

Linien und Polygone

4.1 Parameter	45
4.2 <code>\psline</code>	58
4.3 <code>\qline</code>	58
4.4 <code>\pspolygon</code>	59
4.5 <code>\psframe</code> und <code>\psTextFrame</code>	59
4.6 <code>\psdiamond</code>	61
4.7 <code>\pstriangle</code>	61
4.8 Beispiele	62

Linien stellen einen Schwerpunkt einer jeden grafischen Software dar und haben auch in PSTricks eine große Bedeutung. Entsprechend umfangreich ist auch die Zahl der möglichen Parameter, die alle in Tabelle 4.1 zusammengefasst sind und im folgenden erläutert werden.

4.1 Parameter

Tabelle 4.1 enthält sämtliche Parameter, die im Zusammenhang mit Linien von Interesse sind. Ein Großteil von ihnen kann auch für nicht-linientypische Makros eingesetzt werden, beispielsweise für `\pscircle`.

Tabelle 4.1: Zusammenfassung aller Parameter für Linien und Polygonzüge

<i>Name</i>	<i>Werte</i>	<i>Vorgabe</i>
<code>linewidth</code>	Wert <i>Einheit</i>	0.8pt
<code>linecolor</code>	Farbe	black
<code>linestyle</code>	none solid dotted dashed	solid symbol
<code>symbolStep</code>	Wert <i>Einheit</i>	20pt
<code>symbolWidth</code>	Wert <i>Einheit</i>	10pt
<code>symbolFont</code>	PS-Font	Dingbats

Fortsetzung...

... Fortsetzung

<i>Name</i>	<i>Werte</i>	<i>Vorgabe</i>
rotateSymbol	<i>Boolean</i>	false
startAngle	<i>Winkel</i>	0
linejoin	0 1 2	0
linecap	0 1 2	0
dash	<i>Wert Einheit</i> <i>Wert Einheit</i> ...	5pt 3pt
dotsep	<i>Wert Einheit</i>	3pt
doubleline	<i>Boolean</i>	false
doublesep	<i>Wert Einheit</i>	1.25\pslinewidth
doublecolor	<i>Farbe</i>	white
dimen	outer inner middle	outer
arrows	<i>Pfeiltyp</i>	-
showpoints	<i>Boolean</i>	false
linearc	<i>Wert Einheit</i>	0pt
framearc	<i>Wert</i>	0
cornersize	relative absolute	relative
gangle	<i>Winkel</i>	0
border	<i>Wert Einheit</i>	0pt
bordercolor	<i>Farbe</i>	white
shadow	<i>Boolean</i>	false
shadowsize	<i>Wert Einheit</i>	3pt
shadowangle	<i>Winkel</i>	-45
shadowcolor	<i>Farbe</i>	darkgray
linetype	<i>Wert</i>	0
liftpen	0 1 2	0

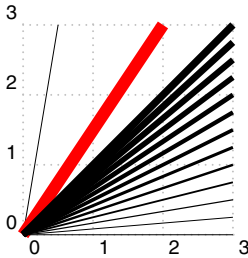
Im Folgenden wird zu jedem der angegebenen Parameter ein Beispiel angegeben, wobei sich die Reihenfolge an Tabelle 4.1 orientiert. Eine Beschreibung der Parameter zu den Fülloptionen finden sich in Kapitel 7 auf Seite 93.

4.1.1 linewidth

Grundsätzlich kann jede beliebige Liniendicke gewählt werden. Sowohl die größte als auch die kleinste Dicke orientieren sich an dem zugrundeliegenden PS-Treiber, über den \TeX bzw. PS keinerlei Informationen vorliegen, sodass an dieser Stelle keine Entscheidung über Sinn oder Unsinn der Liniendicke getroffen werden kann. Variable Liniendicken sind nur für Kurvenzüge möglich (\Rightarrow Abschnitt 5.1.4 auf Seite 67).

Zu beachten ist in jedem Fall, dass es bei der PDF-Ausgabe auf dem Bildschirm Probleme mit zu dünnen Linien geben kann, denn die Bildschirmauflösung setzt hier Grenzen. Erst der Ausdruck zeigt in der Regel die korrekte Liniendicke.

04-01-1



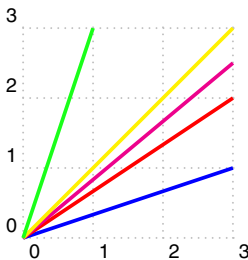
```
\usepackage{pstricks,multido}

\begin{pspicture}[showgrid](3,3)
  \psline[linewidth=0.01pt](0.5,3)
  \psline[linewidth=5pt,linestyle=red](2,3)
  \multido{\rA=0.0+0.25}{13}{%
    \psline[linewidth=\rA pt](3,\rA)}
\end{pspicture}
```

4.1.2 linecolor

Wie im Abschnitt 2.1 auf Seite 10 erwähnt, kennt PSTricks ohne externe Pakete bereits insgesamt 11 vordefinierte Farben, deren Zahl vom Anwender beliebig erweitert werden kann.

04-01-2



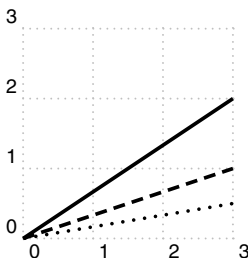
```
\usepackage{pstricks}

\begin{pspicture}[showgrid,
  linewidth=1.5pt](3,3)
  \psline[linestyle=blue](3,1)
  \psline[linestyle=red](3,2)
  \psline[linestyle=magenta](3,2.5)
  \psline[linestyle=yellow](3,3)
  \definecolor{LColor}{rgb}{0.1,1,0.1}
  \psline[linestyle=LColor](1,3)
\end{pspicture}
```

4.1.3 linestyle

Dem Beispiel kann entnommen werden, dass die erste Linie mit dem Linienstil none nicht gezeichnet wird. Ein derartiges Verhalten ist insbesondere dann interessant, wenn man beispielsweise Flächen ohne eine Randlinie füllen oder Endpunkte (Knoten) einer Linie setzen will, ohne dass diese gezeichnet wird.

