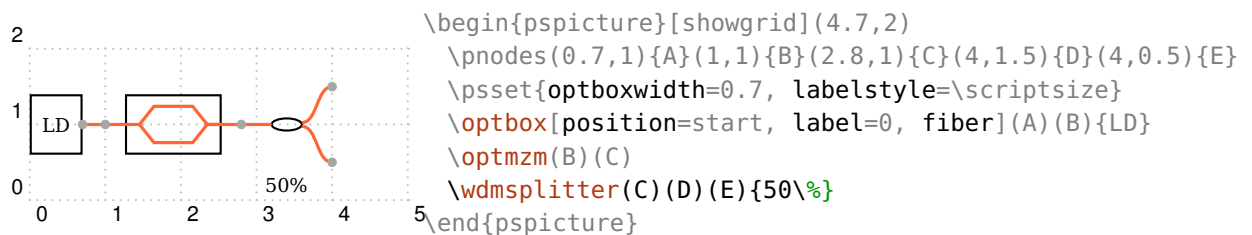
Der Laser wird wieder an den Anfang gesetzt. Der Abstand der Beschriftung wird mit dem Parameter **label** eingestellt, mit dem alle Beschriftungsparameter gesammelt eingestellt werden können. Da eine **\optbox** sowohl als Freistrahls- als auch als Faserkomponente verwendet werden kann, wird in der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



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



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

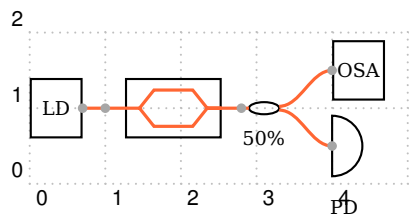


An die Enden des Kopplers wird ein optischer Spektrumanalysator (OSA) und eine Leistungsmessgerät (PD) gesetzt. Der `\optdetector` wird nach Voreinstellung ans Ende der Referenzlinie gesetzt (`position=end`). Diese beiden Komponenten werden ohne weitere Einstellungen ohne Faserverbindungen gezeichnet. 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. 2.2



```
\begin{pspicture}[showgrid](4.7,2)
  \pnodes(0.7,1){A}(1,1){B}(2.8,1){C}(4,1.5){D}(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}
```

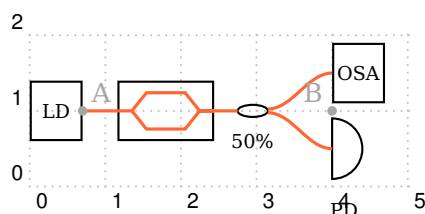
## 2.5.4. Faseraufbau (Alternative)

Dieses Beispiel ist vom Ergebnis her identisch mit dem vorangegangenen (Bsp. 2.2), 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. 2.3

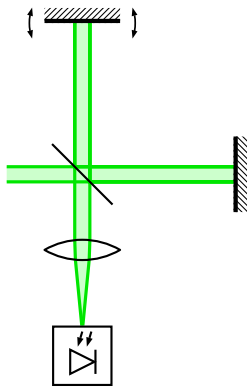


```
\begin{pspicture}[showgrid](4.9,2)
  \pnode(0.7,1){A}\pnode(4,1){B}
  \psset{optboxwidth=0.7, labelstyle=\scriptsize,
    fiber=None}
  \optbox[position=start, labeloffset=0](A)(B){LD}
  \optmzm[abspos=1.1](A)(B)
  \wdmsplitter[abspos=2.25, label=0.4](A)(B)(B){50\%}
  \optbox[position=end, label=0,
    compshift=0.5](A)(B){OSA}
  \optdetector[compshift=-0.5](A)(B){PD}
  \drawfiber{1-4}
  \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. 2.4



```

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

```

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

Dann werden der Reihe nach alle Komponenten platziert (Zeile 5–9). 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 10) und der Strahl gezeichnet (Zeile 11).

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

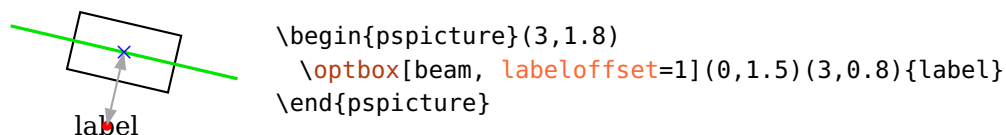
Dieses Kapitel enthält alle Informationen zu allgemeinen Parametern und Konzepten. Das beinhaltet das Setzen von Beschriftungen (3.1), Positionierung (3.2), Drehen und Verschieben von Komponenten (3.3) und Anpassen der Aussehens (3.4 und 3.5).

### 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. 8.4).

`labeloffset=<num>` default: 0.8

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

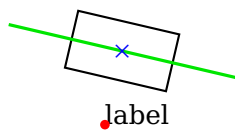


`labelstyle=<macros>`

Der Schriftstil der Beschriftungen.

`labelalign=<refpoint>` default: c

Definiert die Ausrichtung der Beschriftung bezüglich des Beschriftungsknotens.

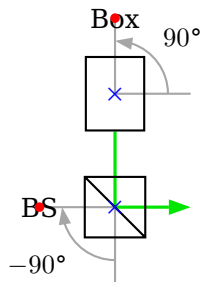


```
\begin{pspicture}(3,1.8)
\psset{beam, labeloffset=1}
\optbox[labelalign=bl](0,1.5)(3,0.8){label}
\end{pspicture}
```

`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`).



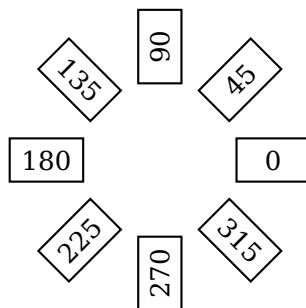
```
\begin{pspicture}(-0.3,0)(2,3.7)
\psset{labeloffset=1, optboxwidth=1, arrowscale=1.5, arrowinset=0}
\optbox[position=end, labelangle=90](1,1)(1,2){Box}
\beamsplitter[labelangle=-90](1,2)(1,1)(2,1){BS}
\drawbeam[arrows=->]{1}{2}(2,1)
\end{pspicture}
```

`labelref=relative, relgrav, global, absolute`

default: relgrav

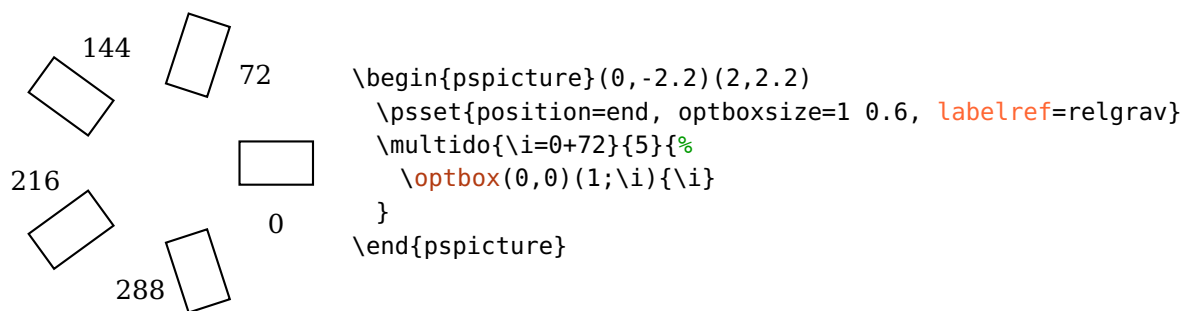
Wählt das Bezugssystem für `labelangle` und die Orientierung der Beschriftung.

**relative** Die Beschriftung ist parallel zu der Referenzlinie der Komponente und der Winkel wird immer auf den Bereich zwischen  $-90^\circ$  und  $+90^\circ$  zurückgerechnet.

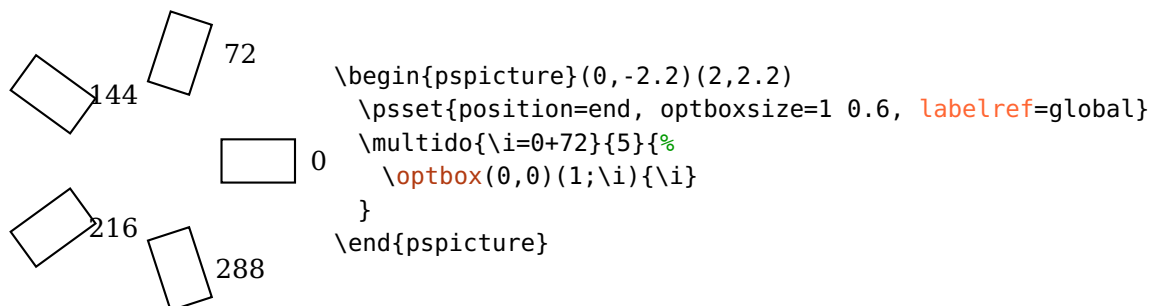


```
\begin{pspicture}(0,-2.2)(2,2.2)
\psset{position=end, optboxsize=1 0.6, labeloffset=0, }
\labelref=relative}
\multido{\i=0+45}{8}{%
\optbox(0,0)(1;\i){\i}
}
\end{pspicture}
```

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

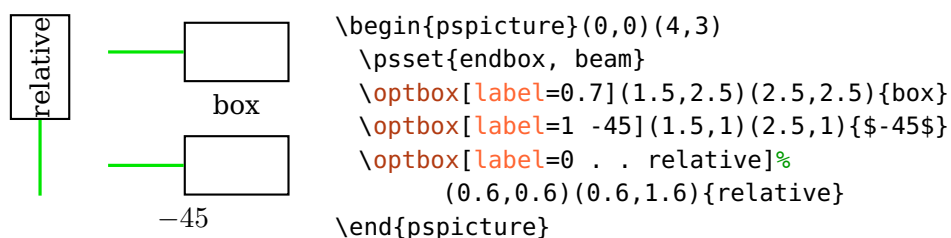


**global, absolute** Die Beschriftung ist unabhängig von der Komponentenausrichtung.



**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.



**innerlabel**=true

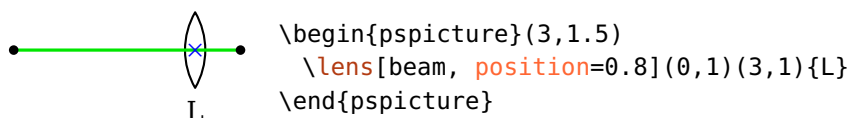
Das ist ein Alias für **label=0 . c relative**

## 3.2. Positionierung

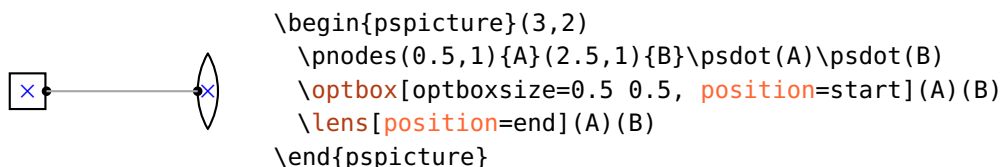
Alle Komponenten außer den Freistrahl-Dreipolen können zwischen ihren beiden Referenzknoten `\oenodeRefA` und `\oenodeRefB` positioniert werden.

`position=<num>`, start, end

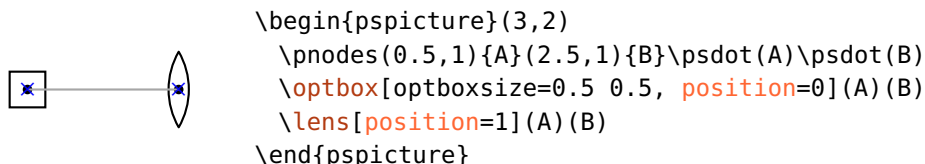
Ist der Wert eine Zahl, so wird damit die relative Position der Mitte eines Objektes (blaues Kreuz) bezüglich seiner beiden Referenzknoten (schwarze Punkte) gesetzt. Das ist äquivalent zum `npos` Parameter von `\ncput`, d.h. eine Zahl im Intervall  $[0, 1]$  (siehe `pst-node` Dokumentation<sup>1</sup>).



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.

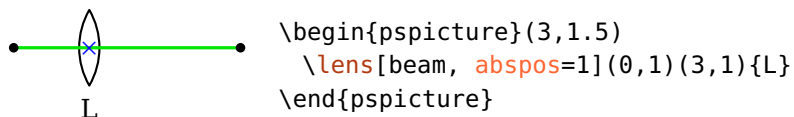


Hier ist im Vergleich dazu die Positionierung mit `position=0/1`.



`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.



<sup>1</sup><http://mirror.ctan.org/help/Catalogue/entries/pst-node.html>

`endbox=true, false`

Dieser Parameter sollte nicht mehr verwendet werden, er ist äquivalent zu `position=end`.

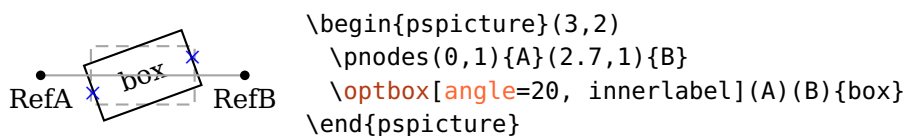
### 3.3. Drehen und Verschieben

Alle Komponenten können versetzt und um verschiedene Bezugspunkte gedreht werden.

Beachten Sie, dass mit den Komponenten auch die Grenzflächenknoten (Kap. 8.6, in den folgenden Beispielen mit blauen Kreuzen markiert) transformiert werden, wodurch Verbindungen beeinflusst werden. Die transformierten Referenzknoten sind als eigene Knoten verfügbar (Kap. 8.2).

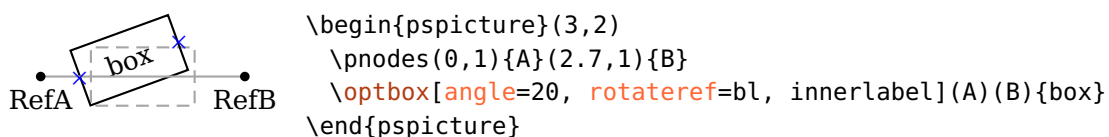
`angle=<psnum>` default: 0

Dreht eine Komponente um den Winkel  $\langle num \rangle$  (in Grad). Die Ausgangsposition ist grau gestrichelt eingezeichnet.



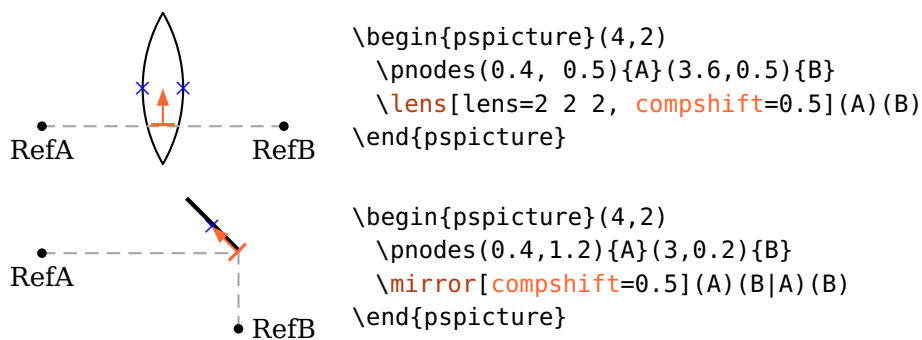
`rotateref=<refpoint>` default: c

Setzt den Bezugspunkt für die Drehung der Komponente. Bitte lesen Sie Kap. 8.7 für eine genauere Beschreibung und entnehmen Sie Kap. 12.2.1 die möglichen Bezugspunkte für alle Komponenten. Die Ausgangsposition ist grau gestrichelt eingezeichnet.



`compshift=<num>` default: 0

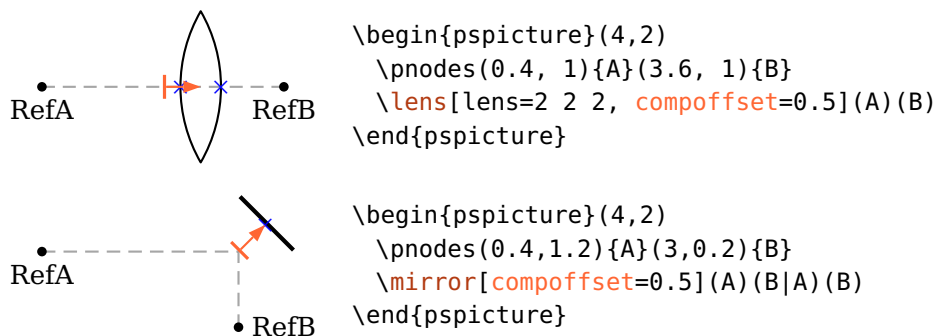
Verschiebt eine Komponente vertikal bezüglich ihrer Referenzlinie (siehe Kap. 8.2). Reflektive Komponenten werden entlang der reflektiven Grenzfläche verschoben.



`compoffset`= $\langle num \rangle$

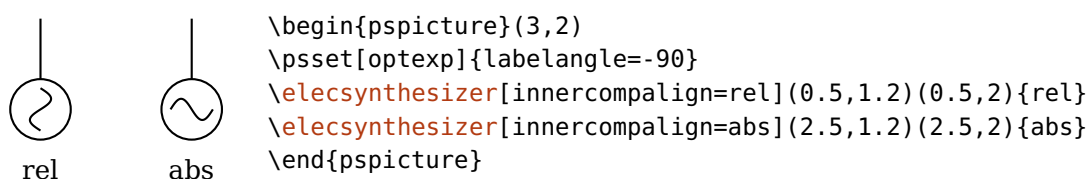
default: 0

Verschiebt eine Komponente vertikal bezüglich ihrer Referenzlinie (siehe Kap. 8.2). Reflektierende Komponenten werden entlang der reflektiven Grenzfläche verschoben. Die Verschiebung erfolgt also senkrecht zu einer Verschiebung mit `compshift`. Ein Anwendungsbeispiel ist Bsp. 11.25.



`innercompalign`=rel, relative, abs, absolute

Ist dieser Parameter auf absolute gesetzt, so wird das „Innere“ einer Komponente nicht mitgedreht. Diese Option wird derzeit von `\optfilter`, `\elecsynthesizer` und `\optdipole` verwendet.



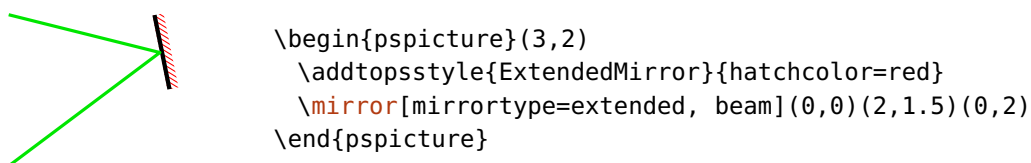


### 3.4. Verwendung von Stilen

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:

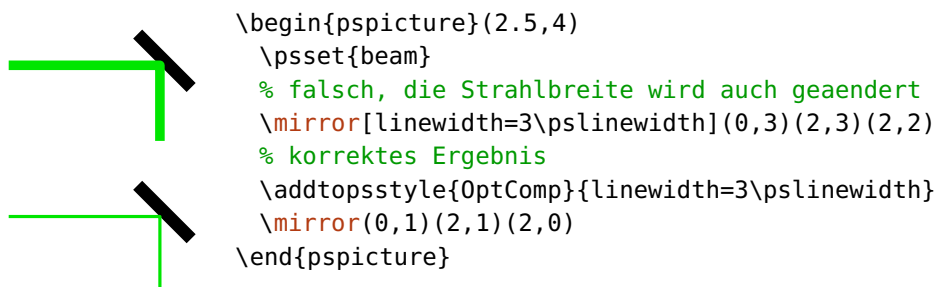


Die veränderbaren Unterbereiche einer Komponente sind in der jeweiligen Referenz zu finden.

### 3.5. Aussehen der Komponenten

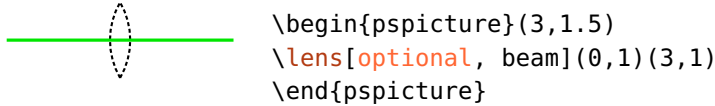
`OptComp`  $\langle psstyle \rangle$

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.

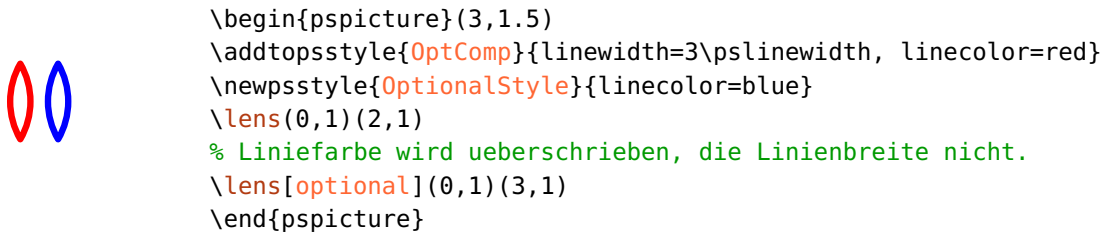


**OptionalStyle**  $\langle psstyle \rangle$  default: linestyle=dashed,dash=1.5pt 1pt

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



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



**VariableStyle**  $\langle psstyle \rangle$  default: linewidth=0.8\pslinewidth, arrowinset=0, arrowscale=0.8, arrows=<->

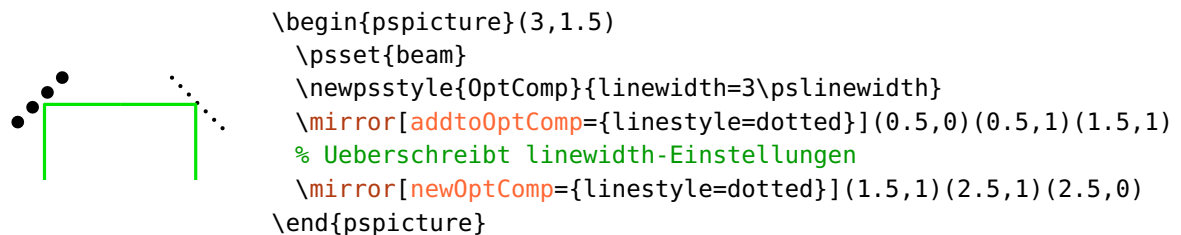
Der Stil der Pfeile für verstellbare Komponenten (**\mirror** und **\optgrating**).

**addtoOptComp**= $\langle list \rangle$

Der vorhandene **OptComp**-Stil wird lokal um die Parameter in  $\langle list \rangle$  erweitert,  $\langle list \rangle$  muss mit geschweiften Klammern gekapselt werden.

**newOptComp**= $\langle list \rangle$

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



**optional**=true, false default: false

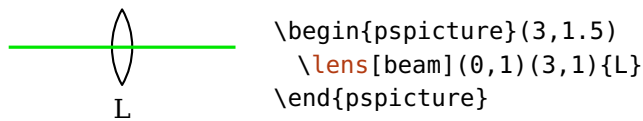
Markiert eine Komponente als optional indem der Stil **OptionalStyle** angewendet wird.

## 4. Freistrahalkomponenten

In diesem Kapitel werden alle Freistrahalkomponenten und deren Parameter beschrieben. Freistrahalkomponenten 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 (4.1–4.13), wohingegen z.B. Spiegel drei Knoten benötigen um korrekt ausgerichtet zu werden (4.14–4.23).

### 4.1. Linse

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



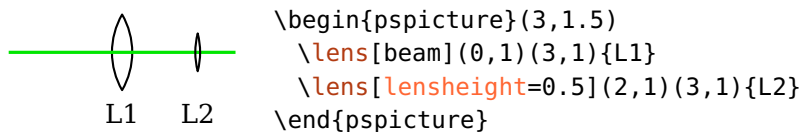
```
\begin{pspicture}(3,1.5)
  \lens[beam](0,1)(3,1){L}
\end{pspicture}
```

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⟩`

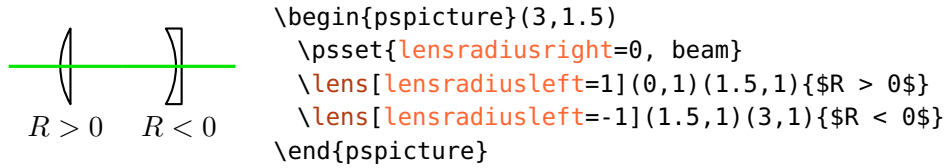
default: 1

Setzt den Radius der linken Linsengrenzfläche. Ein positiver Wert  $\langle num \rangle$  ist für eine konvexe, ein negativer für eine konkave Krümmung. Null ergibt eine ebene Fläche.

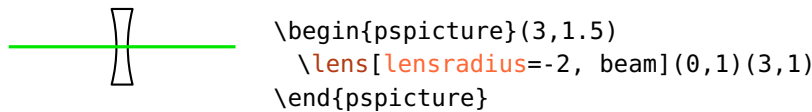
**lensradiusright**= $\langle num \rangle$ 

default: 1

Das Gleiche wie **lensradiusleft**, nur für die rechte Grenzfläche.

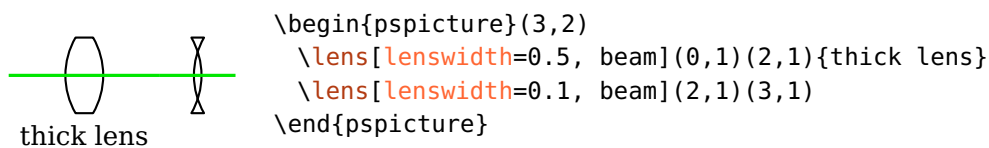
**lensradius**= $\langle left \rangle$  [ $\langle right \rangle$ ]

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

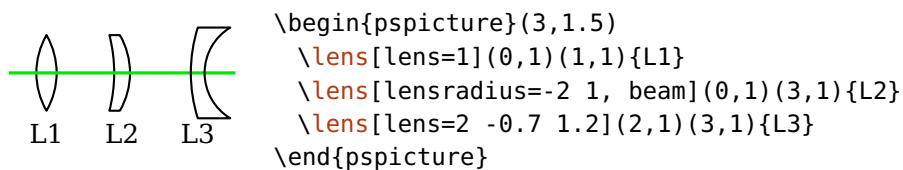
**lenswidth**= $\langle num \rangle$ 

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.

**lens**= $\langle radiusleft \rangle$  [ $\langle radiusright \rangle$ ] [ $\langle height \rangle$ ] [ $\langle width \rangle$ ]

Eine Option zum gleichzeitigen Setzen mehrerer Linsenparameter. Werte mit-tendrin können mit einem Punkt übersprungen und Werte am Ende können weg-gelassen werden (z.B. lens=. 1 1).

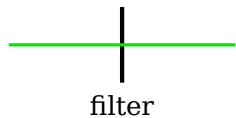


`thicklens=true, false`

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}

```

filter

`plateheight=⟨num⟩`

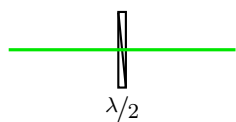
default: 1

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){$\frac{\lambda}{2}$}
\end{pspicture}

```

$\lambda/2$

`plateheight=⟨num⟩`

default: 1

Die Höhe des Plättchens.

`platewidth=⟨num⟩`

default: 0.1

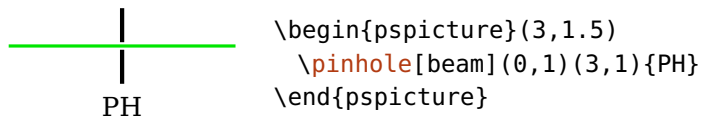
Die Breite des Plättchens.

`platesize=<width> <height>`

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

## 4.4. Lochblende

`\pinhole[<options>](<in>)(<out>){<label>}`



`outerheight=<num>`

default: 1

Die Gesamthöhe der Lochblende.

`innerheight=<num>`

default: 0.1

Die Höhe des Lochs.

`phlinewidth=<num> or <dimen>`

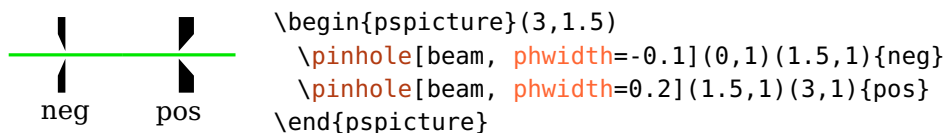
default:  $2\text{\pslinewidth}$

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>`

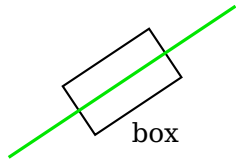
default: 0

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



## 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

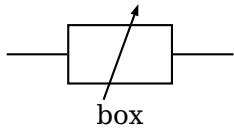
Die Höhe der Box.

`optboxsize=⟨width⟩⟨height⟩`

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

## 4.6. Pfeilkomponente

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



```
\begin{pspicture}(3,1.7)
  \optarrowcomp[wire](0,1)(3,1){box}
\end{pspicture}
```

Eine Komponente für den allgemeinen, flexiblen Gebrauch ähnlich zu `\fiberdelayline`, aber deutlich variabler.

`arrowcompwidth=⟨num⟩`

default: 1.4

Die Breite der Box.

`arrowcompheight=⟨num⟩`

default: 0.8

Die Höhe der Komponente.

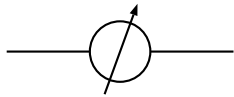
`arrowcompsize=⟨size⟩ or ⟨width⟩⟨height⟩`

Die Breite und Höhe der Komponente, äquivalent zum Aufruf von `arrowcompwidth` und `arrowcompheight`.

**arrowcompangle**= $\langle num \rangle$  default: 70

Der Winkel des Pfeils bezüglich der  $x$ -Achse.

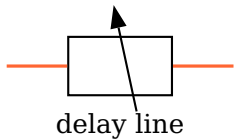
**arrowcompshape**=rectangle, circle default: rectangle



```
\begin{pspicture}(3,1)
\optarrowcomp[arrowcompshape=circle, wire](0,0.5)(3,0.5)
\end{pspicture}
```

**ArrowCompStyle**  $\langle psstyle \rangle$  default: arrowinset=0, arrows=->

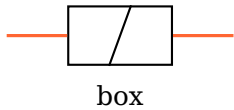
Der Stil für den Pfeil. Das kann insbesondere nützlich sein um die Länge und Ausrichtung des Pfeils zu verändern.



```
\begin{pspicture}(3,1.5)
\addtopsstyle{ArrowCompStyle}{xunit=-1, arrowscale=2}
\optarrowcomp[fiber](0,1)(3,1){delay line}
\end{pspicture}
```

## 4.7. Balkenkomponente

**\optbarcomp**[ $\langle options \rangle$ ]( $\langle in \rangle$ )( $\langle out \rangle$ ){ $\langle label \rangle$ }



```
\begin{pspicture}(3,1.5)
\optbarcomp[fiber](0,1)(3,1){box}
\end{pspicture}
```

Eine Komponente für den allgemeinen, flexiblen Gebrauch ähnlich zu **\optfiberpolarizer**, aber deutlich variabler.

**barcompwidth**= $\langle num \rangle$  default: 1.4

Die Breite der Box.

**barcompheight**= $\langle num \rangle$  default: 0.8

Die Höhe der Box.

**barcompsize**= $\langle size \rangle$  or  $\langle width \rangle$   $\langle height \rangle$

Die Breite und Höhe der Box, äquivalent zum Aufruf von **barcompwidth** und **barcompheight**.

**barcompangle**= $\langle num \rangle$  default: 70

Der Winkel des Balkens bezüglich der  $x$ -Achse.

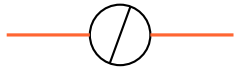




```
\begin{pspicture}(3,1)
\optbarcomp[barcompangle=90, fiber](0,0.5)(3,0.5)
\end{pspicture}
```

`barcompshape=rectangle, circle`

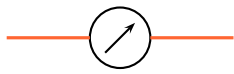
default: rectangle



```
\begin{pspicture}(3,1)
\optbarcomp[barcompshape=circle, fiber](0,0.5)(3,0.5)
\end{pspicture}
```

`BarCompStyle`  $\langle psstyle \rangle$

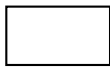
Der Stil für den Balken.



```
\begin{pspicture}(3,1)
\newpsstyle{BarCompStyle}{unit=0.7, arrows=->}
\optbarcomp[barcompshape=circle, barcompangle=45, 2
fiber](0,0.5)(3,0.5)
\end{pspicture}
```

## 4.8. Optische Quelle

`\optsource` [ $\langle options \rangle$ ] ( $\langle in \rangle$ ) ( $\langle out \rangle$ ) { $\langle label \rangle$ }

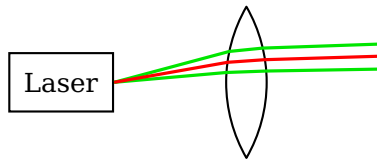


Laser

```
\begin{pspicture}(3,2)
\optsource(1.5,1)(3,1){Laser}
\end{pspicture}
```

Eine optische Quelle gibt ein paar Eigenschaften für alle von ihr ausgehenden Strahlen vor, falls es die erste Komponente in einem Strahlengang ist. Die verwendbaren Eigenschaften sind `beamangle`, `beamwidth`, `beamdiv` und `beamalign`. Alle Eigenschaften, die von einer optischen Quelle definiert werden, können nicht später durch Strahlparameter überschrieben werden. Wird ein Parameter nicht explizit von der Quelle definiert, so werden die Parameter von dem Strahl-Stil verwendet.

Die Voreinstellung für die Positionierung ist `position=start`.



```
\begin{pspicture}(5,2)
  \pnodes(1.5,1){A}(5,1){B}
  \optsource[innerlabel, beamdiv=10, 2
    beamangle=10](A)(B){Laser}
  \lens[lens=2 2 2](A)(B)
  \optplane(B)
  \drawwidebeam{-}
  \drawbeam[linecolor=red]{-}
\end{pspicture}
```

**sourcewidth**= $\langle num \rangle$  default: 1.4

Die Breite der Box.

**sourceheight**= $\langle num \rangle$  default: 0.8

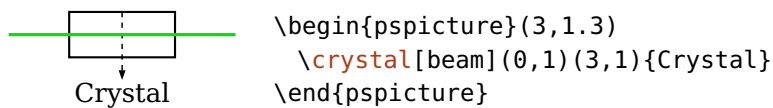
Die Höhe der Box.

**sourcesize**= $\langle width \rangle \langle height \rangle$

Die Breite und Höhe der Box, äquivalent zum Aufruf von **sourcewidth** und **sourceheight**.

## 4.9. Kristall

**\crystal**[ $\langle options \rangle$ ]( $\langle in \rangle$ )( $\langle out \rangle$ ){ $\langle label \rangle$ }



**crystalwidth**= $\langle num \rangle$  default: 1.4

Die Breite des Kristalls.

**crystalheight**= $\langle num \rangle$  default: 0.6

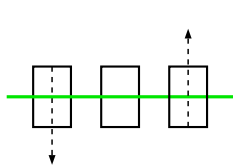
Die Höhe des Kristalls.

**crystalsize**= $\langle width \rangle \langle height \rangle$

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`.



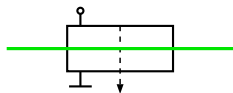
```
\begin{pspicture}(0,0.4)(3,1.5)
\psset{crystalsize=0.5 0.8, label=-45 . l}
\crystal[position=0.2, caxislength=0.5](0,1)(3,1){pos}
\crystal[beam, caxislength=0](0,1)(3,1){0}
\crystal[position=0.8, caxislength=-0.5](0,1)(3,1){neg}
\end{pspicture}
```

`caxisinv=true, false` default: false

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

`voltage=true, false` default: false

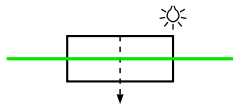
Zeichnet einen Spannungsanschluss und ein Erdungszeichen.



```
\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.



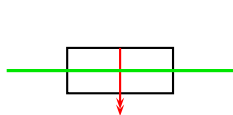
```
\begin{pspicture}(0,0.4)(3,1.7)
\crystal[lamp, beam](0,1)(3,1)
\end{pspicture}
```

`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.

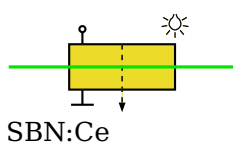


```
\begin{pspicture}(0,0.5)(3,1.6)
\newpsstyle{CrystalCaxis}{linecolor=red, arrows=->}
\crystal[beam](0,1)(3,1)
\end{pspicture}
```

**Crystallamp**  $\langle psstyle \rangle$ 

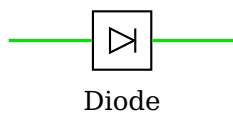
default: linewidth=0.6\pslinewidth

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}
```

## 4.10. Optische Diode

**\optdiode**  $\langle options \rangle (\langle in \rangle) (\langle out \rangle) \{ \langle label \rangle \}$ 

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

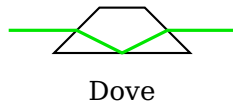
Diode

Die optische Diode hat die Voreinstellung `allowbeaminside=false`.**optdiodesize**  $= \langle num \rangle$ 

default: 0.8

Die Seitenlänge der optischen Diode.

## 4.11. Doveprisma

**\doveprism**  $\langle options \rangle (\langle in \rangle) (\langle out \rangle) \{ \langle label \rangle \}$ 

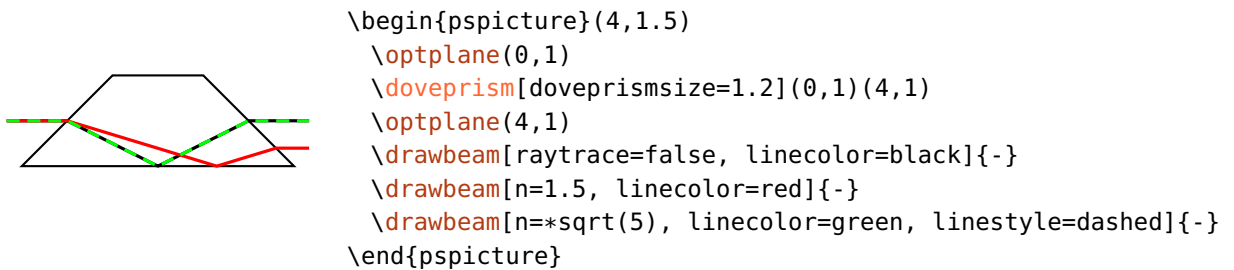
Dove

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

**doveprismsize**  $= \langle num \rangle$  or  $\langle width \rangle \langle height \rangle$ 

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°.



## 4.12. Glan-Thompson-Prisma

`\glanthompson` [*options*] (*in*) (*out*) {*label*}



`glanthompsonwidth`=*num* default: 1.0

Die Breite des Prisma.

`glanthompsonheight`=*num* default: 0.5

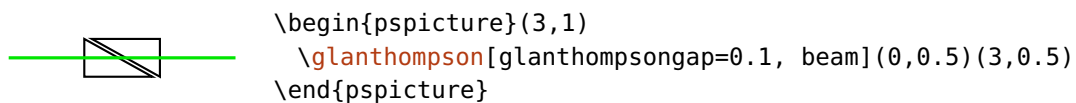
Die Höhe des Prisma.

`glanthompsonsize`=*width* *height*

Die Breite und Höhe des Prisma, äquivalent zum Aufruf von `glanthompsonwidth` und `glanthompsonheight`.

`glanthompsongap`=*num* default: 0

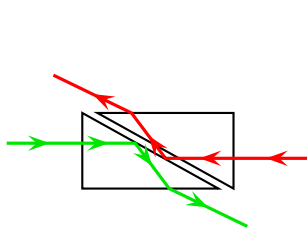
Der Abstand der beiden Teilprismen in  $x$ -Richtung, die Linienbreiten werden nicht berücksichtigt. Die Gesamtbreite ist von diesem Wert unabhängig, aber der Winkel der reflektierenden Fläche wird davon beeinflusst.



Von der Implementierung entspricht das Glan-Thompson-Prisma für `glanthompsongap=0` einem Strahlteiler, der nur zwei Knoten zur Positionierung benötigt und nicht quadratisch ist.

Beachten Sie, dass das Prisma für `glanthompsongap > 0` nicht von beiden Seiten gleich verwendet werden kann, da in der derzeitigen Implementierung nur eine

uneindeutige Grenzfläche (reflektierend oder transmittierend) zugelassen ist. In dem folgenden Beispiel sieht man, dass der rote Strahl an der falsche Fläche reflektiert wird.



```
\begin{pspicture}(4,2)
  \pnodes(0,1){A}(4,1){B}
  \glanthompson[glanthompsongap=0.2, glanthompsonsiz=2
    1](A)(B)
  \optplane[angle=90](2,0)\optplane[angle=90](2,2)
  \addtopsstyle{Beam}{ArrowInside=->, arrowscale=1.5}
  \drawbeam[beampos=0.1](A){1-2}
  \drawbeam[beampos=0.1, linecolor=red](B){1}{3}
\end{pspicture}
```

Bsp. 11.21 enthält ein weiterführendes Beispiel mit einem Glan-Thompson-Prisma.

### 4.13. 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 polymisc ist halb so groß.

`pollinewidth`=*<num>* or *<dimen>*

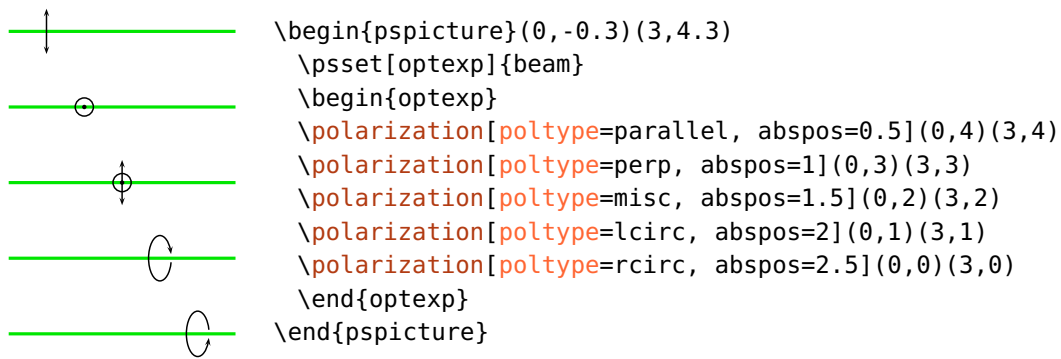
default: 0.7\pslinewidth

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.

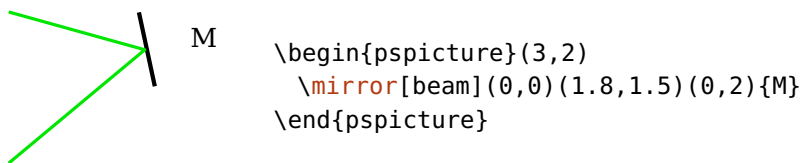


**Polarization**  $\langle psstyle \rangle$  default: `arrowscale=0.8, linewidth=0.7\pslinewidth, dotsize=3\pslinewidth`

Bestimmt das Aussehen der Polarisationsymbole.

## 4.14. Spiegel

**\mirror** [ $\langle options \rangle$ ] ( $\langle in \rangle$ ) ( $\langle center \rangle$ ) ( $\langle out \rangle$ ) { $\langle label \rangle$ }



**mirrorwidth** =  $\langle num \rangle$  default: 1

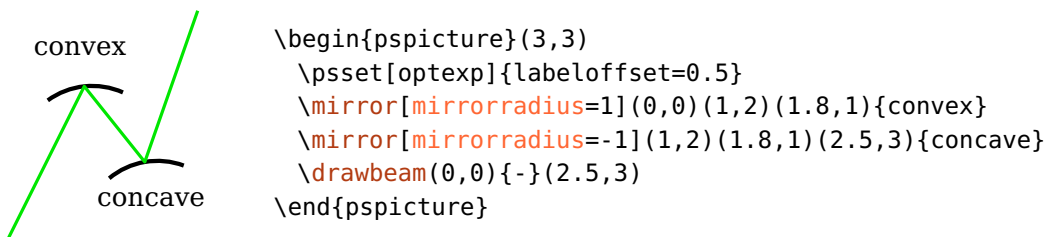
Die Breite des Spiegels.

**mirrorlinewidth** =  $\langle num \rangle$  or  $\langle dimen \rangle$  default: `2\pslinewidth`

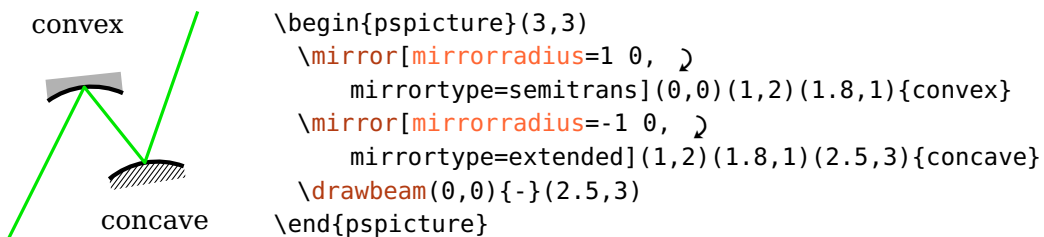
Die Linienbreite des Spiegels. Diese könnte ebenfalls über den `linewidth` Parameter gesetzt werden. Mit diesem Parameter kann aber die Linienbreite aller Spiegel global eingestellt werden.

**mirrorradius** =  $\langle radius \rangle$  [ 0] default: 0

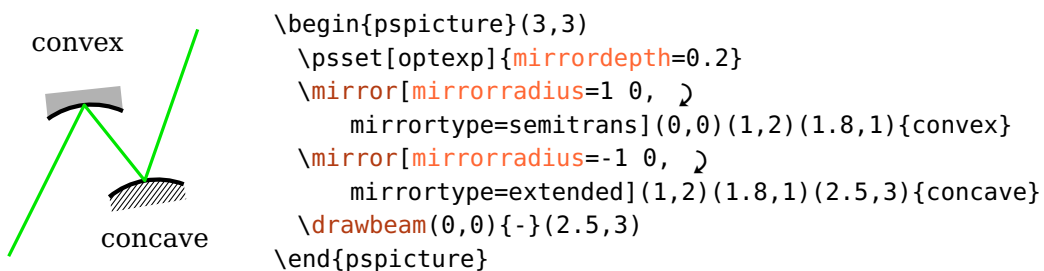
Diese Parameter setzt die Krümmung des Spiegels. Null ergibt einen ebenen, ein negativer Radius einen konvexen und ein positiver Radius einen konkaven Spiegel.



Wird kein zweiter Wert angegeben, so ist die Krümmung der zweiten Fläche eines extended oder semitrans Spiegels die Gleiche wie bei der Hauptgrenzfläche (siehe vorheriges Beispiel). Wird die Null als zweiter Wert angegeben (andere Werte werden nicht unterstützt), so ist die zweite Fläche flach:



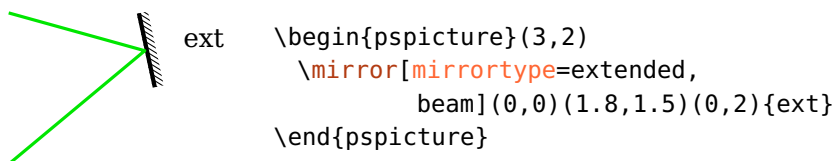
Die Angabe von `mirrordepth` bezieht sich immer auf die Mindesttiefe des Spiegels.



`mirrortype=plain, piezo, extended, semitrans`

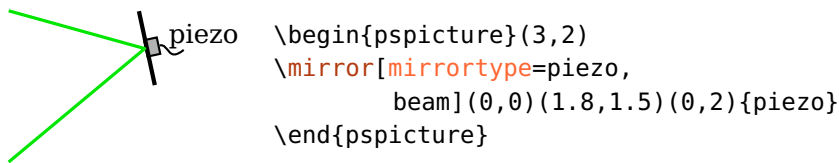
default: plain

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

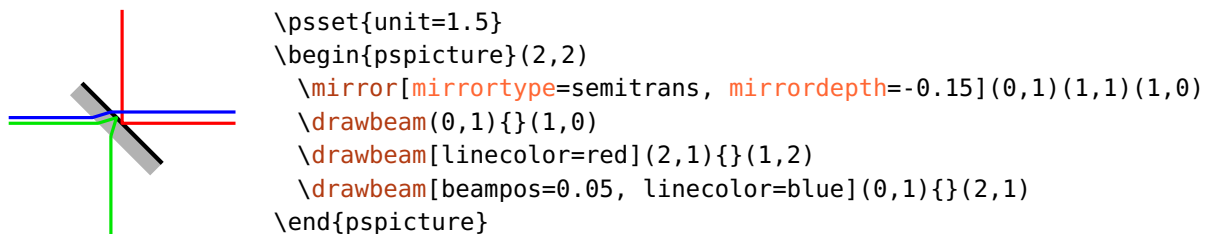
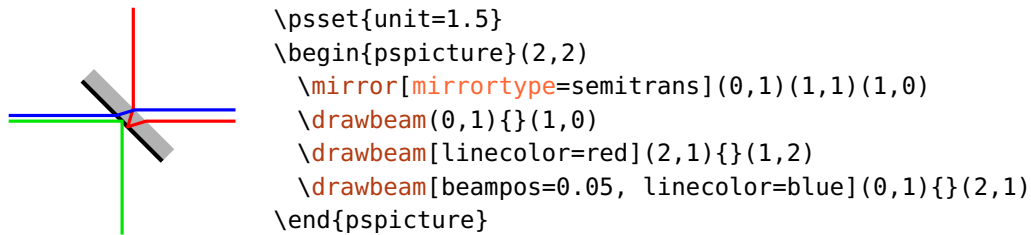


Beachten Sie, dass der Anschlussdraht für den Piezospiegel nicht gezeichnet wird, falls `extnode` verwendet wird.





Ein halbdurchlässiger Spiegel (semitrans) hat, im Gegensatz zu den anderen Spiegeltypen, zwei Grenzflächen, deren Abstand und Position vom Wert und Vorzeichen von `mirrordepth` abhängt.

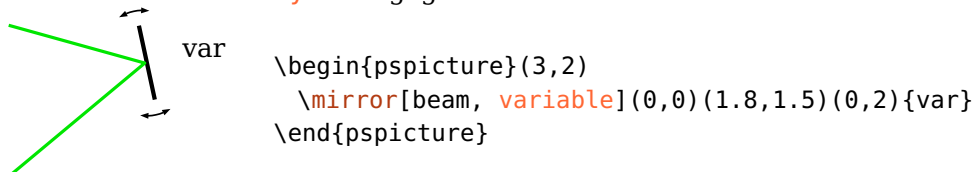


Eine negative Spiegeltiefe `mirrordepth` ist äquivalent zu einer positiven Tiefe kombiniert mit `angle=180`.

`variable=true, false`

default: false

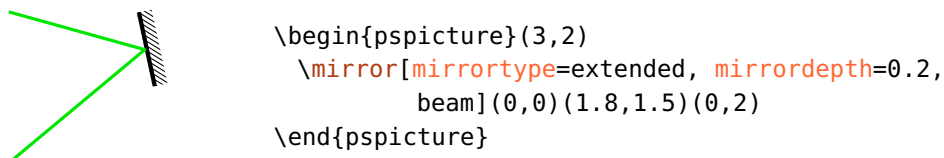
Zeichnet einen verstellbaren Spiegel, der mit zwei zusätzlichen gekrümmten Pfeilen an beiden Seiten angedeutet wird. Das Aussehen, inklusive der Pfeile, wird von `VariableStyle` vorgegeben.



`mirrordepth=<num>`

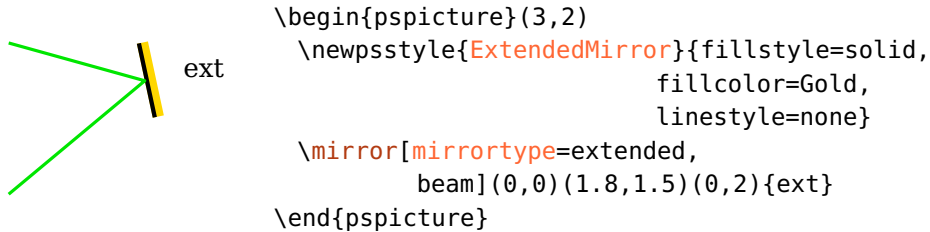
default: 0.1

Die Gesamttiefe eines breiten Spiegels.



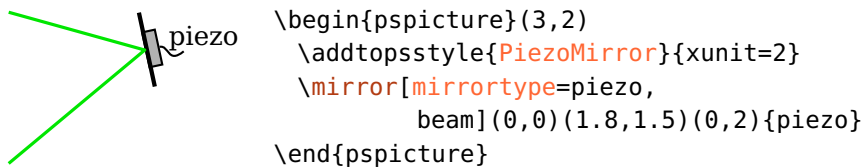
**ExtendedMirror**  $\langle psstyle \rangle$  default: `linestyle=None, hatchwidth=0.5\pslinewidth, hatchsep=1.4\pslinewidth, fillstyle=hlines`

Der Stil für den extended Spiegel.



**PiezoMirror**  $\langle psstyle \rangle$  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.

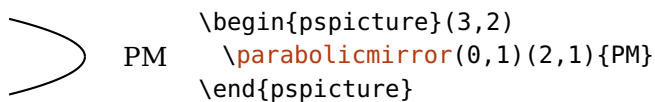


**SemitransMirror**  $\langle psstyle \rangle$  default: `linestyle=None, fillstyle=solid, fillcolor=black!30`

Der Stil für den semitrans Spiegel.

## 4.15. Parabolspiegel

**\parabolicmirror**  $[\langle options \rangle](\langle in \rangle)(\langle out \rangle)\{\langle label \rangle\}$

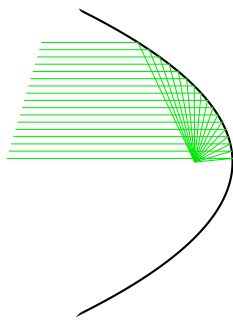


**parmirrorwidth**  $=\langle num \rangle$  default: 1

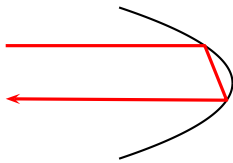
Die Breite des Parabolspiegels.

**parmirrorheight**  $=\langle num \rangle$  default: 1

Die Höhe des Parabolspiegels.



```
\begin{pspicture}(0,-2)(3,2)
\parabolicmirror[parmirrorheight=4, parmirrorwidth=2](0,0)(3,0)
\addtopsstyle{Beam}{beamalign=abs, linewidth=0.2\pslinewidth, 2
beaminsidelast, stopinsidecount=1}
\multido{\r=-1.55+0.1}{33}{%
\drawbeam[beampos=\r](0,0){1}}
\end{pspicture}
```

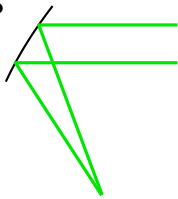


```
\begin{pspicture}(0,-1)(3,1)
\optplane(0,0)
\parabolicmirror[parmirrorheight=2, 2
parmirrorwidth=1.5](0,0)(3,0)
\addtopsstyle{Beam}{arrows=->, beamalign=abs}
\drawbeam[beampos=0.5, linecolor=red]{1}{2}{1}
\end{pspicture}
```

## 4.16. Off-axis-Parabolspiegel

`\oapmirror[⟨options⟩](⟨in⟩)(⟨center⟩)(⟨focus⟩){⟨label⟩}`

OAP



```
\begin{pspicture}(3,2.5)
\oapmirror(3,2)(1,2)(2,0){OAP}
\drawwidebeam[beamwidth=0.5](3,2){1}(2,0)
\end{pspicture}
```

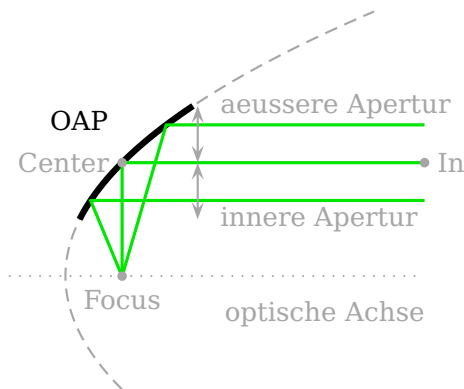
`oapmirroraperture=⟨num⟩ or ⟨inner⟩ ⟨outer⟩`

default: 1

Apertur des Spiegels bezogen auf die Eingangsrichtung. Wird eine Zahl angegeben, dann gibt diese die gesamte Apertur an, die vertikal und symmetrisch zur Verbindung zwischen `⟨in⟩` und `⟨center⟩` gemessen wird. Bei zwei Zahlen geben diese die innere und äußere Apertur an.

Die Konstruktion des Spiegels anhand der vorgegebenen Punkte geschieht wie folgt: Die Gerade von Knoten `<in>` zu `<center>` liegt parallel zur optischen Achse, `<focus>` gibt den Fokuspunkt der Parabel an. Die Apertur des Spiegels wird mittels des Parameters `oapmirroraperture` angegeben, gemessen senkrecht zum einfallenden Strahl.

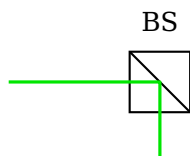
Die unterschiedlichen Parameter und die zugrundeliegende ganze Parabel sind in der folgenden Skizze dargestellt:



```
\begin{pspicture}(6,5)
\nodes(5.5,3){In}(1.5,3){Center}(1.5,1.5){Focus}
\oapmirror[oapmirroraperture=1.5,
linewidth=3\pslinewidth,
plotpoints=200](In)(Center)(Focus){OAP}
\multido{\r=-0.5+0.5}{3}{%
\drawbeam[beampos=\r](In){}(Focus)}
\end{pspicture}
```

## 4.17. Strahlteiler

`\beamsplitter`[`<options>`](`<in>`)(`<center>`)(`<out>`){`<label>`}



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

`bssize`=`<num>`

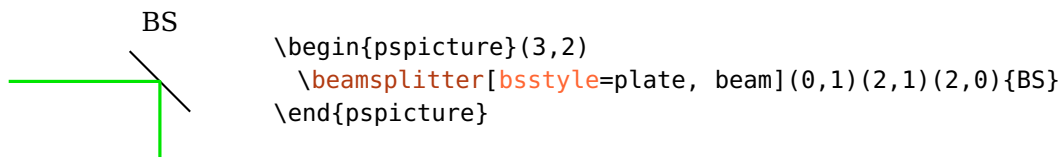
default: 0.8

Die Größe des Strahlteilers.

`bsstyle`=cube, plate

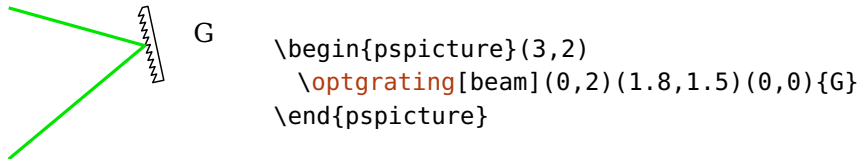
default: cube

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`.



## 4.18. Optisches Gitter

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



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

`gratingwidth`=*num* default: 1

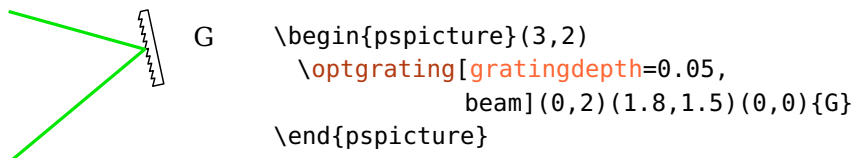
Die Breite des Gitters.

`gratingheight`=*num* default: 0.15

Die Gesamthöhe des Gitters.

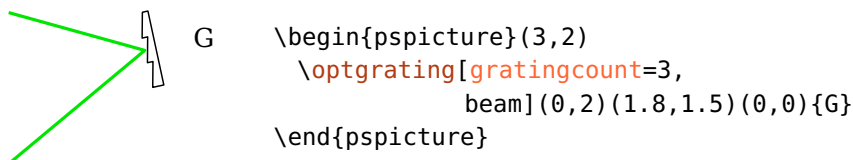
`gratingdepth`=*num* default: 0.075

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



`gratingcount`=*int* default: 10

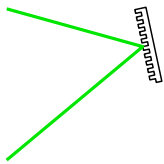
Die Anzahl der Gitterlinien.



**gratingtype**=blazed, binary

default: blazed

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

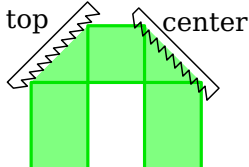


G

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

**gratingalign**=t, top, c, center

default: top

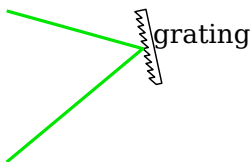
Wählt, ob die Reflektionsebene auf den Gitterspitzen (top), oder in der Mitte des Gitters (center) liegt. Wählt man center aus, macht es Sinn die **optexp**-Umgebung oder eine halbtransparente Strahlfüllung zu verwenden.

```
\begin{pspicture}(3,2)
  \pnodes(0.5,0){A}(0.5,1.5){B}(2,1.5){C}(2,0){D}
  \psset{unit=1.5, labeloffset=0.4}
  \begin{optexp}
    \optgrating[gratingalign=top](A)(B)(C){top}
    \optgrating[gratingalign=center](B)(C)(D){center}
    \drawwidebeam[fillstyle=solid, fillcolor=green!50,
      beamwidth=0.5](A){-}(D)
  \end{optexp}
\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** Diese Parameter wurden in die entsprechenden grating\* Parameter umbenannt und sind seit Version 3.0 veraltet.

**optgridheight**

**optgriddepth**

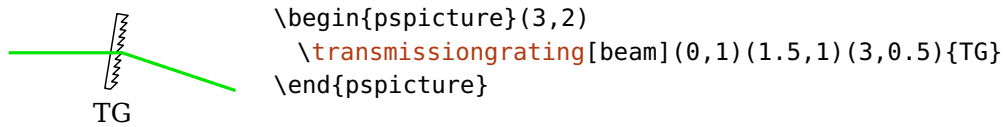
**optgridcount**

**optgridtype**

**optgridlinewidth**

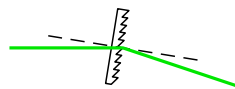
## 4.19. Transmissionsgitter

`\transmissiongrating`[*<options>*](*<in>*)(*<center>*)(*<out>*){*<label>*}

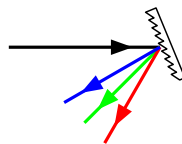


Das Transmissionsgitter unterstützt alle Optionen von `\optgrating`. Der Strahlengang wird so berechnet, dass der Strahl an einer Ebene senkrecht zum dargestellten Gitter reflektiert wird. Diese Ebene ist im folgenden Beispiel dargestellt:

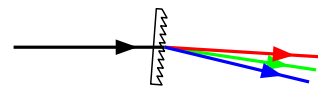
Virtuelle  
Reflexionsebene



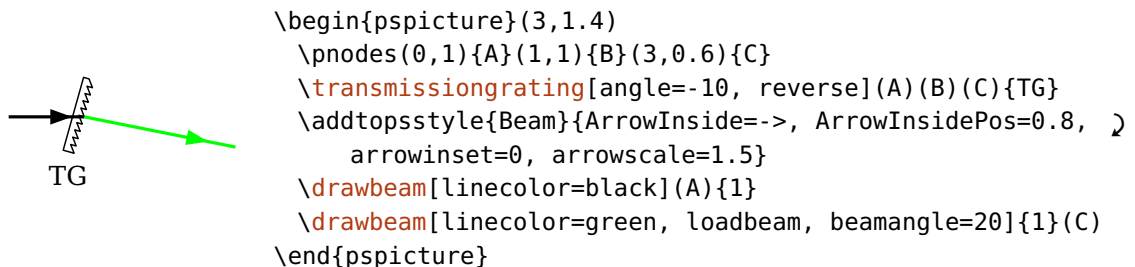
Reflexionsgitter  
`\optgrating`



Transmissionsgitter  
`\transmissiongrating`

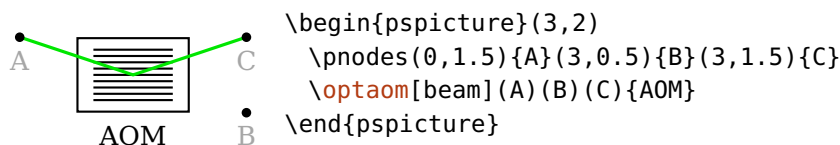


Ist dieses Verhalten nicht gewünscht, so kann man z.B. die Komponente mit `angle` weiter drehen, und den Strahlwinkel mit `beamangle` entsprechend korrigieren.



## 4.20. Akustooptischer Modulator

`\optaom`[*<options>*](*<in>*)(*<trans>*)(*<diff>*){*<label>*}



Dieses Makro stellt einen akustooptischen Modulator dar. Dieser kann, ähnlich einem `\beamsplitter` entweder transmittierend oder reflektierend sein. Beachten Sie, dass die Knoten nicht so angeordnet sind, wie bei den üblichen Tripolen wie z. B. dem `\optgrating`. Der mittlere Knoten gibt die Richtung eines transmittierten, und der dritte Knoten die Richtung der ersten Beugungsordnung an.

Diese erste Umsetzung des AOM in Version 4.8 ist noch etwas experimentell, da ich nicht genügend Testfälle hatte um alle Eventualitäten zu prüfen. Es könnten sich also noch Änderungen ergeben, die nicht abwärts kompatibel sind.

`aomheight`= $\langle num \rangle$  default: 1

Die Höhe des Modulators.

`aomwidth`= $\langle num \rangle$  default: 1.5

Die Breite des Modulators.

`aomsize`= $\langle width \rangle \langle height \rangle$

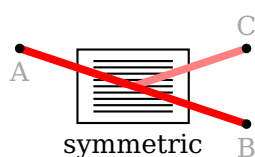
Die Breite und Höhe des Modulators, äquivalent zum Aufruf von `aomwidth` und `aomheight`.

`aomgratingcount`= $\langle int \rangle$  default: 9

Gibt die Anzahl der Gitterlinien an. Werden weitere Änderungen des Gitters oder des Modulators allgemein gewünscht, dann muss der Parameter `aomcomp` verwendet werden.

`aomalign`=symmetric, straight default: symmetric

Die Ausrichtung eines `\optaom` bezüglich der drei Referenzknoten kann mit diesem Parameter eingestellt werden. Die symmetrische (symmetric) Ausrichtung ist im folgenden Beispiel zu sehen:



```

\begin{pspicture}(3,2)
\pnodes(0,1.5){A}(3,0.5){B}(3,1.5){C}
\optaom[aomalign=symmetric](A)(B)(C){symmetric}
\addtopsstyle{Beam}{fillstyle=solid, linestyle=none, }
beamwidth=0.1}
\drawwidebeam[fillcolor=red!50](A){}(C)
\drawwidebeam[fillcolor=red](A){}(B)
\end{pspicture>

```

In der geraden (straight) Ausrichtung sieht der Strahlengang wie folgt aus:

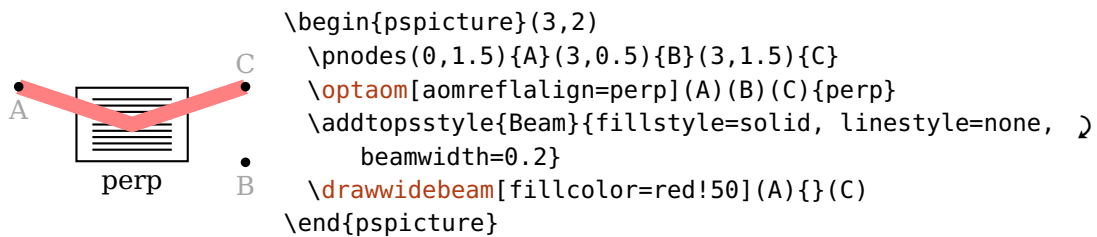
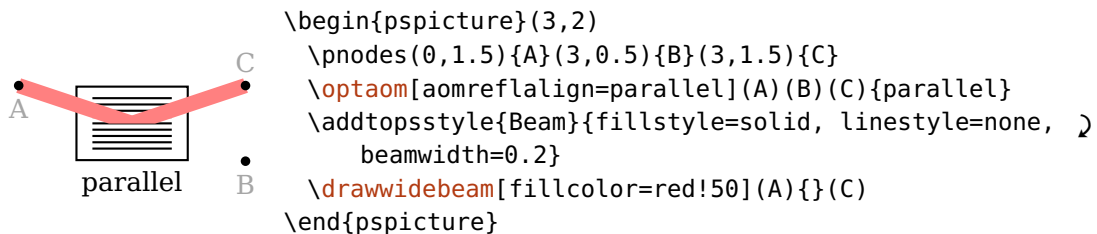




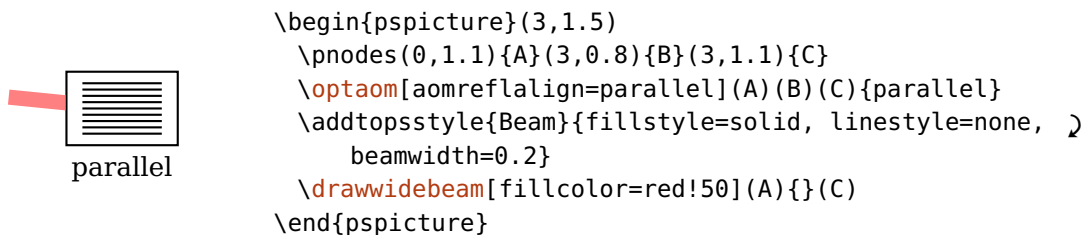
`aomreflalign=perp, parallel`

default: parallel

Die Ausrichtung der reflektiven Grenzfläche kann entweder parallel oder senkrecht (perp) zur Komponente liegen. Diese Option ist lediglich aus technischer Sicht notwendig, die Richtung des reflektierten Strahls wird dadurch nicht verändert.



Das folgende Beispiel zeigt einen Fall, der mit `aomreflalign=parallel` nicht funktioniert, da der Einfallswinkel so flach ist, dass die Schnittpunkte mit der reflektierenden Grenzfläche außerhalb der Komponente liegen:



`aomcomp=default, <macro>`

default: default

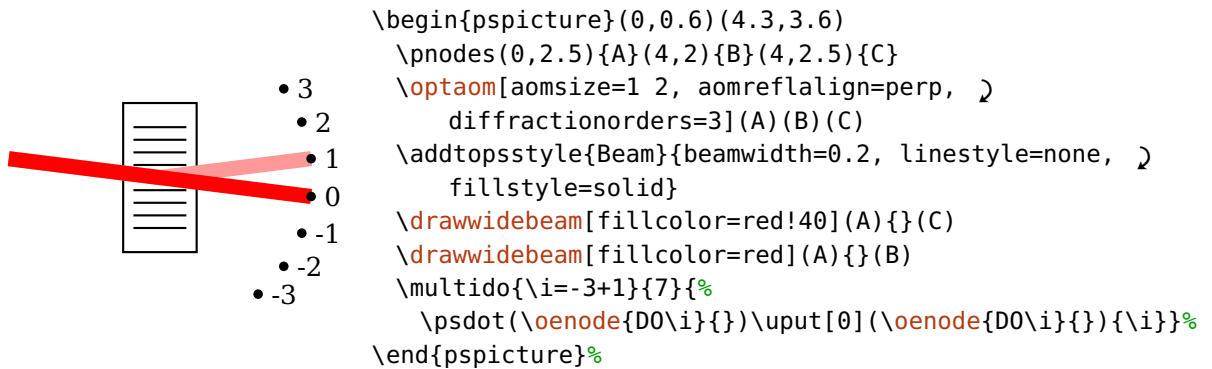
Tauscht die Gitterlinien durch eine beliebige Zeichnung aus.

`diffractionorders=<int>`

default: 2

Die Anzahl der Beugungsordnungen (positive und negative), die mit `beamdiffractionorder` ausgewählt werden können, also  $-\langle num \rangle, -\langle num \rangle + 1, \dots, \langle num \rangle$ .

Gleichzeitig werden entsprechende Knoten `\oencode{D0-1}{<comp>}`, `\oencode{D00}{<comp>}`, `\oencode{D01}{<comp>}` usw. definiert.



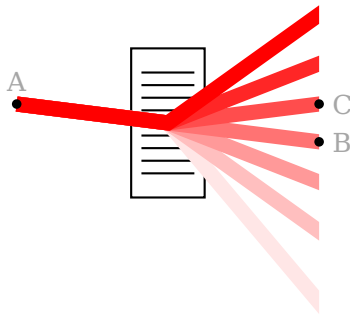
`beamdiffractionorder=<int>`

default: {}

Die Beugungsordnung, die für einen reflektierten Strahl durch den AOM verwendet wird. Wenn der Wert leer ist, {} dann wird die Option ignoriert.

Mit Verwendung dieser Option wird unmittelbar das Verhalten der Grenzfläche festgelegt, hängt also nicht von der relativen Position von AOM und der folgenden Komponente ab. Für `beamdiffractionorder`  $\neq 0$  wird der Strahl reflektiert, für den Wert 0 hingegen transmittiert. Der Parameter `beammode` wird dann ignoriert.

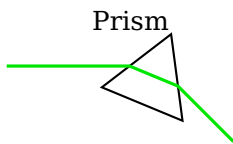
Es können nur die Beugungsordnungen verwendet werden, die für die entsprechende Komponente mit `diffractionorders` definiert wurden. Wird eine nicht-definierte Ordnung verwendet, dann wird die nächstliegende verwendet und eine Postscript-Warnung (siehe `pswarning`) ausgegeben.



```
\begin{pspicture}(0,-0.2)(4.4,3.8)
  \pnodes(0,2.5){A}(4,2){B}(4,2.5){C}
  \optaom[aomsize=1 2, aomreflalign=perp, diffractionorders=3](A)(B)(C)
  \optplane(B)
  \addtopsstyle{Beam}{beamwidth=0.2, linestyle=none, fillstyle=solid}
  \multido{\i=-3+1,\r=10+15}{7}{%
    \drawwidebeam[fillcolor=red!\r, beamdiffractionorder=\i](A){-}}
\end{pspicture}%
```

## 4.21. Prisma

`\optprism`[*<options>*](*<in>*)(*<center>*)(*<out>*){*<label>*}



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

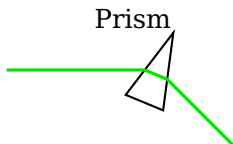
Das Prisma wird immer symmetrisch bezüglich der *<in>* und *<out>* Knoten ausgerichtet. Für asymmetrischen Strahlengang siehe Kap. 9.

`prismsize`=*<num>* default: 1

Die Höhe des Prismas.

`prismangle`=*<num>* default: 60

Der obere Winkel des Prismas.

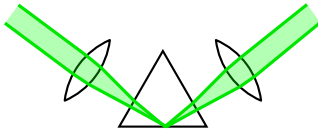


```
\begin{pspicture}(3,2)
  \optprism[prismangle=30, beam](0,1)(2,1)(3,0){Prism}
\end{pspicture}
```

**prismtype**=transmittive, reflective

default: transmittive

Ist dieser Parameter auf reflective gesetzt, so wird die Grundfläche als zusätzliche, reflektierende Grenzfläche verwendet.

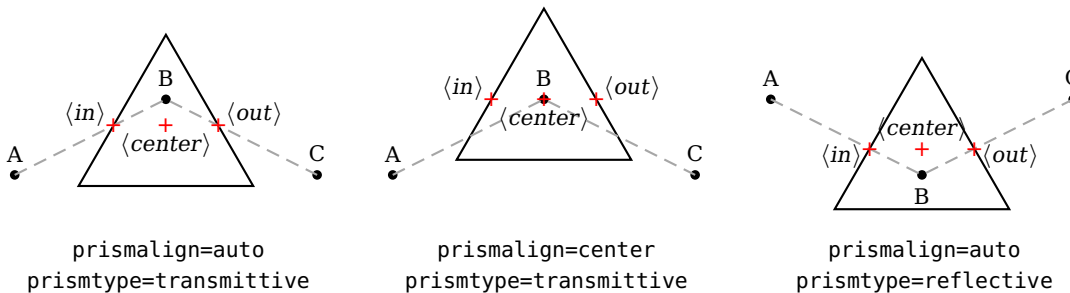


```
\begin{pspicture}(1.5,0.6)(5.1,2.2)
\optprism[prismtype=reflective](0,1)(3,0.9)(6,1)
\lens(1,2)(\oenableIfc{2}{1})
\lens(\oenableIfc{2}{1})(5,2)
\drawwidebeam[beamwidth=0.3, fillstyle=solid,
fillcolor=green, opacity=0.3](1,2){2}{1}{3}(5,2)
\end{pspicture}
```

**prismalign**=auto, center

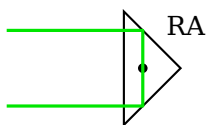
default: auto

Wählt die vertikale Ausrichtung des Prisma bezüglich des „Reflektionsknoten“  $\langle center \rangle$ . 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.



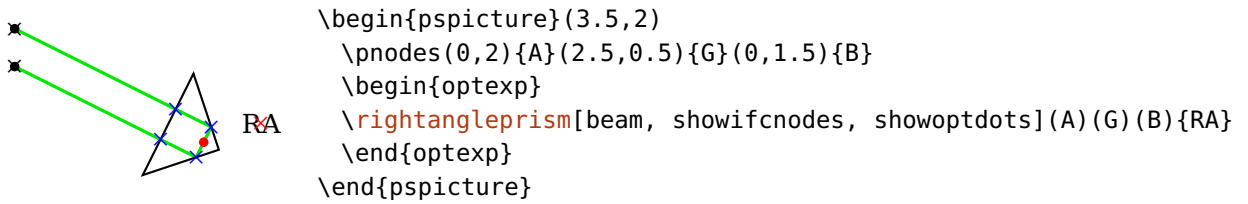
## 4.22. Umkehrprisma

**\rightangleprism**[ $\langle options \rangle$ ]( $\langle in \rangle$ )( $\langle center \rangle$ )( $\langle out \rangle$ ){ $\langle label \rangle$ }



```
\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  $\langle center \rangle$  Knoten vertikal zentriert in dem Prisma liegt.



`raprismsize=<num>`

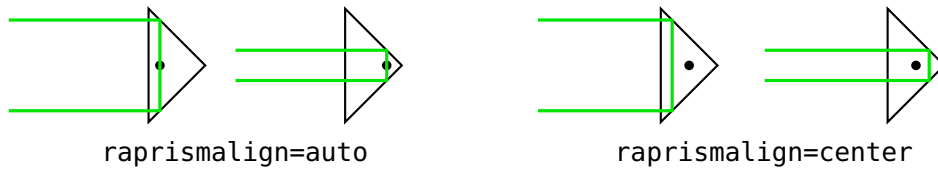
default: 1.5

Die Länge der Eingangsfläche.

`raprismalign=auto, center`

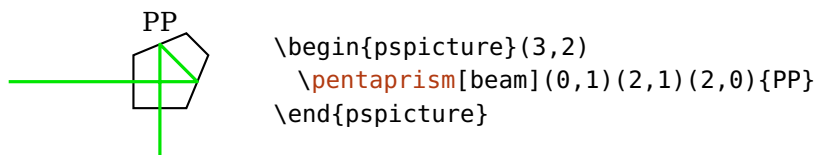
default: auto

Schaltet die automatische Positionierung der Komponente vertikal zur Eingangsfläche ein oder aus:



## 4.23. Pentaprisma

`\pentaprism[<options>](<in>)(<center>)(<out>){<label>}`



`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. Anstelle der automatischen Faserverbindungen können die Komponenten auch manuell mit Fasern verbunden werden, siehe Kap. 9.6.

Die meisten Komponenten benötigen nur zwei Referenzknoten und werden wie die Freistrahlezweipole 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. Hat Vorrang vor der Option `usewirestyle`, sind beide Optionen gesetzt, dann wird eine Faser gezeichnet.

`usewirestyle=true, false`

Kann verwendet werden, um interne Verbindungen als `Wire` darzustellen. Setzt `usefiberstyle` auf `false`, daher kann es sinnvoll sein, z.B. einen elektrischen Filter zu definieren:

```
\newpsobject{elecfilter}{optfilter}{usewirestyle}
```

### 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: 0.2

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}
```

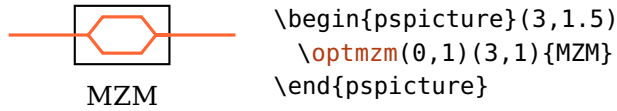
`optampsize=<num> or <width> <height>`

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*}

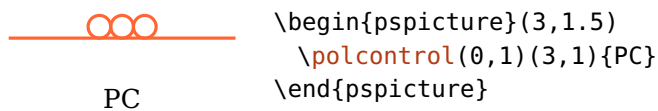


`optmzmsize`=*num* or *width* *height* default: 0.8

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*}

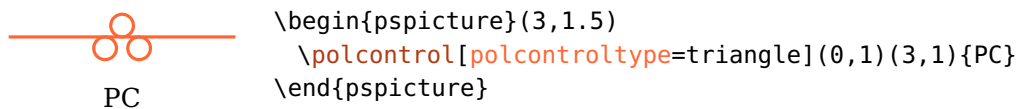


`polcontrolsize`=*num* default: 0.15

Der Radius der Polarisationssteller-Kreise.

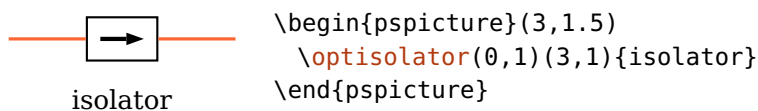
`polcontroltype`=linear, triangle default: linear

Der Typ des Polarisationsstellers.



### 5.5. Isolator

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



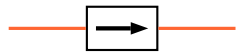


**isolatorsize**= $\langle num \rangle$  or  $\langle width \rangle \langle height \rangle$  default: 0.6

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**  $\langle psstyle \rangle$  default: linewidth=2\pslinewidth, arrowinset=0

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

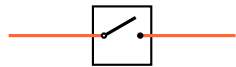


isolator

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

## 5.6. Optischer Schalter

**\optswitch**[ $\langle options \rangle$ ]( $\langle in \rangle$ )( $\langle out \rangle$ ){ $\langle label \rangle$ }



opened switch

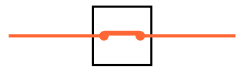
```
\begin{pspicture}(3,1.5)
\optswitch(0,1)(3,1){opened switch}
\end{pspicture}
```

**switchsize**= $\langle num \rangle$  or  $\langle width \rangle \langle height \rangle$  default: 0.8

Wird nur eine Zahl angegeben, so ist das die Seitenlänge des Schalters, andernfalls können Breite und Höhe getrennt angegeben werden.

**switchstyle**=opened, closed default: opened

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

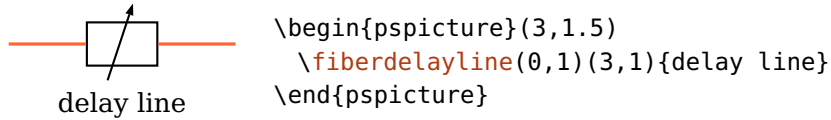


closed switch

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

## 5.7. Faserverzögerungstrecke

`\fiberdelayline` [`<options>`] (`<in>`) (`<out>`) {`<label>`}



`fdlsize`=`<num>` or `<width>` `<height>` default: 0.6

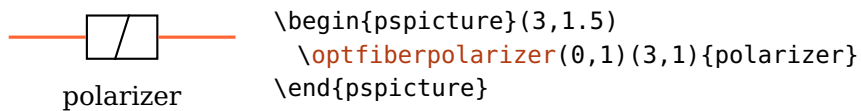
Eine einzelne Zahl gibt die Höhe der Komponente an, die Breite ist 1.6 mal die Höhe. Zwei Zahlen geben Breite und Höhe der Komponente direkt an.

`FdlArrow` `<psstyle>` default: `arrowinset=0`, `arrows=->`

Siehe `ArrowCompStyle`.

## 5.8. Polarisator

`\optfiberpolarizer` [`<options>`] (`<in>`) (`<out>`) {`<label>`}

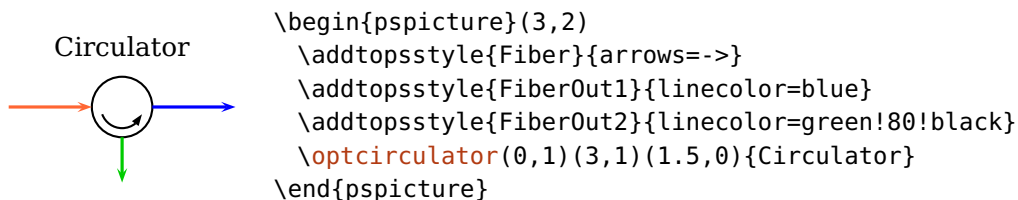


`fiberpolsize`=`<num>` or `<width>` `<height>` default: 0.6

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>`}



`optcircsize=<num>` default: 0.8

Der Durchmesser des Zirkulators.

`optcircangleA=<num>` default: -160

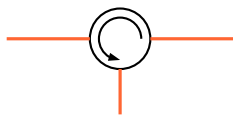
Der Startwinkel des internen Pfeils.

`optcircangleB=<num>` default: -20

Der Endwinkel des internen Pfeils.

`optcircangle=<num> <num>`

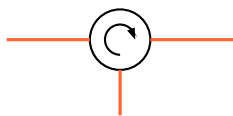
Kurzschreibweise um beide Winkel gleichzeitig zu setzen.



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

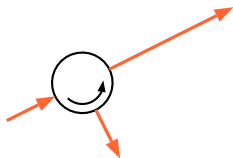
`OptCircArrow <psstyle>` default: unit=0.7, arrows=->, arrowinset=0

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



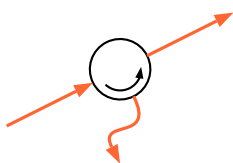
```
\begin{pspicture}(3,1.5)
\newpsstyle{OptCircArrow}{unit=0.5, arrows=<-, arrowinset=0}
\optcirculator[optcircangle=0 -90](0,1)(3,1)(1.5,0)
\end{pspicture}
```

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.

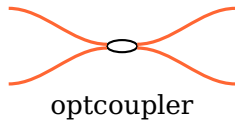


```
\begin{pspicture}(3,2)
\addtopsstyle{Fiber}{arrowscale=1.3, arrows=->, arrowinset=0,
ncurv=1.5}
\optcirculator[position=0.5](0,0.5)(3,2)(1.5,0)
\end{pspicture}
```

## 5.10. Faserkoppler

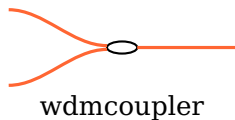
Es stehen drei Faserkoppler zur Verfügung, die alle die gleichen Formen und Parameter verwenden.

`\optcoupler`( $\langle tl \rangle$ )( $\langle bl \rangle$ )( $\langle tr \rangle$ )( $\langle br \rangle$ ){ $\langle label \rangle$ }



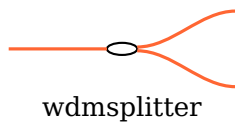
```
\begin{pspicture}(0,-0.4)(3,1)
  \optcoupler(0,1)(0,0)(3,1)(3,0){optcoupler}
\end{pspicture}
```

`\wdmcoupler`( $\langle tl \rangle$ )( $\langle bl \rangle$ )( $\langle r \rangle$ ){ $\langle label \rangle$ }



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

`\wdmsplitter`( $\langle l \rangle$ )( $\langle tr \rangle$ )( $\langle br \rangle$ ){ $\langle label \rangle$ }



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

`couplersize`= $\langle num \rangle$  or  $\langle width \rangle$   $\langle height \rangle$

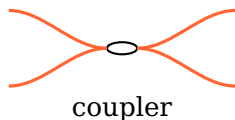
default: 0.2

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`= $\langle num \rangle$

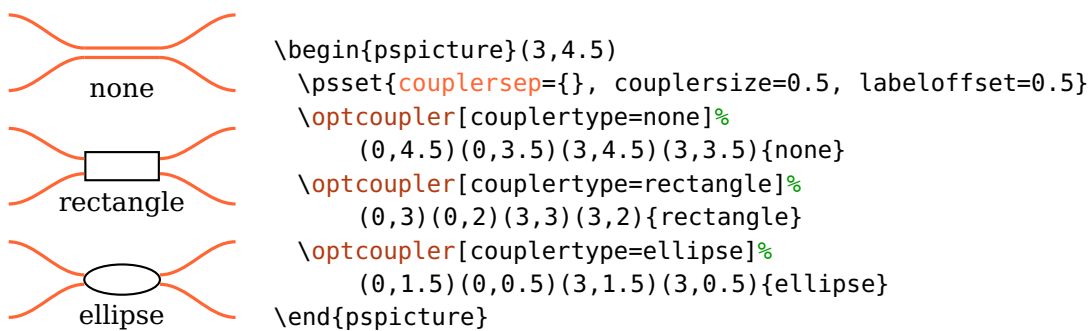
default: 0.05

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}
```

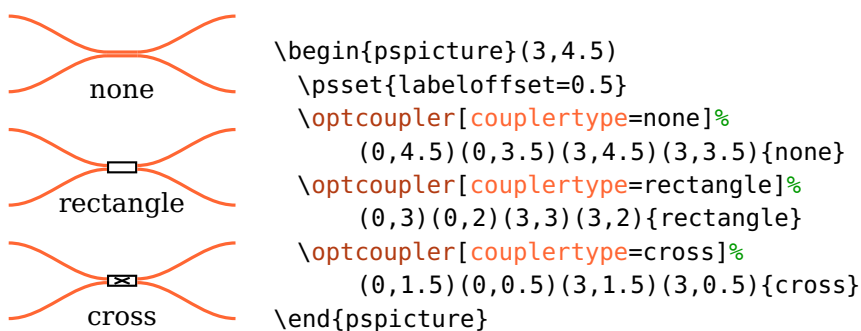
Wird dieser Wert auf {} gesetzt, dann wird der Abstand automatisch berechnet, abhängig von `couplertype` und `couplersize`.



`couplertype`=none, ellipse, rectangle, cross

default: ellipse

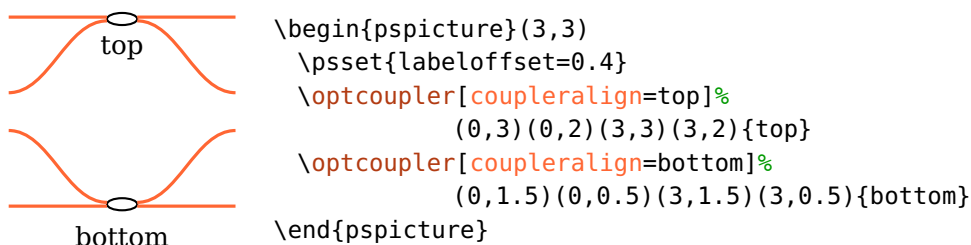
Wählt zwischen unterschiedlichen Kopplertypen.



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

default: center

Die Ausrichtung des Kopplers bezüglich der Referenzknoten. Beachten Sie, dass die Position der Komponentenmitte und damit der Beschriftung von dieser Einstellung abhängt (siehe 8.3).



`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.

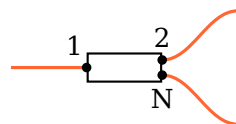
Jeder Koppler hat eine Variante, die nur zwei Knoten benötigt, einen Eingangs- und einen Ausgangsknoten. `\optcoupler` verwendet jeden dieser Knoten zweimal, `\wdmsplitter` verwendet den Knoten *<out>* zweimal und `\wdmcoupler` verwendet *<in>* zweimal. Das kann sehr praktisch sein, wenn man für einen Koppler die automatischen Verbindungen nicht verwendet, sondern die Komponente manuell verbindet.



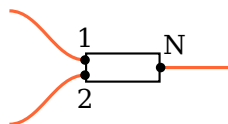
```
\begin{pspicture}(4,2)
  \pnodes(0,1){A}(4,1){B}
  \psset[optexp]{fiber=none}
  \wdmsplitter[position=0.2, coupleralign=b](A)(B)
  \wdmcoupler[position=0.8, coupleralign=t](A)(B)
  \drawfiber{(A)}{1}{2}{(B)}
  \drawfiber{1}{([offset=0.5, Xnodesep=1]\oenableIfc{2}{1})}
  \drawfiber{([offset=-0.5, Xnodesep=-1]\oenableIfc{2}{2})}{2}
\end{pspicture}
```

### 5.10.1. Eingangs- und Ausgangsknoten

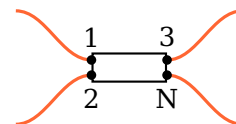
Die Definition der Eingangs- und Ausgangsknoten aus Kap. 8.6 ist für Koppler nicht anwendbar. Hier werden die Knoten einfach von 1 (links oben) bis N (rechts unten) durchnummeriert.



`\wdmsplitter`



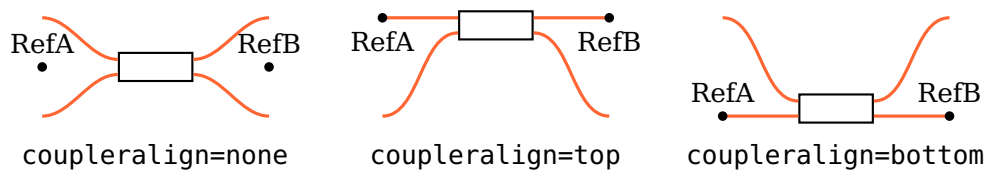
`\wdmcoupler`



`\optcoupler`

### 5.10.2. Referenzknoten

Die Definition der Referenzknoten aus Kap. 8.2 ist für Koppler nicht anwendbar und hängt von `coupleralign` ab.



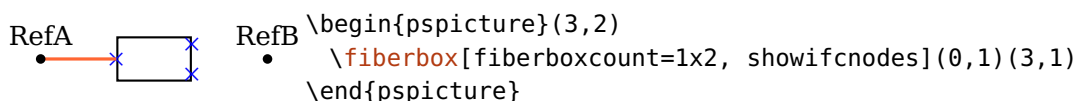
## 5.11. Faserbox

`\fiberbox(<in>)(<out>){<label>}`

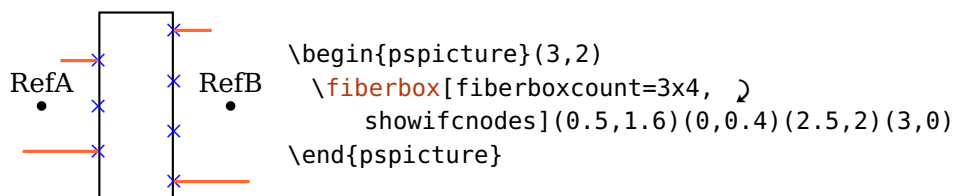
`\fiberbox(<tl>)(<bl>)(<tr>)(<br>){<label>}`

Eine `\fiberbox` kann entweder mit zwei, oder mit vier Knoten positioniert werden. Die Wahl beeinflusst die Position der Komponente und der Komponentenknoten und die automatischen Faserverbindungen.

Werden nur zwei Knoten verwendet, dann sind das die Referenzknoten und der Abstand zwischen zwei benachbarten Eingangs- oder Ausgangsknoten muss manuell angegeben werden (siehe `fiberboxsepin` und `fiberboxsepout`). Nur die Seiten der Komponente wird automatisch mit Fasern verbunden, die nur einen einzigen Knoten hat (siehe `fiberboxcount` und das folgende Beispiel).



Werden vier Knoten verwendet, dann wird die Lage der Komponentenknoten und deren Abstand automatisch bestimmt. Die Komponente wird zwischen der Mitte der beiden Eingangsknoten und der Mitte der beiden Ausgangsknoten positioniert. Der erste Eingangsknoten wird an dem `<tl>`-Knoten ausgerichtet, der letzte Eingangsknoten an dem `<bl>`-Knoten. Die übrigen Eingangsknoten werden dazwischen gleichverteilt. Die Ausgangsknoten werden entsprechend behandelt. Die vier angegebenen Knoten werden automatisch mit den jeweils zugeordneten Komponentenknoten verbunden.

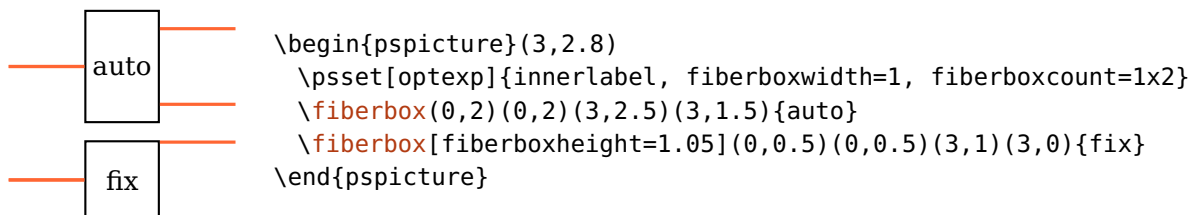


`fiberboxwidth`= $\langle num \rangle$  default: 1

Die Breite der Faserbox.

`fiberboxheight`= $\langle num \rangle$  default: 0

Die Höhe der Faserbox. Ist die Höhe kleiner als der Platz der von den Eingangs- oder Ausgangsknoten benötigt wird, dann wird die Höhe automatisch berechnet. Die berechnete Höhe ist  $\max(\text{sepin} \times (N - 1), \text{sepout} \times (M - 1))$ .



`fiberboxsize`= $\langle width \rangle \langle height \rangle$

Die Breite und Höhe der Faserbox, ist äquivalent zum Aufruf von `fiberboxwidth` und `fiberboxheight`.

`fiberboxsepin`= $\langle num \rangle$  default: 0.2

Der Abstand zwischen zwei Eingangsknoten. Dieser Wert wird nur verwendet, falls die Faserbox mit zwei Knoten positioniert wird. Ansonsten wird der Abstand automatisch ermittelt wird.

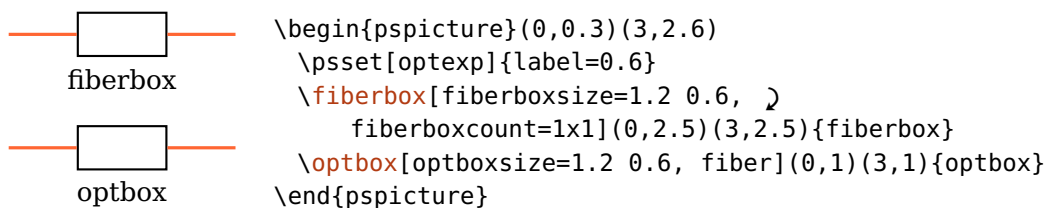
`fiberboxsepout`= $\langle num \rangle$  default: 0.2

Analog zu `fiberboxsepin`, aber für die Ausgangsknoten.

`fiberboxcount`= $\langle N \rangle \times \langle M \rangle$  default: 2x2

Anzahl der Eingangs- ( $\langle N \rangle$ ) und Ausgangsknoten ( $\langle M \rangle$ ).

Für bestimmte Parameterkombinationen ist eine `\fiberbox` äquivalent zu einer `\optbox`.

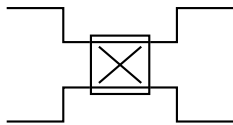




## 6. Elektrische Komponenten

### 6.1. Koppler

`\eleccoupler` ( $\langle tl \rangle$ ) ( $\langle bl \rangle$ ) ( $\langle tr \rangle$ ) ( $\langle br \rangle$ ) { $\langle label \rangle$ }

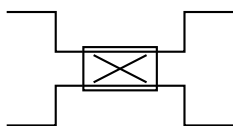


```
\begin{pspicture}(3,1.5)
\eleccoupler(0,1.5)(0,0)(3,1.5)(3,0)
\end{pspicture}
```

`eleccouplersize`= $\langle size \rangle$  or  $\langle width \rangle$   $\langle height \rangle$

default: 0.8

Die Breite und Höhe des Kopplers, wird nur ein Wert angegeben dann ist die Komponente quadratisch.