04-01-3



```
\usepackage{pstricks}

\begin{pspicture}[showgrid](3,3)
  \psset{linewidth=1.5pt}
  \psline[linestyle=none](3,3)%<-- keine Linie!
  \psline[linestyle=solid](3,2)
  \psline[linestyle=dashed](3,1)
  \psline[linestyle=dotted](3,0.5)
\end{pspicture}
```

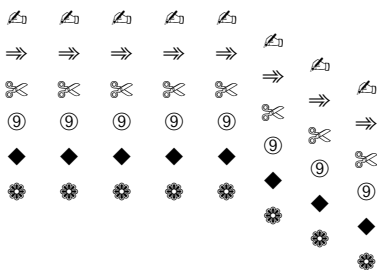
Der Linienstil `symbol` kann über weitere Parameter gesteuert werden, die in Tabelle 4.1 auf Seite 45 zusammengefasst sind. Standardmäßig wird das Symbol der Oktalnummer 141 (dezimal 97), welches dem Kleinbuchstaben »a« entspricht. Die Symbole können über das entsprechende Zeichen aus einem Roman-Zeichensatz oder alternativ über die

Angabe der Oktalzahl ausgewählt werden. Der sinnvolle Bereich ist dabei 041...176 und 241...376, der sich aus Tabelle 4.2 ersehen lässt, wenn die Werte links und oberhalb der Auflistung zugrunde gelegt werden.

Tabelle 4.2: Die Zuordnung zwischen Oktal- bzw. Hexadezimalzahl und Symbol beim PostScript-Zeichensatz Zapf-Dingbats.

	'0	'1	'2	'3	'4	'5	'6	'7	
'04x									"2x
'05x									
'06x									"3x
'07x									
'10x									"4x
'11x									
'12x									"5x
'13x									
'14x									"6x
'15x									
'16x									"7x
'17x									
'24x									"Ax
'25x									
'26x									"Bx
'27x									
'30x									"Cx
'31x									
'32x									"Dx
'33x									
'34x									"Ex
'35x									
'36x									"Fx
'37x									
	"8	"9	"A	"B	"C	"D	"E	"F	

Grundsätzlich kann für die Symbole jeder der standardmäßig vorhandenen PS-Zeichensätze benutzt werden. Eine entsprechende Liste wurde bereits in Abschnitt 3.2.5 auf Seite 36 angegeben.



```
\usepackage{pstricks}
```

```
\pspicture(0,-1)(5,3) \psset{linestyle=symbol}
\psline(0,0)(3,0)(5,-1)
\psline[symbol=u](0,0.5)(3,0.5)(5,-.5)
\psline[symbol=310](0,1)(3,1)(5,0)
\psline[symbol=044](0,1.5)(3,1.5)(5,.5)
\psline[symbol=376](0,2)(3,2)(5,1)
\psline[symbol=055](0,2.5)(3,2.5)(5,1.5)
\endpspicture
```

04-01-4

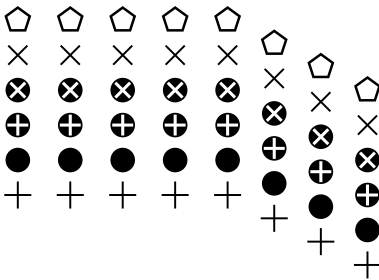
Alternativ zu den Symbolen aus den PS-Zeichensätzen können die intern definierten Symbole des Zeichensatzes `PSTricksDotFont` benutzt werden (siehe Tabelle 6.2 auf Seite 84). Es handelt sich dabei um Type-3-Vektorzeichen, auf die nur über einen zugeordneten Buchstaben zugegriffen werden kann. Eine Zusammenstellung zeigt Tabelle 4.3.

Tabelle 4.3: Die Zuordnung zwischen Zeichen und Symbol beim `PSTricks`-Zeichensatz `PSTricksDotFont`. Leere Felder sind dabei nicht belegt.

04-01-5

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
+	●	○	◇	⊕	⊖	⊗	⊘		*	◆	⊕	⊗		◇	◆	■	□	△	▲						×
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
+	○	◇	⊗	⊕	⊖	⊗		⊗	*		⊕	⊗		◇		□	△								×

04-01-6

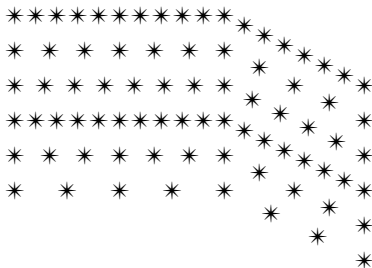


```
\usepackage{pstricks}

\pspicture(0,-1)(5,3)
\psset{linestyle=symbol,
symbolFont=PSTricksDotFont}
\psline(0,0)(3,0)(5,-1)
\psline[symbol=b](0,0.5)(3,0.5)(5,-.5)
\psline[symbol=e](0,1)(3,1)(5,0)
\psline[symbol=E](0,1.5)(3,1.5)(5,.5)
\psline[symbol=x](0,2)(3,2)(5,1)
\psline[symbol=P](0,2.5)(3,2.5)(5,1.5)
\endpspicture
```

Der Abstand der Symbole kann über das optionale Argument `symbolStep` beeinflusst werden, der standardmäßig auf 20 pt festgelegt. Längenangaben ohne Einheit werden auf die aktuell gültige Maßeinheit bezogen, im Standardfall auf die vorgegebenen cm.