```
\begin{pspicture}(3,1.5)
\eleccoupler[eleccouplersize=1 0.6](0,1.5)(0,0)(3,1.5)(3,0)
\end{pspicture}
```

`eleccouplersep`= $\langle num \rangle$

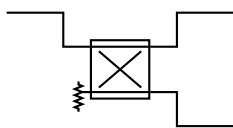
default: {}

Der vertikale Abstand zwischen den beiden Eingangs- bzw. Ausgangsanschlüssen. Ist der Wert leer, bzw. gleich {}, dann wird der Abstand auf 75% der Höhe gesetzt.

`eleccouplertype`=standard, directional

default: standard

Wählt zwischen unterschiedlichen Kopplertypen.

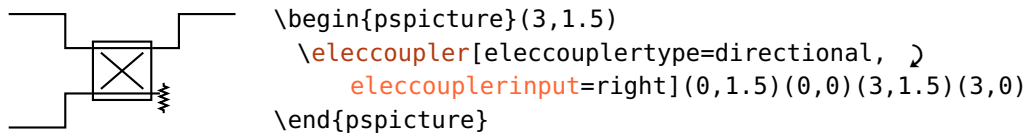


```
\begin{pspicture}(3,1.5)
\eleccoupler[eleccouplertype=directional](0,1.5)(0,0)(3,1.5)(3,0)
\end{pspicture}
```

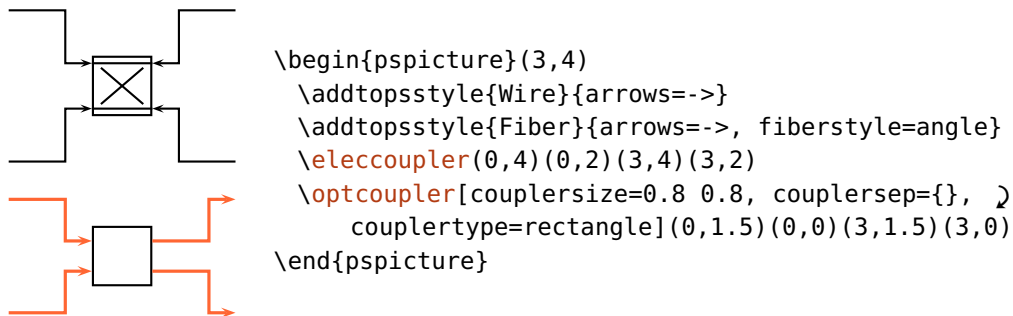
`eleccouplerinput`=left, right

default: left

Gibt beim direktionalen Koppler die Eingangsseite an.

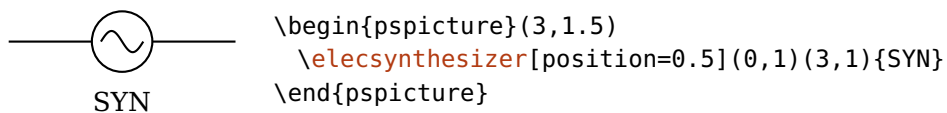
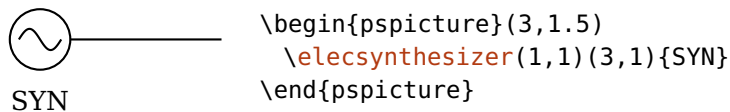


Die automatischen Verbindungen des elektrischen Kopplers sind, anders als beim optischen Koppler, alle auf die Komponente gerichtet. Das ergibt ein symmetrisches Layout der automatischen Verbindungen des elektrischen Kopplers bei Verwendung der Vorgabe `wirestyle=angle`.



## 6.2. Synthesizer

`\elecsynthesizer`( $\langle in \rangle$ )( $\langle out \rangle$ )( $\langle label \rangle$ )

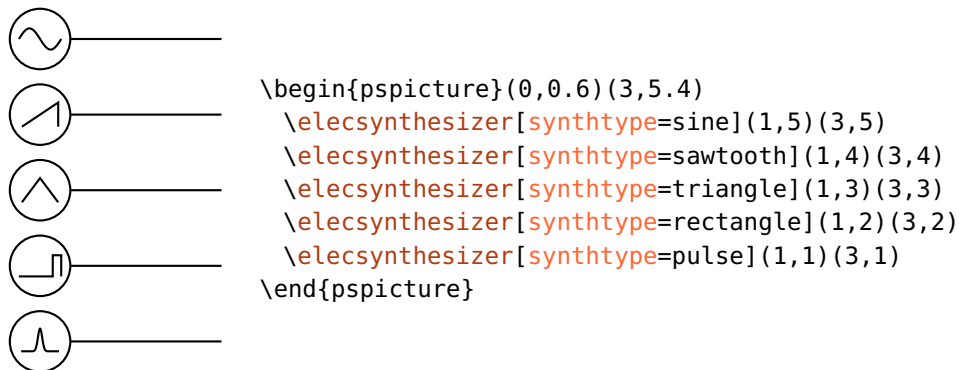


`synthsize`= $\langle size \rangle$  or  $\langle width \rangle$   $\langle height \rangle$  default: 0.8

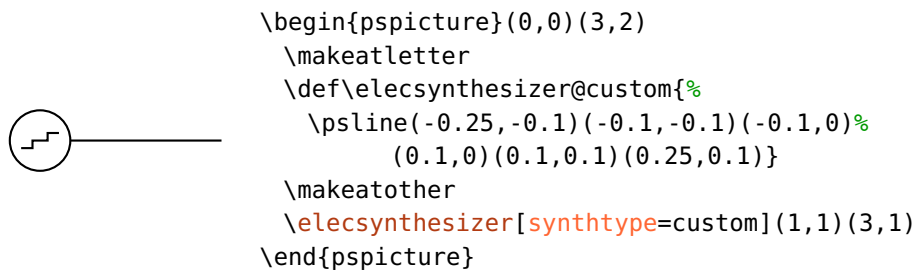
Die Größe der Komponente, ist diese rund und wurden zwei Werte angegeben, dann wird der kleinere als Durchmesser verwendet.

`synthtype`=sine, pulse, sawtooth, rectangle, triangle, custom default: sine

Wählt zwischen unterschiedlichen Typen.



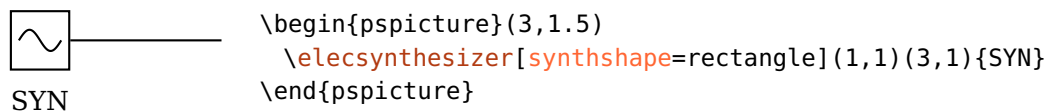
Es kann auch eine frei wählbare Wellenform gewählt werden.



`synthshape=circle, rectangle`

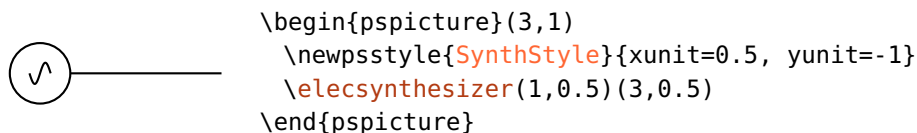
default: circle

Wählt zwischen kreisförmiger und rechteckiger Außenform.



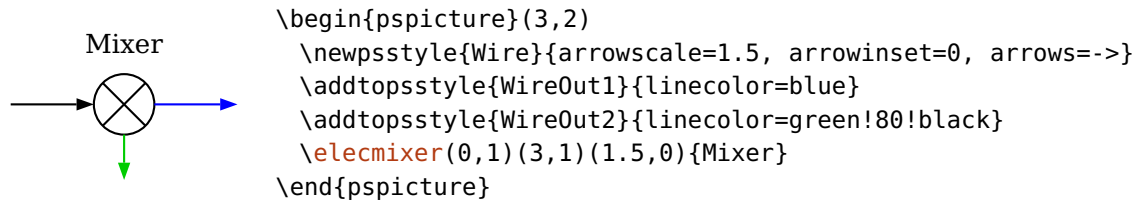
`SynthStyle`  $\langle psstyle \rangle$

Beinflusst das Aussehen der internen Symbole.



## 6.3. Mixer

`\elec mixer` ( $\langle left \rangle$ ) ( $\langle right \rangle$ ) ( $\langle bottom \rangle$ ) { $\langle label \rangle$ }



`elecmmixer`size= $\langle num \rangle$

default: 0.8

Der Durchmesser des Mixers.

Weitere Details bezüglich der Referenzknoten und Positionierung finden Sie unter `\optccirculator`.


## 7. Hybridkomponenten

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

Der `\optfilter` wird mit den Voreinstellungen zwar als Faserkomponente gehandhabt, kann jedoch im Gegensatz zu den reinen Faserkomponenten (Kap. 5) auch als Freistrahlskomponente verwendet werden. Der `\fibercollimator` hat nur eine Faser, die andere Verbindung ist Freistrahls und der `\optdetector` hat eine Freistrahls- und eine elektrische Verbindung.

### 7.1. 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

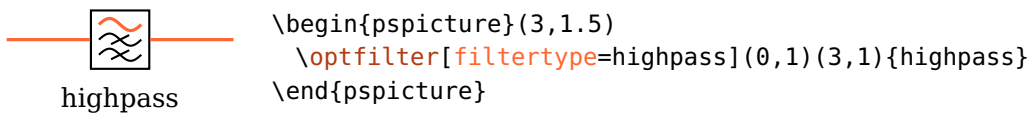
Wähle zwischen unterschiedlichen Filtertypen.

  
bandstop

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

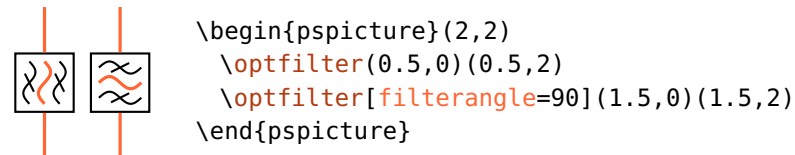
  
lowpass

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



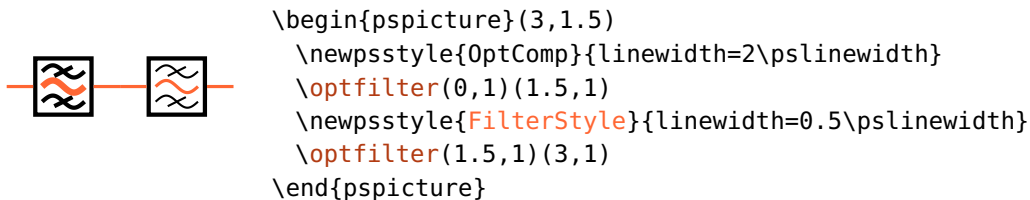
`filterangle`= $\langle num \rangle$  default: 0

Dreht das „Innere“ des Filters relativ zum seinem Rahmen. Anstatt dessn kann auch `innercompalign` verwendet werden.

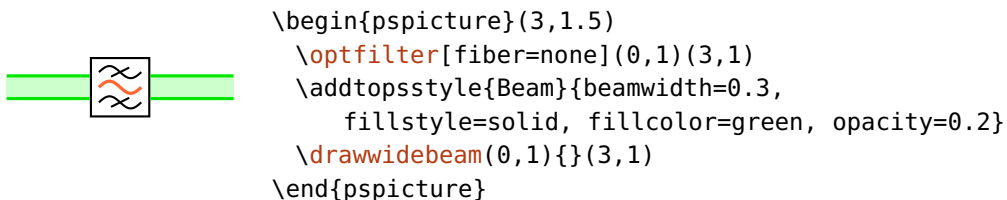


`FilterStyle`  $\langle psstyle \rangle$

Beinflusst das Aussehen der internen Filterlinien.

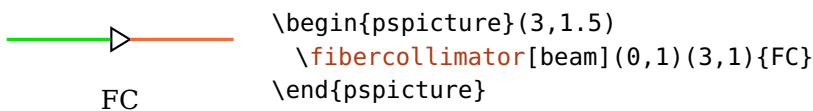


Verwendung als Freistrahalkomponenten, `allowbeaminside` ist auf false voreingestellt:

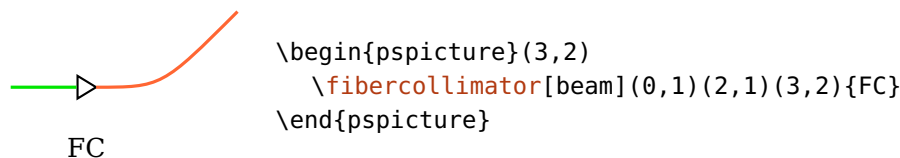


## 7.2. Faserkollimator

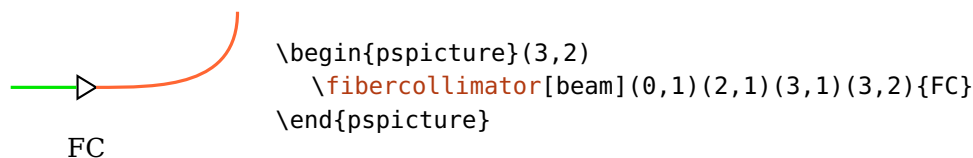
`\fibercollimator` ( $\langle in \rangle$ ) ( $\langle A \rangle$ ) ( $\langle B \rangle$ ) ( $\langle out \rangle$ ) { $\langle label \rangle$ }



Der Faserkollimator kann mit zwei, drei oder vier Punkten verwendet werden. Bei zwei Punkten wird der Kollimator wie jeder andere Zweipol zwischen  $\langle in \rangle$  und  $\langle out \rangle$  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  $\langle in \rangle$  und  $\langle A \rangle$  Knoten auszurichten.



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.



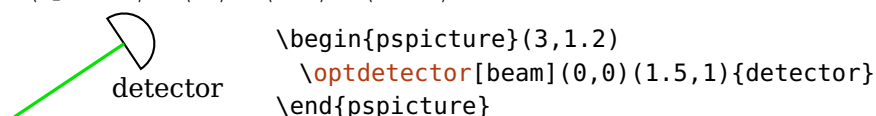
`\fibercolsize`= $\langle num \rangle$  or  $\langle width \rangle \langle height \rangle$

default: 0.3

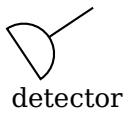
Eine einzelne Zahl gibt die Seitenlänge des Kollimators an, zwei Zahlen die Breite und Höhe. Beachten Sie, dass `\fibercolsize=1` und `\fibercolsize=1 1` nicht dasselbe Ergebnis liefern. Alternativ kann das Verhältnis zwischen Höhe und Breite mit `xunit` und `yunit` geändert werden.

## 7.3. Detektor

`\optdetector`[ $\langle options \rangle$ ]( $\langle in \rangle$ )( $\langle out \rangle$ ){ $\langle label \rangle$ }



Der Detektor wird in der Voreinstellung ans Ende der Referenzlinie platziert (`\position=end`). Wird dieser Parameter überschrieben, dann wird die ausgehende Verbindung wie bei einer elektrischen Komponente behandelt.



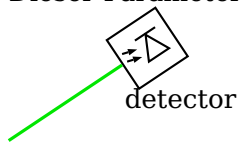
```
\begin{pspicture}(3,1.2)
  \optdetector[position=0.5](0,0)(1.5,1){detector}
\end{pspicture}
```

**detsize**= $\langle num \rangle$  or  $\langle width \rangle \langle height \rangle$  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 `detype=round` nicht dasselbe ist.

**detype**=round, diode default: round

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



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

**DetectorStyle**  $\langle psstyle \rangle$

Der Stil der Diode für `detype=diode`.



```
\begin{pspicture}(3,1.5)
  \newpsstyle{OptComp}{linewidth=3\pslinewidth}
  \newpsstyle{DetectorStyle}{linewidth=0.333\pslinewidth}
  \optdetector[detype=diode](0,1)(2,1)
\end{pspicture}
```



## 8. 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. 8.1). 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`

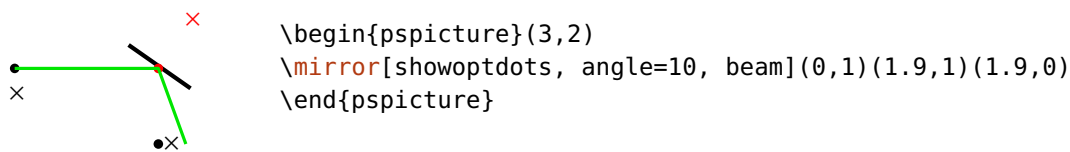
default: new

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. 8.2), der rote Punkt ist der Mittelpunkt-knoten (Kap. 8.3) und das rote Kreuz der Beschriftungsknoten (Kap. 8.4).



## 8.1. Komponenten-Bezeichner

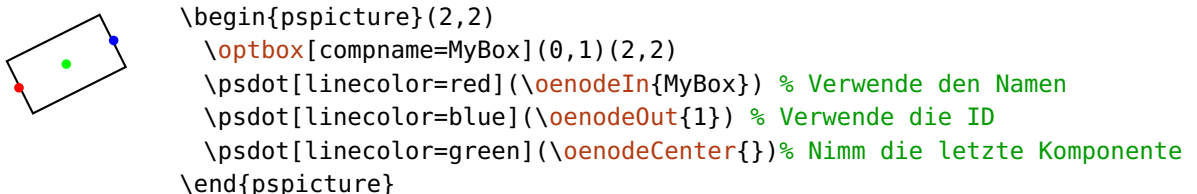
Alle Komponenten einer Aufbauskitze werden in Reihenfolge ihrer Definition im Code aufsteigend nummeriert, angefangen bei 1. Die Komponenten und deren Spezialknoten können immer über diese Nummer (ID) referenziert werden.

IDs können auch relativ zur aktuellen ID über die Syntax  $\langle num \rangle$  angegeben werden:  $\langle 1$  ist die vorletzte ID,  $\langle 2$  die vorvorletzte ID usw. Die vorletzte ID kann abgekürzt auch mit  $\langle$  erhalten werden, und die aktuelle ID mit  $\langle 0$ , was äquivalent zu einem leeren Parameter ist.

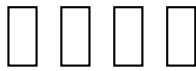
`compname`= $\langle string \rangle$

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.

Beispiele für unterschiedliche Methoden auf ein und dieselbe Komponente zuzugreifen.



Die Verwendung von relativen IDs ist u.a. nützlich, um eine neue Komponente relativ zur vorangegangenen zu definieren. Die Verwendung von z.B. `\oenameOut{}` in den Referenzknoten ist nicht möglich, da hiermit zum Zeitpunkt der Auswertung der Koordinaten schon auf die neue, nicht fertige Komponente referenziert wird. Ein Aufruf der Art `\optbox(\oenameOut{})(B)` ist daher nicht möglich, und führt zu einem Fehler. Mit relativen IDs kann das allerdings mit `\optbox(\oenameOut{<})(B)` erreicht werden:



```
\begin{pspicture}(4,2)
  \psset{optboxsize=0.4 0.8, abspos=0.5}
  \optbox(0,1)(4,1)
  \multido{}{3}{\optbox(\oenableOut{<})(4,1)}
\end{pspicture}
```

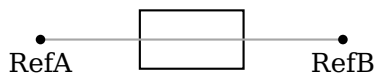
## 8.2. Referenzknoten

`\oenableRefA{<comp>}`

`\oenableRefB{<comp>}`

Die Eingangs- und Ausgangs-Referenzknoten.

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



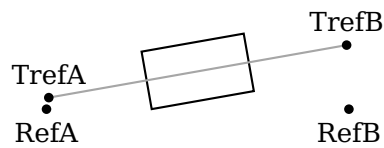
```
\begin{pspicture}(5,1)
  \pnodes(0.5,0.5){A}(4.5,0.5){B}
  \optbox(A)(B)
  \psline[style=Refline](\oenableRefA{ })(\oenableRefB{ })
  \psdot(\oenableRefA{ })\uput[-90](\oenableRefA{ }){RefA}
  \psdot(\oenableRefB{ })\uput[-90](\oenableRefB{ }){RefB}
\end{pspicture}
```

`\oenableTrefA{<comp>}`

`\oenableTrefB{<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.

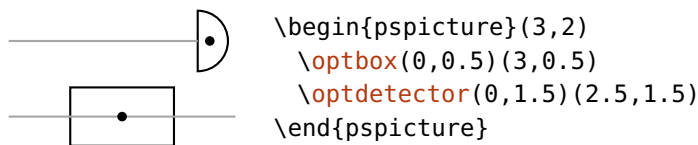


```
\begin{pspicture}(5,1.8)
  \pnodes(0.5,0.5){A}(4.5,0.5){B}
  \optbox[compshift=0.5, angle=10](A)(B)
  \psdot(\oencodeRefA{})\uput[-90](\oencodeRefA){RefA}
  \psdot(\oencodeRefB{})\uput[-90](\oencodeRefB){RefB}
  \psdot(\oencodeTrefA{})\uput[90](\oencodeTrefA){TrefA}
  \psdot(\oencodeTrefB{})\uput[90](\oencodeTrefB){TrefB}
\end{pspicture}
```

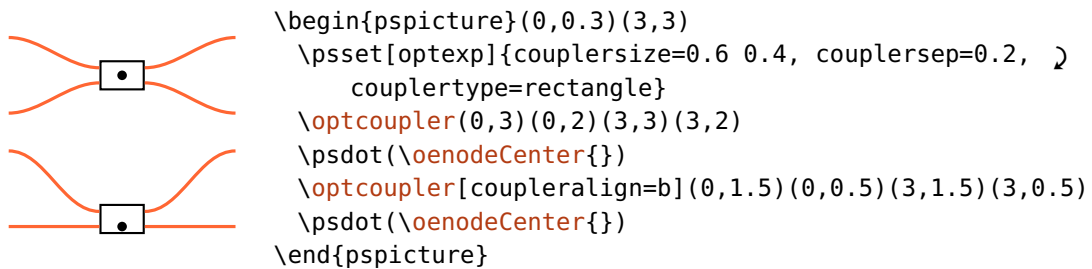
### 8.3. Mittelpunktknoten

`\oencodeCenter{<comp>}`

Dieser Knoten liegt, bis auf wenige Ausnahmen, wie z.B. `\optdetector` und `\optcoupler` (siehe unten), im Mittelpunkt der Komponente.



Bei `\optcoupler` hängt die Position der Komponentenmitte von den `coupleralign` Einstellungen ab.

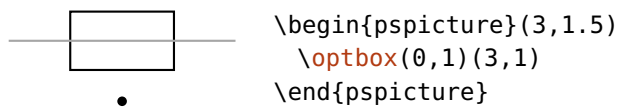


### 8.4. Beschriftungsknoten

`\oencodeLabel{<comp>}`

Auf diesen Knoten wird die Beschriftung platziert. Der Knoten ist auch verfü-

bar wenn keine Beschriftung angegeben wurde. Für `labeloffset=0` ist dieser Knoten identisch mit dem Mittelpunktsknoten (Kap. 8.3).



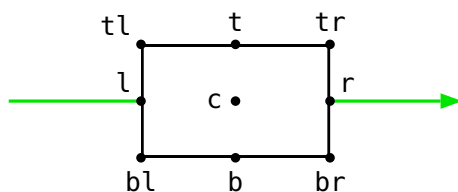
## 8.5. Externe Knoten

`\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. 12.2.1 zusammengefasst.

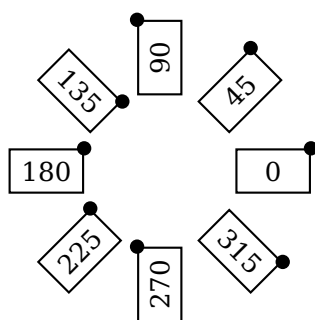


`extnodealign=rel, relative, abs, absolute`

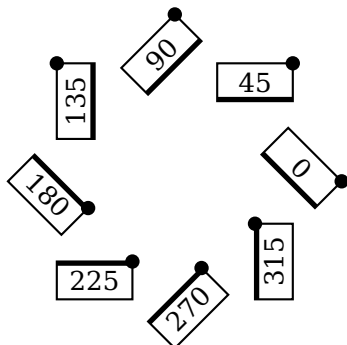
default: abs

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.

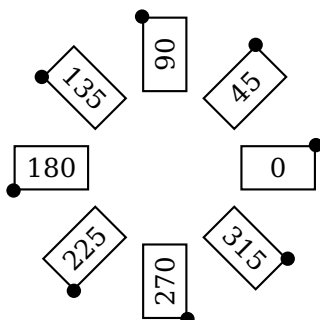


```
\begin{pspicture}(-2,-2)(2,2)
\psset{endbox, optboxsize=1 0.6, dotscale=1.5}
\psset{extnodealign=abs, extnode=tr}
\multido{\i=0+45}{8}{%
\optbox[innerlabel](0,0)(1;\i){\i}
\psdot(\oenodeExt{ })
}%
\end{pspicture}
```

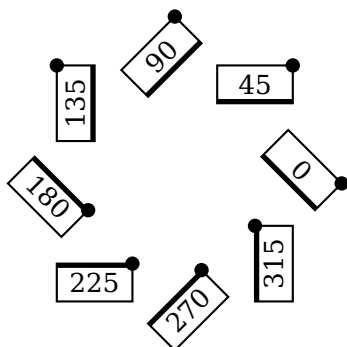


```
\begin{pspicture}(-2.2,-2.2)(2.2,2.2)
\psset{dotscale=1.5, mirrortype=extended, 2
mirrordepth=0.5, label=0.25 . . relative, 2
extnodealign=abs, extnode=tr}
\newpsstyle{ExtendedMirror}{}
\multido{\i=0+45,\ii=-45+45}{8}{%
\mirror(0,0)(1.5;\i)(2.1212;\ii){\i}
\psdot(\oenodeExt{ })
}%
\end{pspicture}
```

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



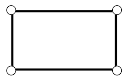
```
\begin{pspicture}(-2,-2)(2,2)
\psset{endbox, optboxsize=1 0.6, dotscale=1.5}
\psset{extnodealign=rel, extnode=tr}
\multido{\i=0+45}{8}{%
\optbox[innerlabel](0,0)(1;\i){\i}
\psdot(\oenodeExt{ })
}%
\end{pspicture}
```



```
\begin{pspicture}(-2.2,-2.2)(2.2,2.2)
\psset{dotscale=1.5, mirrortype=extended, 2
mirrordepth=0.5, label=0.25 . . relative, 2
extnodealign=abs, extnode=tr}
\newpsstyle{ExtendedMirror}{}
\multido{\i=0+45,\ii=-45+45}{8}{%
\mirror(0,0)(1.5;\i)(2.1212;\ii){\i}
\psdot(\oenodeExt{ })
}%
\end{pspicture}
```

`extnodes=<list>`

Erlaubt die Definition mehrerer externer Knoten,  $\langle list \rangle$  ist eine Liste von  $\langle refpoint \rangle$ , wie sie von `extnode` verwendet werden. Der erste Knoten kann wie gewohnt mit `\oenodeExt` angesprochen werden (ist äquivalent zu `\oenode{Ext}` und `\oenode{Ext1}`). Für alle weiteren Knoten muss `\oenode` mit den Knotenbezeichnungen `Ext<num>` verwendet werden: `\oenode{Ext1}` für den ersten, `\oenode{Ext2}` für den zweiten usw.

Ext1		Ext2	<code>\begin{pspicture}(3,2)</code>
			<code>\optbox[extnodes={tl,tr,br,bl}](0,1)(3,1)</code>
Ext4		Ext3	<code>\multido{\i=1+1}{4}{\psdot[dotstyle=o](\oenode{Ext\i})}%</code>
			<code>\end{pspicture}</code>

## 8.6. Grenzflächenknoten

`\oenodeIfc{<num>}{<comp>}`

Der Grenzflächenknoten  $\langle num \rangle$ , wobei  $\langle num \rangle$  eine positive Ganzzahl oder der Buchstabe N ist. Der letzte Knoten ist immer N, kann aber auch über die entsprechende Zahl angesprochen werden.

`\oenodeIn{<comp>}`

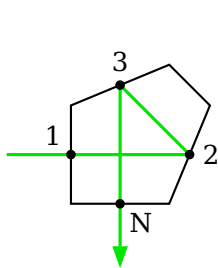
Der Knoten  $\langle 1 \rangle$  ist der Eingangsknoten und sollte immer über `\oenodeIn` angesprochen werden.

`\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 des Ausgangsknotens. Diese Bezeichnung kann nur für Faserkopppler (siehe Kap. 5.10.1) und Strahlteiler (4.17) 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.



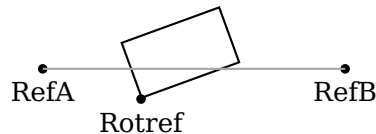
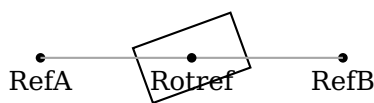
```
\begin{pspicture}(2.8,3)
\pentaprism[pentaprismsize=1.3](0,1.5)(1.5,1.5)(1.5,0)
\drawbeam[arrows=->, arrowinset=0, 2
arrowscale=1.5](0,1.5){}(1.5,0)
\psdot(\oenableIn{})\uput[135](\oenableIn{}){1}
\psdot(\oenableIfc{2}{})\uput[0](\oenableIfc{2}{}){2}
\psdot(\oenableIfc{3}{})\uput[90](\oenableIfc{3}{}){3}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

In Kap. 12.2.2 finden Sie eine vollständige Liste aller Komponenten mit ihren Grenzflächenknoten.

## 8.7. Referenzknoten für die Rotation

`\oenableRotref{<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. 12.2.1.



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

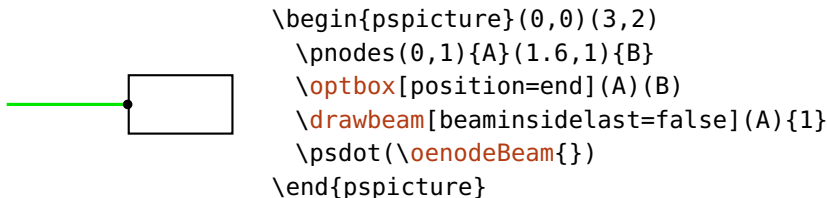
## 8.8. Strahlknoten

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

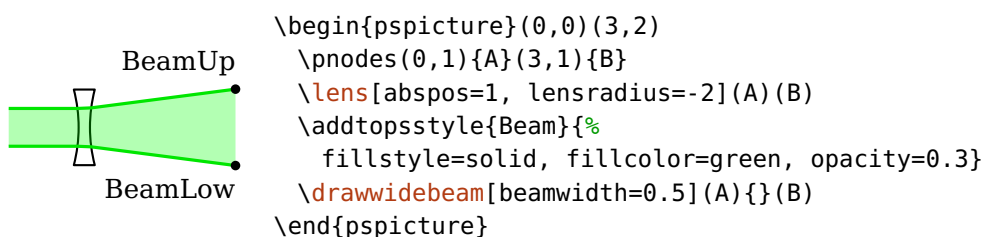


`\oenodeBeam{⟨num⟩}`

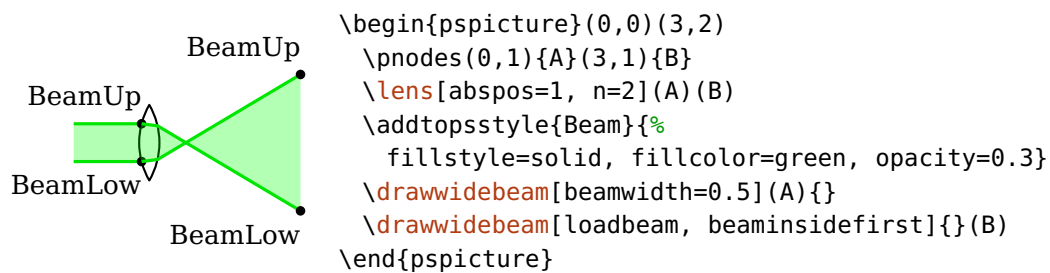
Damit wird der Endknoten eines vorangegangenen `\drawbeam` Kommandos angesprochen. Wird `⟨num⟩` leer gelassen so wird 1 genommen, was in der Regel dem letzten Strahlengang entspricht. Auf welchen Strahl genau Bezug genommen wird, hängt i.a. von den gewählten Einstellungen von `savebeampoints` und entsprechender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

`\oenodeBeamUp{⟨num⟩}``\oenodeBeamLow{⟨num⟩}`

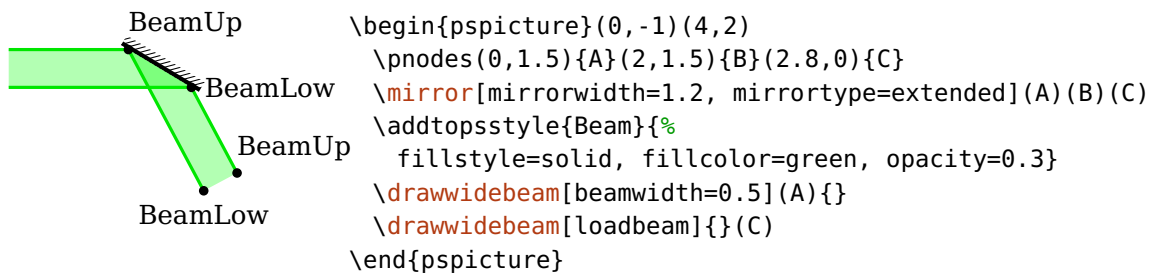
Damit wird der obere („upper“) oder untere („lower“) Endknoten eines vorangegangenen `\drawwidebeam` Kommandos angesprochen. Wird `⟨num⟩` leer gelassen so wird 1 genommen, womit man sich in der Regel auf den letzten Strahlengang bezieht. Auf welchen Strahl genau Bezug genommen wird, hängt i.a. von den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



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:



Bei einer Reflexion können ebenfalls die Rollen vertauscht werden.



Beachten Sie, dass sich die Orientierung von „oben“ und „unten“ in Version 3.3 geändert hat, siehe Kap. 12.3.5.

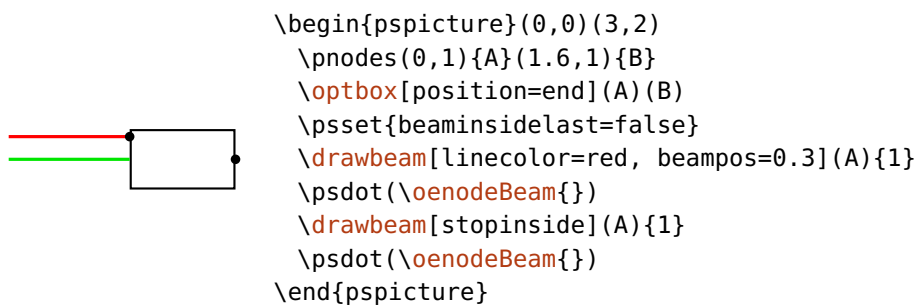
`\oeBeamCenter{<num>}`

Damit erhält man die Mitte zwischen oberem und unterem Endknoten, die Koordinaten sind nur aus Postscript-Ebene verfügbar, im Gegensatz zu `\oenodeBeam*`, daher auch die an `\oeBeamVec*` angelehnte Nomenklatur. Ein Anwendungsbeispiel ist Bsp. 8.1.

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 undefiniert werden:

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

Desweiteren werden die Endknoten durch `stopinside` beeinflusst. Siehe Kap. 9.5 für weitere Details.



## 8.9. Strahlvektor

Zusätzlich zu den Strahlknoten (siehe Kap. 8.8) kann auf Postscript-Ebene auf die normierten Strahlvektoren der letzten gespeicherten Strahlen (siehe `savebeam`) zugegriffen werden.

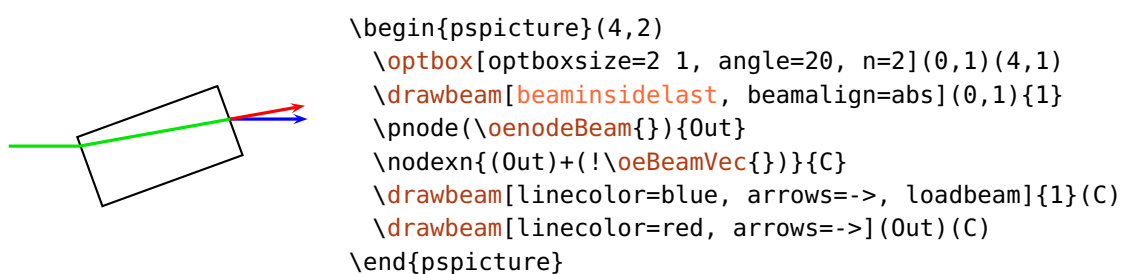
Auch hier wird zwischen Einzelstrahlen, und oberem bzw. unterem Strahl unterschieden. `<num>` hat dieselbe Bedeutung wie bei `\oenodeBeam`.

`\oeBeamVec{<num>}`

`\oeBeamVecUp{<num>}`

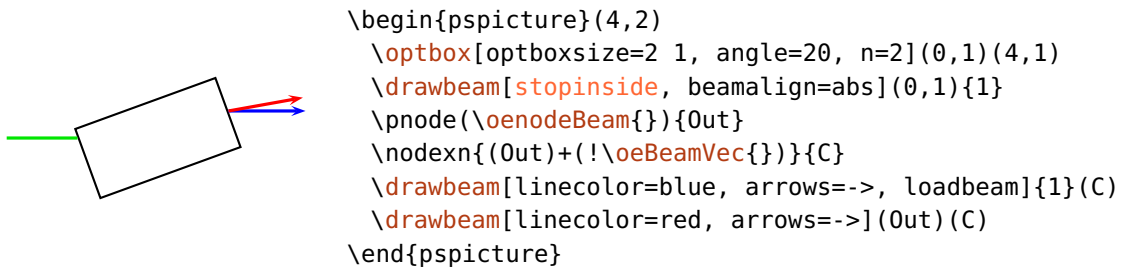
`\oeBeamVecLow{<num>}`

Bei den gespeicherten Werten handelt es sich um den normierten Richtungsvektor des letzten Strahlsegments, dieser ist nur auf Postscript-Ebene verfügbar. Dieser Richtungsvektor ist im folgenden Beispiel in rot zu sehen. Im Gegensatz dazu wird mit `savebeam` der Richtungsvektor gespeichert, der nach der letzten Grenzfläche gelten würde, also für einen nachfolgenden Strahl verwendet würde. Dieser ist im Beispiel in blau eingezeichnet.

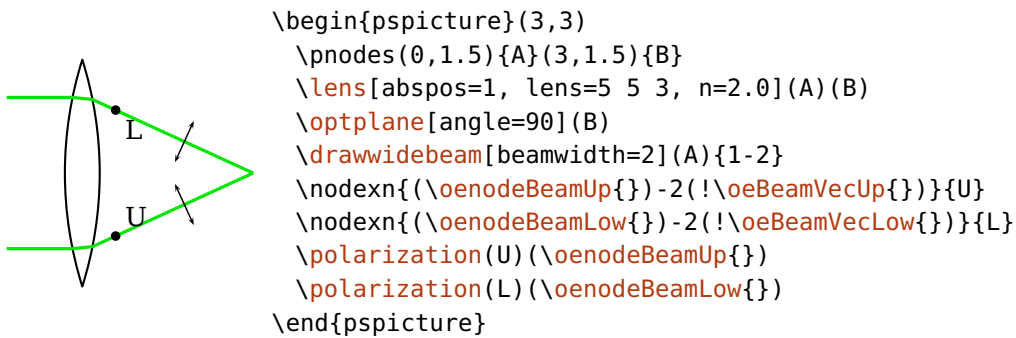


Das Verhalten ist unabhängig davon, ob der letzte Strahl nur berechnet, oder auch gezeichnet wurde. Das nächste Beispiel verwendet `stopinside` anstelle

vom Parameter `beaminsidelast` im vorangegangenen Beispiel; die Strahlvektoren bleiben davon unbeeinflusst.

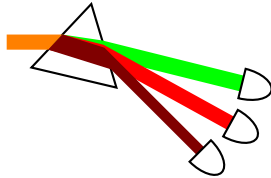


Das kann dazu genutzt werden, um z.B. die Polarisation eines Strahls zu kennzeichnen. Im folgenden Beispiel werden die neuen Knoten  $\langle U \rangle$  und  $\langle L \rangle$  definiert, indem mit `\nodexn` die Endpunkte des oberen bzw. unteren Strahls entlang ihrer Trajektorie zurück verschoben werden.



`\oeBeamVecMedian` $\{\langle num \rangle\}$  Der Mittelwert aus oberem und unterem Strahlvektor.

## Bsp. 8.1



```

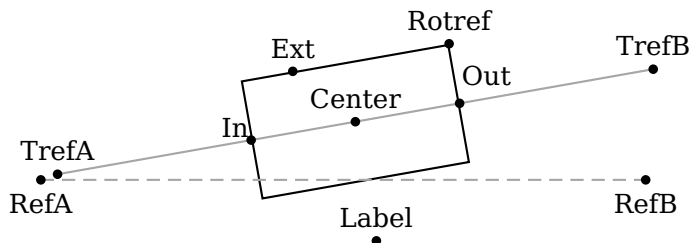
\begin{pspicture}(3.5,2.6)
  \pnodes(0,2){A}(1,2){B}(3,1){C}
  \optprism(A)(B)(C)
  \newsstyle{Beam}{linestyle=none, fillstyle=solid,
    beamwidth=0.2, beamnodealign=vector}
  \drawwidebeam[fillcolor=orange](A){}
  \psset[optexp]{beampathskip=1}
  \drawwidebeam[fillcolor=green, n=1.2, savebeam=2](A){}(C)
  \drawwidebeam[fillcolor=red, n=1.4, savebeam=3](A){}(C)
  \drawwidebeam[fillcolor=red!50!black, n=1.57,
    savebeam=4](A){}(C)
  \multido{\i=2+1}{3}{%
    \backlayer{%
      \pnode(!\oeBeamCenter{\i}){Det\i}
      \nodexn{(Det\i)-(!\oeBeamVecMedian{\i})}{Det\i'}}
    \optdetector[detsize=0.4 0.4](Det\i')(Det\i)}
\end{pspicture}

```

Weitere Anwendungsbeispiele sind Bsp. 11.21 und Bsp. 11.27.

## 8.10. Knotenübersicht

Dies ist eine Übersicht über alle verfügbaren Komponentenknoten.



```

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

```

## 9. Verbinden von Komponenten

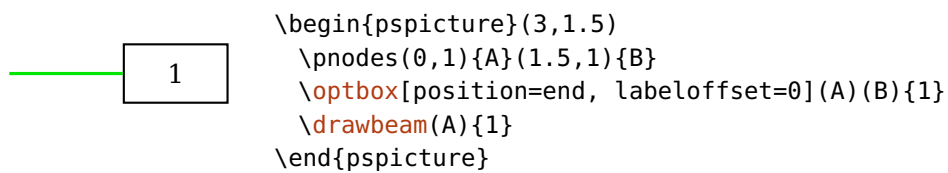
Das `pst-optexp`-Paket stellt unterschiedlichen Methoden bereit, Komponenten vollautomatisch oder manuell, mit Lichtstrahlen, Fasern oder Drähten zu verbinden.

- In Kap. 9.1 wird beschrieben wie Komponenten und Knoten für Verbindungen angesprochen werden.
- Strahlverbindungen im Allgemeinen und das Verhalten von Einzelstrahlen wird in Kap. 9.2 beschrieben.
- Kap. 9.3 erweitert diese Beschreibung von Strahlverbindungen auf ausgehende Strahlen und deren Eigenheiten.
- In Kap. 9.4 wird gezeigt, wie fehlerhafte Strahlverbindungen (verfehlen von Grenzflächen) gehandhabt wird).
- Kap. 9.5 erläutert, wie bestimmte, stückweise definierte Strahlengänge, wie z.B. bei eine Teleskop, einfach realisiert werden können.
- Manuelle Faserverbindungen werden in Kap. 9.6 behandelt, manuelle elektrischen Verbindungen, die sehr ähnlich funktionieren, in Kap. 9.7.
- Automatische Verbindungen werden in Kap. 9.8 erklärt.
- Kap. 9.9 erklärt das Konzept der „front“ und „back“-Ebene.

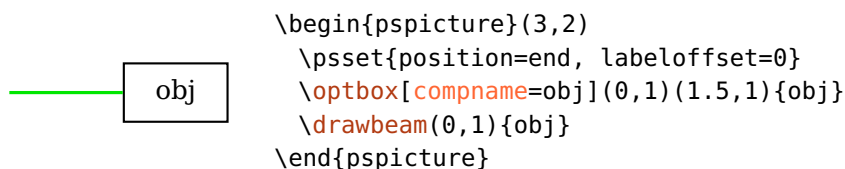
### 9.1. Zugriff auf Komponenten

Alle Makros für Verbindungen akzeptieren entweder eine ID, ein `compname` Bezeichner (siehe Kap. 8.1) 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 entweder in runde Klammern innerhalb der geschweiften Klammern eingeschlossen werden oder dürfen nur runde Klammern haben: `\drawbeam{(node)}{comp}`, oder `\drawbeam(node){comp}`.

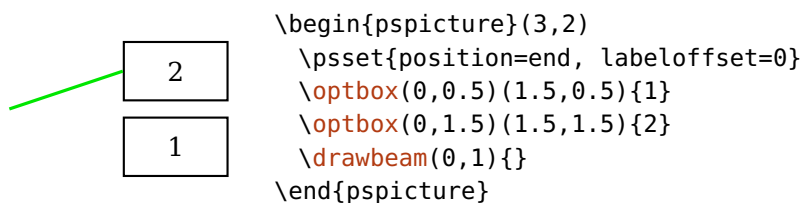
1. Auf alle Komponenten kann immer unter Verwendung der ID zugegriffen werden.



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



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



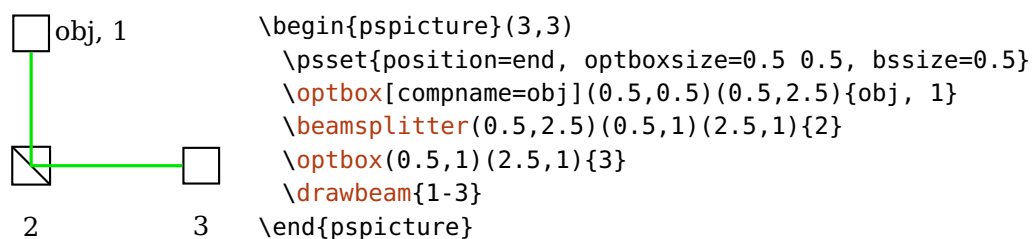
4. Es können auch Zahlintervalle verwendet werden, gültige Intervallangaben sind

**x-y** Von x bis y, ist x größer als y, so wird heruntergezählt. Ist der gesamte Bereich nicht definiert, dann wird ein Fehler ausgegeben.

**x-** Von x bis zur letzten Komponente.

**-y** Von der ersten Komponente bis y. Ist y außerhalb des definierten Bereichs, dann wird eine Warnung ausgegeben.

**-** Verbinde alle Komponenten.



## 9.2. 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. 9.3 beschrieben.

```
\drawbeam[⟨options⟩]{⟨obj1⟩}{⟨obj2⟩}...
```

```
\drawwidebeam[⟨options⟩]{⟨obj1⟩}{⟨obj2⟩}...
```

Diese beiden Makros sind für die Strahlverbindungen zuständig, sie akzeptieren eine variable Anzahl an Argumenten (mindestens zwei), die entweder eine ID, ein ID-Intervall, ein `compname` oder ein PSTricks-Knoten sein können (siehe Kap. 9.1). Um zwischen Knoten und Komponenten unterscheiden zu können, müssen Knoten entweder in runde Klammern innerhalb der geschweiften Klammern eingeschlossen werden oder dürfen nur runde Klammern haben: `\drawbeam{(node)}{comp}`, oder `\drawbeam(node){comp}`.

### 9.2.1. 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. 9.2.2) lediglich als Mittel zur Optimierung des gewünschten Lichtweges sehen und nicht zu sehr an den physikalisch richtigen Werten festhalten.

```
raytrace=true, false
```

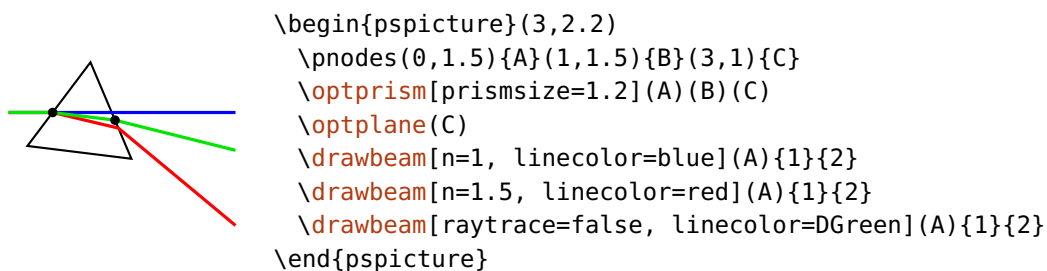
default: true

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

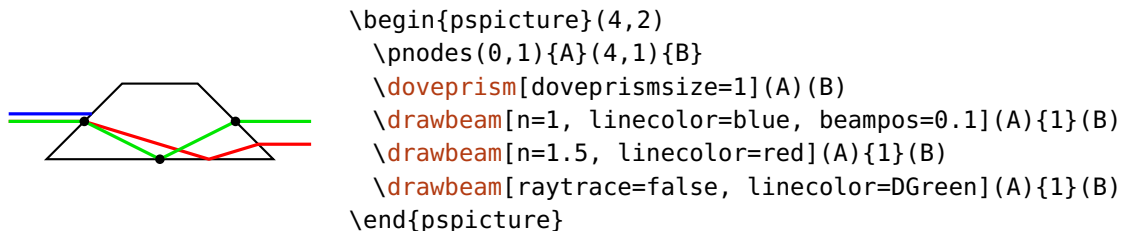


In vielen Fällen ist „connect“ identisch mit „raytracing“ für  $n=1$ , außer für `\optprism` oder `\doveprism` bei denen die Grenzflächenknoten (Kap. 8.6) dann einfach nur verbunden werden. Die folgenden beiden Beispiele zeigen die Unterschiede der beiden Methoden für diese Komponenten. Die Grenzflächenknoten sind zum besseren Verständnis hervorgehoben.

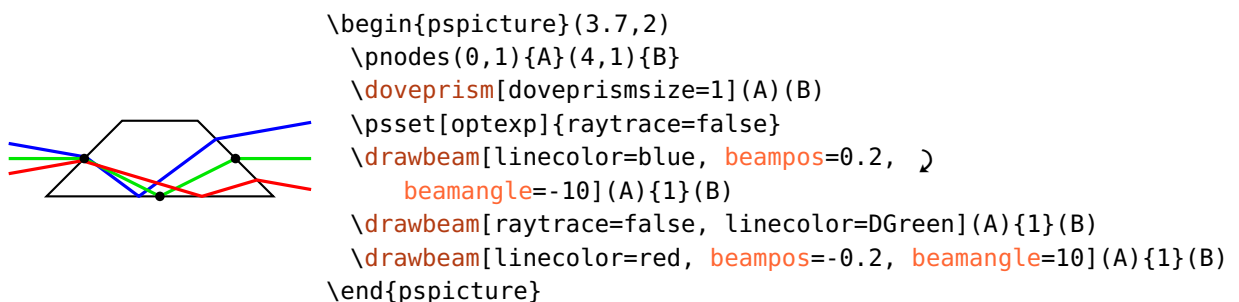
Der „raytracing“ Strahl mit  $n=1$  (blau) wird durch die Komponente transmittiert ohne die Richtung zu verändern, für  $n=1.5$  (rot) wird die Brechung an den Grenzflächen berechnet, und der „connect“ Strahl (grün) durchläuft genau die Grenzflächenknoten.

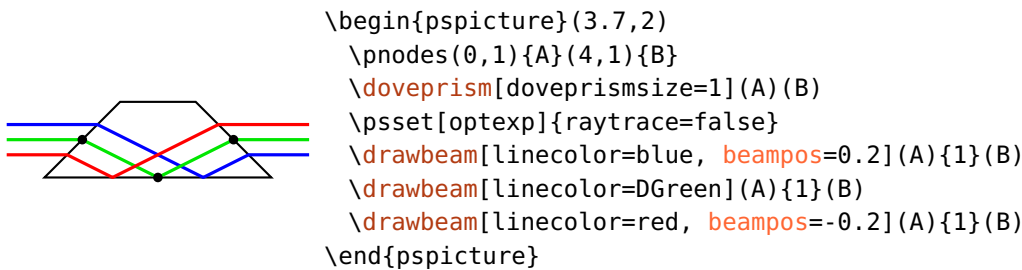


Das `\doveprism` wird im Prinzip genauso behandelt wie das `\optprism`, der „raytracing“ Strahl mit  $n=1$  (blau) trifft wegen fehlender Brechung die zweite Grenzfläche gar nicht, und endet daher schon an der ersten Grenzflächen (siehe Kap. 9.4). Der grüne „connect“ Strahl verläuft wieder durch die Grenzflächenknoten.



Für „connect“ können auch unterschiedliche Anfangsbedingungen (9.2.3) verwendet werden.

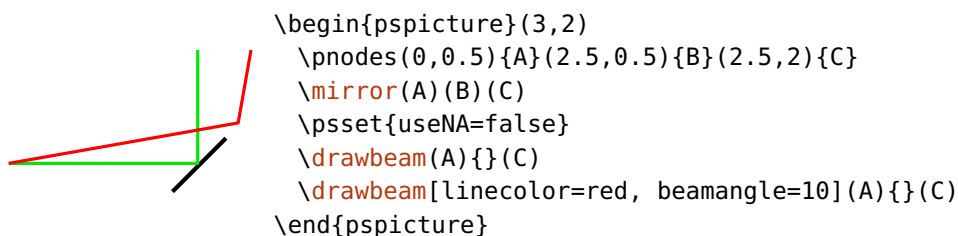




`useNA=true, false`

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.



## 9.2.2. 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:

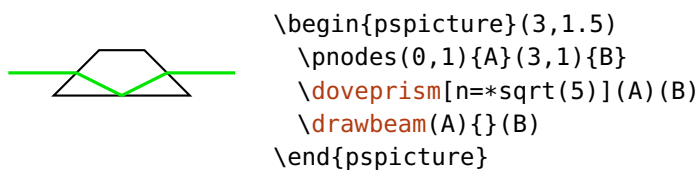
1. Wird `n` als globaler Parameter oder als Komponentenparameter verwendet, so wird der Index für alle betroffenen Komponenten gesetzt. Ohne weitere Änderungen wird jeder Lichtstrahl diese Werte verwenden.
2. Als Parameter zu `\drawbeam` oder `\drawwidebeam` kann der Brechungsindex der Komponenten für 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.

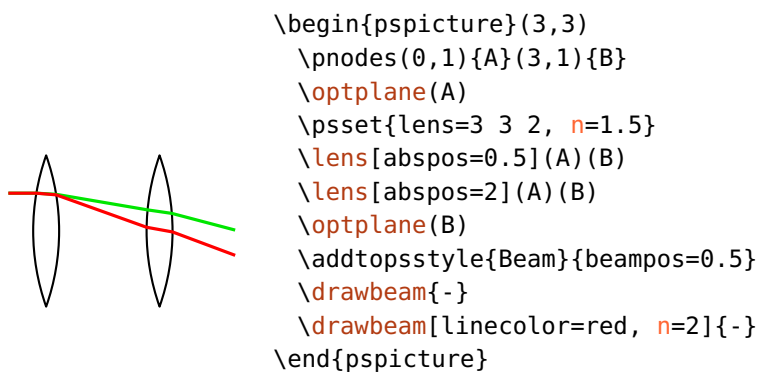
Der `<code>` kann jeglicher Postscript-Kode sein, der zu einer Zahl ausgewertet werden kann, also z.B. `1.5` oder `5 sqrt`. Beginnt `<code>` mit einem `*`, so wird der Code als algebraischer Ausdruck<sup>1</sup> ausgewertet, d.h. `*sqrt(5)` ist äquivalent zu `5 sqrt`. Innerhalb von `<code>` kann mit `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.

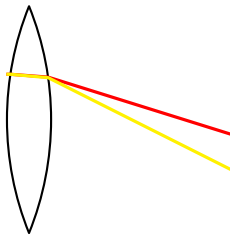


2. Ein Strahl (grün) verwendet die voreingestellten Werte der Komponenten, der rote überschreibt diese mit 2.



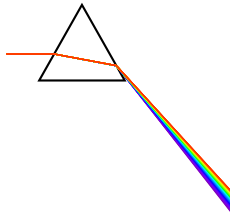
3. Der rote Strahl verwendet die voreingestellten Werte, der grüne ändert diese um 0.5.

<sup>1</sup><http://mirror.ctan.org/graphics/pstricks/base/doc/pst-news08.pdf>



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

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<sup>2</sup> für die unterschiedlichen Wellenlängen berechnet.



```
\begin{pspicture}(3,3)
  \pnodes(-1,0){A}(1,2.2){B}(3,0){C}
  \optplane(0,2.15)
  \optprism[prismalign=center, prismangle=59](A)(B)(C)
  \optplane(C)
  \definecolor[ps]{bl}{rgb}{%
    tx@addDict begin tx@OptexpDict begin
      Wavelength wavelengthToRGB
      Red Green Blue
    end end }%
  \addtopsstyle{Beam}{linecolor=bl, 2
    linewidth=0.4\pslinewidth, beamalign=abs}
  \multido{\iLambda=400+4}{60}{%
    \pstVerb{/Wavelength \iLambda\space def }%
    \drawbeam[n=Wavelength Sellmaier]{-}%
  }%
\end{pspicture}
```

5. Die numerische Apertur einer Lochblende als spektraler Filter.

<sup>2</sup>[http://en.wikipedia.org/wiki/Sellmeier\\_equation](http://en.wikipedia.org/wiki/Sellmeier_equation)



```
\begin{pspicture}(0.3,0)(3.3,5.2)
  \pnodes(0.3,1){A}(2,1){B}(3,2){C}(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}
```

**refractiveindex**= $\langle \text{code} \rangle$

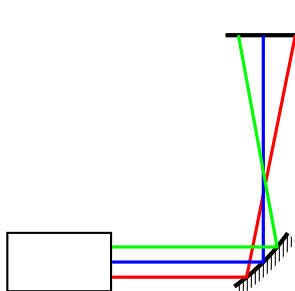
Dieser Parameter ist seit Version 3.0 veraltet, verwenden Sie stattdessen **n**.

### 9.2.3. Anfangsbedingungen

**beampos**=[ $\langle x \rangle$ ] [ $\langle y \rangle$ ]

default: 0

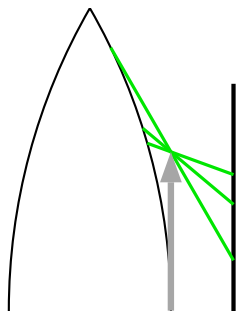
Die Startposition ( $\langle x \rangle$ ,  $\langle y \rangle$ ) des Strahls an der ersten Grenzfläche. Beide Zahlen sind vom Typ  $\langle \text{psnum} \rangle$ . Wird eine einzelne Zahl angegeben, wird die  $x$ -Koordinate auf 0 gesetzt.



```
\begin{pspicture}(4,3.8)
  \pnodes(1.5,0.5){A}(3.5,0.5){B}(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.



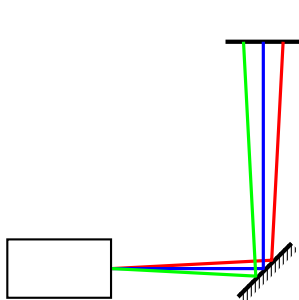
```
\begin{pspicture*}(0,2)(3.05,6)
\lens[lens=8 8 8, abspos=0.6](0.5,2)(3,2)
\optplate[position=1, plateheight=6](0.5,2)(3,2)
\psset{beampos=2.1}
\drawbeam[beamangle=-60]{1}{2}
\drawbeam[beamangle=-40]{1}{2}
\drawbeam[beamangle=-20]{1}{2}
\end{pspicture*}
```

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

`beamangle`= $\langle psnum \rangle$

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.



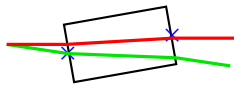
```
\begin{pspicture}(4,4)
\pnodes(1.5,0.5){A}(3.5,0.5){B}(3.5,3.5){C}
\optbox[position=end](B)(A)
\mirror[mirrortype=extended](A)(B)(C)
\optplate[position=end](B)(C)
\drawbeam[linecolor=red, beamangle=3]{-3}
\drawbeam[linecolor=blue, beamangle=0]{1-}
\drawbeam[linecolor=green, beamangle=-3]{-}
\end{pspicture}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

`beamalign`=rel, relative, abs, absolute, firstcomp

default: relative

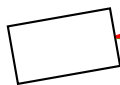
Gibt vor, ob der Winkel `beamangle` relativ zu der Verbindung zwischen den ersten beiden Komponenten ist, ob es sich um einen absoluten Winkel handelt, oder ob der Winkel durch die Ausrichtung der ersten Komponente vorgegeben wird.

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

Im folgenden Beispiel liegen beide Komponenten auf der horizontalen Linie  $\overline{AB}$ . Der grüne Strahl, mit `beamalign=rel` verläuft daher auch horizontal. Der rote Strahl ist hingegen mit `beamalign=firstcomp` entsprechend der Drehung der ersten Komponente ausgerichtet.

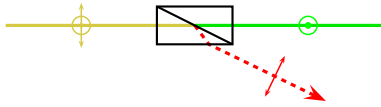


```
\begin{pspicture}(4,3)
  \pnodes(1.5,1.5){A}(4,1.5){B}
  \optbox[position=start, rotateref=r, angle=10](A)(B)
  \optplate[position=end, plateheight=3](A)(B)
  \addtopsstyle{Beam}{beamdiv=20}
  \drawwidebeam{-}
  \drawwidebeam[beamalign=firstcomp, linecolor=red]{-}
\end{pspicture}
```

`beampathskip=<num>`

default: 0

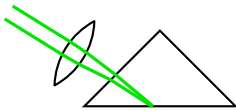
Überspringt  $\langle num \rangle$  Segmente am Anfang des Strahlengangs. Diese Option beeinflusst lediglich ab welchem Segment der Strahlgang gezeichnet wird, das Raytracing beginnt immer noch an der ersten Grenzfläche. Um das zu ändern können Sie `startinsidecount` verwenden.



```
\begin{pspicture}(5,2)
\nodes(0,1){A}(5,1){B}(2,0){C}
\begin{optexp}
\glanthompson(A)(B)
\optplane(B)\optplane[angle=90](C)
\polarization[abspos=1, poltype=misc, linecolor=yellow!80!black](A)(B)
\drawbeam[beaminsidelast, linecolor=yellow!80!black](A){1}
\drawbeam[beampathskip=2](A){1-2}
\drawbeam[beampathskip=2, savebeam, linestyle=dashed, dash=2pt 2pt, 
linecolor=red,arrowscale=1.5, arrows=->](A){1}{3}
\end{optexp}
\nodexn{(\oenameBeam{})-1.5(!\oeBeamVec{})}{D}
\polarization[poltype=parallel, linecolor=red](D)(\oenameBeam{})
\polarization[abspos=4, poltype=perp, linecolor=green, dotscale=1.5](A)(B)
\end{pspicture}
```

`beampathcount=<num>` default: -1

Anzahl der Strahlsegmente, die gezeichnet werden sollen. Werden am Anfang Segmente mit `beampathskip` übersprungen, so werden diese nicht mitgezählt.



```
\begin{pspicture}(3,2)
\nodes(0,2){A}(2,0.8){B}(4,2){C}
\optprism[comname=P, prismtype=reflective, 
prismangle=90](A)(B)(C)
\lens[abspos=1.33](\oenameIfc{2}{P})(A)%
\drawwidebeam[beamwidth=0.2, beaminsidelast, 
beampathcount=4](A){2}{1}
\end{pspicture}
```

## 9.2.4. Interner Strahlengang

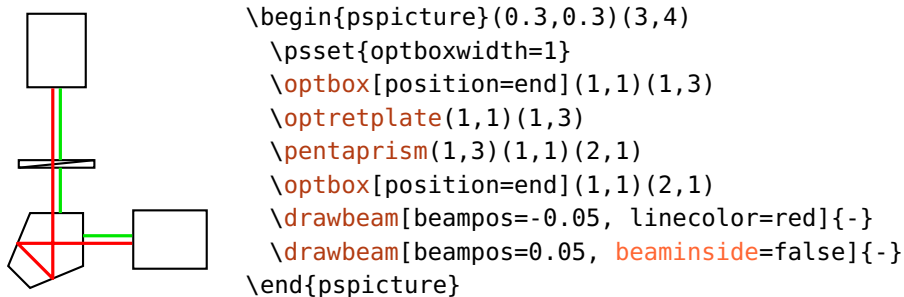
Ein Strahlengang setzt sich aus Strahlen zwischen den Komponenten und innerhalb der Komponenten zusammen. Ob diese internen Strahlen gezeichnet werden, kann mit mehreren Parametern festgelegt werden. Die erste, letzte und die übrigen Komponenten werden separat gehandhabt.



**beaminside**=true, false

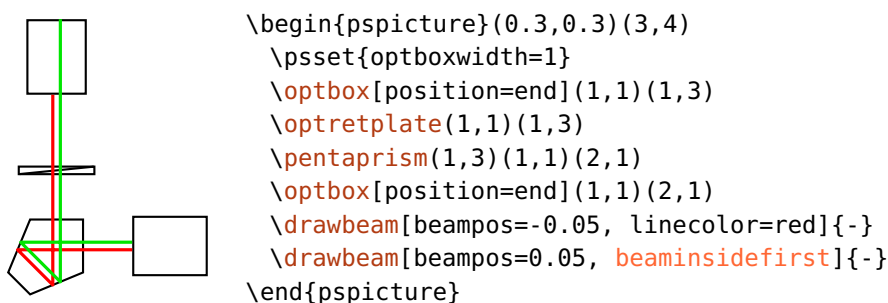
default: true

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

**beaminsidefirst**=true, false

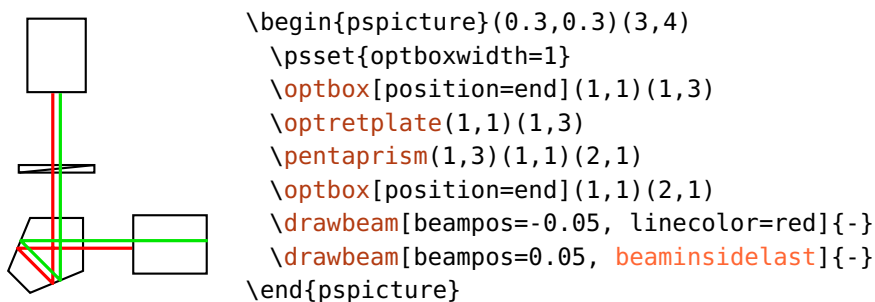
default: false

Zeichne den Strahlengang innerhalb der ersten Komponenten.

**beaminsidelast**=true, false

default: false

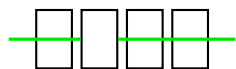
Zeichne den Strahlengang innerhalb der letzten Komponenten.

**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**,

`beaminsidelfirst` 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.



```

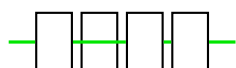
\begin{pspicture}(3,2)
  \psset{optboxwidth=0.5}
  \pnodes(0,1){A}(3,1){B}
  \optbox[position=0.2](A)(B)
  \optbox[position=0.4, allowbeaminside=false](A)(B)
  \optbox[position=0.6](A)(B)
  \optbox[position=0.8](A)(B)
  \drawbeam(A){-}(B)
\end{pspicture}

```

`forcebeaminside=true, false`

default: false

Wenn auf true gesetzt, wird der Strahlengang innerhalb der betroffenen Komponenten immer gezeichnet, unabhängig von allen anderen Strahleinstellungen mit `allowbeaminside`, `beaminside`, `beaminsidelfirst` und `beaminsidelast`. Damit kann der Strahl innerhalb einzelner Komponenten gezeichnet werden, auch wenn alle anderen Komponenten in dem Strahlengang keinen internen Strahl haben sollen.



```

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

```

`startinsidecount=<num>`

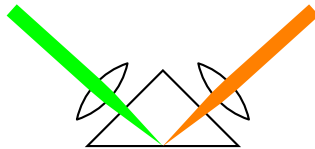
default: -1

Anzahl der Segmente, die innerhalb der ersten Komponente verwendet werden. Diese Option hat nur eine Wirkung, wenn `beaminsidelfirst` oder `startinside` auf true gesetzt sind.

`stopinsidecount=<num>`

default: -1

Anzahl der Segmente, die innerhalb der letzten Komponente verwendet werden. Diese Option hat nur eine Wirkung, wenn `beaminsidelast` oder `stopinside` auf true gesetzt sind.



```
\begin{pspicture}(4,2)
  \pnodes(0,2){A}(2,0){B}(4,2){C}
  \optprism[comname=P, n=1, prismtype=reflective,
    prismangle=90](A)(B)(C)
  \lens[abspos=1.08](\oenableIfc{2}{P})(A)
  \lens[abspos=1.08](\oenableIfc{2}{P})(C)
  \newsstyle{Beam}{linestyle=none, fillstyle=solid}
  \drawwidebeam[fillcolor=green, beamwidth=0.2,
    beaminsidelast, stopinsidecount=1](A){2}{1}
  \drawwidebeam[fillcolor=orange, beaminsidefirst,
    startinsidecount=1, loadbeam]{1}{3}(C)
\end{pspicture}
```

**beammode**=refl, trans, reflective, transmittive, auto

default: auto

Mit dieser Option kann das Strahlverhalten bei Komponenten wie z. B. dem `\beamsplitter` auf reflektierend oder transmittierend festgelegt werden, unabhängig von den verwendeten Knoten und deren relative Position bezüglich der Komponente.

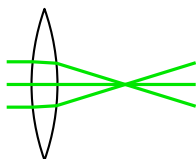
### 9.2.5. Verbinden mit Knoten

Wie vorher schon einmal erwähnt können Komponenten auch mit Knoten verbunden werden. Für Raytracing wird in der Regel eine Ebene benötigt, die auf unterschiedliche Weise automatisch oder manuell erstellt werden kann.

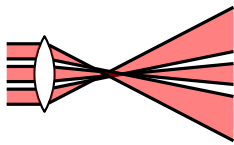
**beamnodealign**=vec, conn, vector, connection

default: connection

Dieser Parameter bestimmt die Ausrichtung der automatisch erstellen Grenzflächen. In der Voreinstellung (connection) wird die Ebene vertikal zur direkten Verbindung zwischen dem Knoten und dem vorangegangenen Grenzflächenknoten ausgerichtet.

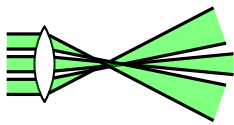


```
\begin{pspicture}(3,2)
  \pnodes(0,1){A}(2.5,1){B}
  \lens[lens=3 3 2, n=2.5, abspos=0.5](A)(B)
  \multido{\r=-0.3+0.3}{3}{%
    \drawbeam[beampos=\r](A){}(B)}
\end{pspicture}
```



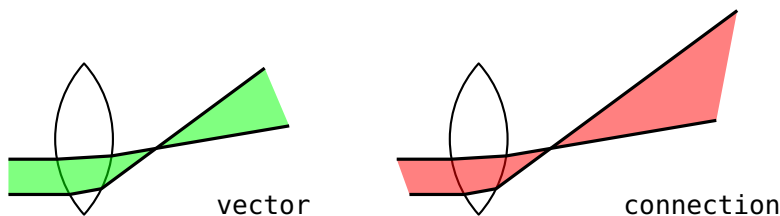
```
\begin{pspicture}(3,2)
  \pnodes(0,1){A}(3,1){B}
  \lens[abspos=0.5](A)(B)
  \addtopsstyle{Beam}{linecolor=black, fillstyle=solid,
    beamwidth=0.2, fillcolor=red, opacity=0.5, beaminside=false}
  \multido{\r=-0.3+0.3}{3}{\drawwidebeam[beampos=\r](A){}(B)}
\end{pspicture}
```

Ist der Wert auf `vector` gesetzt, dann wird die Ebene vertikal zum einfallenden Strahlvektor ausgerichtet. Bei weiten Strahlen wird der gemittelte Strahlenvektor (siehe `\oeBeamVecMedian`) als Referenz verwendet. Beispiele in dem das sehr praktisch ist, sind Bsp. 8.1 und Bsp. 11.27.



```
\begin{pspicture}(3,2)
  \pnodes(0,1){A}(3,1){B}
  \lens[abspos=0.5](A)(B)
  \addtopsstyle{Beam}{linecolor=black, fillstyle=solid,
    beamwidth=0.2, beamnodealign=vector, fillcolor=green,
    opacity=0.5, beaminside=false}
  \multido{\r=-0.3+0.3}{3}{\drawwidebeam[beampos=\r](A){}(B)}
\end{pspicture}
```

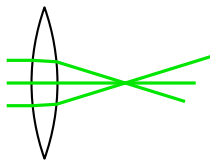
Die automatische Ausrichtung ist in vielen Fällen sehr praktisch, kann aber wiederum unerwünschte Ergebnisse bei gedrehten und verschobenen Komponenten ergeben und die Ebenenausrichtung kann nicht beliebig gesetzt werden. Im folgenden Beispiel ist beim grünen Strahl `beamnodealign` auf `vector` gesetzt, bei dem roten Strahl hingegen auf `connection`:



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)
  \pnodes(0,1){A}(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}
```

### 9.2.6. 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](\oencodeRefA){}{(\oencodeRefB{})}
```

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

### 9.2.7. Strahlaussehen

`Beam`  $\langle psstyle \rangle$

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`= $\langle list \rangle$

Der vorhandene `Beam`-Stil wird lokal um die Parameter in  $\langle list \rangle$  erweitert,  $\langle list \rangle$  muss mit geschweiften Klammern gekapselt werden.

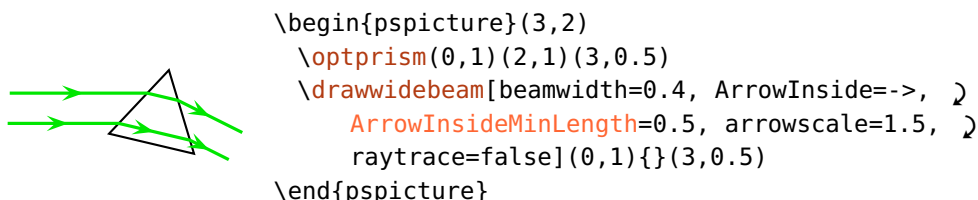
`newBeam`= $\langle list \rangle$

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

**ArrowInsideMinLength**= $\langle psnum \rangle$

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 Pfeile gezeichnet werden. In dem folgenden Beispiel ist der Wert so gewählt, dass nur der untere interne Strahl einen Pfeil hat (siehe auch Bsp. 11.16).

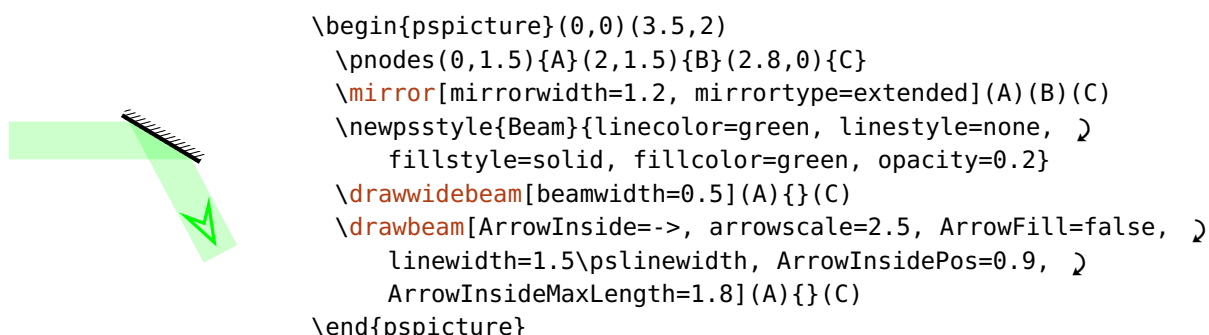


**ArrowInsideMaxLength**= $\langle psnum \rangle$

default: -1

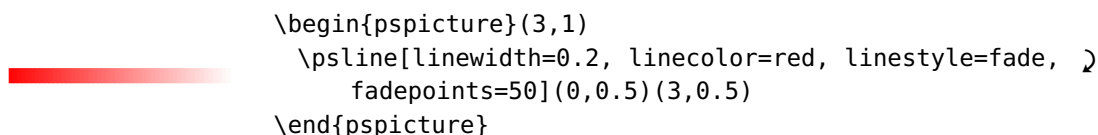
Gibt analog zu **ArrowInsideMinLength** die maximale Länge eines Segments an, für das Pfeile gezeichnet werden. Ist der Wert negativ, dann gibt es keine obere Grenze.

`ArrowInside` kann auch verwendet werden ohne dass ein Linientyp definiert ist.




**fade**  $\langle linestyle \rangle$

Erzeugt einen Farbverlauf von der aktuellen `linecolor` zu einer anderen Farbe, definiert durch **fadeto**. Der Linienstil funktioniert momentan nur vernünftig mit Linien, nicht jedoch mit Kurven. Der gesamte Pfad wird in **fadepoints**-1 Segmente unterteilt.



**fadeto**=white, black, transparency default: white

Die Endfarbe des Farbverlaufs, hängt auch von **fadefuncname** und **fadefunc** ab.



```

\begin{pspicture}(3,1)
\psset{linewidth=0.2, linecolor=red, linestyle=fade, }
fadepoints=50}
\psline[fadeto=white](0,0.9)(3,0.9)
\psline[fadeto=black](0,0.5)(3,0.5)
\psline[fadeto=transparency](0,0.1)(3,0.1)
\end{pspicture}

```

**fadepoints**=*<num>* default: 200


Anzahl der Punkte, die für den Linienstil **fade** verwendet werden.

**fadefuncname**=gauss, linear, squared, exp, custom default: linear

Ein paar vordefinierte Funktionen, die den Farbverlauf bestimmen. Wird **custom** angegeben, dann wird der Postscript-Code aus **fadefunc** verwendet.

**fadefunc**=*<PS code>*

Eine benutzerdefinierte Funktion für den Farbverlauf des Linienstils **fade**. Der *<PS code>* muss eine Zahl auf dem Stack im Intervall  $[0, 1]$  (0: Pfadanzfang, 1: Pfadende) auf das Intervall  $[0, 1]$  abbilden (0: **fadeto**-Farbe, 1: aktuelle Linienfarbe). Die Funktionswerte werden auf das Intervall  $[0, 1]$  abgeschnitten.



```

\begin{pspicture}(3,1)
\psset{linewidth=0.2, linecolor=red, linestyle=fade, }
fadepoints=50}
\psline[fadefuncname=gauss](0,1)(3,1)
\psline[fadefuncname=exp](0,0.7)(3,0.7)
\psline[fadefuncname=custom, fadefunc={360 mul cos 1 add 0.5 }
mul}}(0,0.4)(3,0.4)
\psline[fadefuncname=custom, fadefunc={neg 1 add 2 }
mul}}(0,0.1)(3,0.1)
\end{pspicture}

```

Ein schönes Beispiel ist Bsp. 11.30.

## 9.3. Aufgeweitete Strahlen

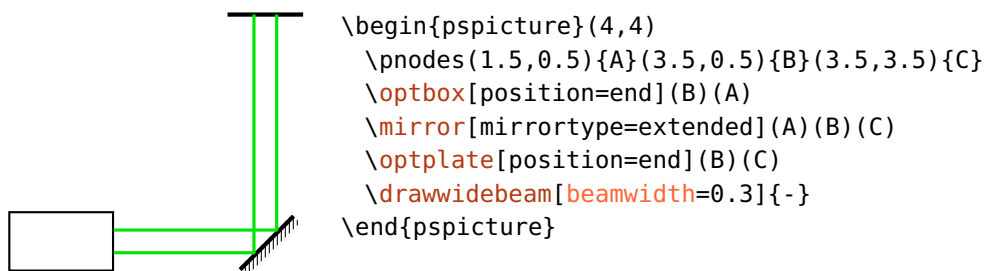
Alle Informationen über einfache Strahlen (siehe Kap. 9.2) betreffen auch aufgeweitete Strahlen. Hier werden nur zusätzliche Einstellmöglichkeiten aufgeweiteter Strahlen behandelt.

`\drawwidebeam[⟨options⟩]{⟨obj1⟩}{⟨obj2⟩}...`

`beamwidth=⟨psnum⟩`

default: 0

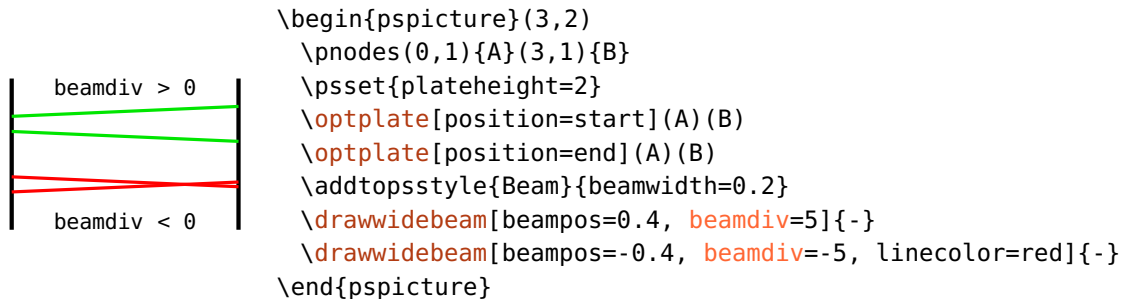
Die Anfangsstrahlbreite an der mit `beampos` angegebenen Position, gemessen vertikal zur Verbindungslinie zur nächsten Komponente.



`beamdiv=⟨psnum⟩`

default: 0

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



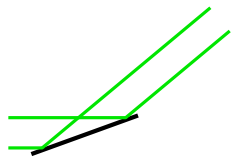
### 9.3.1. Strahlaussehen

Das Aussehen aufgeweiteter Strahlen kann genauso wie bei einfachen Strahlen über `Beam` oder das optionale Argument vorgegeben werden (siehe Kap. 9.2.7).

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



1. Nur die Randstrahlen.



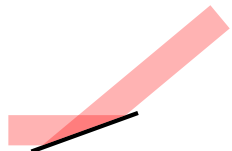
```
\begin{pspicture}(3,2)
  \pnodes(0,0.5){A}(1,0.5){B}(2.8,2){C}
  \mirror[mirrorwidth=1.5](A)(B)(C)
  \drawwidebeam[beamwidth=0.4](A){}(C)
\end{pspicture}
```

2. Randstrahlen, und Füllstil.



```
\begin{pspicture}(3,2)
  \pnodes(0,0.5){A}(1,0.5){B}(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.

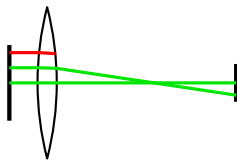


```
\begin{pspicture}(3,2)
  \pnodes(0,0.5){A}(1,0.5){B}(2.8,2){C}
  \mirror[mirrorwidth=1.5](A)(B)(C)
  \addtopsstyle{Beam}{fillstyle=solid, fillcolor=red,
    opacity=0.3, linestyle=none}
  \drawwidebeam[beamwidth=0.4](A){}(C)
\end{pspicture}
```

## 9.4. Fehlerbehandlung

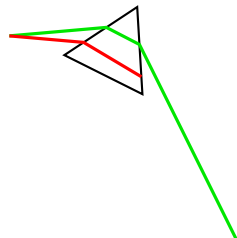
Alle Strahlen werden direkt mit Postscript berechnet und gezeichnet, so dass eine umfassende Fehlerbehandlung zur Zeit der Kompilation nicht möglich ist. Um jedoch trotzdem fehlerfreie Bilder zu bekommen, wird der Strahlengang abgebrochen, sobald eine Situation eintritt, die nicht unterstützt wird. Das kann eintreten, falls der Strahl eine Grenzfläche nicht trifft, Totalreflektion auftritt oder wenn die folgende Komponente keine Freistrahilverbindungen unterstützt. Die folgenden Beispiele zeigen Situationen, in denen diese Fehler auftreten:

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



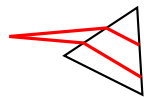
```
\begin{pspicture}(3,2.7)
  \pnodes(0,1){A}(3,1){B}
  \optplate[position=start](A)(B)
  \lens[lens=4 4 2, abspos=0.5, n=2.5](A)(B)
  \optplate[position=end, plateheight=0.5](A)(B)
  \drawbeam{-}
  \drawbeam[beampos=0.2]{-}
  \drawbeam[beampos=0.4, linecolor=red]{-}
\end{pspicture}
```

2. Totale interne Reflexion 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.



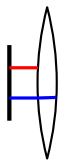
```
\begin{pspicture}(3,2.5)
  \pnodes(0,2){A}(3,0){B}
  \optplane(A)
  \optprism[n=1.8](A)([Xnodesep=1.5]A)(B)
  \optplane(B)
  \drawbeam[beamangle=5]{-}
  \drawbeam[beamangle=-5, linecolor=red]{-}
\end{pspicture}
```

Dasselbe gilt ebenfalls für aufgeweitete Strahlen (siehe Kap. 9.3). Bei diesen wird der Strahlengang abgebrochen, falls einer der beiden Randstrahlen einen Fehler aufweist.



```
\begin{pspicture}(3,2.5)
  \pnodes(0,2){A}(3,0){B}
  \optplane(A)
  \optprism[n=1.8](A)([Xnodesep=1.5]A)(B)
  \optplane(B)
  \drawwidebeam[beamdiv=10, linecolor=red]{-}
\end{pspicture}
```

3. Eine Komponente soll verbunden werden, die jedoch keine Freistrahverbindungen unterstützt. In diesem Fall wird die falsche Komponente und alle Folgenden ignoriert.



```
\begin{pspicture}(3,2.7)
  \pnodes(0,1){A}(2,1){B}
  \optplate[position=start](A)(B)
  \lens[lens=4 4 2, abspos=0.5, n=2.5](A)(B)
  \fiberdelayline[fdlsize=1 0.7, position=end, 2
    fiber=none](A)(B)
  \drawbeam[beampos=0.2, linecolor=red]{-}
  \drawbeam[beampos=-0.2, linecolor=blue, beaminsidelast]{-}
\end{pspicture}
```

`pswarning=true, false`

default: false

Um eine Fehlersuche zu vereinfachen kann ein Hinweis ausgegeben werden sobald ein Strahlengang vorzeitig beendet. Die Fehlermeldung wird auf die Standardausgabe geschrieben, wenn die Datei mit einem Postscript-Interpreter gelesen wird, also mit einem Betrachter geöffnet oder z.B. mit ps2pdf konvertiert wird.

## 9.5. 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,  $\langle int \rangle$`

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, siehe z.B. Bsp. 11.32. In der Regel reicht an- und abschalten vollkommen aus, was den Zahlen 1 und 0 entspricht.

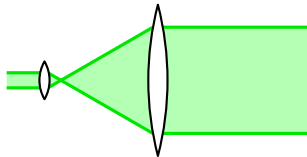
`loadbeampoints=true, false,  $\langle int \rangle$`

default: false

Verwende die Endpunkte eines vorangegangenen Strahls als Startpunkte. Wird keine Zahl  $\langle int \rangle$  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

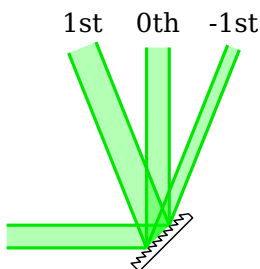
drei Strahlenteile mit den gewünschten Divergenzen, und verwenden die Endpunkte des jeweils vorangegangenen Strahls als neue Ausgangspunkte.