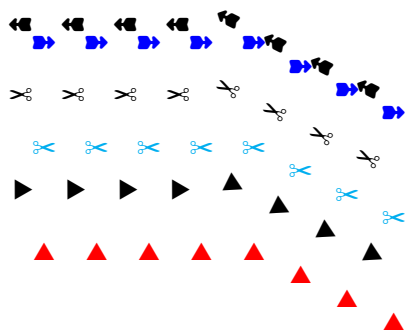
04-01-7



```
\usepackage{pstricks}

\pspicture(0,-1)(5,3)
\psset{linestyle=symbol,symbol=T}
\psline(0,0)(3,0)(5,-1)
\psline[symbolStep=0.5](0,0.5)(3,0.5)(5,-.5)
\psline[symbolStep=8pt](0,1)(3,1)(5,0)
\psline[symbolStep=11pt](0,1.5)(3,1.5)(5,.5)
\psline[symbolStep=14pt](0,2)(3,2)(5,1)
\psline[symbolStep=3mm](0,2.5)(3,2.5)(5,1.5)
\endpspicture
```

Die Ausrichtung der Symbole entspricht dem vorgegebenen Layout des Zeichens, unabhängig von der Steigung der jeweiligen Linie. Insbesondere bei Dreiecksymbolen kann eine Rotation angebracht sein, wenn eine Spitze des Dreiecks in Richtung der Linie zeigt. Für das Symbol der Schere kann dies ebenfalls angebracht sein. Den Winkel der Rotation kann man mit `startAngle` beeinflussen. Dies ist immer dann notwendig, wenn intern die Richtung der Linie nicht berechnet werden konnte.



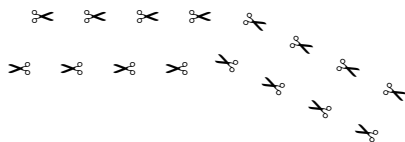
```
\usepackage{pstricks}
```

04-01-8

```
\pspicture(0,-1)(5,3)
\psset{linestyle=symbol}
\psline[symbol=163,linicolor=red](0,0)(3,0)(5,-1)
\psline[symbol=163,rotateSymbol,rot=30](0,1)(3,1)(5,0)
\psline[symbol=042,linicolor=cyan](0,1.5)(3,1.5)(5,0.5)
\psline[symbol=042,rotateSymbol](0,2.5)(3,2.5)(5,1.5)
\psline[symbol=375,linicolor=blue](0,3)(3,3)(5,2)
\psline[symbol=375,rotateSymbol](0,3.5)(3,3.5)(5,2.5)
\endpspicture
```

Linien-
richtung

Zu beachten ist, dass Linienzüge intern in umgekehrter Reihenfolge gezeichnet werden und daher die Richtung der Symbole um genau 180° gedreht erscheint. Über das optionale Argument `rot` können diese bei Bedarf beliebig gedreht werden.

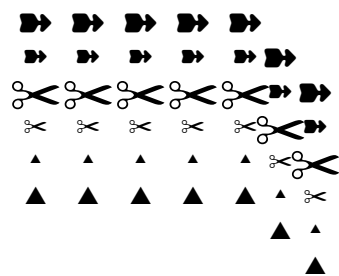


```
\usepackage{pstricks}
```

04-01-9

```
\pspicture(0,-1)(5,0.5) \psset{linestyle=symbol}
\psline[symbol=042,rotateSymbol](0,0)(3,0)(5,-1)
\psline[symbol=042,rotateSymbol,
rot=180](0,.5)(3,.5)(5,-0.5)
\endpspicture
```

Die Symbolgröße kann über den Faktor `symbolWidth` beeinflusst werden, wobei die Skalierung symmetrisch zur Breite/Höhe erfolgt. Eine getrennte Einstellung ist hier nicht möglich. Bei einer Größenangabe ohne Einheit wird diese auf die aktuell gültige Maßeinheit bezogen, im Standardfall auf die vorgegebenen `cm`.



```
\usepackage{pstricks}
```

04-01-10

```
\pspicture(0,-1)(4,3) \psset{linestyle=symbol}
\psline[symbol=163](0,0)(3,0)(4,-1)
\psline[symbol=163,symbolWidth=5pt](0,.5)(3,.5)(4,-.5)
\psline[symbol=042](0,1)(3,1)(4,0)
\psline[symbol=042,
symbolWidth=0.3in](0,1.5)(3,1.5)(4,.5)
\psline[symbol=375](0,2)(3,2)(4,1)
\psline[symbol=375,
symbolWidth=0.5](0,2.5)(3,2.5)(4,1.5)
\endpspicture
```

4.1.4 linejoin

PS kann beim Aufeinandertreffen zweier Linienenden nicht wissen, wie diese verbunden werden sollen, denn dies kann auf unterschiedliche Weise erfolgen und wird auf PS-Ebene durch den Befehl `setlinejoin` kontrolliert. Ein entsprechender Parameter kann von \TeX aus durch die Option `linejoin` an PS übergeben werden, wobei nur die Werte 0, 1 oder 2 eine Wirkung in der Ausgabe zeigen. Dieser Parameter hat insbesondere bei dicken Linien und/oder kleinen Verbindungswinkeln eine große Bedeutung.

- 0 Die Ränder zweier Linien werden bis zu einem Schnittpunkt verlängert.
- 1 Die äußere Kontur wird durch einen Kreisbogen dargestellt.
- 2 Die äußere Kontur wird durch eine horizontale Linie dargestellt.

```
\usepackage{pstricks}
```

```
\psset{linewidth=3mm,unit=0.8}
\begin{pspicture}(4,2) \psline(0,0)(1,2)(2,0)(3,2)(4,0) \end{pspicture}
\begin{pspicture}(4,2) \psline[linejoin=1](0,2)(1,0)(2,2)(3,0)(4,2)\end{pspicture}
\begin{pspicture}(4,2) \psline[linejoin=2](0,0)(1,2)(2,0)(3,2)(4,0)\end{pspicture}
```

04-01-11



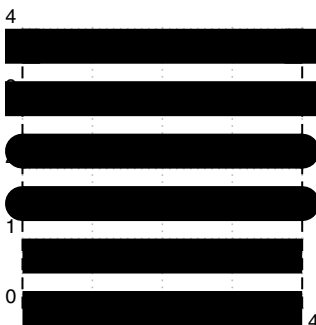
4.1.5 linecap

Das Ende von dicken Linien ist von Bedeutung und kann daher durch die Option `linecap` beeinflusst werden. Mögliche Werte sind 0, 1 oder 2.