```
\begin{pspicture}(4,2)
  \pnodes(0,1){A}(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. 9.1



```
\begin{pspicture}(3.5,3.5)
  \pnodes(0,0.5){A}(2,0.5){B}(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,  $\langle int \rangle$`

default: true

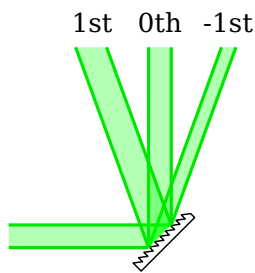
Speichert die Endzustände, d.h. Endpunkte und Ausgangsvektoren, eines `\draw*beam` Makros, vorangegangene Werte werden überschrieben. `\drawbeam` und `\drawwidebeam` werden getrennt behandelt. Wird hier eine Zahl  $\langle int \rangle > 0$  angegeben, so können beliebig viele Strahlen separat gespeichert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

`loadbeam=true, false,  $\langle int \rangle$`

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).

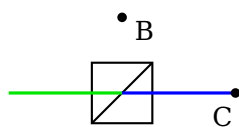
Bsp. 9.2



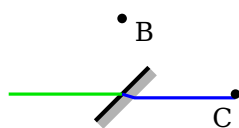
```
\begin{pspicture}(3.5,3.5)
\pnodes(0,0.5){A}(2,0.5){B}(2,3){C}
\optgrating(A)(B)(C)
\addtopsstyle{Beam}{%
  fillstyle=solid, fillcolor=green, opacity=0.3}
\drawwidebeam[savebeam, beamwidth=0.3](A){1}
\psset{savebeam=false, loadbeam}
\drawwidebeam[beamangle=-20]{1}(C)
\drawwidebeam{1}(C)
\drawwidebeam[beamangle=20]{1}(C)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

Beachten Sie, dass `loadbeam` und `savebeam` nicht immer funktionieren, wenn der Strahlengang an einem halbdurchlässigen Spiegel oder einem Strahlteiler endet. In diesen Fällen wird nur der transmittierte Strahlvektor gespeichert, soll der Strahlengang über den reflektierenden Weg fortgesetzt werden, so ist das nicht möglich.

In den beiden folgenden Beispielen wird demnach der rote Strahl, der von der Komponente zum Knoten  $\langle B \rangle$  fortgesetzt werden soll, nicht gezeichnet.



```
\begin{pspicture}(3,1.5)
\pnodes(0,0.5){A}(1.5,1.5){B}(3,0.5){C}
\beamsplitter(A)(1.5,0.5)(B)
\drawbeam[beaminsidelast, savebeam](A){}
\drawbeam[linecolor=red, beaminsidefirst, loadbeam]{}(B)
\drawbeam[linecolor=blue, beaminsidefirst, loadbeam]{}(C)
\end{pspicture}
```



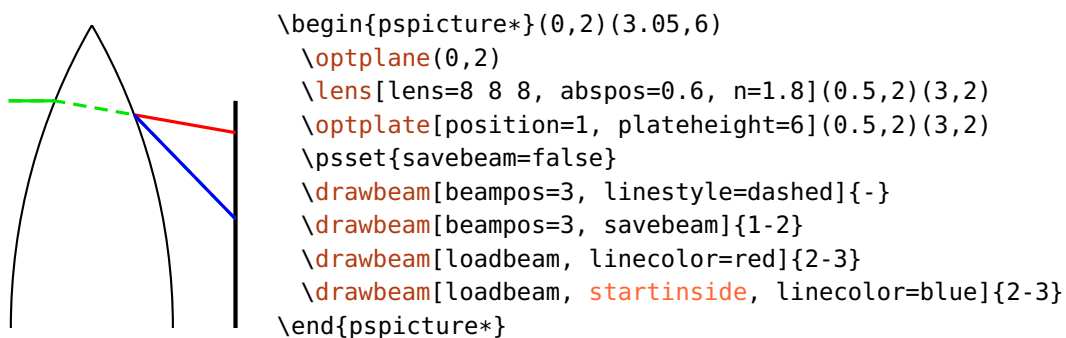
```
\begin{pspicture}(3,1.5)
\pnodes(0,0.5){A}(1.5,1.5){B}(3,0.5){C}
\mirror[mirrortype=semitrans](A)(1.5,0.5)(B)
\drawbeam[beaminsidelast, savebeam](A){}
\drawbeam[linecolor=red, beaminsidefirst, loadbeam]{}(B)
\drawbeam[linecolor=blue, beaminsidefirst, loadbeam]{}(C)
\end{pspicture}
```

`startinside=true, false`

default: false

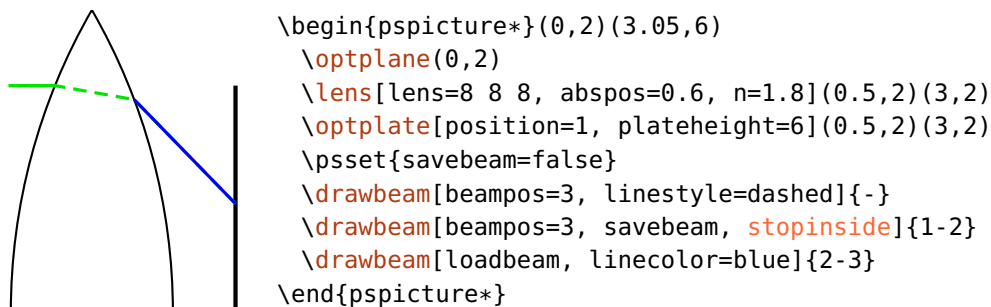
Mit dem Parameter `beaminsidefirst` kann eingestellt werden, ob der interne Strahlengang in der erste Komponente gezeichnet wird. Für angepasste Strahlengänge kann es aber notwendig sein dem internen Strahlengang in der ersten Komponente zu folgen ohne diesen zu zeichnen, z.B. wenn der vorangegangene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

`stopinside=true, false`

default: false

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



## 9.6. Faserverbindungen

Alle Komponenten können mit Fasern verbunden werden, die Faserkomponenten in Kap. 5 werden automatisch verbunden.

`\drawfiber[⟨options⟩]{⟨obj1⟩}{⟨obj2⟩}...`

Dieses Makro zeichnet Faserverbindungen und akzeptiert eine variable Anzahl an Argumenten (mindestens zwei), die entweder eine ID, ein ID-Intervall, ein `comptime` oder ein PSTricks-Knoten sein können (siehe Kap. 9.1). Um zwischen Knoten und Komponenten unterscheiden zu können, müssen Knoten entweder in runde Klammern innerhalb der geschweiften Klammern eingeschlossen werden oder dürfen nur runde Klammern haben: `\drawbeam{(node)}{comp}`, oder `\drawbeam(node){comp}`. Werden mehr als zwei Argumente mitgegeben, werden die Verbindungen  $1 \rightarrow 2$ ,  $2 \rightarrow 3$ , usw. gezeichnet.

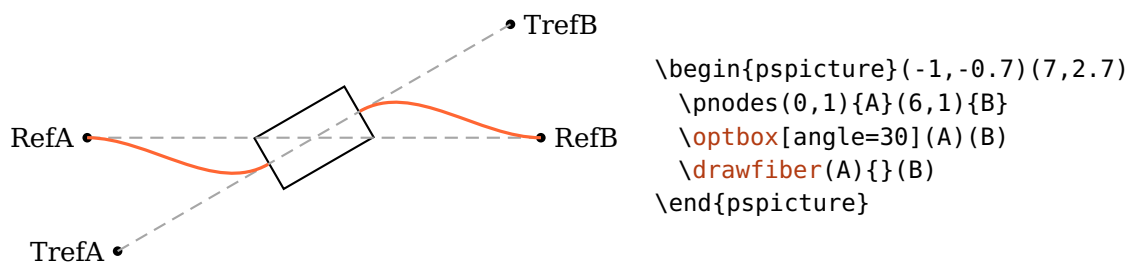
Die automatischen Faserverbindungen werden intern als `\drawfiber` von den Referenzknoten zur Komponente gezeichnet.

### 9.6.1. Faserwinkel

Üblicherweise werden Fasern als `\ncurve`-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.

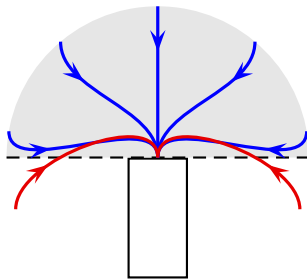
#### Komponente mit Knoten

Eine Verbindung zwischen einer Komponente und einem Knoten ist das, was bei automatischen Verbindungen verwendet wird (siehe Kap. 9.8). Folgendes Beispiel erläutert die Ausrichtung der Kurve:



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  $180^\circ$  gegenüber den blauen Fasern gedreht sind, die Winkel an der Komponente bleiben unverändert.

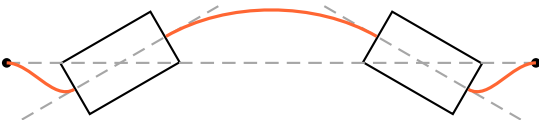


```
\begin{pspicture}(-2,-1.6)(2,2)
\optbox[position=end](0,1)(0,0)
\addtopsstyle{Fiber}{linecolor=blue}
\multido{\i=10+40}{5}{%
\drawfiber(2;\i){}
}
\addtopsstyle{Fiber}{linecolor=red!90!black}
\drawfiber(2;-20){}
\drawfiber(2;200){}
\end{pspicture}
```

### 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. 9.3 und Bsp. 9.4.

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)
\pnodes(0,0){A}(3,0){B}(4,0){C}(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.

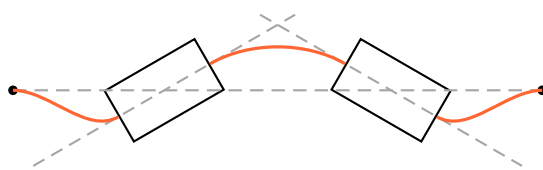
**fiberalign**=rel, relative, center, abs, absolute

default: relative

Wählt die Referenz für die Faserwinkel.

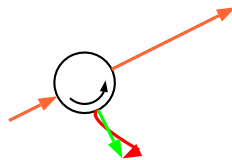
**relative** Die Faserwinkel sind relativ zur Komponente. Wird ein Knoten mit der Komponente verbunden, so entspricht der Winkel am Knoten dem der Referenzlinie der Komponente. Der Winkel an der Komponente entspricht dem der transformierten Referenzlinie.





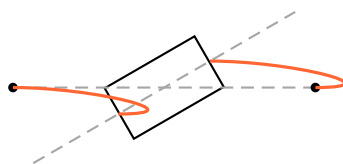
```
\begin{pspicture}(1,-1.7)(7,1.7)
  \pnodes(0,0){A}(4,0){B}(3,0){C}(7,0){D}
  \optbox[angle=30](A)(B)
  \optbox[angle=-30](C)(D)
  \drawfiber(A){1}{2}(D)
\end{pspicture}
```

**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** und **\elecmixer** 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.



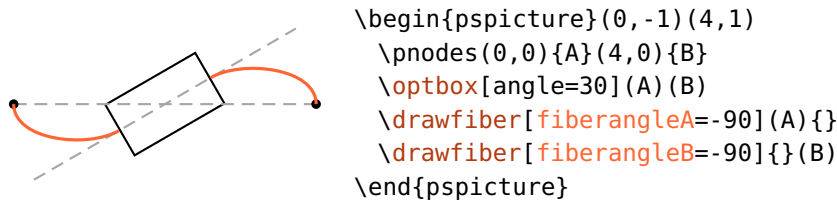
```
\begin{pspicture}(1,-1)(4.5,1)
  \pnodes(0,0){A}(4,0){B}
  \optbox[angle=30](A)(B)
  \drawfiber[fiberalign=abs, ncurv=1.5](A){}(B)
\end{pspicture}
```

**fiberangleA**= $\langle num \rangle$  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**= $\langle num \rangle$  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.

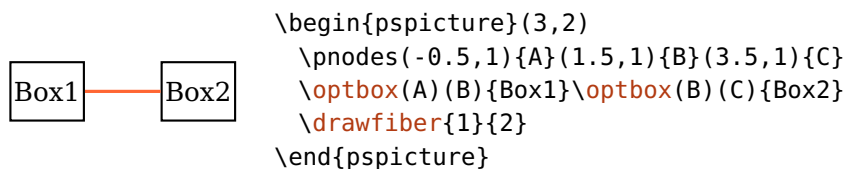
**stopnode**=auto, N, 1, 2, ...

default: auto

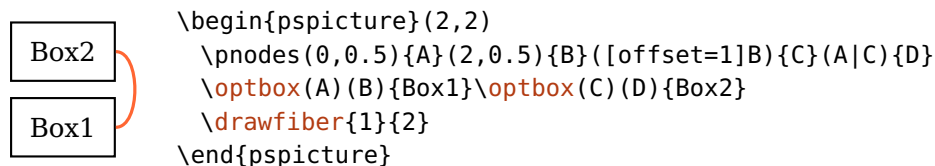
Genau wie **startnode**, aber für den Stopknoten.

Die folgenden Beispiele zeigen Anwendungsmöglichkeiten für **startnode** und **stopnode**.

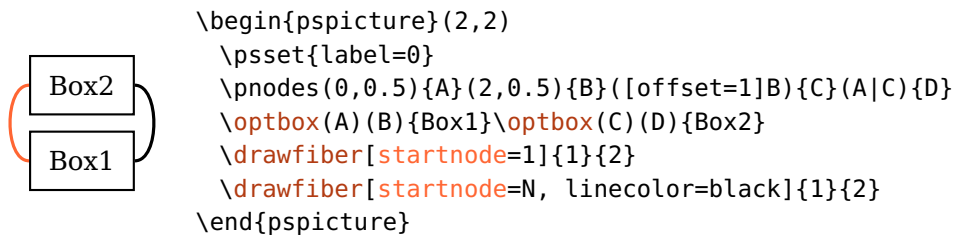
1. In diesem Fall ist der nächstgelegene Knoten auch der richtige.



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

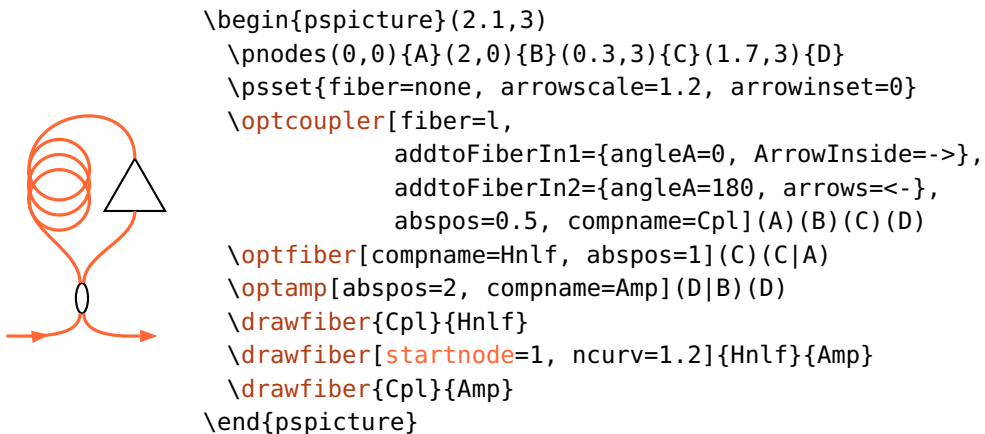


3. Um die gewünschte Verbindung zu wählen ist es meistens ausreichend entweder **startnode** oder **stopnode** zu setzen. Ist ein Knoten fest vorgegeben, wird der anderen wieder als der mit dem kürzesten Abstand gewählt. Nur in Ausnahmefällen sollte es notwendig sein beide Knoten explizit anzugeben.



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

Bsp. 9.3



## 9.6.2. Faseraussehen

**Fiber**  $\langle psstyle \rangle$

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**= $\langle list \rangle$

Der vorhandene **Fiber**-Stil wird lokal um die Parameter in  $\langle list \rangle$  erweitert,  $\langle list \rangle$  muss mit geschweiften Klammern gekapselt werden. Siehe z.B. Bsp. 9.4.

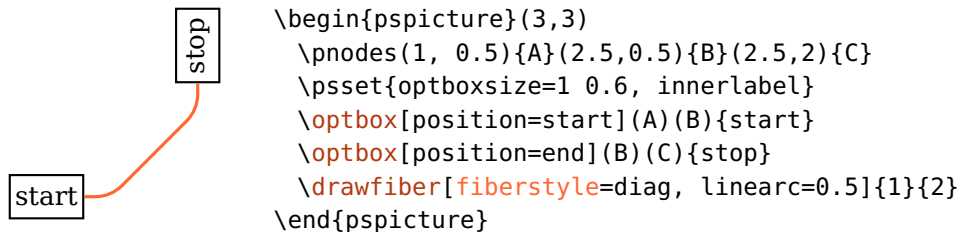
**newFiber**= $\langle list \rangle$

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

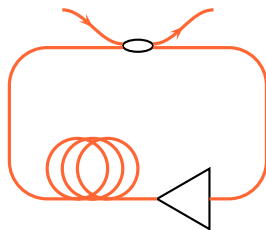
`fiberstyle=<string>`

default: curve

Die Fasern werden in der Regel als `\ncurve` gezeichnet, aber eine beliebige andere Knotenverbindung `\nc<string>` des `pst-node` Paketes kann ausgewählt werden, lesen Sie die entsprechende Dokumentation<sup>3</sup> für mögliche Werte. In den folgenden Beispielen werden Anwendungsmöglichkeiten gezeigt, eine weitere ist in Bsp. 11.13 zu sehen.



Bsp. 9.4



```

\begin{pspicture}(4,2.5)
\pnodes(1,2.5){In}(3,2.5){Out}(0,2){A}(4,2){B}(4,0){C}(0,0){D}
\optcoupler[fiber=t, addtoFiber={ArrowInside=->}, 2
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}

```

## 9.7. Elektrische Verbindungen

Alle Komponenten können mit Drähten verbunden werden, die elektrischen Komponenten in Kap. 6 werden automatisch verbunden. Die `wire*`-Parameter sind äquivalent zu den entsprechenden `fiber*`-Parametern, die detailliertere Informationen enthalten.

`\drawwire[<options>]{<obj1>}{<obj2>}...`

Dieses Makro akzeptiert eine variable Anzahl an Argumenten (mindestens zwei), die entweder eine ID, ein ID-Intervall, ein `compname` oder ein `PSTricks`-Knoten sein können (siehe Kap. 9.1. Um zwischen Knoten und Komponenten

<sup>3</sup><http://mirror.ctan.org/help/Catalogue/entries/pst-node.html>

unterscheiden zu können, müssen Knoten entweder in runde Klammern innerhalb der geschweiften Klammern eingeschlossen werden oder dürfen nur runde Klammern haben: `\drawbeam{(node)}{comp}`, oder `\drawbeam(node){comp}`. Werden mehr als zwei Argumente mitgegeben, werden die Verbindungen  $1 \rightarrow 2$ ,  $2 \rightarrow 3$ , usw. gezeichnet.

Die automatischen Verbindungen werden intern als `\drawwire` von den Referenzknoten zur Komponente gezeichnet.

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

Siehe `fiberalign`.

`wireangleA`= $\langle num \rangle$  default: 0

Siehe `fiberangleA`.

`wireangleB`= $\langle num \rangle$  default: 0

Siehe `fiberangleB`.

`wirestyle`= $\langle string \rangle$  default: angle

Siehe `fiberstyle`, der Vorgabewert ist jedoch angle.

`addtoWire`= $\langle list \rangle$

Der vorhandene `Wire`-Stil wird lokal um die Parameter in  $\langle list \rangle$  erweitert,  $\langle list \rangle$  muss mit geschweiften Klammern gekapselt werden.

`newWire`= $\langle list \rangle$

Ähnlich wie `addtoWire`, nur wird der `Wire`-Stil mit den neuen Parametern überschrieben.

`Wire`  $\langle psstyle \rangle$

Das Aussehen der elektrischen Verbindungen wird durch diesen Stil vorgegeben, analog zu Kap. 9.6.2. Das optionale Argument von `\drawwire` kann ebenfalls verwendet werden, die entsprechenden Parameter können die Einstellungen von `Wire` überschreiben.

## 9.8. Automatische Faser- und Drahtverbindungen

Alle Faserkomponenten, die in Kap. 5 beschrieben werden, werden automatisch über `\drawfiber` mit ihren Referenzknoten verbunden, das Verhalten ist in Kap. 9.6.1 beschrieben. Entsprechend dazu werden die elektrischen Komponenten (Kap. 6) mit `\drawwire` mit Drähten verbunden. Die automatischen Verbindungen können umfangreich kontrolliert werden, ohne diese Makros explizit zu verwenden. Das Aussehen jeder Verbindung kann ebenfalls separat konfiguriert werden, siehe Kap. 9.8.1.

`fiber=[*+]none, all, i, o, <refpoint>` default: all

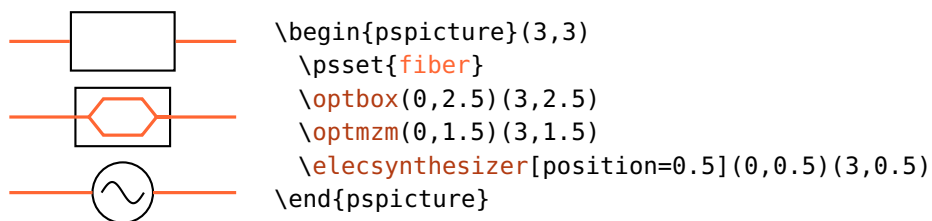
Dieser Parameter gibt an, welche Faserverbindungen gezeichnet werden. Der Parameter kann nur die Faserkomponenten betreffen (falls ein \* vorangestellt ist), nur die elektrischen und Freistrahlskomponenten (+ wird vorangestellt), oder alle Komponententypen. Typische Freistrahlskomponenten, die häufig als Faserkomponenten verwendet werden, sind `\optbox` und `\optdetector`. Beachten Sie, dass elektrische Komponenten nicht automatisch mit Fasern verbunden werden können, Sie müssen dann explizit `\drawfiber` verwenden.

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.

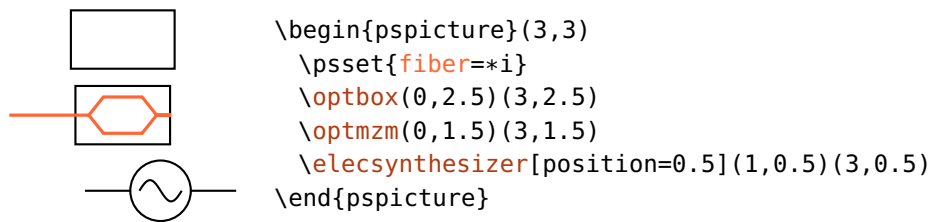
Komponenten, die über mehr als einen Eingang oder Ausgang verfügen (`Koppler` und `\optcirculator`) erlauben weitergehende Auswahl mit t und b, möglicherweise in Verbindung mit i, o, l und r.

Die folgenden Beispiele zeigen einige Möglichkeiten.

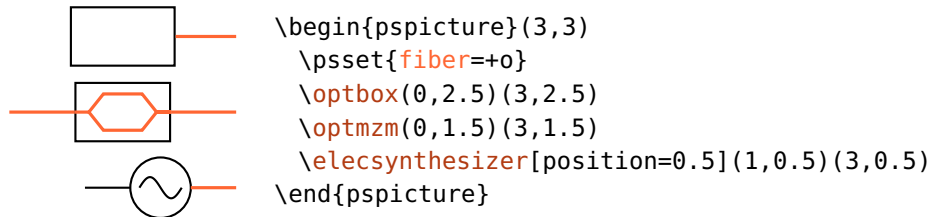
1. Zeichne alle Fasern für alle Komponenten.



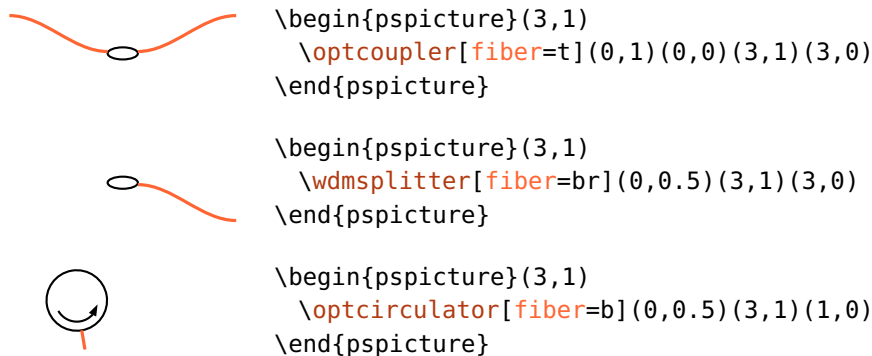
2. Nur die Eingangsverbindung einer Faserkomponente (`\optmzm`) wird gezeichnet, die anderen Komponenten bleiben unbeeinflusst.



3. Von der Faserkomponente werden alle Verbindungen gezeichnet, die anderen Komponenten erhalten nur eine Ausgangsverbindung.



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

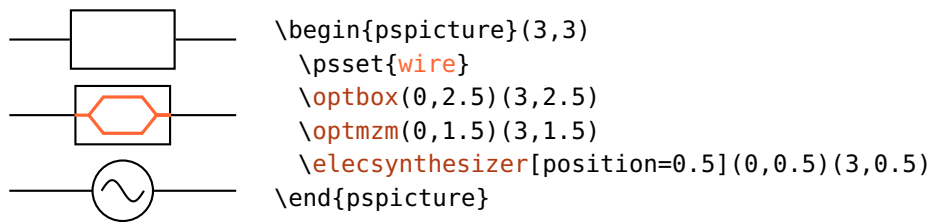


`wire=[*+]none, all, i, o,  $\langle$ refpoint $\rangle$`

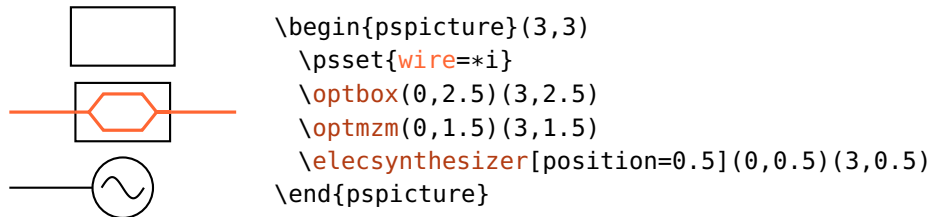
default: all

Dieser Parameter gibt an, welche elektrische Verbindungen gezeichnet werden. Der Parameter kann nur die elektrischen Komponenten betreffen (falls ein \* vorangestellt ist), nur die Faser- und Freistrahlkomponenten (+ wird vorangestellt), oder alle Komponenten.

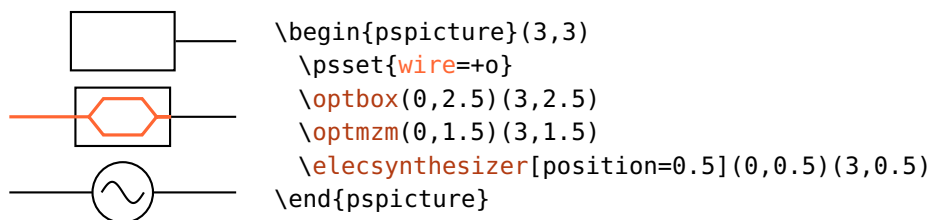
1. Zeichne alle Drähte aller Komponenten.



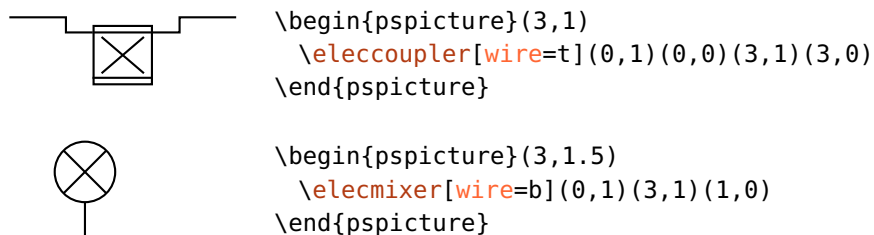
2. Nur die Eingangsverbindung einer elektrischen Komponente (`\elecsynthesizer`) wird gezeichnet, die anderen Komponenten bleiben unbeeinflusst.



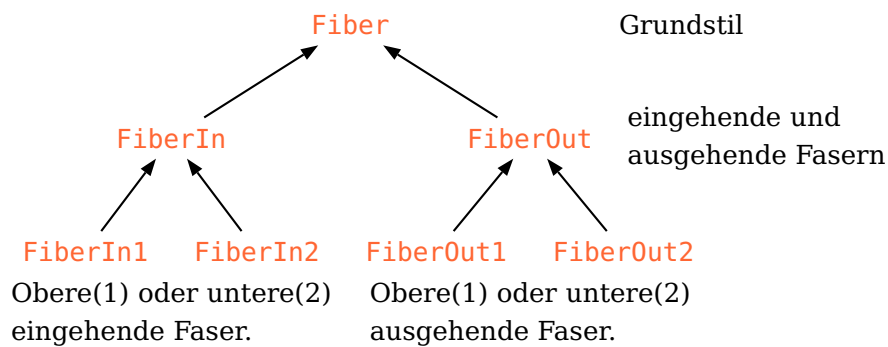
3. Von der Freistrah- und Faserkomponente wird nur die Ausgangsverbindung gezeichnet, die elektrischen Komponente erhält jedoch alle Verbindungen.



4. Die Werte `t` und `b` betreffen nur die Komponenten mit mehr als zwei Verbindungen, also `\elecmixer` und `\eleccoupler`.







**Abbildung 9.1.:** Vererbungsdiagramm der PS-Stile für die automatischen Faserverbindungen. Sie sollten diese Stile mit `\addtopstyle` oder den entsprechenden `addto<style>` Parametern ändern um die Vererbungslinie beizubehalten. Eine identisches Vererbungsdiagramm besteht für die entsprechenden Wire... Stile.

### 9.8.1. Aussehen der automatischen Verbindungen

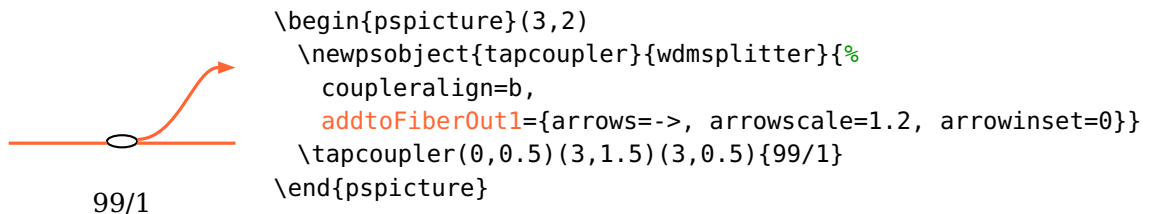
Die Stile aller automatischen Faserverbindungen können separat konfiguriert werden, der zugrundeliegende Stil ist immer **Fiber**, der in Kap. 9.6.2 beschrieben ist.

**FiberIn** 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. 9.1 für ein Vererbungsdiagramm der Faserstile. Beachten Sie, dass diese Vererbung nur erhalten bleibt, falls die Stile nur mit `\addtopstyle` oder den entsprechenden `addto<style>` Parametern verändert werden.

**WireIn** Der Stil **WireIn** bezieht sich auf alle eingehenden elektrischen Verbindungen und erbt alle Eigenschaften von **Wire**. Hat eine Komponente zwei davon, so können diese mit **WireIn1** und **WireIn2** angesprochen werden, die alles von **WireIn** übernehmen. Entsprechend bezieht sich **WireOut** auf alle ausgehenden elektrischen Verbindungen, falls es davon zwei gibt dann können diese mit **WireOut1** und **WireOut2** geändert werden. Siehe Abb. 9.1 für ein Vererbungsdiagramm der Stile. Beachten Sie, dass diese Vererbung nur erhalten bleibt, falls die Stile nur mit `\addtopstyle` oder den entsprechenden `addto<style>` Parametern

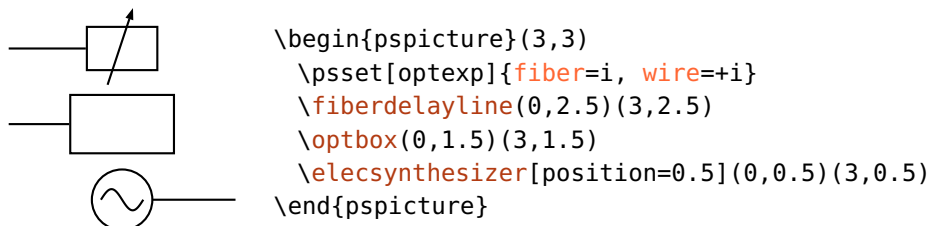
verändert werden.

`new<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 Verbindungen definiert werden (Kap. 10.1), oder explizite Gruppierung kann vermieden werden.



### 9.8.2. Bekannte Einschränkungen

Die Kombination von `fiber` und `wire` erlaubt in der derzeitigen Implementierung (Version v5.1) nicht die beliebige Kombination aller Parametervarianten. Die meisten funktionieren, aber das folgende Beispiel zeigt eine Variante, die nicht korrekt funktioniert (der `\elecsynthesizer` sollte eine Eingangsfaser haben):



In solchen speziellen Fällen muss auf manuelle Verbindungen zurückgegriffen werden.

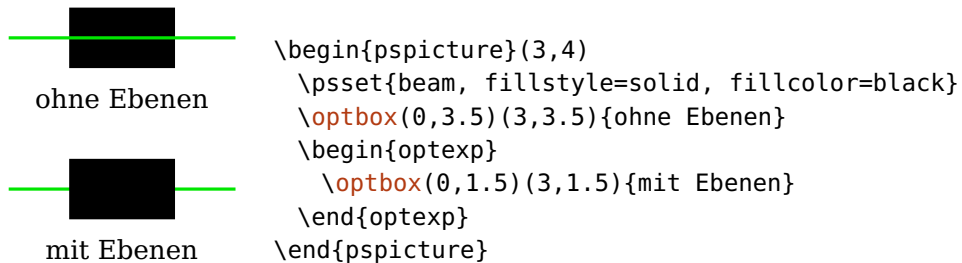
### 9.9. Zeichenebenen

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}
```



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.

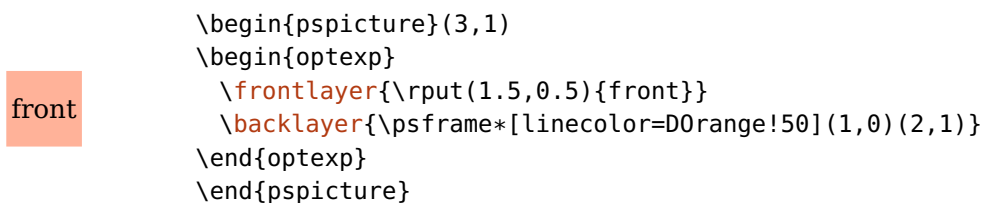
```
\backlayer{<code>}
```

Führe `<code>` nur beim ersten Durchgang aus.

```
\frontlayer{<code>}
```

Führe `<code>` nur beim zweiten Durchgang aus.

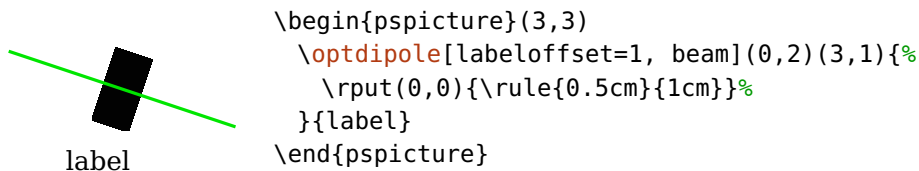
In diesem Beispiel wird der Text über dem Rechteck angezeigt, obwohl der Text zuerst definiert wird:



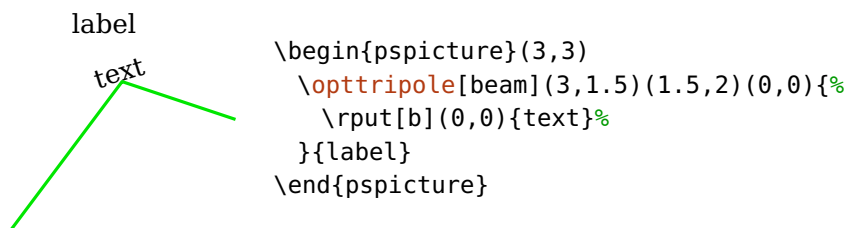
## 10. Benutzerdefinierte Komponenten

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⟩){⟨comp⟩}{⟨label⟩}`



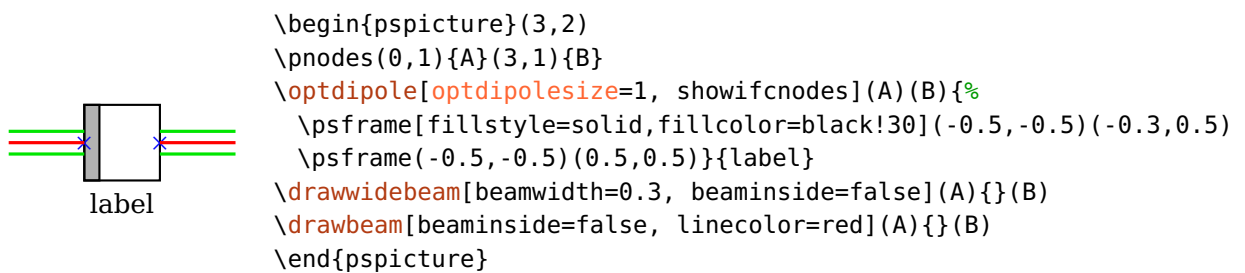
`\opttripole[⟨options⟩](⟨in⟩)(⟨center⟩)(⟨out⟩){⟨comp⟩}{⟨label⟩}`



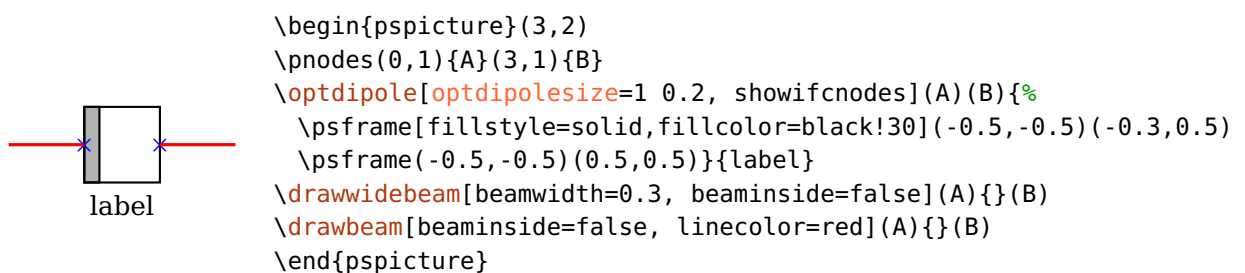
Diese Komponenten können sowohl mit Fasern als auch mit Strahlen verbunden werden, können aber nicht deren volles Potential ausschöpfen. `\opttripole` definiert eine einzelne reflektive Grenzfläche, `\optdipole` kann zwei transmittive Grenzflächen haben:

`optdipolesize=⟨width⟩[⟨height⟩]` default: 0 0

Hiermit können zwei transmittierende Grenzflächen für `\optdipole` definiert werden, die sich bei  $(\pm 0.5\langle width \rangle, 0)$  befinden.



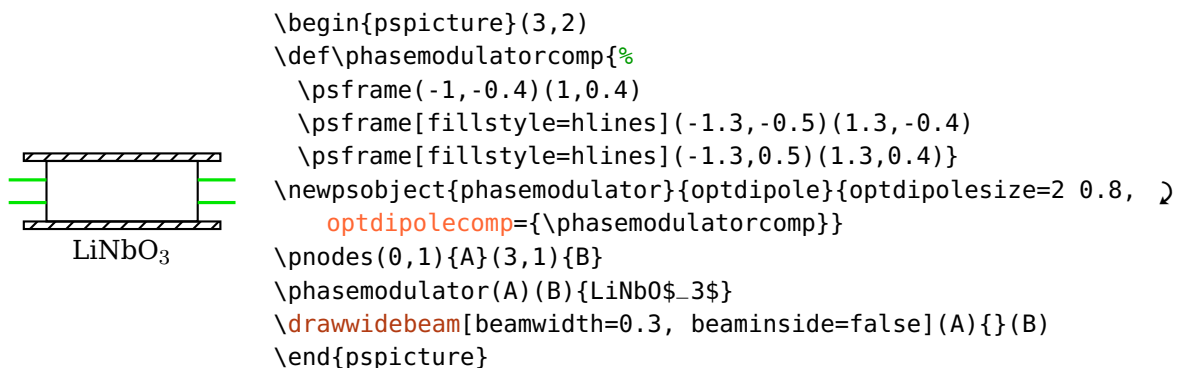
Wenn zusätzlich eine Höhe (*height*) ungleich Null angegeben wird, so wird diese als Apertur für das Raytracing genommen (siehe [useNA](#)). Im folgenden Beispiel ist die Höhe gleich 0.2 und damit kleiner als der aufgeweitete Strahl der deshalb anders als im vorangegangenen Beispiel nicht gezeichnet wird.



Weitere Beispiele sind Bsp. 11.11 und Bsp. 11.24.

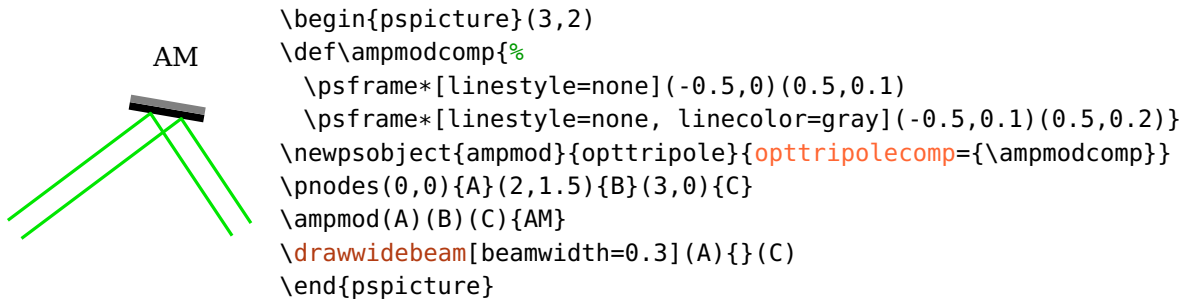
`optdipolecomp`=*(macros)*

Mit dieser Option kann die Zeichnung einer Komponente definiert werden, um dann mit `\newpsobject` basierend auf `\optdipole` eine neue Komponente zu definieren.



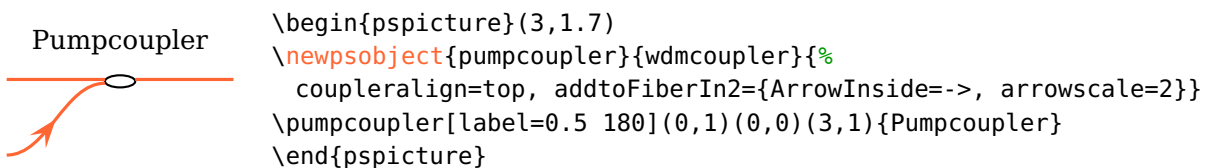
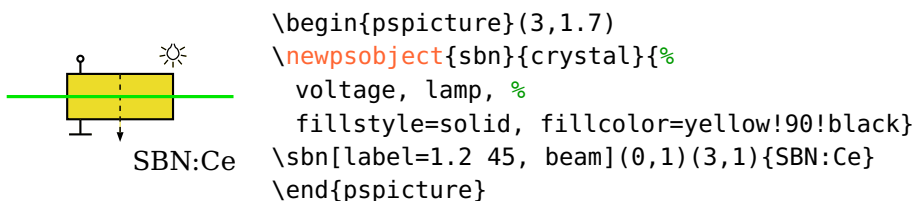
`opttripolecomp=<macros>`

Mit dieser Option kann die Zeichnung einer Komponente definiert werden, um dann mit `\newpsobject` basierend auf `\opttripole` eine neuen Komponente zu definieren.

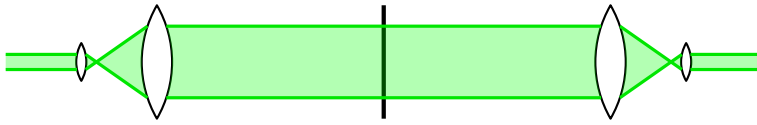


## 10.1. Benutzerdefinierte Version existierender Komponenten

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



Bsp. 10.1



```
\begin{pspicture}(10,2)
  \newpsobject{MOLensIn}{lens}{lens=0.5 0.5 0.5}
  \newpsobject{MOLensOut}{lens}{lens=1.5 1.5 1.5}
  \pnodes(0,1){A}(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}
```

## 10.2. 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 seit Version 3.0 werden grob beschrieben, können sich aber noch ändern und sind nicht zwangsläufig abwärtskompatibel.

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

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

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

```
\newOptexpElecDipole[⟨fixopt⟩]{⟨name⟩}{⟨dftopt⟩}
```

Diese Makros erzeugen den Code für die Positionierung, Drehung, Beschriftung und die Ebenenverwaltung der neuen Komponente  $\backslash\langle name \rangle$ . Im einfachsten Fall müssen Sie lediglich die eigentliche Zeichnung der Komponente erstellen.

Mit dem Argument  $\langle \text{fixopt} \rangle$  können Optionen voreingestellt werden, die später über das optionale Argument der Komponente nicht verändert werden können,  $\langle \text{dftopt} \rangle$  erlaubt Voreinstellungen zu machen, die überschrieben werden können. Beide Argumente  $\langle \text{fixopt} \rangle$  und  $\langle \text{dftopt} \rangle$  sind optional und können auch ganz weggelassen werden.

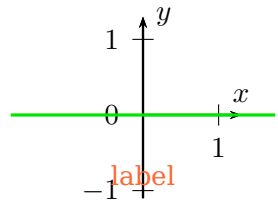
Die Definition der neuen Komponenten kann in unterschiedliche Schritte eingeteilt werden, je nach Komplexität und Anforderung des Benutzers:

1. Die Zeichnung (siehe Kap. 10.2.1), das ist der einzige erforderliche Schritt.
2. Unterstützung von Raytracing (optional).
3. Unterstützung für `rotateref`, `extnode` und `position=start|end` (optional).

Die folgenden Abschnitte enthalten die Entwicklung einer einfachen Beispielkomponente `\testcomp`.

### 10.2.1. 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.

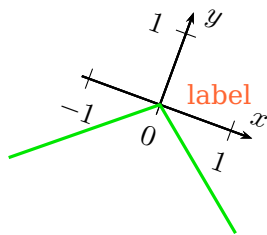


```

\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>

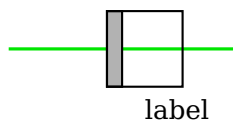
```





```
\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}
```

Unsere Beispielkomponente und ihre Zeichnung wird also wie folgt definiert:



```
\begin{pspicture}(3, 2)
\newOptexpDipole{testcomp}
\makeatletter
\def\testcomp@comp{%
\psframe[fillstyle=solid, 2
fillcolor=black!30](-0.5,-0.5)(-0.3,0.5)
\psframe(-0.5,-0.5)(0.5,0.5)
}%
\makeatother
\begin{optexp}
\testcomp[position=0.6, labelalign=l](0,1)(3,1){label}
\drawbeam[beaminside=false](0,1){}(3,1)
\end{optexp}
\end{pspicture}
```

Die Positionierung der Komponente und der Beschriftung, sowie die Ebenen funktioniert also schon.

## 10.2.2. Unterstützung von Raytracing

Damit die Komponente Raytracing unterstützt, müssen die Grenzflächen mit ihren Eigenschaften definiert werden. Die Definition der Ebenen werden in einem Makro  $\backslash(name)@nodes$  definiert, damit sie auch in den unterschiedlichen Ebenen verfügbar sind.

Die Definition der Ebenen erfolgt dann über das Makro  $\backslash newOptexpComp$