- 0 Die Linien werden an den Koordinaten abgeschnitten (Standard).
- 1 Die Linien werden an den Koordinaten durch Halbkreise vom Radius der halben Liniendicke ($0.5 \cdot \text{\pslinewidth}$) verlängert.
- 2 Die Linien werden durch halbe Quadrate der Liniendicke ($\cdot \text{\pslinewidth}$) verlängert.

Wie dem folgenden Beispiel zu entnehmen ist, kann der Effekt auch ohne `linecap` erreicht werden, indem die entsprechenden Liniendenen durch »Pfeilsymbole« festgelegt werden (siehe dazu Tabelle 8.2 auf Seite 109). Die folgenden Beispiele verwenden übertriebene Liniendicken, um den Effekt besser zeigen zu können. In der Realität wird dies so nicht immer sichtbar werden und manchmal vielleicht beim Betrachten der Grafik so gar nicht auffallen.

04-01-12



```
\usepackage{pstricks}
```

```
\begin{pspicture}[showgrid](4,4)%
\psline[linestyle=dashed](0,4)
\psline[linestyle=dashed](4,0)(4,4)
\psset{linewidth=5mm}
\psline[arrows=C-C](0,3.75)(4,3.75)
\psline[linecap=2](0,3)(4,3)
\psline[arrows=c-c](0,2.25)(4,2.25)
\psline[linecap=1](0,1.5)(4,1.5)
\psline[arrows=-](0,0.75)(4,0.75)
\psline(4,0)
\end{pspicture}
```

Die Anwendung der `linecap`-Option macht nur in ganz speziellen Fällen Sinn. Pfeile sind grundsätzlich nicht Teil eines aktuellen Pfades, sodass Fülloptionen sich nur auf den Linienteil beziehen. Dies gilt nicht für `linecap`, da hier das Linienende mit zum aktuellen Pfad gehört.



```
\usepackage{pstricks}
```

```
\def\curve{\pscurve(-.1,.1)(-.15,.15)(0,.2)(.15,.15)(.1,.1)}
\psset{unit=5cm,linewidth=5mm}
```

```
\begin{pspicture}(-0.2,-0.6)(0.2,0.5)%
```

```
\rput(0,.2){\psset{arrows=c-c}\curve}
```

```
\rput(0,-.2){%
```

```
\psset{fillstyle=solid,fillcolor=red,arrows=c-c}%
```

```
\curve}
```

```
\rput(0,-.6){%
```

```
\psset{fillstyle=solid,fillcolor=red,linecap=1}%
```

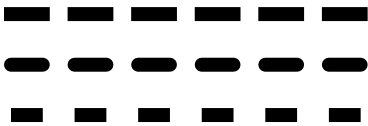
```
\curve}
```

```
\end{pspicture}
```

04-01-13



Für gestrichelte Linien hat `linecap` eine Auswirkung auf jedes einzelne Linienelement und nicht nur auf die äußeren Elemente:



```
\usepackage{pstricks}
```

```
\psset{linewidth=2mm,linestyle=dashed,
dash=5mm 5mm}
```

```
\psline[linecap=2](5,0)\l[3mm]
```

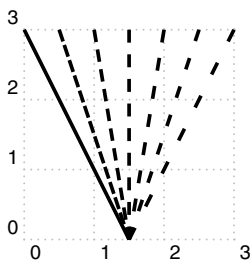
```
\psline[linecap=1](5,0)\l[3mm]
```

```
\psline[linecap=0](5,0)
```

04-01-14

4.1.6 dash

Voraussetzung für die Anwendung des `dash`-Parameters ist der Linienstil `dashed`.



```
\usepackage{pstricks,multido}
```

```
\begin{pspicture}[showgrid](3,3)
```

```
\psset{linewidth=1.5pt,linestyle=dashed}
```

```
\multido{\rA=0.0+1.5,\rB=0.0+0.5}{7}{%
```

```
\psline[dash=5pt \rA pt](1.5,0)(\rB,3)}
```

```
\end{pspicture}
```

04-01-15

PS erlaubt eine unbegrenzte Anzahl an Intervallen für die Definition einer gestrichelten Linie.

`dash=Wert1 Einheit Wert2 Einheit ...`

Die Option kann auch für jedes andere Linien- oder Kurvenmakro angewendet werden.

04-01-16



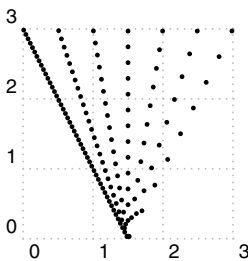
```
\usepackage{pstricks}

\begin{pspicture}(4,1.5)
  \psset{linestyle=dashed,linewidth=2pt}
  \psline[dash=3mm 3mm 1mm 1mm](0,1.5)(4,1.5)
  \psline[dash=5mm 2mm 0.1 0.2](0,1)(4,1)
  \psline[dash=5mm 1mm 1mm 1mm](0,0.5)(4,0.5)
  \psline[dash=5mm 1mm 1mm 1mm 1mm 1mm
    1mm 1mm 1mm 1mm](4,0)
\end{pspicture}
```

4.1.7 dotsep

Voraussetzung für die Anwendung des dotsep-Parameters ist der Linienstil . Die Größe der einzelnen Punkte orientiert sich an der Vorgabe von linewidth und ist nicht abhängig von den Parametern dotsize und dotscale, die sich auf das \psdot-Makro beziehen, welches später behandelt werden wird.

04-01-17



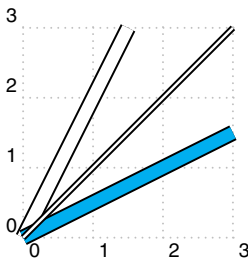
```
\usepackage{pstricks,multido}

\begin{pspicture}[showgrid](3,3)
  \psset{linewidth=2pt,linestyle=dotted}
  \multido{\rA=0.0+1.5,\rB=0.0+0.5}{7}{%
    \psline[dotsep=\rA pt](1.5,0)(\rB,3)}
\end{pspicture}
```

4.1.8 doubleline, doublesep, doublecolor

doublecolor und doublesep beziehen sich nur auf das »Innere« der Linie, die Linienfarbe und Liniendicke an sich kann mit linecolor bzw. linewidth geändert werden.

04-01-18



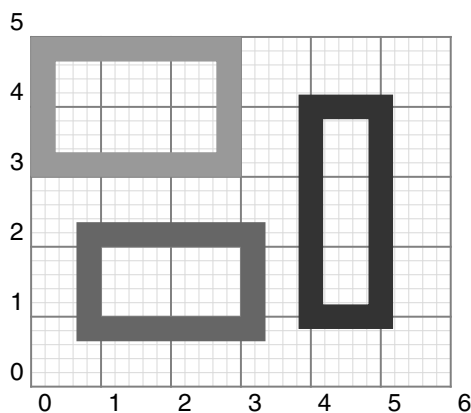
```
\usepackage{pstricks}

\begin{pspicture}[showgrid](3,3)
  \psset{doubleline=true}
  \psline[doublesep=5pt](1.5,3)
  \psline[doublesep=5pt,doublecolor=cyan](3,1.5)
  \psline(3,3)
\end{pspicture}
```

4.1.9 dimen

Diese Option bezieht sich ausschließlich auf geschlossene Linienzüge wie \psframe, \pscircle, \psellipse und \pswedge. Alle anderen sind nicht betroffen, wobei sich dimen bei \pswedge nur auf den Radius bezieht, das Zentrum liegt immer in der Mitte.

`dimen` legt fest, worauf sich die angegebenen Koordinaten beziehen, entweder auf das *Innere* (`inner`), das *Äußere* (`outer`) oder die *Mittellinie* (`middle`) des Grafik-Objekts. Die folgende Abbildung macht dies deutlicher.



```
\usepackage{pstricks}

\begin{pspicture}(6,5)
  \psgrid[subgriddiv=5,griddots=0,
    subgridwidth=0.1pt,
    subgridcolor=black!15,
    gridcolor=black!50]
  \psset{linewidth=10pt}
  \psframe[dimen=outer,
    linecolor=black!40](0,3)(3,5)
  \psframe[dimen=inner,
    linecolor=black!60](1,1)(3,2)
  \psframe[dimen=middle,
    linecolor=black!80](4,1)(5,4)
\end{pspicture}
```

04-01-19

4.1.10 arrows

`PSTricks` hat bereits eine große Zahl an vordefinierten Pfeilen, bzw. Linienendmarkierungen, die in Tabelle 8.2 auf Seite 109 zusammengestellt sind. Diese Pfeile können für einen Großteil der Linien- beziehungsweise Kurvenmakros alternativ über das key-value-Interface oder die spezielle Option gesetzt werden, was im Folgenden für `\psline` gezeigt wird:

```
\psline [arrows=Pfeiltyp] (x,y) ...
\psline {Pfeiltyp} (x,y) ...
```

Hierin steht »*Pfeiltyp*« für einen Ausdruck der Form »*Startpfeil-Endpfeil*«, wobei sowohl *Startpfeil* als auch *Endpfeil* fakultative Angaben sind. `arrows=-` oder `{-}` ergeben somit eine Linie oder Kurve ohne Pfeile an den Enden, was der allgemeinen Vorgabe entspricht.

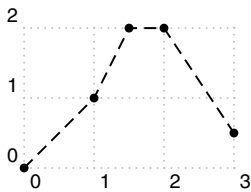


Aufgrund der unterschiedlichen Möglichkeiten zur Festlegung von Linienenden, sind auch widersprüchliche Angaben möglich, die jedoch keine Auswirkung haben, da immer nur die letzte Angabe wirksam ist. Ein `\psline[arrows=->]{-}(3,0)` hat daher dieselbe Wirkung wie ein `\psline(3,0)`, denn die Optionen werden auch intern in derselben Reihenfolge eingelesen und daher überschreibt »-« alle vorhergehenden Definitionen. Besteht die Linie aus einem Linienzug, so bezieht sich die Pfeilangabe auf den Beginn (erste Linie) und das Ende (letzte Linie) des Linienzuges. Per Definition erstellt `\pspolygon` geschlossene Linienzüge, indem eine Linie vom letzten zum ersten Punkt gezogen wird, sodass Pfeilangaben hier prinzipiell keinen Sinn machen.

4.1.11 showpoints

Dies ist primär für Bézierkurve und alle anderen Makros, die Kurven zeichnen von Interesse, um so besser zu erkennen, wo die eigentlichen Punkte liegen. Aber auch bei Linienzügen kann es in manchen Fällen sinnvoll sein, die Option `showpoints` zu setzen.

04-01-20



```
\usepackage{pstricks}

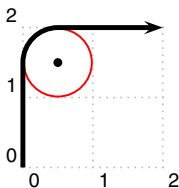
\begin{pspicture}[showgrid](3,2)
  \psline[showpoints,linestyle=dashed]%
    (0,0)(1,1)(1.5,2)(2,2)(3,0.5)
\end{pspicture}
```

Die Größe der Punkte kann über die Parameter `dotsize` und `dotscale` beeinflusst werden. Die Beschreibung dazu findet sich in Kapitel 6 auf Seite 83.

4.1.12 linearc

Mit dieser Option können anspruchsvolle Linienzüge erstellt werden. Prinzipiell macht die Option `linearc` nur bei Linienzügen bzw. Polygonen Sinn. Die erste Abbildung zeigt, dass der Wert für `linearc` den Radius des Kreises angibt, um den die Linie »gebogen« wird.

04-01-21

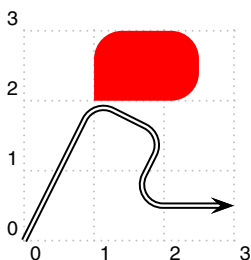


```
\usepackage{pstricks}

\begin{pspicture}[showgrid](2,2)
  \pscicle[linecolor=red](0.5,1.5){0.5}
  \psdot(0.5,1.5)
  \psline[linearc=0.5,linewidth=2pt]{->}(0,0)(0,2)(2,2)
\end{pspicture}
```

Wie der folgenden Abbildung zu entnehmen ist, werden Anfang und Ende eines Linienzuges, wenn sie identisch sind, normal verbunden, eine Anwendung des Parameters `linearc` erfolgt in diesem Fall nicht. Ein anderes Verhalten zeigt `\pspolygon`, welches per Definition geschlossene Kurvenzüge voraussetzt (vergleiche Abschnitt 4.4 auf Seite 59).