```
\def\testcomp@nodes{%
\newOptexpComp{%
% Definition der ersten Grenzflaeche (GF)
```

```

{-0.5 0} % x- und y-Koordinate der GF-Mitte
{0 1} % Richtungsvektor der GF (x und y)
-0.5 0.5 % Grenzen fuer die NA, gesehen als Faktoren des GF-Vektor
trans % Verhalten der GF, kann 'trans', 'refl' oder 'abs' sein
{PlainIfc} % Art der GF
% Dasselbe fuer die rechte GF, nur die Mitte ist anders
{0.5 0} {0 1} -0.5 0.5 trans {PlainIfc}
\POE@key@n % Brechungsindex, koennte jede Zahl sein
}%
}%

```

Unsere Komponent sieht also folgendermaßen aus, und der Strahl verläuft nicht mehr innerhalb der Komponente:



```

\begin{pspicture}(3, 2)
\newOptexpDipole{testcomp}
\makeatletter
\def\testcomp@comp{%
\psframe[fillstyle=solid, 2
fillcolor=black!30](-0.5,-0.5)(-0.3,0.5)
\psframe(-0.5,-0.5)(0.5,0.5)}%
\def\testcomp@nodes{%
\newOptexpComp{%
{-0.5 0} {0 1} -0.5 0.5 trans {PlainIfc}
{0.5 0} {0 1} -0.5 0.5 trans {PlainIfc} \POE@key@n }}%
\makeatother
\begin{optexp}
\testcomp[position=0.6, labelalign=l](0,1)(3,1){label}
\drawbeam[beaminside=false](0,1){}(3,1)
\end{optexp}
\end{pspicture}

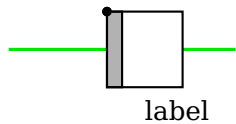
```

### 10.2.3. Unterstützung von externen Knoten

Damit nun, als letzter Schritt auch externe Knoten mit `extnode`, Positionierung an den Anfang oder das Ende mit `position=start|end`, oder auch das Setzen eines anderen Drehpunktes mit `rotateref` möglich wird, müssen die externen Dimensionen der Komponente bekannt gemacht werden. Dafür muss ein weiteres Makro mit dem Name `\<name>@ref` definiert werden, in dem auf Postscript-Ebene die Werte `@@x` (halbe Breite), `@@y` (halbe Höhe), und optional auch `@@x0` und `@@y0` (Verschiebung der Mitte) definiert werden müssen.

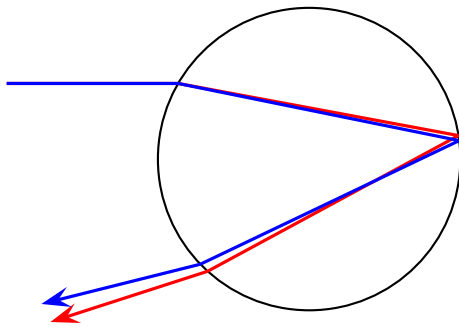
```
\def\testcomp@ref{%
  \POE@setref{ /@@x 0.5 def /@@y 0.5 def }%
}%
```

Der Code für unsere Beispielkomponente `\testcomp` ist damit vollständig, und sieht folgendermaßen aus:



```
\begin{pspicture}(3, 2)
\newOptexpDipole{testcomp}
\makeatletter
\def\testcomp@comp{%
  \psframe[fillstyle=solid, 2
    fillcolor=black!30](-0.5,-0.5)(-0.3,0.5)
  \psframe(-0.5,-0.5)(0.5,0.5)}%
\def\testcomp@nodes{%
  \newOptexpComp{%
    {-0.5 0} {0 1} -0.5 0.5 trans {PlainIfc}
    {0.5 0} {0 1} -0.5 0.5 trans {PlainIfc} \POE@key@n }}%
\def\testcomp@ref{%
  \POE@setref{ /@@x 0.5 def /@@y 0.5 def }}%
\makeatother
\begin{optexp}
\testcomp[position=0.6, labelalign=l, extnode=tl](0,1)(3,1){label}
\drawbeam[beaminside=false](0,1){}(3,1)
\psdot(\oenodeExt{})
\end{optexp}
\end{pspicture}
```

## 10.2.4. Beispiele für benutzerdefinierte Komponenten



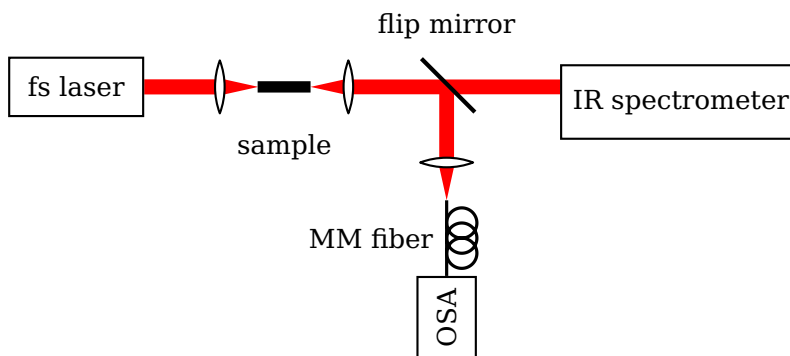
```

\begin{pspicture}(6,4)
  \newOptexpTripole{raindrop}
  \makeatletter
  \def\raindrop@nodes{%
    \newOptexpComp{%
      {0 -2} { 0 2 } -2 2 trans {CurvedIfc}
      {0 2} {0 -2} -2 2 refl {CurvedIfc}
      {0 -2} { 0 2 } -2 2 trans {CurvedIfc}
      \POE@key@n }}%
  \def\raindrop@comp{\pscircle(0,0){2}}%
  \raindrop(0,4)(4,2)(0,0)
  \psset{beamalign=abs, arrows=->, arrowscale=2}
  \drawbeam[linecolor=red](0,3){}(0,1)
  \drawbeam[n=*n*1.05, linecolor=blue](0,3){}(0,1)
\end{pspicture}

```

## 11. Beispiele

Bsp. 11.1

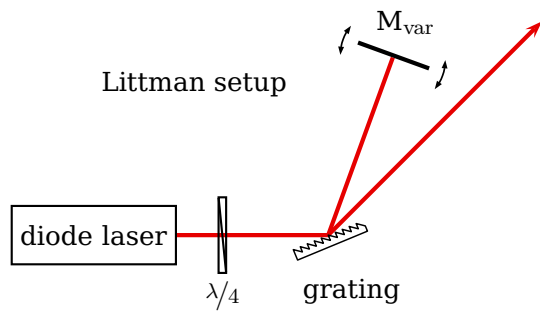


```

\begin{pspicture}(0.1,0.4)(10,4.5)
  \psset{optexp}{lens=1 1 0.7, loadbeampoints}
  \pnodes(2,4){L}([Xnodesep=4]L){M}([Xnodesep=1.5]M){IRSpec}([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[angle=90]([offset=-1.5]M)
    \optfiber[fiberloopradius=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. 11.2

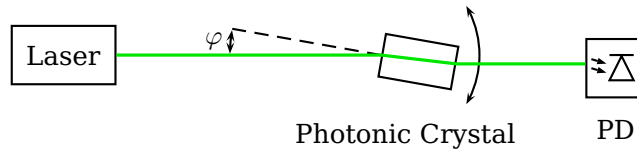


```

\begin{pspicture}(-4.2,-1)(3,3)
\nodes(-2,0){L0ut}(0,0){Grat}(4;45){0ut}(2.5;70){Mvar}
\newpsstyle{Beam}{linewidth=2\pslinewidth, linecolor=red!90!black}
\begin{optexp}
\optbox[optboxwidth=2.2, labeloffset=0, position=start](L0ut)(Grat){diode laser}
\mirror[variable, label=0.5](Grat)(Mvar)(Grat){M$_{\mathrm{var}}$}
\optretplate[position=0.3](L0ut)(Grat){$\frac{\lambda}{4}$}
\optgrating(L0ut)(Grat)(0ut){grating}
\drawbeam[arrows=->]{1}{3}{4}(0ut)
\drawbeam{2}{4}
\end{optexp}
\end{pspicture}
\end{pspicture}

```

Bsp. 11.3

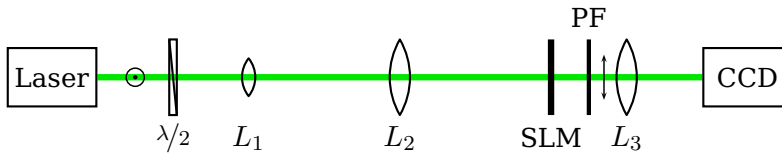


```

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

```

Bsp. 11.4

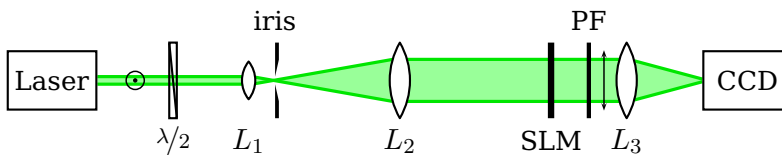


```

\begin{pspicture}(10.4,1.7)
\pnodes(1.2,1){Start}(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){$\frac{\lambda}{2}$}
\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}

```

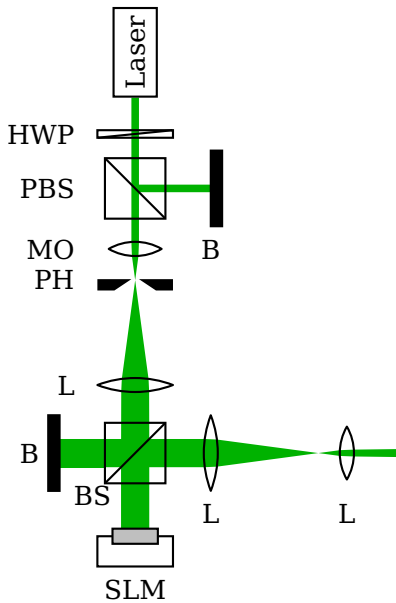
Bsp. 11.5



```

\begin{pspicture}(10.4,1.7)
\pnodes(1.2,1){Start}(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){$\frac{\lambda}{2}$}
\lens[lens=0.4 0.4 0.5,abspos=2](Start)(CCD){$L_1$}
\pinhole[pewidth=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. 11.6: Angepasst von [arXiv:1112.5270v2](https://arxiv.org/abs/1112.5270v2)

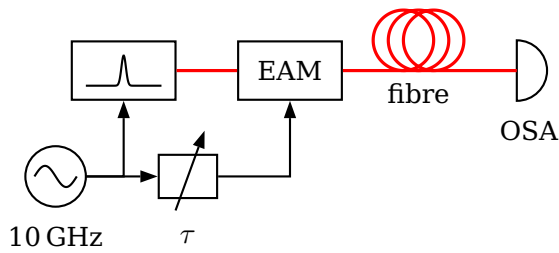
```

\begin{pspicture}(0.3,0.4)(5.5,8.3)
\pnodes(2,7){L}(2,5.8){Pbs}([Xnodesep=1]Pbs){Blk1}(2,2.3){Bs}
\pnodes([Xnodesep=-1]Bs){Blk2}([Xnodesep=3.5]Bs){Out}(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){M0}
\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}
\opttripole(Bs)(Slm)(Bs){%
\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)}{SLM}
\addtopsstyle{Beam}{linestyle=none, fillstyle=solid, fillcolor=green!70!black}
\drawwidebeam[beamwidth=0.1]{1-4}
\drawwidebeam[beamwidth=0.1, beaminsidfirst, beaminsidelast]{3}{5-8}
\psset{savebeampoints=false, loadbeampoints}
\drawwidebeam{8-11}(Out)\drawwidebeam[beaminsidfirst]{8}{12}
\end{optexp}
\end{pspicture}

```



Bsp. 11.7

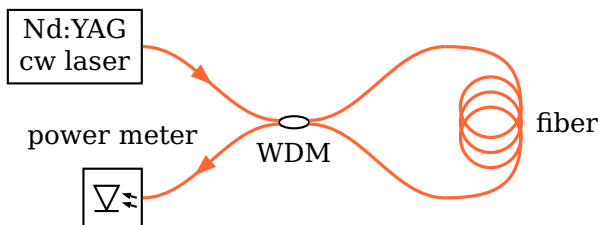


```

\begin{pspicture}(6.4,3.2)
\addtopsstyle{Fiber}{linecolor=red}\psset{fiber=none}
\pnodes(2.3,2.3){Lin}([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=-0.5,offset=-1]\oenableExt{L}){Osc}
\elecsynthesizer[wire=none](Osc)(\oenableExt{EAM}|Osc){10\,GHz}
\optarrowcomp[arrowcompsize=0.8 0.6, 2
arrowscale=1.5](Osc)(\oenableExt{EAM}|Osc){$\tau$}
\drawfiber{1-4}
\addtopsstyle{Wire}{arrows=->, arrowinset=0, arrowscale=1.5}
\drawwire{5}{6}\psset[optexp]{wireangleB=90}
\drawwire{6}(\oenableExt{EAM})\drawwire{5}(\oenableExt{L})
\end{pspicture}

```

Bsp. 11.8

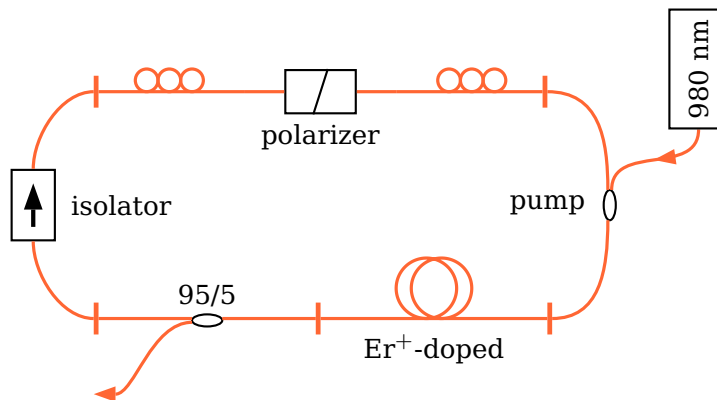


```

\begin{pspicture}(0.2,0.6)(8.2,3.5)
\pnodes(2,3){Laser}(2,1){PwMeter}(6,3){CplTop}(6,1){CplBot}
\psset{arrowscale=1.5, arrowinset=0}
\optbox[position=start, optboxsize=1.8 1, 2
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[detttype=diode]([Xnodesep=0.1]PwMeter)(PwMeter){power meter}
\end{pspicture}

```

Bsp. 11.9

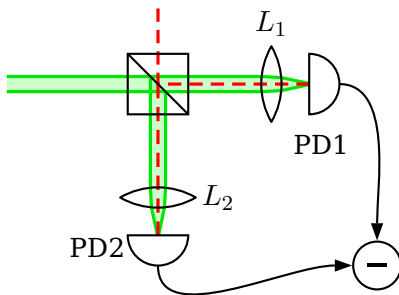


```

\begin{pspicture}(0.9,0.9)(10.4,6.1)
\psset{arrowscale=1.5, arrowinset=0}
\nodes(2,5){PC1in}(4,5){PC1out}(6,5){PC2in}(8,5){PC2out}(2,2){CplSig}
\nodes(5,2){CplIn}(2,1){CplOut}(10,4.5){Pump}(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, 2
        optboxwidth=1.6](Pump)([offset=-0.1]Pump){980~nm}
\optfiber[fiberloops=2, labeloffset=0.4](CplIn)(PumpSig){Er$^+$-doped}
\end{pspicture}

```

Bsp. 11.10

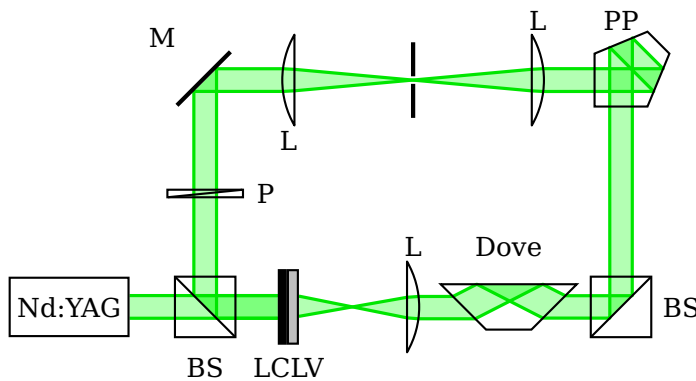


```

\begin{pspicture}(1,1.2)(6.2,5)
\pnodes(1,4){SigIn}(3,4){BS}(3,5){L0}(5,4){Det1}(3,2){Det2}
\begin{optexp}
\optplane[compname=L0, angle=90](L0)
\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{L0}{BS}{L1}{Det1}
\drawbeam[beaminsidefirst]{BS}{L2}{Det2}
\end{optexp}
\cnodeput[framesep=5pt]([Xnodesep=0.5, 2
offset=-0.5]\oenableExt{Det1}|\oenableExt{Det2}){M}{\rule{2.5mm}{1pt}}
\psset{arrows=<- , arrowscale=1.5, arrowinset=0}
\ncurve[angleA=90]{M}{\oenableExt{Det1}}
\ncurve[angleA=180, angleB=-90]{M}{\oenableExt{Det2}}
\end{pspicture}

```

## Bsp. 11.11



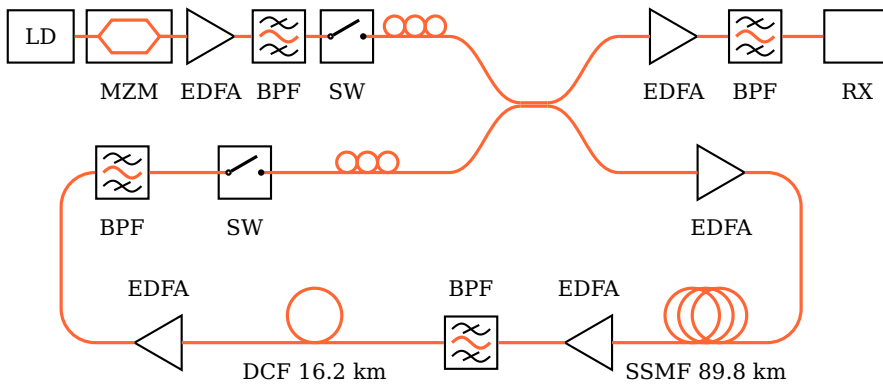
```

\begin{pspicture}(-0.2,0)(9,5)
\psset{optexp}{lens=1.2 0 1.2, n=1.72}
\addtopsstyle{Beam}{fillstyle=solid, fillcolor=green, opacity=0.3}
\nodes(2.4,1){BS1}([offset=3]BS1){M1}([Xnodesep=5.5]M1){PP}(PP|BS1){BS2}
\begin{optexp}
\optbox[label=0, position=start, optboxwidth=1.6]([Xnodesep=-1]BS1)(BS1){Nd:YAG}
\beamsplitter[comname=BS](BS2)(BS1)(M1){BS}
\optdipole[position=0.2, comname=LCLV, optdipolesize=0.28 1](BS1)(BS2){%
\psframe[fillstyle=solid,fillcolor=black,dimen=outer](-0.14,-0.5)(-0.02,0.5)
\psframe[fillstyle=solid,fillcolor=gray!50,dimen=outer](-0.02,-0.5)(0.14,0.5)
}{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[comname=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. 11.13

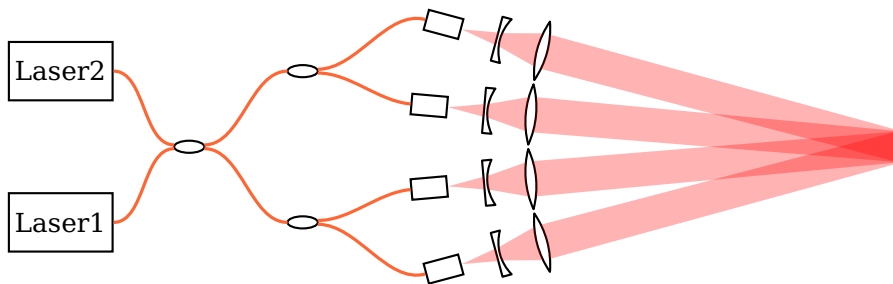


```

\begin{pspicture}(13,5.5)
\psset{optboxwidth=1, labelstyle=\footnotesize, fiber=none}
\pnodes(1,5){LD}([Xnodesep=5.5]LD){CPLin1}([offset=-2]CPLin1){CPLin2}
\pnodes([Xnodesep=2.5]CPLin1){CPLout1}([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{1-6}
\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{8-10}
\pnodes([Xnodesep=-0.5]LD|CPLin2){TL}(RX|TL){TR}([offset=-2.5]TR){BR}(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, lineararc=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{12-16}\drawfiber{17-19}
\drawfiber[fiberstyle=angle, arm=0.5, lineararc=0.5, stopnode=1]{16}{17}
\end{pspicture}

```

Bsp. 11.14

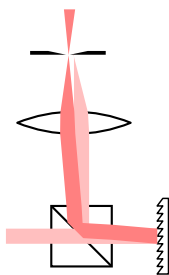


```

\begin{pspicture}(12, 4)
\psset{labeloffset=0.3, fiber=none}
\nodes(1.6,2){In}(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. 11.15

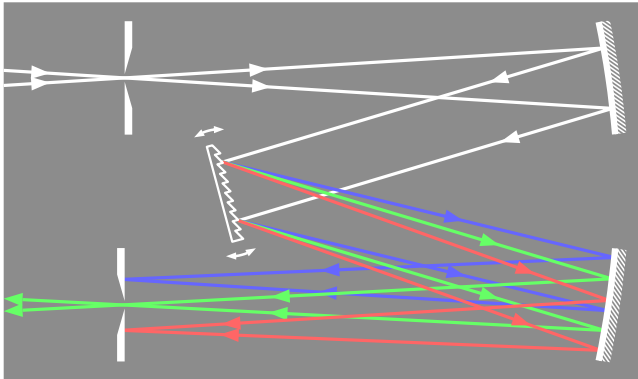


```

\begin{pspicture}(0.5,0)(2.2,4)
\nodes(0,1){A}(1,1){BS}(2,1){G}(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[pewidth=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. 11.16: Angepasst von Wikipedia



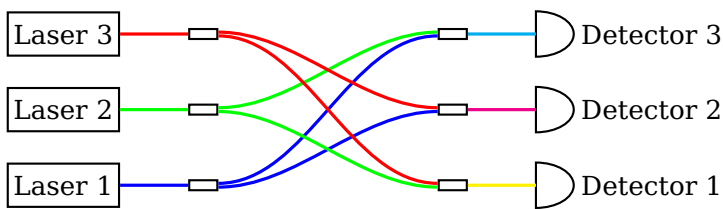
```

\begin{pspicture}(0,-1)(8.5,4)
\psframe[fillstyle=solid,fillcolor=gray!90,linestyle=none](0,-1)(8.5,4)
\addtopsstyle{OptComp}{linecolor=white}
\newsstyle{Beam}{ArrowInside=->, linejoin=2, arrowscale=1.3, arrowinset=0}
\addtopsstyle{ExtendedMirror}{hatchcolor=white, hatchsep=0.5\pslinewidth}
\pnodes(0,3){A}(8,3){B}(3,1.5){C}(8,0){D}(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]{3-5}(E)
  \drawwidebeam[ArrowInsidePos=0.7, linecolor=green!60!white, arrows=->, ]
    ArrowInsideMinLength=2]{3-5}(E)
  \drawwidebeam[ArrowInsidePos=0.8, beamangle=-3, linecolor=red!60!white]{3-5}(E)
\end{optexp}
\end{pspicture}

```



Bsp. 11.17

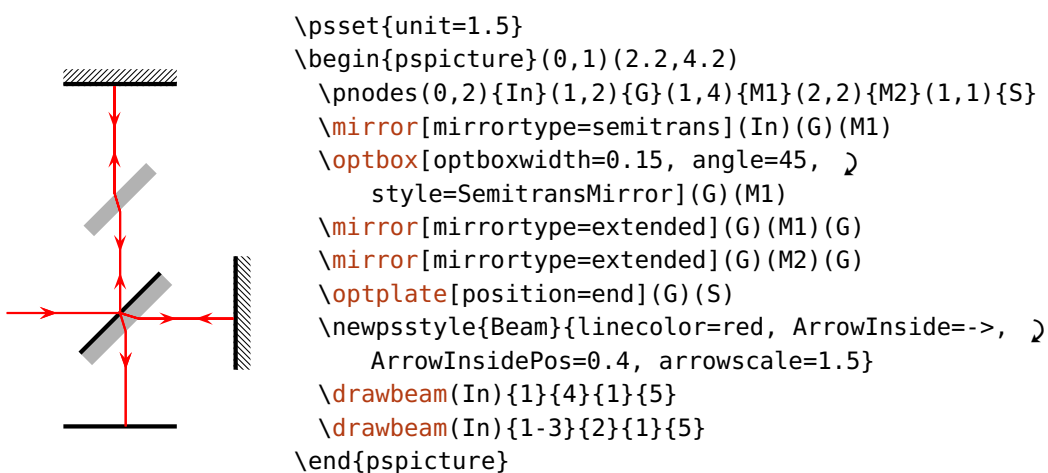


```

\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}

```

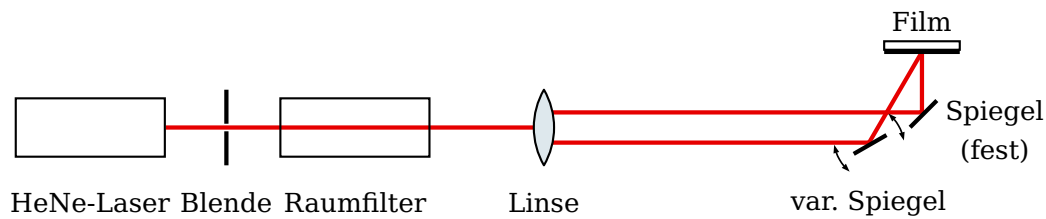
Bsp. 11.18



```

\psset{unit=1.5}
\begin{pspicture}(0,1)(2.2,4.2)
\pnodes(0,2){In}(1,2){G}(1,4){M1}(2,2){M2}(1,1){S}
\mirror[mirrortype=semitrans](In)(G)(M1)
\optbox[optboxwidth=0.15, angle=45, 2
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=->, 2
ArrowInsidePos=0.4, arrowscale=1.5}
\drawbeam(In){1}{4}{1}{5}
\drawbeam(In){1-3}{2}{1}{5}
\end{pspicture}

```

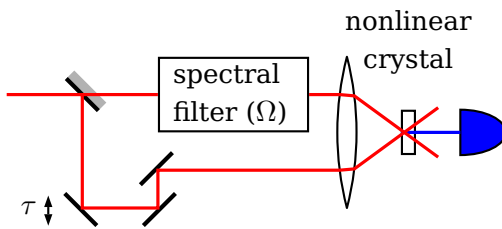
Bsp. 11.19: Angepasst von [semibyte.de](http://semibyte.de)

```

\begin{pspicture}(13.6,2.6)
  \psset{optexp}{labeloffset=1, mirrorwidth=0.5, optboxwidth=2}
  \pnodes(2,1){Laser}{12,2}{Film}{Film|Laser}{Ref}
  \begin{optexp}
    \optbox[position=start](Laser)(Ref){HeNe-Laser}
    \pinhole[abspos=0.8](Laser)(Ref){Blende}
    \optbox[abspos=2.5](Laser)(Ref){Raumfilter}
    \lens[fillstyle=solid, fillcolor={rgb}{0.87,0.91,0.93}](Laser)(Ref){Linse}
    \mirror[label=1 30]([offset=0.2]Laser)([offset=0.2]Ref)(Film){%
      \begin{tabular}{c}Spiegel\\(fest)\end{tabular}}
    \opttripole[label=0.4](Ref)(Film)(Ref){%
      \psframe(0.5,1\pslinewidth)(-0.5,5\pslinewidth)
      \psline[linewidth=2\pslinewidth](0.5,0)(-0.5,0)
    }{Film}
    \newpsstyle{Beam}{linewidth=2\pslinewidth, linecolor=red!90!black, 2
      beamalign=absolute}
    \drawbeam{1-4}
    \drawbeam[beampos=0.2]{4-6}
    \mirror[variable, label=0.8 -30]([offset=-0.2]Laser)([offset=-0.2, 2
      Xnodesep=-0.7]Ref)(Film){var. Spiegel}
    \drawbeam[beampos=-0.2]{4}{7}{6}
  \end{optexp}
\end{pspicture}

```

Bsp. 11.20

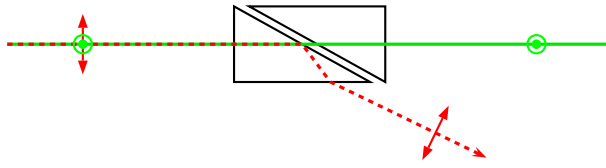


```

\begin{pspicture}(0,0.2)(6.6,3.1)
\psset{mirrorwidth=0.6, useNA=false, beaminsidfirst}
\pnodes(0,2){In}(1,2){BS}(1,0.5){M1}(2,1){M2}(6,1.5){Det}
\mirror[mirrortype=semitrans, n=1](In)(BS)(M1)
\mirror[label=0.6 180 . absolute, compname=M](BS)(M1)(M2|M1)
\mirror(M1)(M2|M1)(M2)
\mirror(M2|M1)(M2)([Xnodesep=1]M2)
\optbox[optboxsize=2 1, abspos=2, label=0, 2
allowbeaminside=false](BS)([offset=0.5]Det){%
\parbox{1.6\psxunit}{spectral\filter ($\Omega$)}}
\lens[lens=4 4 2, n=3.3]([Xnodesep=-3]Det)(Det)
\optbox[optboxsize=0.2 0.6, n=1, label=-1.2](\oencodeOut{6})(Det){%
\begin{tabular}{@{}c@{}}nonlinear\crystal\end{tabular}}
\optplate[linestyle=none](\oencodeOut{7})(Det)
\optdetector[detsize=0.6 0.6, fillstyle=solid, fillcolor=blue](\oencodeOut{6})(Det)
\newsstyle{Beam}{linewidth=1.5\pslinewidth}
\drawbeam[linecolor=blue]{7-9}
\addtopsstyle{Beam}{linecolor=red}
\drawbeam(In){1-4}{6-8}
\drawbeam{1}{5-8}
\rput[r](\oencodeLabel{M}){$\tau$}
\psline[arrows=<->, arrowinset=0](0.15,0.25)(0.15,-0.15)}
\end{pspicture}

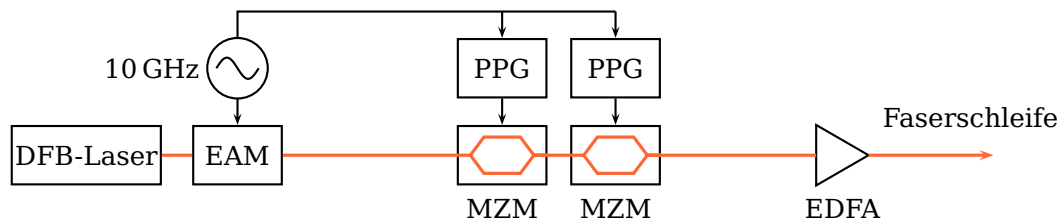
```

Bsp. 11.21



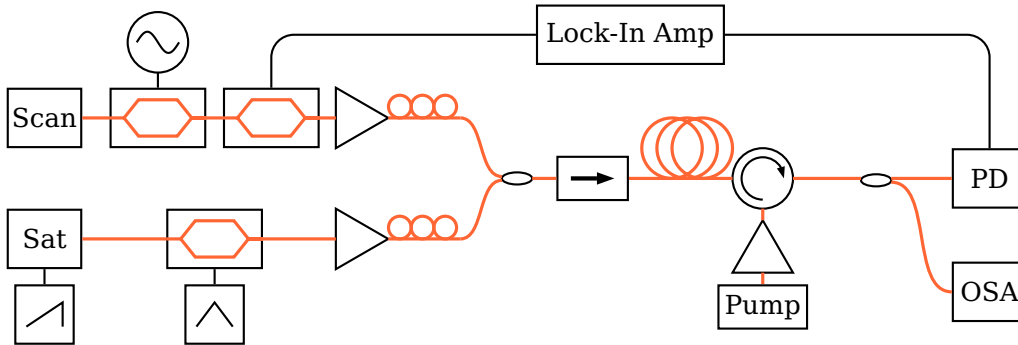
```
\begin{pspicture}(-1,-0.5)(8,1.5)
  \newpsobject{perppol}{polarization}{poltype=perp, linecolor=green, polsize=1}
  \newpsobject{parpol}{polarization}{poltype=parallel, linecolor=red, polsize=0.8}
  \newpsstyle{Polarization}{arrowinset=0, arrowscale=1.2}
  \pnodes(0,1){A}(8,1){B}(4,-0.5){C}
  \glanthompson[glanthompsongap=0.2, glanthompsonsize=2 1](A)(B)
  \optplane(B)\optplane[angle=90](C)
  \drawbeam(A){1-2}
  \drawbeam[linestyle=dashed, dash=2pt 2pt, linecolor=red, arrows=->](A){1}{3}
  \parpol[abspos=1](A)(B)\perppol[abspos=1](A)(B)
  \nodexn{(\oenameBeam)}-1.5(!\oeBeamVec){}{D}
  \perppol[abspos=7](A)(B)\parpol(D)(\oenameBeam){}
\end{pspicture}
```

Bsp. 11.22



```
\begin{pspicture}(13.7,2.9)
  \newpsstyle{Wire}{arm=0.1, arrows=->}
  \psset{optexp}{optboxsize=1.2 0.8, optmzmsize=1.2 0.8, fiber=none, label=0.7}
  \pnodes(2,1){L}([Xnodesep=11]L){A}
  \optbox[position=start, optboxwidth=2, label=0](L)(A){DFB-Laser}
  \optbox[abspos=1, label=0, compname=Eam, extnode=t](L)(A){EAM}
  \optmzm[abspos=4.5, compname=Mzm1, extnode=t](L)(A){MZM}
  \optmzm[abspos=6, compname=Mzm2, extnode=t](L)(A){MZM}
  \optamp[abspos=9](L)(A){EDFA}
  \drawfiber{1-5}\drawfiber[arrows=->]{}(A)
  \pnode([offset=1.5]\oenameExt{Eam}){Osci}
  \elec synthesizer[compname=Syn, position=0.5, wire=o, label=0.5 . 2]
    r](Osci)(\oenameExt{Eam}){10\,GHz}
  \psset{optexp}{label=0, wire, optboxsize=0.8 1.2}
  \optbox[compname=Ppg1](\oenameExt{Mzm1}|Osci)(\oenameExt{Mzm1}){PPG}
  \optbox[compname=Ppg2](\oenameExt{Mzm2}|Osci)(\oenameExt{Mzm2}){PPG}
  \drawwire[arrows=-, wireangleB=-90]{Syn}(\oenameExt{Mzm2}|Osci)
  \rput([offset=0.5, Xnodesep=-0.3]A){Faserschleife}
\end{pspicture}
```

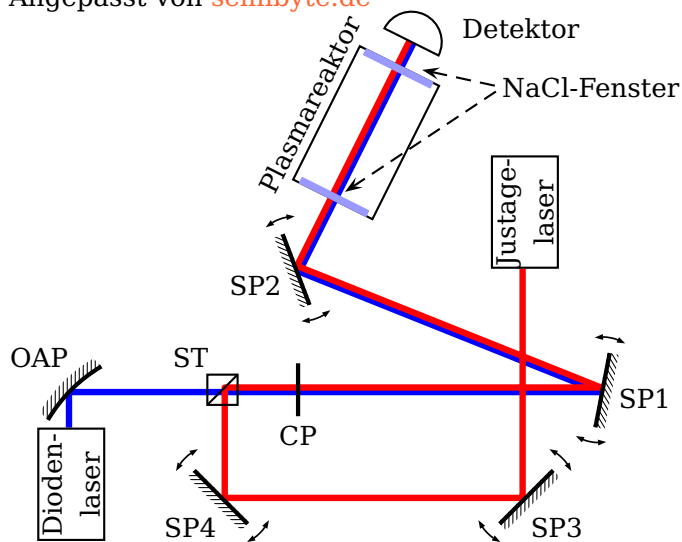
Bsp. 11.23: Angepasst von DOI:10.1364/OL.37.000930



```

\begin{pspicture}(0,0.2)(13.5,4.7)
\psset{optexp}{optboxsize=1 0.8, label=0, fiber=none}
\nodes(1,3.2){Scan}\nnode([offset=-1.6]Scan){Sat}
\node([Xnodesep=5]Scan){ScanCpl}\nnode(ScanCpl|Sat){SatCpl}
\node([Xnodesep=1.5,offset=-0.8]ScanCpl){Cpl}\nnode([Xnodesep=5]Cpl){PD}
\node([Xnodesep=2.5, offset=-1.4]Cpl){Pump}
\optbox[position=start](Scan)(ScanCpl){Scan}
\optmzm[abspos=1, compname=M1, extnode=t](Scan)(ScanCpl)
\optmzm[abspos=2.5, compname=M2, extnode=t](Scan)(ScanCpl)
\optamp[abspos=3.7](Scan)(ScanCpl)
\polcontrol[abspos=4.5, fiber=o](Scan)(ScanCpl)
\drawfiber{1-5}
\optbox[position=start, compname=S, extnode=b](Sat)(SatCpl){Sat}
\optmzm[abspos=1.75, compname=M3, extnode=b](Sat)(SatCpl)
\optamp[abspos=3.7](Sat)(SatCpl)
\polcontrol[abspos=4.5, fiber=o](Sat)(SatCpl)
\drawfiber{6-9}
\wdmcoupler[fiber=i](ScanCpl)(SatCpl)(Cpl)
\optisolator[position=0.1](Cpl)(Pump|Cpl)
\optfiber[position=0.6](Cpl)(Pump|Cpl)
\addtopsstyle{OptCircArrow}{arrows=<-}
\optcirculator[optcircangle=0 -90](Cpl)(PD)(Pump)
\wdmsplitter[position=0.8, coupleralign=t](Cpl)(PD)
\optbox[position=end, compname=Pd, extnode=t](Cpl)(PD){PD}
\optbox[position=end, compshift=-1.5](Cpl)(PD){OSA}
\optamp[fiber](Pump)(\oencodeIfc{2}{13})
\optbox[position=start, optboxsize=0.6 1.2](Pump)(Pump|Cpl){Pump}
\drawfiber{10-15}\drawfiber{14}{16}
\optbox[compshift=0.7, optboxsize=2.5 0.8](\oencodeExt{M2})(\oencodeExt{Pd}){\oencodeExt{M2}}{Lock-In Amp}
\addtopsstyle{Wire}{arm=0.1}
\drawwire[wireangleA=90, linearc=0.3](\oencodeExt{M2}){}
\drawwire[wireangleB=90, linearc=0.3](\oencodeExt{Pd}){}
\elecsynthesizer([offset=0.2]\oencodeExt{M1})(\oencodeExt{M1})
\psset{optexp}{synthshape=rectangle}
\elecsynthesizer[synthtype=sawtooth](\oencodeExt{S}|Pump)(\oencodeExt{S})
\elecsynthesizer[synthtype=triangle](\oencodeExt{M3}|Pump)(\oencodeExt{M3})
\end{pspicture}

```

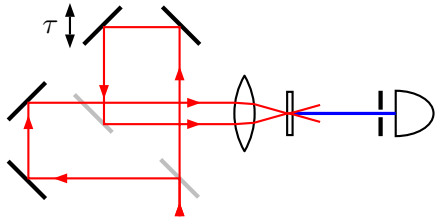
Bsp. 11.24: Angepasst von [semibyte.de](http://semibyte.de)

```

\begin{pspicture}(0.2,0.5)(9,7.4)
  \pnode(1,2){DL}\pnode([offset=0.5]DL){OAP}\pnode([Xnodesep=2]OAP){ST}
  \pnode([offset=-1.4]ST){SP4}\pnode([Xnodesep=4]SP4){SP3}
  \pnode([offset=3]SP3){JL}\pnode([Xnodesep=1]SP3|ST){SP1}
  \pnode([Xnodesep=1]ST|JL){SP2}\pnode([Xnodesep=1.5,offset=3]SP2){Det}
  \psset{optexp}{optboxsize=1.5 0.9, labeloffset=0.6}
  \begin{optexp}
    \optbox[position=start, innerlabel](DL)(OAP){\parbox{1.5\psunit}{\centering }
      Dioden-\\laser}}
    \psset{optexp}{mirrortype=extended}
    \mirror[mirrorradius=2](DL)(OAP)(ST){OAP}
    \psset{optexp}{variable}
    \beamsplitter[bssize=0.4, labelangle=-45](SP4)(ST)(SP1){ST}
    \optplate[abspos=1, plateheight=0.7](ST)(SP1){CP}
    \mirror(ST)(SP1)(SP2){SP1}\mirror(SP1)(SP2)(Det){SP2}
    \optdipole[label=-0.7 . b relative, optdipolesize=2, position=0.6](SP2)(Det){%
      \psframe[dimen=outer](-1,-0.6)(1,0.6)
      \psset{fillstyle=solid, fillcolor=blue!40, linestyle=none}
      \psframe(-1,-0.5)(-0.9,0.5)\psframe(0.9,-0.5)(1,0.5)}{Plasmareaktor}
    \optdetector[label=. 30 l](SP2)(Det){Detektor}
    \optbox[position=end, innerlabel](SP3)(JL){\parbox{1.5\psunit}{\centering }
      Justage-\\laser}}
    \mirror(SP4)(SP3)(JL){SP3}\mirror(ST)(SP4)(SP3){SP4}
    \psset{linewidth=2\pslinewidth}
    \drawbeam[linecolor=blue, beampos=0.03]{1-2}
    \drawbeam[beamalign=abs, loadbeampoints, linecolor=blue]{2-8}
    \drawbeam[linecolor=red, beampos=-0.03]{9-11}{3-8}
  \end{optexp}
  \pnode([Xnodesep=1.2,offset=-0.6]\oenodeIn{8}){NaCl}
  \psset{linestyle=dashed, arrows=->, arrowscale=1.5}
  \ncline[nodesepA=0.1, nodesepB=0.2]{NaCl}{\oenodeIn{7}}
  \ncline[nodesepA=0.1, nodesepB=0.3]{NaCl}{\oenodeOut{7}}
  \rput[l](NaCl){NaCl-Fenster}
\end{pspicture}

```

Bsp. 11.25: Angepasst von DOI:10.1364/AOP.1.000308, Abb. 8

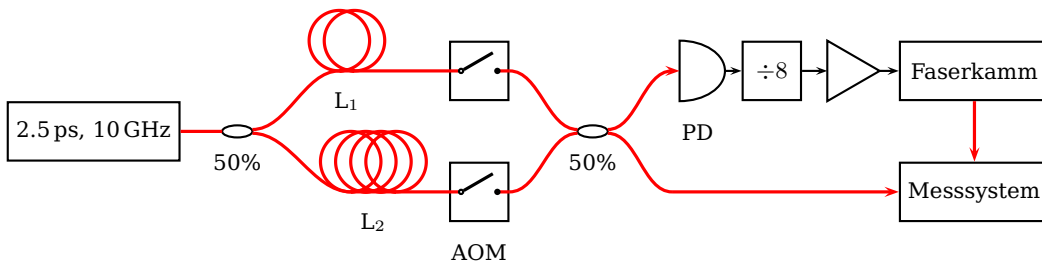


```

\begin{pspicture}(5.7,2.8)
\psset{mirrorwidth=0.7, bsstyle=plate, bssize=0.5, arrowinset=0, arrowscale=1.2}
\newpsobject{splitterplate}{beamsplitter}{linewidth=2\pslinewidth, }
\psnode(2.3,0){In}([offset=0.5]In){STM1}([Xnodesep=-2]STM1){M1}
\psnode([offset=1]M1){M2}([offset=2]STM1){M3}([Xnodesep=-1]M3){M4}(M4|M2){STM2}
\splitterplate(In)(STM1)(M1)
\mirror(STM1)(M1)(M2)
\mirror(M1)(M2)(M3|M2)
\mirror(STM1)(M3)(M4)
\mirror[label=0.6 180 . absolute](M3)(M4)(STM2)
\rput[r](\oencodeLabel){}\tau$
\psline[arrows=<->](0.15, 0.4)(0.15, -0.2)
\splitterplate[compoffset=0.2, compname=STM2](M4)(STM2)([Xnodesep=1]STM2)
\psnode(\oencodeCenter{STM2}){STM2C}
\psnode([Xnodesep=4]STM2C){Out}
\lens[n=2](STM2C)(Out)
\optbox[position=0.65, optboxsize=0.1 0.6](STM2C)(Out)
\optplate[linestyle=none, position=0.75](STM2C)(Out)
\pinhole[position=0.95, outerheight=0.6](STM2C)(Out)
\optdetector[detsize=0.5 0.6](STM2C)(Out)
\drawbeam[linecolor=blue]{8-}
\newpsstyle{Beam}{ArrowInside=->, linecolor=red, ArrowInsideMinLength=1, arrows=>-}
\drawbeam[ArrowInsidePos=0.835](In){1-3}{7-9}
\drawbeam[ArrowInsidePos=0.74](In){1}{4-9}
\end{pspicture}

```

Bsp. 11.26



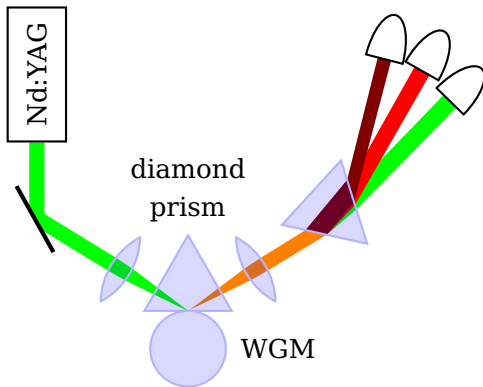
```

\begin{pspicture}(13.8,3.8)
  \node(2.3,2){Signal}
  \node([Xnodesep=1.5,offset=0.8]Signal){UpIn}
  \node([Xnodesep=1.5, offset=-0.8]Signal){LowIn}
  \node([Xnodesep=3.5]UpIn){UpOut}
  \node(UpOut|LowIn){LowOut}
  \node([Xnodesep=1.5]UpOut){UpIn2}
  \node([Xnodesep=3]UpIn2){UpOut2}
  \node([Xnodesep=1.5]LowOut){LowIn2}
  \node(UpOut2|LowIn2){LowOut2}
  \psset{labelstyle=\footnotesize}
  \psset{fiber=none}\addtopsstyle{Fiber}{linecolor=red}
  \psset{wire=none}\addtopsstyle{Wire}{arm=0.1, arrows=->}
  \optbox[position=start, label=0, optboxsize=2.3]{
    0.8](Signal)([Xnodesep=0.1]Signal){2.5\,ps, 10\,GHz}
  }
  \wdmsplitter[label=0.4](Signal)(UpIn)(LowIn){50\%}
  \optfiber[fiberloops=4, label=0.4, position=0.3](LowIn)(LowOut){L$_2$}
  \optswitch[position=0.7](LowIn)(LowOut){AOM}
  \optfiber[fiberloops=2, label=0.4, position=0.2](UpIn)(UpOut){L$_1$}
  \optswitch[position=0.7](UpIn)(UpOut)
  \optcoupler[label=0.4, position=0.3](UpOut)(LowOut)(UpIn2)(LowIn2){50\%}
  \optbox[optboxsize=2 0.8, position=end, label=0,]{
    extnode=t)(LowIn2)(LowOut2){Messsystem}
  }
  \drawfiber{1-4}{7}(LowIn2)\drawfiber[arrows=->](LowIn2){8}
  \optdetector[abspos=0.4, detsize=0.6 0.8](UpIn2)(UpOut2){PD}
  \optbox[optboxsize=0.8 0.8, label=0, position=0.44](UpIn2)(UpOut2){$\div 8$}
  \optamp[position=0.8](UpIn2)(UpOut2)
  \optbox[position=end, extnode=b, label=0, optboxsize=2]{
    0.8](UpIn2)(UpOut2){Faserkamm}
  }
  \drawfiber{2}{5-7}(UpIn2)\drawfiber[arrows=->](UpIn2){9}
  \drawwire{9-12}
  \drawfiber[arrows=->](\oencodeExt{12})(\oencodeExt{8})
\end{pspicture}