04-01-22

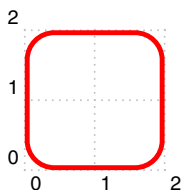


```
\usepackage{pstricks}

\begin{pspicture}[showgrid](3,3)
  \psline*[linecolor=red,linearc=0.4]%
    (1,2)(2.5,2)(2.5,3)(1,3)(1,2)
  \psline[linearc=0.3,doubleline=true]%
    {->}(0,0)(1,2)(2,1.5)(1.5,0.5)(3,0.5)
\end{pspicture}
```

4.1.13 framearc

Dieser Parameter ist letztlich identisch zu `linearc`, mit dem Unterschied, dass er sich auf eine geschlossene Fläche (Rahmen) bezieht, was sich primär auf `\psframe` (Abschnitt 4.5 auf Seite 59) und `\pspolygon` (Abschnitt 4.4 auf Seite 59) bezieht. Weiterhin kann `framearc` nur Werte zwischen 0 und 1 annehmen, wobei 1 sich auf die Hälfte der kürzesten Seite bezieht. Für ein Quadrat würde sich dann für den Wert 1 ein Kreis ergeben.



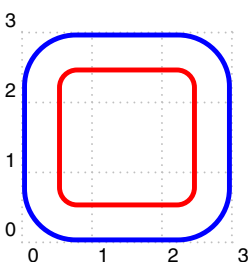
```
\usepackage{pstricks}

\begin{pspicture}[showgrid](2,2)
  \psframe[linewidth=2pt,framearc=0.4,
    linecolor=red](2,2)
\end{pspicture}
```

04-01-23

4.1.14 cornersize

Damit alle flächenförmigen Gebilde gleiches Verhalten an den Kanten zeigen, kann für `cornersize` zwischen den Werten `relative` und `absolute` gewählt werden. `relative` bezieht sich auf die Hälfte der kürzesten Seite, während `absolute` veranlasst, dass anstelle von `framearc` nun der absolute Wert von `linearc` herangezogen wird.



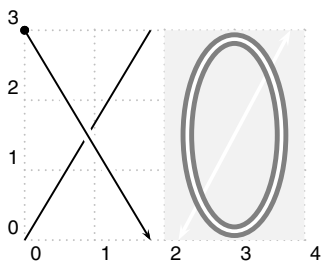
```
\usepackage{pstricks}

\begin{pspicture}[showgrid](3,3)
  \psframe[linewidth=2pt,linearc=0.25,
    cornersize=absolute,
    linecolor=red](0.5,0.5)(2.5,2.5)
  \psframe[linewidth=2pt,framearc=0.5,
    linecolor=blue](3,3)
\end{pspicture}
```

04-01-24

4.1.15 border, bordercolor

Mit dem Parameter `border` lassen sich sehr einfach Kreuzungen von Linienzügen zeigen, indem eine der Linien als oben liegend angesehen und mit einem umlaufenden Rahmen versehen wird. Mit `bordercolor` kann die Farbe festgelegt werden.



```
\usepackage{pstricks}

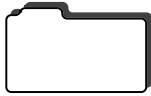
\begin{pspicture}[showgrid](4,3)
  \psline(0,0)(1.8,3)
  \psline[border=2pt]{*->}(0,3)(1.8,0)
  \psframe*[linecolor=gray!10](2,0)(4,3)
  \psset{linecolor=white}
  \psline[linewidth=1.5pt]{<->}(2.2,0)(3.8,3)
  \psellipse[linewidth=1.5pt,bordercolor=gray,
    border=2pt](3,1.5)(.7,1.4)
\end{pspicture}
```

04-01-25

4.1.16 shadow, shadowsize, shadowangle, shadowcolor

Schatteneffekte dienen vorrangig dazu, bestimmte Bereiche besonders hervorzuheben. Dabei sollte insbesondere die Schattengröße sorgfältig gewählt werden. Im Prinzip macht dies erst dann Sinn, wenn man geschlossene Polygonzüge hat, da nur hier die Schattenbildung auch gefüllte Flächen liefert, während sie bei Linien letztlich diese nur doppelt zeichnet, was mit der `doubleline`-Option ebenfalls möglich ist. Weitere Informationen dazu, wie `PSTricks` diese Schattenwirkung erzielt, gibt es im Kapitel 11.3.15 auf Seite 148.

04-01-26



```
\usepackage{pstricks}

\begin{pspicture}(2,1)
  \pspolygon[linearc=2pt,shadow=true,
    shadowangle=45](0,0)(0,1.1)(0.2,1.1)%
    (0.2,1.2)(0.8,1.2)(0.8,1.05)(2,1.05)(2,0)
\end{pspicture}
```

4.1.17 linetype

Die Linienstile `dashed` und `dotted` können nur dann lückenlos an bestehende Pfade (Linien oder Kurven) anschließen, wenn sie etwas über den aktuellen Pfadzustand, bzw. über die Art der Linie/Kurve wissen, die vorher gezeichnet wurde. Dies ist insbesondere für `\pscustom` wichtig (siehe Seite 137), wo beliebige Linien- und Kurvenarten aneinander gefügt werden können. Mit dem Parameter `linetype` kann man der aktuellen Kurve den eigenen Linientyp mitgeben (Tabelle 4.4).

Wert	Typ	Tabelle 4.4: Mögliche Werte für <code>linetype</code>	
0	Offene Kurve ohne Pfeile		
-1	Offene Kurve mit Pfeil am Anfang		
-2	Offene Kurve mit Pfeil am Ende		
-3	Offene Kurve mit Pfeil am Anfang und Ende		
1	Geschlossene Kurve mit verschiedenen Elementen		
n>1	Geschlossene Kurve mit n gleichartigen Elementen		

4.1.18 liftpen

Der Parameter `liftpen` kontrolliert das Verhalten beim Zeichnen von offenen Kurven, was insbesondere für `\pscustom` von Interesse ist (→ 11.3.2 auf Seite 140). Dort finden sich auch entsprechende Beispiele.

4.1.19 labelsep

Der Parameter `labelsep` gibt den Abstand von den Koordinaten und von einem zu setzenden Label an, was insbesondere für das `\uput`-Makro von Interesse ist (siehe Kapitel 9 auf Seite 119). Dort finden sich auch entsprechende Beispiele. Der Wert für

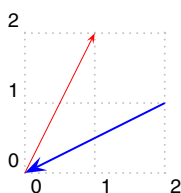
`\pslabelsep` kann über das Längenregister `\pslabelsep` abgefragt werden und wird intern auch von anderen Makros benutzt, beispielsweise von `\psaxes` (siehe Abschnitt 14.1 auf Seite 184).

4.2 `\psline`

Die Syntax für Linien bzw. für polygonartige Linienzüge, die sich durch eine Folge von Koordinaten ergeben, lautet:

```
\psline* [Optionen] {Pfeilart} (x,y)
\psline* [Optionen] {Pfeilart} (x1,y1) (x2,y2) ... (xn,yn)
```

- ▶ Intern werden sämtliche angegebenen Punkte in der umgekehrten Reihenfolge abgearbeitet (LIFO-Prinzip). Dies ist wichtig zu wissen, wenn mit `\pscustom` geschlossene Linien- oder Kurvenzüge erstellt werden sollen.
- ▶ Existiert für `\psline` nur ein Koordinatenpaar, so wird grundsätzlich vom aktuellen Punkt aus zum angegebenen Punkt gezeichnet, wobei der aktuelle Punkt immer auf den Koordinatenursprung (0,0) gesetzt wird, wenn eine `pspicture` Umgebung existiert. Die Sternversionen führen grundsätzlich zu einem geschlossenen Polygonzug, indem vom letzten angegebenen Punkt eine Linie zum ersten gezogen wird. Danach wird der gesamte Bereich mit der Linienfarbe gefüllt.

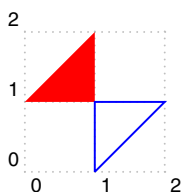


```
\usepackage{pstricks}

\begin{pspicture}[showgrid](2,2)
  \psset{arrowscale=2}
  \psline*[linecolor=red]{->}(1,2)% sinnlos
  \psline*[linecolor=blue]{<-}(2,1)
\end{pspicture}
```

04-02-1

Sind für Sternversionen nur ein oder zwei Punkte angegeben, so ergibt sich eine Linie mit einer komplementären Farbe, die auf dem Bildschirm sichtbar ist, aber im Ausdruck in der Regel nicht zu sehen ist. Wie in obigem Beispiel deutlich zu sehen ist, haben die Angaben zu den Pfeilen ebenfalls keine Wirkung. Dies wird hier ohnehin nur der Vollständigkeit halber gezeigt, denn obiger Fall ist praktisch sinnlos, aber eben möglich.



```
\usepackage{pstricks}

\begin{pspicture}[showgrid](2,2)
  \psline*[linecolor=red](0,1)(1,2)(1,1)
  \psline*[linecolor=blue](1,0)(1,1)(2,1)(1,0)
\end{pspicture}
```

04-02-2

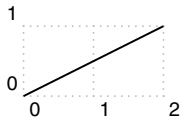
4.3 `\qline`

Dies stellt sozusagen die Minimalversion (**quick line**) von `\psline` dar, denn es werden keine lokalen Optionen ausgewertet und es müssen *grundsätzlich* zwei Punkte angegeben werden:

$$\backslashqline(x_1,y_1)(x_2,y_2)$$

Alle mit `\psset` gesetzten Parameter werden jedoch von `\qline` beachtet, da diese `\psset` Angaben für die aktuelle und tiefer liegende Umgebungen global gelten.

04-03-1




```
\usepackage{pstricks}
\begin{pspicture}[showgrid](2,1)
  \qline(0,0)(2,1)
\end{pspicture}
```

4.4 \pspolygon

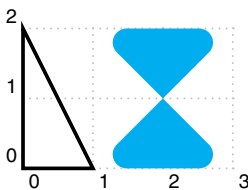
Im Gegensatz zu `\psline` stellt `\pspolygon` grundsätzlich einen *geschlossenen* Kurvenzug dar, wozu eventuell die angegebenen Koordinatenpaare entsprechend ergänzt werden. Eine Angabe von Pfeilen ist daher nicht zulässig beziehungsweise sinnlos.

$$\backslashpspolygon * [Optionen] (x_1,y_1)(x_2,y_2)$$

$$\backslashpspolygon * [Optionen] (x_1,y_1)(x_2,y_2) \dots (x_n,y_n)$$

Werden nur zwei Punkte übergeben, so wird als Start- *und* Endpunkt standardmäßig $(0,0)$ hinzugefügt. Ist der Endpunkt ungleich dem Startpunkt, so wird automatisch vom Endpunkt eine direkte Linie zum Startpunkt gezogen, um den Kurvenzug auf diese Weise zu schließen. Die Sternversion füllt das Innere des Polygonzugs mit der aktuellen Linienfarbe und dem aktuellen Füllmuster. 

04-04-1



```
\usepackage{pstricks}
\begin{pspicture}[showgrid](3,2)
  \pspolygon[linewidth=1.5pt](0,2)(1,0)
  \pspolygon*[linearc=.2,linecolor=cyan,
    swapaxes=true](0,1)(0,3)(2,1)(2,3)
\end{pspicture}
```


4.5 \psframe und \psTextFrame

`\psframe` zeichnet ein horizontal liegendes Rechteck, welches durch zwei diagonal gegenüberliegende Punkte gegeben ist.