```

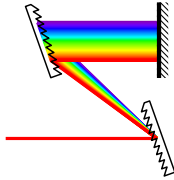


Bsp. 11.27: Angepasst von DOI:10.1103/PhysRevLett.105.263904



```
\begin{pspicture}(0.6,2.8)(6.8,7.8)
\pnodes(1,6){Laser}(1,5){M}(3,4){PR}(5,5){PT}(6,7){D1}
\newsstyle{Glass}{fillcolor=blue, fillstyle=solid, opacity=0.15,
  \linecolor=blue!30!white}
\begin{optexp}
\optbox[position=start, innerlabel, optboxwidth=1.8](Laser)(M){Nd:YAG}
\optprism[style=Glass, compname=P, n=1, prismtype=reflective, prismangle=60,
  \labeloffset=1.2](M)(PR)(PT){\begin{tabular}{@{}c@{}}diamond\\
  prism\end{tabular}}
\mirror(Laser)(M){\oenableIfc{2}{P}}%
\lens[abspos=1.05, style=Glass]{\oenableIfc{2}{P}}(M)%
\optprism[compname=PDet, style=Glass]{\oenableIfc{2}{P}}(PT)(D1)%
\lens[abspos=1.05, style=Glass]{\oenableIfc{2}{P}}{\oenableOut{PDet}}%
\newsstyle{Beam}{linestyle=none, fillstyle=solid}%
\drawwidebeam[fillcolor=green, beamwidth=0.2, beaminsidelast,
  \beampathcount=5]{1}{3}{4}{2}%
\drawwidebeam[fillcolor=orange, beaminsidefirst, startinsidecount=1, loadbeam,
  \beaminsidelast]{2}{6}%
\psset{savebeam=false, loadbeam, beamnodealign=vector}%
\drawwidebeam[fillcolor=orange]{6}{5}%
\drawwidebeam[fillcolor=green, n=1.2, beampathskip=1, savebeam=2]{6}{5}(D1)
\drawwidebeam[fillcolor=red, beampathskip=1, n=1.4, savebeam=3]{6}{5}(D1)
\drawwidebeam[fillcolor=red!50!black, beampathskip=1, n=1.57, savebeam=4]{6}{5}(D1)
\multido{\i=2+1}{3}{%
  \backlayer{%
    \pnode(!\oeBeamCenter{\i}){Det\i}
    \nodexn{(Det\i)-(!\oeBeamVecMedian{\i})}{Det\i'}}
  \optdetector[detsize=0.6 0.5](Det\i')(Det\i)}
\end{optexp}
\pscircle[style=Glass]([offset=-0.5]\oenableIfc{2}{P}){0.5}
\rput[l]([offset=-0.5,Xnodesep=0.7]\oenableIfc{2}{P}){WGM}
\end{pspicture}
```

Bsp. 11.28

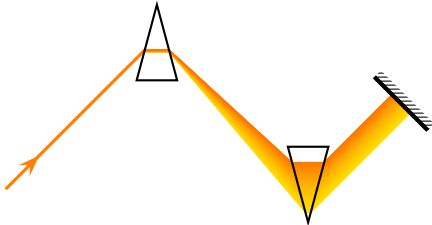


```

\begin{pspicture}(2.2,2.2)
\definecolor[ps]{bl}{rgb}{tx@addDict begin Red Green Blue end}%
\pnode(0,0.5){In}\pnode(2,0.5){G1}
\pnode([Xnodesep=-1.5, offset=1.3]G1){G2}\pnode(G1|G2){MG}
\begin{optexp}
\optgrating[angle=40, gratingalign=c](In)(G1)(G2)
\optgrating[reverse, angle=40, gratingalign=c](G1)(G2)(MG)
\mirror[mirrorwidth=1, mirrortype=extended](G2)(MG)(G2)
\drawbeam[linecolor=red](In){1}
\multido{\i=0+1}{40}{%
\pstVerb{\i\space 650 400 sub 39 div mul 400 add
tx@addDict begin wavelengthToRGB end }%
\drawbeam[linecolor=bl, beamangle=\i\space 16 sub 0.2 mul]{1-2}%
\drawbeam[linecolor=bl, loadbeampoints]{2-3}
}%
\end{optexp}
\end{pspicture}

```

Bsp. 11.29

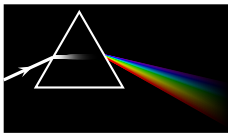


```

\begin{pspicture}(5.6,3)
\pnodes(0,0.5){S1}(2,2.5){S2}(3,1.5){S3}(4,0.5){S4}(4.5,1){S5}
\psset{prismangle=30, mirrortype=extended}
\definecolor[ps]{bl}{rgb}{tx@addDict begin Red Green Blue end}%
\begin{optexp}
\optprism(S1)(S2)(S3)\optprism(S3)(S4)(S5)
\mirror[compshift=-0.07](\oenableIn{2})([offset=1,Xnodesep=1]\oenableIn{2})(\oenableIn{2})
\drawbeam[linecolor=orange, ArrowInside=->, ArrowInsidePos=0.2, ↘
arrowscale=1.5](S1){1}
\addtopsstyle{Beam}{linecolor=bl, linewidth=0.35\pslinewidth, beampathskip=1}
\multido{\r=0+0.5}{120}{%
\pstVerb{\r\space 620 590 sub 59 div mul 590 add
tx@addDict begin wavelengthToRGB end }%
\drawbeam[n=3.42 -0.002 \r\space mul add, beampos=\r\space 60 sub 0.0002 ↘
mul](S1){1-3}
}%
\end{optexp}
\end{pspicture}

```

Bsp. 11.30

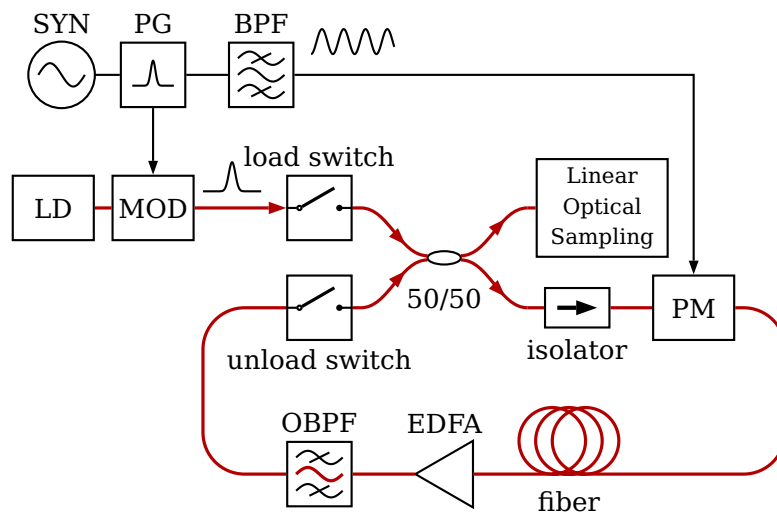


```

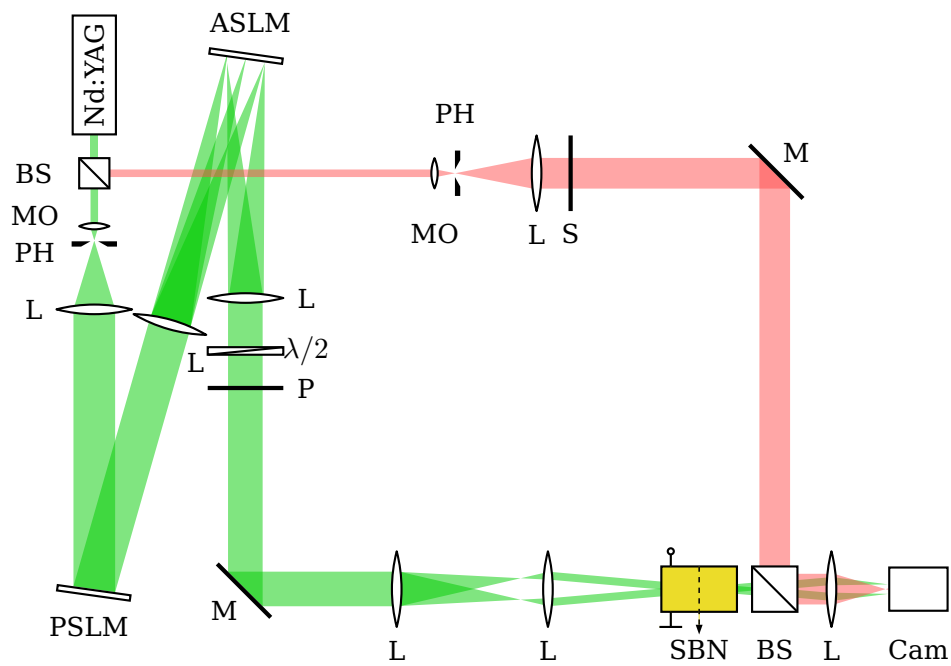
\begin{pspicture}(3,1.7)
\pnodes(-1,0){A}(1,1){B}(3,0){C}
\psframe*(0,0)(3,1.7)
\begin{optexp}
\optplane(0,0.7)
\optprism[prismalign=center, linecolor=white](A)(B)(C)
\optplane(C)
\addtopsstyle{Beam}{linecolor=white}
\drawbeam[ArrowInside=->]{1-2}
\definecolor[ps]{bl}{rgb}{%
tx@addDict begin Red Green Blue end}%
\addtopsstyle{Beam}{linestyle=fade, beaminsidelast, fadeto=black}
\multido{\i=0+1}{60}{%
\drawbeam[n=1.5 \i\space 0.003 mul add, linecolor=white, beampathskip=1, 2
fadefuncname=gauss]{1-2}
\pstVerb{\i\space 650 400 sub 59 div mul 400 add
tx@addDict begin wavelengthToRGB end }%
\drawbeam[n=1.5 \i\space 0.003 mul add, linecolor=bl, beampathskip=2, 2
linewidth=0.3\pslinewidth]{-}%
}%
\end{optexp}
\end{pspicture}

```

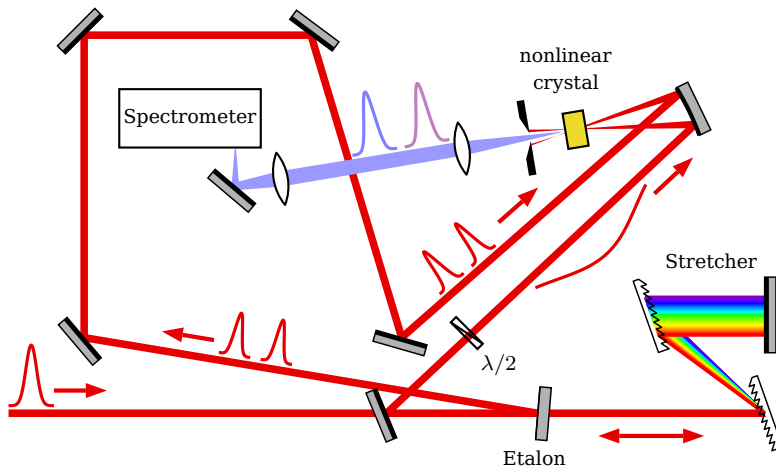
Bsp. 11.31



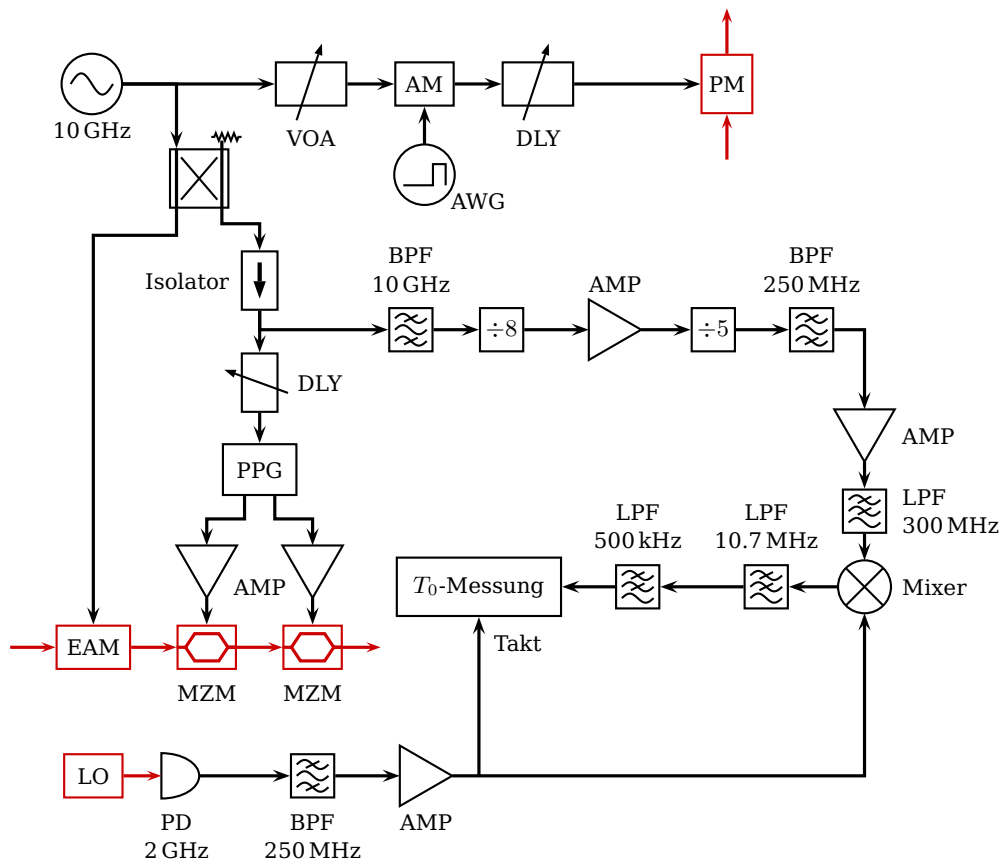
Bsp. 11.32: Angepasst von DOI:10.1364/OL.37.000797



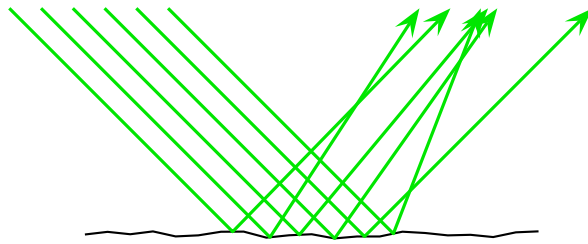
Bsp. 11.33: Angepasst von DOI:10.1364/AOP.1.000308, Abb. 37



Bsp. 11.34



Bsp. 11.35



```

\begin{pspicture}(10,3)
\makeatletter
\newOptexpTripole{roughsurface}
\def\roughsurface@nodes{%
  \pssavepath[linestyle=none, arrows=-,ArrowInside=-]{\oename@Path{A}}{%
    \code{0 srand}
    \moveto(!-3 Rand 0.5 sub 0.1 mul)
    \multido{\r=-2.7+0.3}{20}{%
      \lineto(!\r\space Rand 0.5 sub 0.1 mul)}%
    }%
  \newOptexpComp{%
    {0 0} tx@IntersectDict /\PIT@name{\oename@Path{A}} get 0 0 refl {PathIfc}
    1 }%
  }%
\def\roughsurface@comp{%
  \pstracecurve{\oename@Path{A}}
}%
\makeatother
\optplane[angle=90](1,3)
\roughsurface(0,3)(5,0)(10,3)
\addtopsstyle{Beam}{beamalign=abs, beamangle=-45, arrows=->, arrowscale=2}
\multido{\r=0+0.3}{6}{%
  \drawbeam[beampos=\r]{1}{2}{1}}
\end{pspicture}

```

## 12. Zusatzinformationen

### 12.1. Interne Struktur der Komponenten

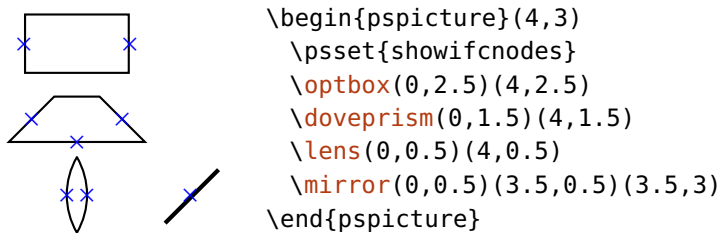
`showifcnodes=true, false`

default: false

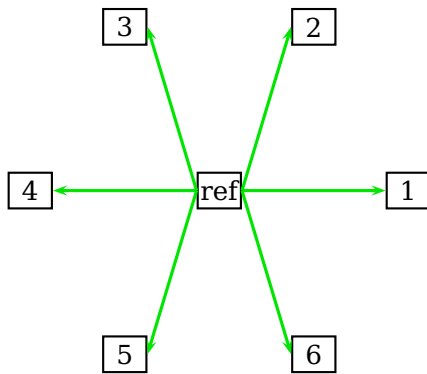
`IfcNodeStyle<psstyle>`

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.



`\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}
```

`showinterfaces=true, false`

default: false

Zeigt alle Grenzflächen einer Komponente an.



`IfcStyle`(*psstyle*)

default: `linestyle=dashed, dash=1pt 1pt,`  
`linecolor=red!70!black`

Bestimmt das Aussehen der mit `showinterfaces` angezeigten Grenzflächen.



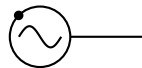
## 12.2. Übersicht der Spezialknoten

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.

### 12.2.1. Externe und Rotationsreferenzknoten

Hier finden Sie eine Übersicht über alle möglichen externen Knoten (Kap. 8.5) und Rotationsknoten (Kap. 8.7) 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  $+1 \cdot \Delta x$ . Das kann gerade im Fall einer runden Außenform sehr hilfreich sein, wie das folgende Beispiel zeigt.

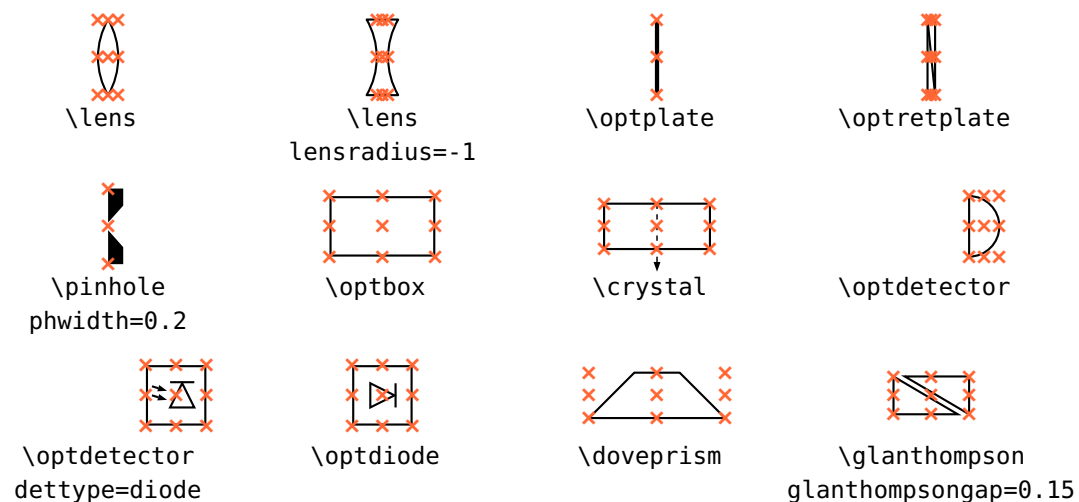


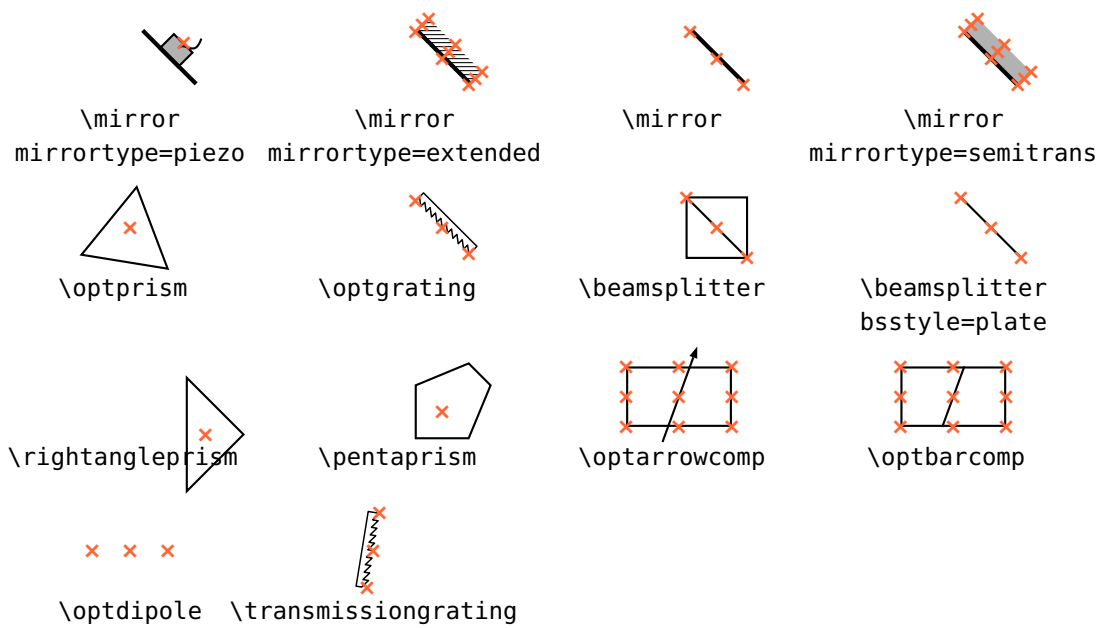
```

\begin{pspicture}(2,1)
  \elecsynthesizer[extnode={-0.707,0.707}](1,0.5)(2,0.5)
  \psdot(\oencodeExt{})
\end{pspicture}

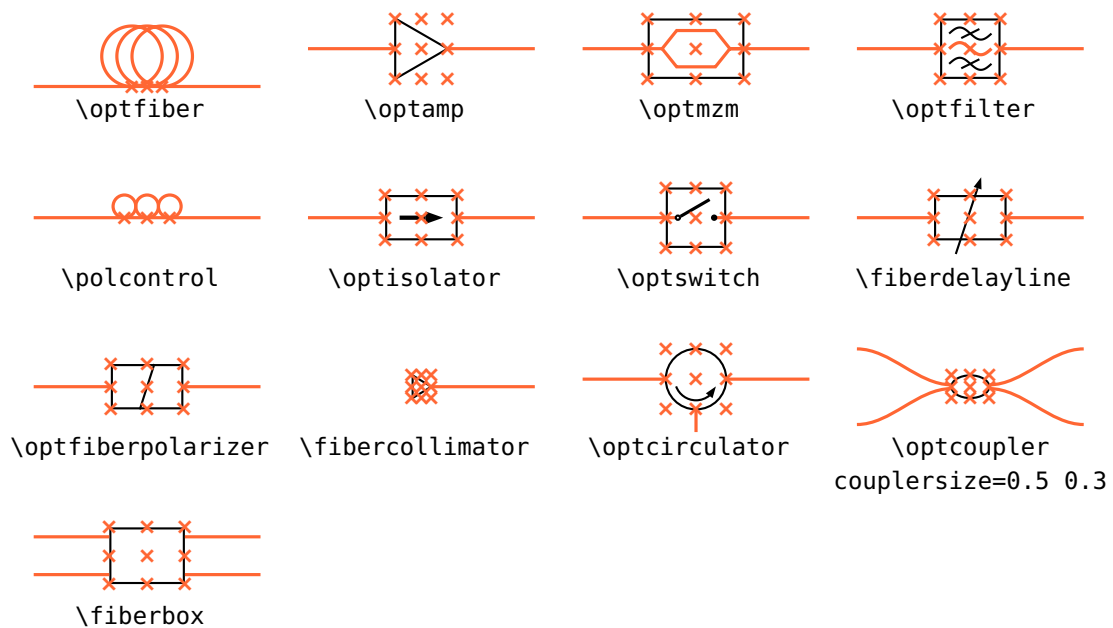
```

### Freistrahlkomponenten

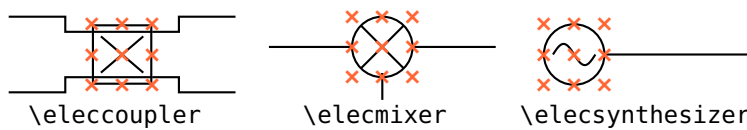




### Faserkomponenten



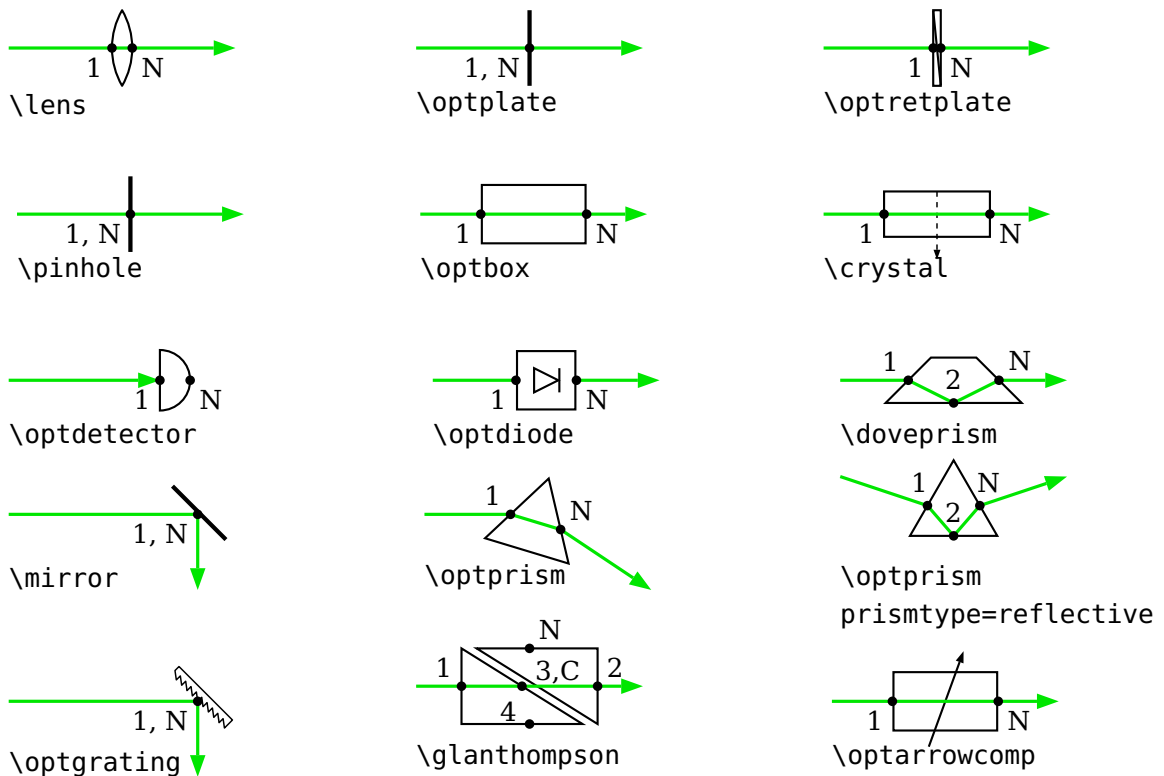
### Elektrische Komponenten

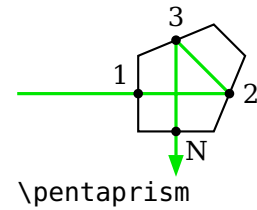
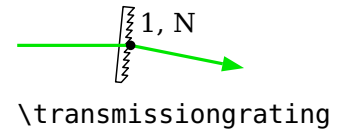
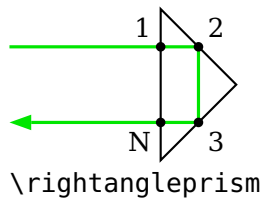
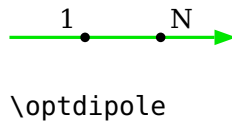
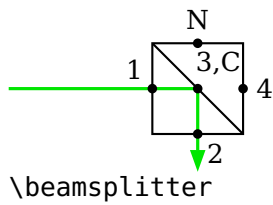
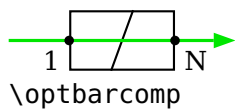


### 12.2.2. Grenzflächenknoten

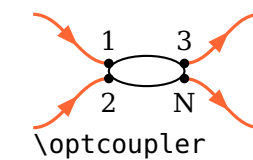
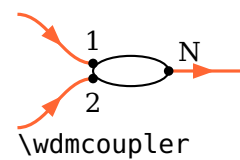
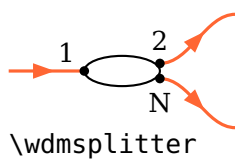
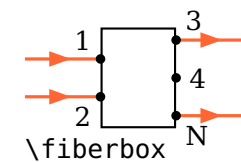
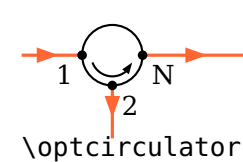
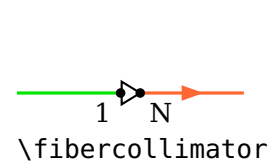
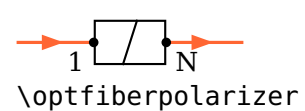
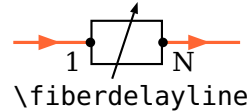
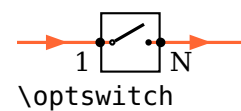
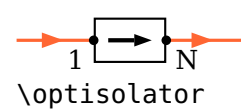
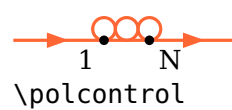
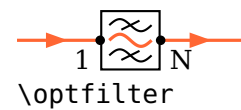
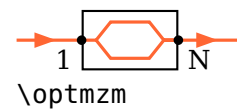
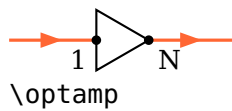
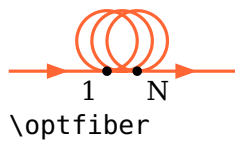
Hier sind alle Komponenten und deren Grenzflächenknoten aufgelistet. Ist ein Knoten mit „1, N“ beschriftet, dann fallen beide Knoten zusammen.

### Freistrahalkomponenten

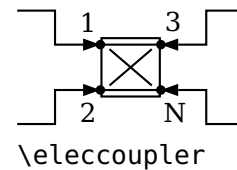
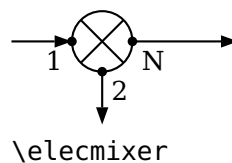
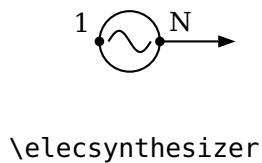




### Faserkomponenten



### Elektrische Komponenten



## 12.3. 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.

### 12.3.1. Version 4.3a

- `prismtype` Die Option `prismtype` war immer noch „zusammengeschustert“ und die automatische Positionierung war recht willkürlich. Dies sollte nun endlich korrigiert sein, unter Option `prismalign` ist ein entsprechendes Beispiel zu sehen.
- `\optprism` Die Positionierung der Beschriftung war nicht konsistent mit anderen Komponenten, der Bezugspunkt war nicht der Mittelpunkt und auch nicht fest. Das ist jetzt behoben, könnte aber leichte Korrekturen an bestehenden Skizzen erfordern.

### 12.3.2. Version 4.3

- `\rightangleprism` Die Ausrichtung des `\rightangleprism` und die Positionierung der Beschriftung war falsch. Das kommt aber nur zum tragen, wenn Eingangs- und Ausgangsvektor sehr unterschiedliche Längen haben.

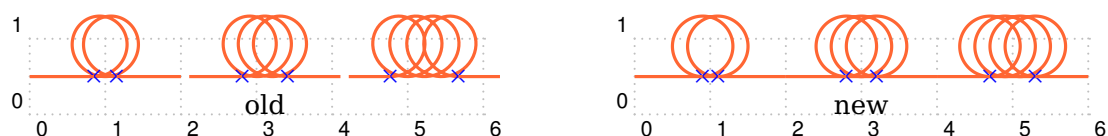
### 12.3.3. Version 4.2

**prismtype** Die in Version 4.1 eingeführte Option **prismtype** benötigte immer eine zusätzliche Drehung der Komponente um 180°. Das entfällt nun, da die Komponenten-ausrichtung korrigiert wurde, erfordert aber gegebenenfalls eine Korrektur von Skizzen, die mit Version 4.1 erstellt wurden.

### 12.3.4. Version 4.0

Die Option `conn` und das alte, fehlerhafte Verfahren Strahlverbindungen zu zeichnen, wurde nun entfernt. Zur Migration älterer Dokumente siehe Kap. 12.3.6. Damit wurde auch die Option `connjoin` überflüssig und wurde entfernt.

Das Aussehen von `\optfiber` hat sich etwas geändert, da in der vorherigen Implementierung der Abstand zwischen zwei Schleifen von der Anzahl der Schleifen abhing (siehe das Beispiel unten). Beachten Sie, dass die Voreinstellung für **fiberloopsep** von 0.3 auf 0.2 geändert wurde um ein ähnliches Aussehen bei Verwendung der Voreinstellungen zu erhalten. Dadurch sitzen die Eingangs- und Ausgangsknoten jedoch etwas näher zusammen (siehe unten).



Das Makro `\newOptexpDipoleNoLabel` wurde entfernt und kann ohne weitere Änderungen durch `\newOptexpDipole` ersetzt werden.

Die alte Syntax mit `\mycomp@iii` zur Definition eigener Komponenten wird nicht mehr unterstützt.

### 12.3.5. Version 3.3

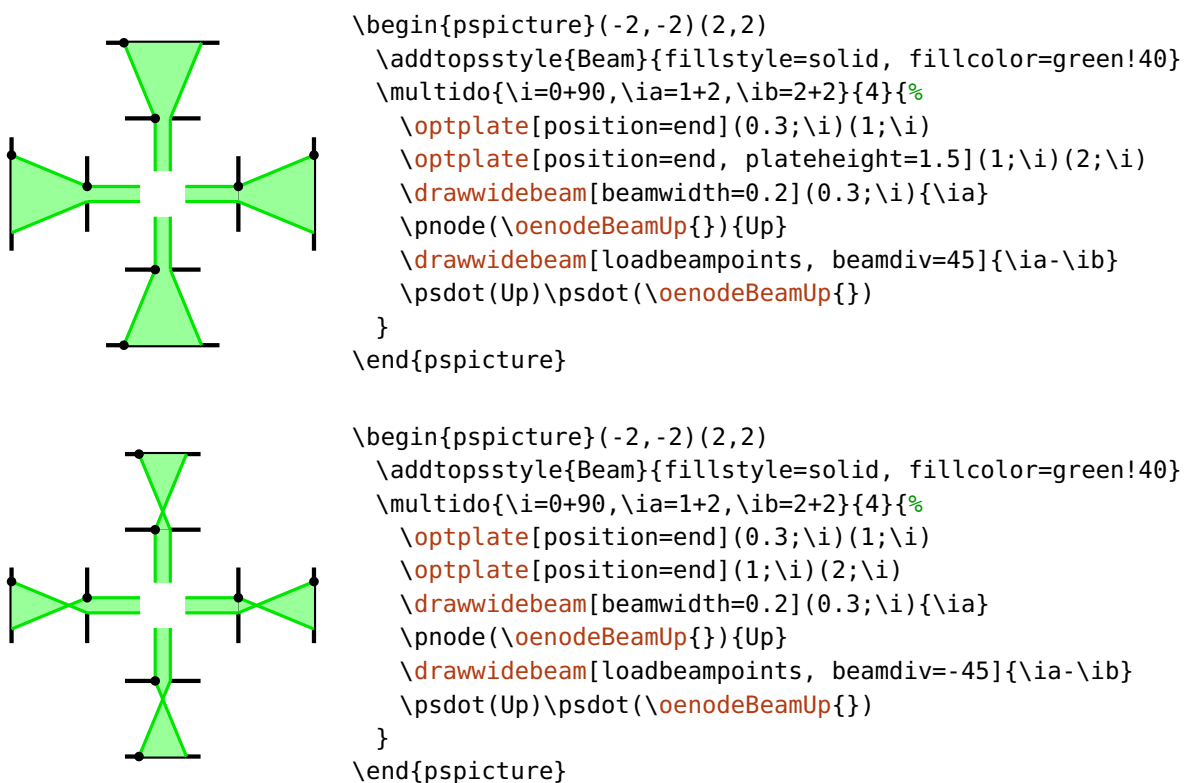
**\opttripole** Der Ausgangswinkel für die Beschriftung des **\opttripole** wurde um 180° geändert um konsistent zu den anderen Dreipolen zu sein.

## Strahlknoten

Vor Version 3.3 sind the Strahlknoten `\oenodeBeamUp` und `\oenodeBeamLow` nicht korrekt ausgerichtet gewesen, „upper“ und „lower“ war immer relativ zur Strahlrichtung (analog zu `extnodealign=relative`).

Das konnte zu Problemen führen wenn `loadbeampoints` zusammen mit `beamdiv` verwendet wurde, da das Vorzeichen von `beamdiv` nicht eindeutig kontrahierenden bzw. expandierenden Strahlen zugeordnet war.

Seit Version 3.3 werden die Strahlknoten analog zu `extnodealign=absolute` definiert und die Zuordnung eines negativen `beamdiv` mit kontrahierenden Strahlen sowie positiven `beamdiv` mit expandierenden Strahlen ist immer gültig. In den folgenden beiden Beispielen ist immer der `\oenodeBeamUp` gekennzeichnet.



## 12.3.6. 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. 8

`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.

`\optbox` Eine `\optbox` hat immer einen Eingangs- und Ausgangsknoten an den entsprechenden Grenzflächen. In Version 2.1 fielen Eingangs- und Ausgangsknoten zusammen falls `endbox` auf `true` gesetzt war.

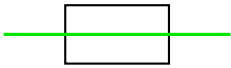
`\fibercollimator` Der `\fibercollimator` hat in den Voreinstellungen nur eine Faserverbindung, der Strahl muss manuell oder durch Angabe des `beam` Parameters gezeichnet werden.

## Strahlverbindungen

In Version 2.1 konnten Strahlverbindungen intern sein, d.h. die Verbindung wurde zusammen mit der Komponente 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. 9.9).

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

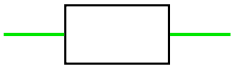




```

\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \optbox[beam](A)(B)
\end{pspicture}
conn=o-i, conn=i-o

```

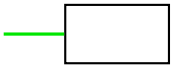


```

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

```


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:



```

\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \optbox(A)(B)
  \drawbeam(A){}
\end{pspicture}
conn=o-, conn=a-

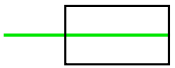
```



```

\begin{pspicture}(3,2)
  \pnode(0,1){A}\pnode(3,1){B}
  \optbox(A)(B)
  \drawbeam{ }(B)
\end{pspicture}
conn=-o, conn=-b


```



```

\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-, A-

```



```

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

```

# Dokumentationsindex

## A

abspos, 9, 10, 13–15, 17, 22, **22**, 39, 75, 81, 82, 84, 91–94, 96, 99–101, 106–108, 110, 115, 127, 133–137, 139, 141–143, 146–150, 152, 153

addtoBeam, 101, **101**

addtoFiber, 115, **115**, 116

addtoFiberIn, 122, 137, 138

addtoFiberIn1, 115, 122, 137, 138

addtoFiberIn2, 115, 122, 126, 137, 138

addtoFiberOut, 122, 137, 138

addtoFiberOut1, 122, 138

addtoFiberOut2, 122, 138

addtoOptComp, 25, 26, **26**

\addtopsstyle, 25, 121

addtoWire, 117, **117**

addtoWireIn, 122

addtoWireIn1, 122

addtoWireIn2, 122

addtoWireOut, 122

addtoWireOut1, 122

addtoWireOut2, 122

align, **61**

allowbeaminside, 36, 70, **97**, 98, 147

angle, 23, **23**, 38, 41, 47, 66, 74–76, 80, 83–85, 95, 96, 100, 101, 111–114, 133, 134, 139, 143–145, 148, 154, 158

aomalign, 48, **48**, 49, 160

aomcomp, 48, **49**

aomgratingcount, **48**, 160

aomheight, 48, **48**

aomreflalign, 49, **49**, 50, 51

aomsize, **48**, 50, 51

aomwidth, 48, **48**

arrowcompangle, **31**

arrowcompheight, [31](#), [31](#)  
arrowcompshape, [32](#), [32](#)  
arrowcompsize, [31](#), [137](#)  
ArrowCompStyle, [32](#), [32](#), [58](#)  
arrowcompwidth, [31](#), [31](#)  
ArrowInside, [102](#)  
ArrowInsideMaxLength, [102](#), [102](#)  
ArrowInsideMinLength, [101](#), [102](#), [144](#), [151](#)

## B

\backlayer, [85](#), [123](#), [153](#)  
barcompangle, [32](#), [33](#)  
barcompheight, [32](#), [32](#)  
barcompshape, [33](#), [33](#)  
barcompsize, [32](#)  
BarCompStyle, [33](#), [33](#)  
barcompwidth, [32](#), [32](#)  
Beam, [15](#), [18](#), [38](#), [43](#), [47–51](#), [70](#), [81](#), [82](#), [85](#), [91–93](#), [95](#), [99–101](#), [101](#), [102](#), [104](#),  
[105](#), [108](#), [109](#), [127](#), [133–136](#), [139–141](#), [143–147](#), [151](#), [153–155](#), [158](#),  
[160](#), [167](#)  
beam, [21](#), [25–31](#), [34–42](#), [44–47](#), [51–53](#), [70](#), [71](#), [74](#), [101](#), [124](#), [126](#), [168](#), [169](#)  
beamalign, [33](#), [43](#), [83](#), [84](#), [92](#), [94](#), [94](#), [95](#), [132](#), [133](#), [146](#), [150](#), [158](#)  
beamangle, [11](#), [33](#), [34](#), [47](#), [89](#), [90](#), [93](#), [94](#), [94](#), [106](#), [109](#), [143](#), [144](#), [154](#), [158](#)  
beamdiffractionorder, [50](#), [50](#), [51](#)  
beamdiv, [33](#), [34](#), [95](#), [104](#), [104](#), [106](#), [108](#), [127](#), [133](#), [135](#), [143](#), [144](#), [167](#)  
beaminside, [11](#), [96](#), [97](#), [98](#), [100](#), [125](#), [129–131](#), [143](#), [169](#)  
beaminsidefirst, [82](#), [94](#), [97](#), [97](#), [98](#), [110](#), [136](#), [139](#), [141](#), [147](#), [169](#)  
beaminsidelast, [81](#), [83](#), [84](#), [97](#), [97](#), [98](#), [107](#), [136](#), [153](#), [169](#)  
beammode, [50](#), [99](#)  
beamnodealign, [85](#), [99](#), [100](#), [153](#)  
beampathcount, [96](#), [96](#), [153](#)  
beampathskip, [85](#), [95](#), [96](#), [153–155](#)  
beamos, [11](#), [38](#), [41](#), [43](#), [44](#), [83](#), [89–93](#), [93](#), [94](#), [97](#), [99–101](#), [104](#), [106](#), [107](#), [110](#),  
[146](#), [150](#), [154](#), [158](#)  
\beamsplitter, [18](#), [20](#), [44](#), [44](#), [45](#), [48](#), [87](#), [99](#), [109](#), [136](#), [139–141](#), [143](#), [150](#),  
[151](#), [159](#)  
beamwidth, [11](#), [15](#), [18](#), [33](#), [43](#), [46](#), [48–52](#), [70](#), [81](#), [82](#), [84](#), [85](#), [96](#), [99](#), [100](#), [102](#),  
[104](#), [104](#), [105](#), [108](#), [109](#), [125–127](#), [133](#), [135](#), [136](#), [139](#), [140](#), [143](#), [144](#),  
[153](#), [167](#)  
bssize, [44](#), [87](#), [141](#), [150](#), [151](#)

bsstyle, 18, **44**, 45, 151

## C

caxisinv, **35**

caxislength, **34**, 35

compname, 18, 74, **74**, 86–88, 92, 96, 99, 111, 115, 116, 134, 137, 139, 140, 143, 147–149, 151, 153, 160

compoffset, 24, **24**, 151

compshift, 17, **23**, 24, 75, 76, 85, 133, 137, 138, 143, 149, 154

conn, 166, 168

connjoin, 166

coupleralign, 61, **61**, 62, 76, 116, 122, 126, 138, 149

couplersep, 60, **60**, 61, 66, 76

couplersize, 60, **60**, 61, 66, 76

couplertype, 60, 61, **61**, 66, 76, 142, 145

\crystal, 34, **34**, 35, 36, 126, 141, 147

CrystalCaxis, 35, **35**

crystalheight, 34, **34**, 35

CrystalLamp, 35, **36**

crystalsize, **34**, 35, 141

crystalwidth, 34, **34**

## D

DetectorStyle, 72, **72**

detsize, **72**, 85, 145, 147, 151–153

detype, 13, 18, 72, **72**, 134, 137

diffractionorders, **49**, 50, 51

\doveprism, 36, **36**, 37, 89–91, 140, 159

doveprismsize, **36**, 37, 89, 90

\drawbeam, 7, 10, 11, 13, 20, 34, 37, 38, 40, 41, 43, 44, 47, 80, 81, 83, 84, 86–99, 101, 102, 106–111, 117, 125, 128–132, 134, 135, 139, 141, 145–148, 150, 151, 154, 155, 158–160, 168, 169

\drawfiber, 12, 17, 62, 110–116, 118, 137, 142, 143, 145, 148, 149, 152

\drawwidebeam, 11, 15, 18, 34, 43, 46, 48–52, 70, 80–82, 84, 85, 88, 90, 95, 96, 99, 100, 102, 104–109, 125–127, 133, 135, 136, 139, 140, 143, 144, 153, 159, 167

\drawwire, 116–118, 137, 148, 149, 152

## E

\eleccoupler, 65, **65**, 66, 120

eleccouplerinput, **65**, 66

eleccouplersep, **65**  
eleccouplersize, **65**, **65**  
eleccouplertype, **65**, **65**, **66**  
\elecmmixer, **67**, **68**, **113**, **120**  
elecmixersize, **68**  
\elecsynthesizer, **24**, **66**, **66**, **67**, **118–120**, **122**, **137**, **148**, **149**, **161**  
endbox, **23**, **168**  
environ, **6**  
ExtendedMirror, **25**, **40**, **42**, **42**, **78**, **136**, **144**  
extnode, **40**, **77**, **77**, **78–80**, **85**, **128**, **130**, **131**, **137**, **139**, **143**, **148**, **149**, **152**,  
**161**, **168**  
extnodealign, **77**, **78**, **139**, **167**  
extnodes, **78**, **79**

## F

fade, **102**, **103**  
fadefunc, **103**, **103**  
fadefuncname, **103**, **103**, **155**  
fadepoints, **102**, **103**, **103**  
fadeto, **102**, **103**, **103**, **155**  
FdlArrow, **58**  
fdlsize, **58**, **107**  
Fiber, **54**, **58**, **59**, **66**, **112**, **113**, **115**, **115**, **116**, **121**, **137**, **152**  
fiber, **15–17**, **25**, **32**, **33**, **62**, **64**, **70**, **107**, **113**, **115**, **116**, **118**, **118**, **119**, **122**,  
**133**, **137**, **142**, **143**, **145**, **148**, **149**, **152**  
fiber\*, **116**  
fiberalign, **112**, **113**, **117**  
fiberangleA, **113**, **113**, **114**, **117**  
fiberangleB, **113**, **113**, **114**, **117**  
\fiberbox, **63**, **63**, **64**  
fiberboxcount, **63**, **64**, **64**  
fiberboxheight, **64**, **64**  
fiberboxsepin, **63**, **64**, **64**  
fiberboxsepout, **63**, **64**  
fiberboxsize, **64**, **64**  
fiberboxwidth, **64**, **64**  
\fibercollimator, **69**, **70**, **70**, **71**, **168**  
fibercolsize, **71**  
\fiberdelayline, **31**, **58**, **58**, **107**, **122**  
FiberIn, **121**

FiberIn1, 121  
FiberIn2, 121  
fiberloopradius, 55, 55, 133  
fiberloops, 55, 55, 138, 142, 152  
fiberloopsep, 55, 55, 166  
FiberOut, 121  
FiberOut1, 58, 121  
FiberOut2, 58, 121  
fiberpolsize, 58  
fiberstyle, 12, 66, 115, 116, 117, 142  
fillstyle, 11  
filterangle, 70, 70  
filtersize, 69  
FilterStyle, 70, 70  
filtertype, 69, 69, 70  
forcebeaminside, 98, 98  
\frontlayer, 123

## G

\glanthompson, 37, 37, 38, 96, 148  
glanthompsongap, 37, 37, 38, 148  
glanthompsonheight, 37, 37  
glanthompsonsize, 37, 38, 148  
glanthompsonwidth, 37, 37  
gratingalign, 46, 46, 154  
gratingcount, 45, 45  
gratingdepth, 45, 45  
gratingheight, 45  
gratinglinewidth, 46  
gratingtype, 45, 46  
gratingwidth, 45, 144

## I

IfcNodeStyle, 159  
IfcStyle, 160, 160  
innercompalign, 24, 24, 70  
innerheight, 30, 93, 143  
innerlabel, 21, 23, 78, 116, 141, 150  
IsolatorArrow, 57, 57  
isolatorsizes, 56

**K****Komponenten**

- \beamsplitter, 18, 20, 44, **44**, 45, 48, 87, 99, 109, 136, 139–141, 143, 150, 151, 159
- \crystal, 34, **34**, 35, 36, 126, 141, 147
- \doveprism, 36, **36**, 37, 89–91, 140, 159
- \eleccoupler, 65, **65**, 66, 120
- \elecmmixer, **67**, 68, 113, 120
- \elecsynthesizer, 24, 66, **66**, 67, 118–120, 122, 137, 148, 149, 161
- \fiberbox, 63, **63**, 64
- \fibercollimator, 69, 70, **70**, 71, 168
- \fiberdelayline, 31, 58, **58**, 107, 122
- \glanthompson, 37, **37**, 38, 96, 148
- \lens, **27**
- \mirror, 10, 18, 24–26, 39, **39**, 40–42, 44, 74, 78, 82, 90, 93, 94, 102, 104, 105, 109, 133, 134, 136, 140, 141, 144–147, 150, 151, 153, 154, 159, 161
- \oapmirror, 43, **43**, 44
- \optamp, 55, **55**, 115, 116, 142, 148, 149, 152
- \optaom, 47, **47**, 48–51, 160
- \optarrowcomp, 31, **31**, 32, 137
- \optbarcomp, 32, **32**, 33
- \optbox, 10–17, 19–23, 31, **31**, 64, 74–81, 83–85, 87, 93–95, 97, 98, 104, 111–116, 118–120, 122, 123, 133–138, 140–143, 145–153, 159, 160, 168, 169
- \optcirculator, 58, **58**, 59, 68, 113, 118, 119, 149
- \optcoupler, 60, **60**, 61, 62, 66, 76, 115, 116, 119, 137, 142, 143, 152
- \optdetector, 13, 16–18, 69, 71, **71**, 72, 76, 85, 118, 134, 137, 139, 145, 147, 150–153, 168
- \optdiode, 36, **36**, 98
- \optdipole, 24, 124, **124**, 125, 140, 150
- \optfiber, **54**, 55, 115, 116, 133, 137, 138, 142, 149, 152, 166
- \optfiberpolarizer, 32, 58, **58**, 138
- \optfilter, 24, 54, 69, **69**, 70, 98, 142
- \optgrating, 26, 45, **45**, 46–48, 93, 108, 109, 134, 143, 144, 154
- \optgrid, 45
- \optisolator, 56, **56**, 57, 138, 149
- \optmzm, 16, 17, 54, 56, **56**, 118–120, 142, 148, 149
- \optplane, 34, 37, 38, 43, 51, 84, 89, 91–93, 96, 100, 101, 106, 110, 133, 139, 143, 148, 155, 158
- \optplate, 13, 29, **29**, 93–95, 104, 106, 107, 110, 127, 135, 143, 145, 147, 150, 151, 161, 167

- \optprism, 51, **51**, 52, 85, 89, 92, 96, 99, 102, 106, 153–155
- \optretplate, 13, 29, **29**, 97, 134–136, 140, 141
- \optsource, 33, **33**, 34
- \optswitch, 57, **57**, 142, 152
- \opttripole, 124, **124**, 126, 136, 146, 166
- \parabolicmirror, 42, **42**, 43
- \pentaprism, 53, **53**, 80, 97, 140
- \pinhole, 30, **30**, 93, 135, 136, 140, 141, 143, 144, 146, 151
- \polarization, 38, **38**, 39, 84, 96, 135, 141, 148
- \polcontrol, 56, **56**, 138, 142, 149
- \rightangleprism, 52, **52**, 53, 165
- \transmissiongrating, 47, **47**
- \wdmcoupler, 60, **60**, 62, 126, 138, 145, 149
- \wdmsplitter, 16, 17, 60, **60**, 62, 119, 122, 138, 143, 145, 149, 152

**L**

label, 15–17, 19–21, **21**, 35, 36, 55, 64, 78, 115, 124–126, 128–131, 133–138, 140–143, 145–152

labelalign, **19**, 20, 21, 129–131, 136

labelangle, 10, 20, **20**, 21, 24, 129, 133, 135, 136, 141, 142, 150

labeloffset, 10, 13–15, 17, 19, **19**, 20, 21, 40, 46, 61, 77, 87, 124, 134, 136–138, 141, 143, 146, 150, 153, 160

labelref, 10, 20, **20**, 21, 77

labelstyle, 16, 17, **19**, 141, 142, 152

lamp, **35**

lampscale, **35**

\lens, **27**

lens, 9–11, 14, 15, 18, 22, 24, 26–28, **28**, 34, 52, 81, 82, 84, 91, 92, 94, 96, 99–101, 106–108, 110, 127, 133, 135, 136, 139–141, 143, 146, 147, 151, 153, 159

lensheight, 11, 14, 15, 27, **27**, 141

lensradius, 11, 13–15, 28, **28**, 81, 136

lensradiusleft, **27**, 28

lensradiusright, **27**, 28

lenswidth, 28, **28**, 29

linestyle, 11

loadbeam, 82–84, 99, **108**, 109

loadbeampoints, **107**, 108, 127, 133, 135, 136, 140, 154, 167

**M**



## Makros

- \addtopsstyle, 25, 121
- \backlayer, 85, 123, 153
- \drawbeam, 7, 10, 11, 13, 20, 34, 37, 38, 40, 41, 43, 44, 47, 80, 81, 83, 84, 86–99, 101, 102, 106–111, 117, 125, 128–132, 134, 135, 139, 141, 145–148, 150, 151, 154, 155, 158–160, 168, 169
- \drawfiber, 12, 17, 62, 110–116, 118, 137, 142, 143, 145, 148, 149, 152
- \drawwidebeam, 11, 15, 18, 34, 43, 46, 48–52, 70, 80–82, 84, 85, 88, 90, 95, 96, 99, 100, 102, 104–109, 125–127, 133, 135, 136, 139, 140, 143, 144, 153, 159, 167
- \drawwire, 116–118, 137, 148, 149, 152
- \frontlayer, 123
- \nccurve, 12, 111, 116
- \ncput, 22
- \newOptexpComp, 129
- \newOptexpDipole, 127–131, 166
- \newOptexpDipoleNoLabel, 166
- \newOptexpElecDipole, 127
- \newOptexpFiberDipole, 127
- \newOptexpTripole, 127, 129, 132, 158
- \newpsobject, 25, 125, 126
- \newpsstyle, 8, 25
- \nodexn, 84
- \oeBeamCenter, 85, 153
- \oeBeamVec, 83, 84, 96, 148
- \oeBeamVec\*, 82
- \oeBeamVecLow, 84
- \oeBeamVecMedian, 85, 100, 153
- \oeBeamVecUp, 84
- \oenode, 50, 79
- \oenodeBeam, 81–84, 96, 148
- \oenodeBeam\*, 82
- \oenodeBeamLow, 84, 167
- \oenodeBeamUp, 84, 167
- \oenodeCenter, 74, 76, 151
- \oenodeExt, 78, 79, 131, 137, 139, 148, 149, 152, 161
- \oenodeIfc, 52, 62, 80, 96, 99, 149, 153
- \oenodeIn, 74, 80, 134, 150, 154
- \oenodeLabel, 147, 151
- \oenodeOut, 74, 75, 80, 134, 147, 150, 153

- \oenodeRefA, 22, 75, 76, 79, 101
- \oenodeRefB, 22, 75, 76, 101
- \oenodeRotref, 80
- \oenodeTrefA, 76
- \oenodeTrefB, 76
- \psbezier, 71

\mirror, 10, 18, 24–26, 39, **39**, 40–42, 44, 74, 78, 82, 90, 93, 94, 102, 104, 105, 109, 133, 134, 136, 140, 141, 144–147, 150, 151, 153, 154, 159, 161

mirrordepth, 18, 40, 41, **41**, 44, 78

mirrorlinewidth, **39**

mirrorradius, **39**, 40, 93, 144, 150

mirrortype, 18, 25, 40, **40**, 41, 42, 44, 78, 82, 93, 94, 102, 104, 109, 136, 141, 144, 145, 147, 150, 154

mirrorwidth, **39**, 82, 102, 105, 141, 144, 146, 147, 151, 154

multido, 6

## N

n, 10, 11, 37, 82–85, 89, 90, **90**, 91–93, 99, 101, 106, 107, 110, 127, 132, 134, 136, 139, 140, 143, 147, 151, 153–155

namingscheme, **73**, 168

\nccurve, 12, 111, 116

\ncput, 22

newBeam, **101**

newFiber, **115**, 133

newFiberIn, 122

newFiberIn1, 122

newFiberIn2, 122

newFiberOut, 122

newFiberOut1, 122

newFiberOut2, 122

newOptComp, 25, 26, **26**

\newOptexpComp, 129

\newOptexpDipole, 127–131, 166

\newOptexpDipoleNoLabel, 166

\newOptexpElecDipole, 127

\newOptexpFiberDipole, 127

\newOptexpTripole, 127, 129, 132, 158

\newpsobject, 25, 125, 126

\newpsstyle, 8, 25

newWire, **117**

newWireIn, 122  
newWireIn1, 122  
newWireIn2, 122  
newWireOut, 122  
newWireOut1, 122  
newWireOut2, 122  
\nodexn, 84  
npos, 22

## O

\oapmirror, 43, 43, 44  
oapmirroraperture, 43, 44  
\oeBeamCenter, 85, 153  
\oeBeamVec, 83, 84, 96, 148  
\oeBeamVec\*, 82  
\oeBeamVecLow, 84  
\oeBeamVecMedian, 85, 100, 153  
\oeBeamVecUp, 84  
\oenode, 50, 79  
\oenodeBeam, 81–84, 96, 148  
\oenodeBeam\*, 82  
\oenodeBeamLow, 84, 167  
\oenodeBeamUp, 84, 167  
\oenodeCenter, 74, 76, 151  
\oenodeExt, 78, 79, 131, 137, 139, 148, 149, 152, 161  
\oenodeIfc, 52, 62, 80, 96, 99, 149, 153  
\oenodeIn, 74, 80, 134, 150, 154  
\oenodeLabel, 147, 151  
\oenodeOut, 74, 75, 80, 134, 147, 150, 153  
\oenodeRefA, 22, 75, 76, 79, 101  
\oenodeRefB, 22, 75, 76, 101  
\oenodeRotref, 80  
\oenodeTrefA, 76  
\oenodeTrefB, 76  
\optamp, 55, 55, 115, 116, 142, 148, 149, 152  
optampsize, 55  
\optaom, 47, 47, 48–51, 160  
\optarrowcomp, 31, 31, 32, 137  
\optbarcomp, 32, 32, 33

`\optbox`, 10–17, 19–23, 31, **31**, 64, 74–81, 83–85, 87, 93–95, 97, 98, 104, 111–116, 118–120, 122, 123, 133–138, 140–143, 145–153, 159, 160, 168, 169

`optboxheight`, 31, **31**

`optboxsize`, 11, 12, 20–22, **31**, 64, 75, 78, 83, 84, 87, 116, 133, 134, 136, 137, 141, 143, 145, 147–152, 160

`optboxwidth`, 13, 16, 17, 20, 31, **31**, 97, 98, 133–135, 138, 140–142, 145, 146, 148, 153

`optcircangle`, 59, **59**, 149

`optcircangleA`, **59**

`optcircangleB`, **59**

`OptCircArrow`, 59, **59**, 149

`optcirsize`, **58**

`\optcirculator`, 58, **58**, 59, 68, 113, 118, 119, 149

`OptComp`, 25, **25**, 26, 70, 72, 144

`\optcoupler`, 60, **60**, 61, 62, 66, 76, 115, 116, 119, 137, 142, 143, 152

`\optdetector`, 13, 16–18, 69, 71, **71**, 72, 76, 85, 118, 134, 137, 139, 145, 147, 150–153, 168

`\optdiode`, 36, **36**, 98

`optdiodesize`, **36**

`\optdipole`, 24, 124, **124**, 125, 140, 150

`optdipolecomp`, 125, **125**

`optdipolesize`, **124**, 125, 140, 150

`optexp`, 11, 18, 46, 123, 169

`\optfiber`, **54**, 55, 115, 116, 133, 137, 138, 142, 149, 152, 166

`\optfiberpolarizer`, 32, 58, **58**, 138

`\optfilter`, 24, 54, 69, **69**, 70, 98, 142

`\optgrating`, 26, 45, **45**, 46–48, 93, 108, 109, 134, 143, 144, 154

`\optgrid`, 45

`optgridcount`, 46

`optgriddepth`, 46

`optgridheight`, 46

`optgridlinewidth`, 46

`optgridtype`, 46

`optgridwidth`, 46

`optional`, 26, **26**

`OptionalStyle`, 26, **26**

`\optisolator`, 56, **56**, 57, 138, 149

`\optmzm`, 16, 17, 54, 56, **56**, 118–120, 142, 148, 149

`optmzmsize`, **56**, 148

\optplane, 34, 37, 38, 43, 51, 84, 89, 91–93, 96, 100, 101, 106, 110, 133, 139, 143, 148, 155, 158  
\optplate, 13, 29, **29**, 93–95, 104, 106, 107, 110, 127, 135, 143, 145, 147, 150, 151, 161, 167  
\optprism, 51, **51**, 52, 85, 89, 92, 96, 99, 102, 106, 153–155  
\optretplate, 13, 29, **29**, 97, 134–136, 140, 141  
\optsource, 33, **33**, 34  
\optswitch, 57, **57**, 142, 152  
\opttripole, 124, **124**, 126, 136, 146, 166  
opttripolecomp, **125**, 126  
outerheight, **30**, 141, 144, 151

## P

### Pakete

- environ, 6
- multido, 6
- pst-eucl, 6
- pst-intersect, 6
- pst-node, 6, 22, 116
- pst-optexp, 6, 25, 123, 124
- pst-plot, 6
- pstricks, 6
- pstricks-add, 6, 102

\parabolicmirror, 42, **42**, 43

### Parameter

- abspos, 9, 10, 13–15, 17, 22, **22**, 39, 75, 81, 82, 84, 91–94, 96, 99–101, 106–108, 110, 115, 127, 133–137, 139, 141–143, 146–150, 152, 153
- addtoBeam, 101, **101**
- addtoFiber, 115, **115**, 116
- addtoFiberIn, 122, 137, 138
- addtoFiberIn1, 115, 122, 137, 138
- addtoFiberIn2, 115, 122, 126, 137, 138
- addtoFiberOut, 122, 137, 138
- addtoFiberOut1, 122, 138
- addtoFiberOut2, 122, 138
- addtoOptComp, 25, 26, **26**
- addtoWire, 117, **117**
- addtoWireIn, 122
- addtoWireIn1, 122
- addtoWireIn2, 122

- addtoWireOut, 122
- addtoWireOut1, 122
- addtoWireOut2, 122
- align, 61
- allowbeaminside, 36, 70, 97, 98, 147
- angle, 23, 23, 38, 41, 47, 66, 74–76, 80, 83–85, 95, 96, 100, 101, 111–114, 133, 134, 139, 143–145, 148, 154, 158
- aomalign, 48, 48, 49, 160
- aomcomp, 48, 49
- aomgratingcount, 48, 160
- aomheight, 48, 48
- aomreflalign, 49, 49, 50, 51
- aomsize, 48, 50, 51
- aomwidth, 48, 48
- arrowcompangle, 31
- arrowcompheight, 31, 31
- arrowcompshape, 32, 32
- arrowcompsize, 31, 137
- arrowcompwidth, 31, 31
- ArrowInside, 102
- ArrowInsideMaxLength, 102, 102
- ArrowInsideMinLength, 101, 102, 144, 151
- barcompangle, 32, 33
- barcompheight, 32, 32
- barcompshape, 33, 33
- barcompsize, 32
- barcompwidth, 32, 32
- beam, 21, 25–31, 34–42, 44–47, 51–53, 70, 71, 74, 101, 124, 126, 168, 169
- beamalign, 33, 43, 83, 84, 92, 94, 94, 95, 132, 133, 146, 150, 158
- beamangle, 11, 33, 34, 47, 89, 90, 93, 94, 94, 106, 109, 143, 144, 154, 158
- beamdiffractionorder, 50, 50, 51
- beamdiv, 33, 34, 95, 104, 104, 106, 108, 127, 133, 135, 143, 144, 167
- beaminside, 11, 96, 97, 98, 100, 125, 129–131, 143, 169
- beaminsidefirst, 82, 94, 97, 97, 98, 110, 136, 139, 141, 147, 169
- beaminsidelast, 81, 83, 84, 97, 97, 98, 107, 136, 153, 169
- beammode, 50, 99
- beamnodealign, 85, 99, 100, 153
- beampathcount, 96, 96, 153
- beampathskip, 85, 95, 96, 153–155

- beampos, 11, 38, 41, 43, 44, 83, 89–93, **93**, 94, 97, 99–101, 104, 106, 107, 110, 146, 150, 154, 158
- beamwidth, 11, 15, 18, 33, 43, 46, 48–52, 70, 81, 82, 84, 85, 96, 99, 100, 102, 104, **104**, 105, 108, 109, 125–127, 133, 135, 136, 139, 140, 143, 144, 153, 167
- bssize, **44**, 87, 141, 150, 151
- bsstyle, 18, **44**, 45, 151
- caxisinv, **35**
- caxislength, **34**, 35
- compname, 18, 74, **74**, 86–88, 92, 96, 99, 111, 115, 116, 134, 137, 139, 140, 143, 147–149, 151, 153, 160
- compoffset, 24, **24**, 151
- compshift, 17, **23**, 24, 75, 76, 85, 133, 137, 138, 143, 149, 154
- conn, 166, 168
- connjoin, 166
- coupleralign, 61, **61**, 62, 76, 116, 122, 126, 138, 149
- couplersep, 60, **60**, 61, 66, 76
- couplersize, 60, **60**, 61, 66, 76
- couplertype, 60, 61, **61**, 66, 76, 142, 145
- crystalheight, 34, **34**, 35
- crystalsize, **34**, 35, 141
- crystalwidth, 34, **34**
- detsize, **72**, 85, 145, 147, 151–153
- dettype, 13, 18, 72, **72**, 134, 137
- diffractionorders, **49**, 50, 51
- doveprismsize, **36**, 37, 89, 90
- eleccouplerinput, **65**, 66
- eleccouplersep, **65**
- eleccouplersize, 65, **65**
- eleccouplertype, 65, **65**, 66
- elecmixersize, **68**
- endbox, **23**, 168
- extnode, 40, 77, **77**, 78–80, 85, 128, 130, 131, 137, 139, 143, 148, 149, 152, 161, 168
- extnodealign, **77**, 78, 139, 167
- extnodes, **78**, 79
- fade, 103
- fadefunc, 103, **103**
- fadefuncname, 103, **103**, 155
- fadepoints, 102, 103, **103**

- fadeto, 102, 103, **103**, 155
- fdlsiz, **58**, 107
- fiber, 15–17, 25, 32, 33, 62, 64, 70, 107, 113, 115, 116, 118, **118**, 119, 122, 133, 137, 142, 143, 145, 148, 149, 152
- fiber\*, 116
- fiberalign, **112**, 113, 117
- fiberangleA, 113, **113**, 114, 117
- fiberangleB, 113, **113**, 114, 117
- fiberboxcount, 63, 64, **64**
- fiberboxheight, 64, **64**
- fiberboxsepin, 63, 64, **64**
- fiberboxsepout, 63, **64**
- fiberboxsize, 64, **64**
- fiberboxwidth, 64, **64**
- fibercolsize, **71**
- fiberloopradius, 55, **55**, 133
- fiberloops, 55, **55**, 138, 142, 152
- fiberloopsep, 55, **55**, 166
- fiberpolsize, **58**
- fiberstyle, 12, 66, **115**, 116, 117, 142
- fillstyle, 11
- filterangle, 70, **70**
- filtersize, **69**
- filtertype, 69, **69**, 70
- forcebeaminside, 98, **98**
- glanthompsongap, 37, **37**, 38, 148
- glanthompsonheight, 37, **37**
- glanthompsonsize, **37**, 38, 148
- glanthompsonwidth, 37, **37**
- gratingalign, 46, **46**, 154
- gratingcount, 45, **45**
- gratingdepth, 45, **45**
- gratingheight, **45**
- gratinglinewidth, **46**
- gratingtype, **45**, 46
- gratingwidth, **45**, 144
- innercompalign, 24, **24**, 70
- innerheight, **30**, 93, 143
- innerlabel, **21**, 23, 78, 116, 141, 150
- isolatorsiz, **56**



- label, 15–17, 19–21, **21**, 35, 36, 55, 64, 78, 115, 124–126, 128–131, 133–138, 140–143, 145–152
- labelalign, **19**, 20, 21, 129–131, 136
- labelangle, 10, 20, **20**, 21, 24, 129, 133, 135, 136, 141, 142, 150
- labeloffset, 10, 13–15, 17, 19, **19**, 20, 21, 40, 46, 61, 77, 87, 124, 134, 136–138, 141, 143, 146, 150, 153, 160
- labelref, 10, 20, **20**, 21, 77
- labelstyle, 16, 17, **19**, 141, 142, 152
- lamp, **35**
- lampscale, **35**
- lens, 9–11, 14, 15, 18, 22, 24, 26–28, **28**, 34, 52, 81, 82, 84, 91, 92, 94, 96, 99–101, 106–108, 110, 127, 133, 135, 136, 139–141, 143, 146, 147, 151, 153, 159
- lensheight, 11, 14, 15, 27, **27**, 141
- lensradius, 11, 13–15, 28, **28**, 81, 136
- lensradiusleft, **27**, 28
- lensradiusright, **27**, 28
- lenswidth, 28, **28**, 29
- linestyle, 11
- loadbeam, 82–84, 99, **108**, 109
- loadbeampoints, **107**, 108, 127, 133, 135, 136, 140, 154, 167
- mirrordepth, 18, 40, 41, **41**, 44, 78
- mirrorlinewidth, **39**
- mirrorradius, **39**, 40, 93, 144, 150
- mirrorrtype, 18, 25, 40, **40**, 41, 42, 44, 78, 82, 93, 94, 102, 104, 109, 136, 141, 144, 145, 147, 150, 154
- mirrorwidth, **39**, 82, 102, 105, 141, 144, 146, 147, 151, 154
- n, 10, 11, 37, 82–85, 89, 90, **90**, 91–93, 99, 101, 106, 107, 110, 127, 132, 134, 136, 139, 140, 143, 147, 151, 153–155
- namingscheme, **73**, 168
- newBeam, **101**
- newFiber, **115**, 133
- newFiberIn, 122
- newFiberIn1, 122
- newFiberIn2, 122
- newFiberOut, 122
- newFiberOut1, 122
- newFiberOut2, 122
- newOptComp, 25, 26, **26**
- newWire, **117**

- newWireIn, 122
- newWireIn1, 122
- newWireIn2, 122
- newWireOut, 122
- newWireOut1, 122
- newWireOut2, 122
- npos, 22
- oapmirroraperture, 43, 44
- optampsize, 55
- optboxheight, 31, 31
- optboxsize, 11, 12, 20–22, 31, 64, 75, 78, 83, 84, 87, 116, 133, 134, 136, 137, 141, 143, 145, 147–152, 160
- optboxwidth, 13, 16, 17, 20, 31, 31, 97, 98, 133–135, 138, 140–142, 145, 146, 148, 153
- optcircangle, 59, 59, 149
- optcircangleA, 59
- optcircangleB, 59
- optcircsize, 58
- optdiodesize, 36
- optdipolecomp, 125, 125
- optdipolesize, 124, 125, 140, 150
- optgridcount, 46
- optgriddepth, 46
- optgridheight, 46
- optgridlinewidth, 46
- optgridtype, 46
- optgridwidth, 46
- optional, 26, 26
- optmzmsize, 56, 148
- opttripolecomp, 125, 126
- outerheight, 30, 141, 144, 151
- parmirrorheight, 42, 43
- parmirrorwidth, 42, 43
- pentaprismsize, 53, 80
- phlinewidth, 30, 30
- phwidth, 30, 30, 93, 135, 136, 143, 144
- plateheight, 29, 29, 30, 93–95, 104, 106, 110, 127, 141, 143, 150, 167
- platelinewidth, 29, 135
- platesize, 29
- platewidth, 29, 30

- polcontrolsize, **56**
- polcontroltype, **56**, **56**
- pollinewidth, **38**
- polsize, **38**, **135**, **148**
- poltype, **38**, **39**, **96**, **135**, **148**
- position, **10–17**, **20–22**, **22**, **23**, **33**, **35**, **59**, **62**, **66**, **71**, **72**, **81**, **83**, **87**, **93–95**, **97**, **98**, **104**, **106**, **107**, **110**, **112**, **116**, **118–120**, **122**, **128–131**, **133–138**, **140–146**, **148–153**, **167**
- prismalign, **52**, **92**, **155**, **165**
- prismangle, **51**, **51**, **92**, **96**, **99**, **153**, **154**
- prismsize, **51**, **89**
- prismtype, **51**, **52**, **96**, **99**, **153**, **165**, **166**
- pswarning, **50**, **107**
- raprismalign, **53**
- raprismsize, **53**
- raytrace, **37**, **88**, **89**, **90**, **101**, **102**, **133**
- refractiveindex, **93**
- reverse, **46**, **47**, **93**
- rotateref, **23**, **23**, **80**, **85**, **95**, **128**, **130**, **134**
- savebeam, **83**, **85**, **108**, **109**, **110**, **153**
- savebeampoints, **80**, **81**, **107**, **108**, **133**, **136**, **144**
- showifcnodes, **63**, **95**, **125**, **159**, **159**
- showinterfaces, **160**, **160**
- showoptdots, **53**, **73**
- sourceheight, **34**, **34**
- sourcesize, **34**
- sourcewidth, **34**, **34**
- startinside, **94**, **98**, **109**, **110**, **133**
- startinsidecount, **95**, **98**, **99**, **153**
- startnode, **114**, **114**, **115**, **142**, **145**
- stopinside, **82**, **83**, **98**, **108**, **110**, **110**, **127**, **133**, **135**
- stopinsidecount, **43**, **98**, **99**
- stopnode, **114**, **114**, **116**, **142**, **145**
- switchsize, **57**
- switchstyle, **57**, **57**
- synthshape, **67**, **67**, **149**
- synthsize, **66**
- synthtype, **66**, **67**, **149**
- thicklens, **28**
- usefiberstyle, **54**, **54**

- useNA, 90, **90**, 125, 147
- usewirestyle, 54, **54**
- variable, 18, 41, **41**, 61, 62, 144, 150
- voltage, **35**
- wire, 31, 32, **119**, 120, 122, 137, 148, 152
- wire\*, 116
- wirealign, **117**
- wireangleA, **117**, 149
- wireangleB, **117**, 137, 148, 149
- wirestyle, 66, **117**

parmirrorheight, **42**, 43

parmirrorwidth, **42**, 43

\pentaprism, 53, **53**, 80, 97, 140

pentaprismsize, **53**, 80

phlinewidth, 30, **30**

phwidth, 30, **30**, 93, 135, 136, 143, 144

PiezoMirror, 40, 42, **42**

\pinhole, 30, **30**, 93, 135, 136, 140, 141, 143, 144, 146, 151

plateheight, 29, **29**, 30, 93–95, 104, 106, 110, 127, 141, 143, 150, 167

platelinewidth, **29**, 135

platesize, **29**

platewidth, **29**, 30

Polarization, 38, **39**, 148

\polarization, 38, **38**, 39, 84, 96, 135, 141, 148

\polcontrol, 56, **56**, 138, 142, 149

polcontrolsize, **56**

polcontroltype, 56, **56**

polllinewidth, **38**

polsize, **38**, 135, 148

poltype, **38**, 39, 96, 135, 148

position, 10–17, 20–22, **22**, 23, 33, 35, 59, 62, 66, 71, 72, 81, 83, 87, 93–95, 97, 98, 104, 106, 107, 110, 112, 116, 118–120, 122, 128–131, 133–138, 140–146, 148–153, 167

prismalign, **52**, 92, 155, 165

prismangle, 51, **51**, 92, 96, 99, 153, 154

prismsize, **51**, 89

prismtype, **51**, 52, 96, 99, 153, 165, 166

\psbezier, 71

pspicture, 74, 82

pst-eucl, 6

pst-intersect, 6  
pst-node, 6, 22, 116  
pst-optexp, 6, 25, 123, 124  
pst-plot, 6  
pstricks, 6  
pstricks-add, 6, 102  
pswarning, 50, 107

## R

raprismalign, 53  
raprismsize, 53  
raytrace, 37, 88, 89, 90, 101, 102, 133  
refractiveindex, 93  
reverse, 46, 47, 93  
\rightangleprism, 52, 52, 53, 165  
rotateref, 23, 23, 80, 85, 95, 128, 130, 134

## S

savebeam, 83, 85, 108, 109, 110, 153  
savebeampoints, 80, 81, 107, 108, 133, 136, 144  
SemitransMirror, 40, 42, 145  
showifcnodes, 63, 95, 125, 159, 159  
showinterfaces, 160, 160  
showoptdots, 53, 73  
sourceheight, 34, 34  
sourcesize, 34  
sourcewidth, 34, 34  
startinside, 94, 98, 109, 110, 133  
startinsidecount, 95, 98, 99, 153  
startnode, 114, 114, 115, 142, 145  
Stile

- ArrowCompStyle, 32, 32, 58
- BarCompStyle, 33, 33
- Beam, 15, 18, 38, 43, 47–51, 70, 81, 82, 85, 91–93, 95, 99–101, 101, 102, 104, 105, 108, 109, 127, 133–136, 139–141, 143–147, 151, 153–155, 158, 160, 167
- CrystalCaxis, 35, 35
- CrystalLamp, 35, 36
- DetectorStyle, 72, 72
- ExtendedMirror, 25, 40, 42, 42, 78, 136, 144

- FdlArrow, **58**
- Fiber, **54, 58, 59, 66, 112, 113, 115, 115, 116, 121, 137, 152**
- FiberIn, **121**
- FiberIn1, **121**
- FiberIn2, **121**
- FiberOut, **121**
- FiberOut1, **58, 121**
- FiberOut2, **58, 121**
- FilterStyle, **70, 70**
- IfcNodeStyle, **159**
- IfcStyle, **160, 160**
- IsolatorArrow, **57, 57**
- OptCircArrow, **59, 59, 149**
- OptComp, **25, 25, 26, 70, 72, 144**
- OptionalStyle, **26, 26**
- PiezoMirror, **40, 42, 42**
- Polarization, **38, 39, 148**
- SemitransMirror, **40, 42, 145**
- SynthStyle, **67, 67**
- VariableCoupler, **62**
- VariableStyle, **26, 41**
- Wire, **54, 66, 68, 117, 117, 121, 137, 148, 149, 152**
- WireIn, **121, 122**
- WireIn1, **121, 122**
- WireIn2, **121, 122**
- WireOut, **121, 122**
- WireOut1, **68, 121, 122**
- WireOut2, **68, 121, 122**
- stopinside, **82, 83, 98, 108, 110, 110, 127, 133, 135**
- stopinsidecount, **43, 98, 99**
- stopnode, **114, 114, 116, 142, 145**
- switchsize, **57**
- switchstyle, **57, 57**
- synthshape, **67, 67, 149**
- synthsize, **66**
- SynthStyle, **67, 67**
- synthtype, **66, 67, 149**

**T**

thicklens, **28**

\transmissiongrating, 47, 47

## U

Umgebungen

– optexp, 11, 18, 46, 123, 169

– pspicture, 74, 82

usefiberstyle, 54, 54

useNA, 90, 90, 125, 147

usewirestyle, 54, 54

## V

variable, 18, 41, 41, 61, 62, 144, 150

VariableCoupler, 62

VariableStyle, 26, 41

voltage, 35

## W

\wdmcoupler, 60, 60, 62, 126, 138, 145, 149

\wdmsplitter, 16, 17, 60, 60, 62, 119, 122, 138, 143, 145, 149, 152

Wire, 54, 66, 68, 117, 117, 121, 137, 148, 149, 152

wire, 31, 32, 119, 120, 122, 137, 148, 152

wire\*, 116

wirealign, 117

wireangleA, 117, 149

wireangleB, 117, 137, 148, 149

WireIn, 121, 122

WireIn1, 121, 122

WireIn2, 121, 122

WireOut, 121, 122

WireOut1, 68, 121, 122

WireOut2, 68, 121, 122

wirestyle, 66, 117

## 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 dass 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. 12.3 zu finden.

### 5.1 2014-11-19

Added component `\oapmirror`.

Added support for filled beams with path interfaces.

### 5.0 2014-10-30

Added support for arbitrary interface shapes (experimental, requires the `pst-intersect` package and `pstricks` version 2.53.

Added component `\parabolicmirror`.

### 4.10 2014-06-04

Added value `firstcomp` to parameter `beamalign` . . . . . 9.2.3

Modified `\optsource` to handle also the beam alignment . . . . . 4.8

### 4.9 2014-06-03

Added new component `\optsource` . . . . . 4.8

Improved naming of the last component name, which can now be accessed also by its number besides using `N`.

Fixed bug related to `extnodes` together with `compname`



**4.8 2014-02-28**

Added component <code>\optaom</code> . . . . .	4.20
Added option <code>beammode</code> . . . . .	9.2.4
Added option <code>showinterfaces</code> . . . . .	12.1

**4.7 2014-01-12**

Added syntax <code>&lt;num&gt;</code> for accessing relative component IDs . . . . .	8.1
--	-----

**4.6 2013-11-27**

Added option <code>extnodes</code> . . . . .	8.5
Added option <code>optdipolecomp</code> . . . . .	10
Added option <code>opttripolecomp</code> . . . . .	10

**4.5 2013-07-21**

Added option <code>usewirestyle</code> . . . . .	5
Added line style <code>fade</code> , with options <code>fadeto</code> , <code>fadepoints</code> , <code>fadefunc</code> , and <code>fadefuncname</code> . . . . .	9.2.7 .

**4.4 2013-07-14**

Added component <code>\transmissiongrating</code> . . . . .	4.19
Modified <code>innerlabel</code> to explicitly set <code>labelalign=c</code>	
Fixed trailing spaces in <code>savebeampoints</code> (and <code>savebeam</code> ).	
Allow Postscript code in <code>angle</code> .	

**4.3a 2013-07-02**

Fixed tripole alignment bugs introduced in version 4.3.	
Again fixed the alignment of reflective <code>\optprism</code> . . . . .	4.21
Fixed improper alignment of <code>\optprism</code> label . . . . .	12.3.1

**4.3 2013-06-28**

Added option <code>raprismalign</code> . . . . .	4.22
Fixed alignment of <code>\rightangleprism</code> and rotation of its label.	

**4.2 2013-06-07**

Added option startinsidecount . . . . .	9.2.4
Added option stopinsidecount . . . . .	9.2.4
Added option beampathcount . . . . .	9.2.3
Added option beamnodealign . . . . .	9.2.5
Added \oeBeamVecMedian . . . . .	8.9
Added \oeBeamCenter . . . . .	8.8
Modified the orientation of a reflective \optprism . . . . .	12.3.3
Fixed bug of \optprism and \rightangleprism inside an optexp environment.	
Fixed bug of wrong \rightangleprism rotation in special cases.	
Fixed buggy handling of multiple saved beams . . . . .	9.5

**4.1 2013-05-21**

Added option prismtype . . . . .	4.21
----------------------------------	------

**4.0 2013-04-15**

Added electrical components . . . . .	6
Added wire connections with \drawwire . . . . .	9.7
Added component \optarrowcomp . . . . .	4.6
Added component \optbarcomp . . . . .	4.7
Improved \optdetector . . . . .	7.3
Extended \drawfiber to support component ranges . . . . .	9.6
Extended couplersep to support automatic calculation . . . . .	5.10
Fixed appearance of \optfiber . . . . .	12.3.4
Fixed extnodealign to affect also multipoles . . . . .	8.5
Fixed pswarning.	
Removed old, buggy beam connection code . . . . .	12.3.4
Removed option conn . . . . .	12.3.4
Removed option connjoin . . . . .	12.3.4
Removed \newOptexpDipoleNolabel . . . . .	12.3.4

**3.6 2013-03-20**

Added option compoffset . . . . .	3.3
Added option beampathskip . . . . .	9.2.3
Added option innercompalign . . . . .	3.3
Fixed some bugs in the raytracing and beam code.	

**3.5 2013-02-22**

Added option `filterangle` . . . . . 7.1  
 Fixed wrong output fiber style in `\optcoupler`  
 Fixed strange Postscript error which occurred with some interpreters.

**3.4 2013-02-03**

Fixed bugs when using `fillstyle` for some components.  
 Extended option `switchsize` . . . . . 5.6  
 Modified coupler center node to be on the base line . . . . . 8.3  
 Extended error checking for `\drawbeam` and `\drawfiber` . . . . . 9.1  
 Fixed some bugs in the connection code.

**3.3a 2012-09-18**

Fixed bug which was exposed by an update of `pst-node`.  
 Fixed trailing spaces.

**3.3 2012-08-17**

Extended `\opttripole` and `\optdipole` . . . . . 10  
 Modified reference angle for label of `\opttripole` . . . . . 12.3.5  
 Added option `optdipolesize` to support two interfaces for `\optdipole` . 10  
 Added option `gratingalign` . . . . . 4.18  
 Added option `forcebeaminside` . . . . . 9.2.4  
 Extended option `mirrorradius` . . . . . 4.14  
 Improved `\drawbeam` and `\drawwidebeam` to be able to use `ArrowInside`  
   without `linestyle` . . . . . 9.2.7  
 Fixed orientation of `\oenodeBeamUp` and `\oenodeBeamLow` . . . . . 8.8  
 Fixed a bug related to `beaminside` and ambiguous components.  
 Fixed some trailing spaces.  
 Added more examples . . . . . 11

**3.2 2012-07-26**

Added component `\glanthompson` . . . . . 4.12  
 Added access to beam vectors with `\oeBeamVec`, `\oeBeamVecUp` and  
   `\oeBeamVecLow` . . . . . 8.9  
 Fixed wrong computation of node distance in `\fiberbox`  
 Fixed bug in `\wdmsplitter` for `coupleralign=b` and `couplertype=none`  
 Fixed bug when filling components placed with `position=end/start`  
 Added value `absolute` to parameter `labelref` . . . . . 3.1

Added examples . . . . .	11
--------------------------	----

### 3.1 2012-07-17

Added component \fiberbox . . . . .	5.11
Extended connection macros to not required curly braces around node parenthesis any more . . . . .	8.1
Modified \fibercollimator to have allowbeaminside=false by default	7.2
Extended fiber couplers to allow using only two nodes . . . . .	5.10
Fixed a bug concerning node expressions with \drawfiber	

### 3.0 2012-07-09

Modified beam connections with \drawbeam to support raytracing . . . .	9.2
Added wide beams with \drawwidebeam . . . . .	9.3
Added \drawfiber for fiber connections . . . . .	9.6
Added optexp environment for layering of components and connections	9.9
Added german documentation.	
Modified naming of component nodes . . . . .	8
Modified extnode to work with more components . . . . .	8.5
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 . . . . .	4.9
Deprecated option lampscale . . . . .	4.9
Added style CrystalCaxis . . . . .	4.9
Added style CrystalLamp . . . . .	4.9
Added option optboxsize . . . . .	4.5
Extended option detsize . . . . .	7.3
Added style DetectorStyle . . . . .	7.3
Removed deprecated \detector, use \optdetector . . . . .	7.3
Extended option doveprismsize . . . . .	4.11
Added style Polarization . . . . .	4.13
Deprecated option pollinewidth . . . . .	4.13
Removed option polwidth . . . . .	4.13
Removed option pol . . . . .	4.13

Added style VariableStyle . . . . .	4.14
Added mirror type semitrans . . . . .	4.14
Renamed \optgrid to \optgrating . . . . .	4.18
Renamed optgridwidth to gratingwidth . . . . .	4.18
Renamed optgridheight to gratingheight . . . . .	4.18
Renamed optgriddepth to gratingdepth . . . . .	4.18
Renamed optgridcount to gratingcount . . . . .	4.18
Renamed optgridtype to gratingtype . . . . .	4.18
Renamed optgridlinewidth to gratinglinewidth . . . . .	4.18
Added option prismalign . . . . .	4.21
Modified faulty alignment of \rightangleprism . . . . .	4.22
Extended option optampsize . . . . .	5.2
Extended option optmzsize . . . . .	5.3
Added option polcontroltype . . . . .	5.4
Extended option isolatorsizes . . . . .	5.5
Added style IsolatorArrow . . . . .	5.5
Extended option fdlsizes . . . . .	5.7
Added style FdlArrow . . . . .	5.7
Extended option fiberpolsize . . . . .	5.8
Added component \optcirculator . . . . .	5.9
Extended option couplersizes . . . . .	5.10
Extended option couplertypes . . . . .	5.10
Renamed option align to coupleralign . . . . .	5.10
Added style VariableCoupler . . . . .	5.10
Added style FilterStyle . . . . .	7.1
Extended option fibercolsize . . . . .	7.2
Removed deprecated option labelrelative . . . . .	3.1
Removed deprecated option iwidth . . . . .	4.4
Removed deprecated option owidth . . . . .	4.4
Removed deprecated option bswidth . . . . .	4.17
Deprecated \newOptexpDipoleNoLabel, use \newOptexpDipole . . . . .	10.2
Deprecated option refractiveindex . . . . .	9.2.2
Deprecated option conn . . . . .	9.2
Extended option fiber . . . . .	9.8

## 2.1 2009-11-05

Added component \optfiberpolarizer . . . . .	5.8
Added option compshift . . . . .	3.3
Added option label . . . . .	3.1

Added option connjoin	
Added options addtoBeam and newBeam	9.2
Added style OptComp and related options addtoOptComp and newOptComp	3.5
Added option bsstyle	4.17
Extended \fibercollimator to use up to four reference nodes	7.2
Improved thicklens to work also with plain lenses	4.1
Use pst-doc class for the documentation.	

## 2.0 2008-07-27

Added fiber-optical components	5
Added component \optdiode	4.10
Added component \pentaprism	4.23
Added component \rightangleprism	4.22
Added component \doveprism	4.11
Added component \optprism	4.21
Added \drawbeam	9.2
Added component connections (options fiber, conn and beam)	9
Added option compname	8.1
Added option extrnode	8.5
Renamed \detector to \optdetector	7.3

## 1.2 2008-06-17

Modified lens design to use interface curvatures	4.1
Added options lensradiusleft and lensradiusright	4.1
Added option thicklens	4.1
Added option lenstype	4.1
Added option mirrorradius (curved mirrors)	4.14
Added option optgridtype (binary gratings)	4.18
Added \newOptexpDipole	10.2
Added \newOptexpDipoleNolabel	10.2
Added \newOptexpTripole	10.2
Added \newOptexpFiberDipole	10.2
General improvements of T <sub>E</sub> X and Postscript code	

## 1.1 2007-09-06

Improved labeling features	3.1
Added parameter labelref	3.1
Replaced labelrelative by labelref=relative	3.1
Renamed \polarisation to \polarization	4.13

Renamed polwidth to polsize . . . . .	4.13
Renamed pol to poltype . . . . .	4.13
Renamed bswidth to bssize . . . . .	4.17
Renamed iwidth to innerheight . . . . .	4.4
Renamed owidth to outerheight . . . . .	4.4
Added support for fillstyle for all components	

## **1.0 2007-07-18**

First CTAN version