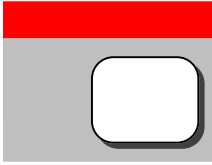
$$\backslashpsframe * [Optionen] (x,y)$$

$$\backslashpsframe * [Optionen] (x_1,y_1)(x_2,y_2)$$

$$\backslashpsTextFrame * [Optionen] (x_1,y_1)(x_2,y_2)\{Text\}$$

Wird für `\psframe` nur ein Punkt übergeben, so wird als zweiter Punkt automatisch $(0,0)$ genommen. Die Sternversion füllt das Innere des Rechtecks mit der aktuellen 

Linienfarbe und dem aktuellen Füllmuster. Für das Rechteck existieren die speziellen Optionen `framearc` und `cornersize`.



```
\usepackage{pstricks}

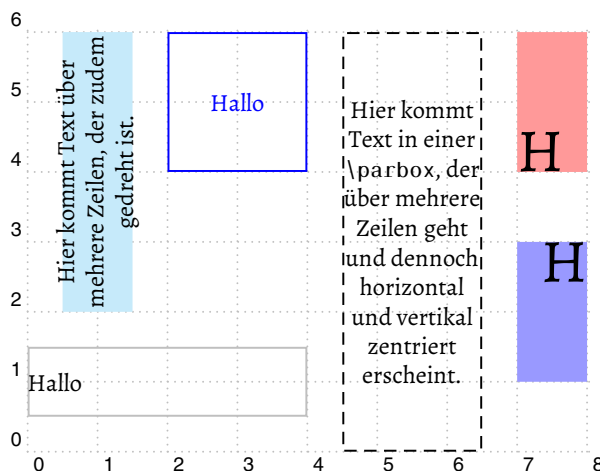
\begin{pspicture}(3,2)
  \psframe*[shadowsize=15pt,linecolor=lightgray,
    shadow=true,shadowcolor=red,shadowangle=90](3,1.75)
  \psframe[fillcolor=white,fillstyle=solid,
    framearc=0.5,shadow=true](1.25,0.25)(2.8,1.5)
\end{pspicture}
```

04-05-1

Das Argument *Text* bei `\psTextFrame` kann keine Zeilenumbrüche aufweisen. Werden diese benötigt, ist eine `\parbox` in der üblichen Weise anzuwenden. Die `ref`-Option erlaubt eine unterschiedliche Anordnung bezogen auf die Box und die `rot`-Option dient zum Rotieren von *Text*. Das Makro selbst benutzt `\psframe` und `\rput`.

```
\usepackage{pstricks}
```

```
\begin{pspicture}[showgrid](0,-0.5)(8,6)
  \psTextFrame[linecolor=lightgray,ref=l](0,0.5)(4,1.5){Hallo}
  \psTextFrame[linecolor=blue](2,4)(4,6){\color{blue}Hallo}
  \psTextFrame*[linecolor=red!40,ref=LB](7,4)(8,6){\Huge H}
  \psTextFrame*[linecolor=blue!40,ref=rt](7,1)(8,3){\Huge H}
  \psTextFrame[linestyle=dashed](4.5,0)(6.5,6){\parbox{2cm}{\centering
    Hier kommt Text in einer \texttt{\textbackslash parbox}, der über mehrere
    Zeilen geht und dennoch horizontal und vertikal zentriert erscheint.}}
  \psTextFrame*[linecolor=cyan!20,rot=90](.5,2)(1.5,6){\parbox{4cm}{\centering
    Hier kommt Text über mehrere Zeilen, der zudem gedreht ist.}}
\end{pspicture}
```



04-05-2

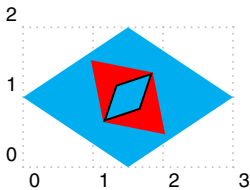
4.6 `\psdiamond`

`\psdiamond` zeichnet eine horizontal liegende Raute, welche durch ihren Mittelpunkt und ihre zwei rechtwinklig aufeinander stehenden Diagonalen gegeben ist, wobei dx und dy jeweils nur die Hälfte der Längen angeben.

```
\psdiamond * [Optionen] (dx,dy)
\psdiamond * [Optionen] (x_M,y_M) (dx,dy)
```

Wird nur ein Punkt übergeben, so wird als Mittelpunkt automatisch der Koordinatenursprung $(0,0)$ angenommen, unabhängig davon, ob dieser Punkt innerhalb oder außerhalb der `PSTricks`-Box liegt. Die Sternversion füllt das Innere der Raute mit der aktuellen Linienfarbe und dem aktuellen Füllmuster. Mit dem Parameter `gangle=Winkel` kann die Raute beliebig rotiert werden.

04-06-1



```
\usepackage{pstricks}

\begin{pspicture}[showgrid](3,2)
  \psdiamond*[linecolor=cyan](1.5,1)(1.5,1)
  \psdiamond*[linecolor=red,gangle=45]%
    (1.5,1)(0.5,0.75)
  \psdiamond[fillstyle=solid,fillcolor=cyan,
    gangle=-45](1.5,)(0.25,0.5)
\end{pspicture}
```

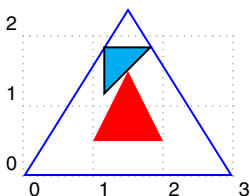
4.7 `\pstriangle`

`\pstriangle` zeichnet ein gleichschenkliges Dreieck, welches durch den Mittelpunkt der Grundlinie (Basis), die Länge dieser Grundlinie und die dazugehörige Höhe gegeben ist, wobei dx und dy die ganze Länge von Basis bzw. Höhe angeben.

```
\pstriangle * [Optionen] (dx,dy)
\pstriangle * [Optionen] (x_M,y_M) (dx,dy)
```

Wird nur ein Punkt übergeben, so wird als Mittelpunkt der Grundlinie automatisch der Koordinatenursprung $(0,0)$ angenommen, unabhängig davon, ob dieser Punkt innerhalb oder außerhalb der `PSTricks`-Box liegt. Die Sternversion füllt das Innere des Dreiecks mit der aktuellen Linienfarbe und dem aktuellen Füllmuster. Mit dem Parameter `gangle=Winkel` kann das Dreieck beliebig rotiert werden.

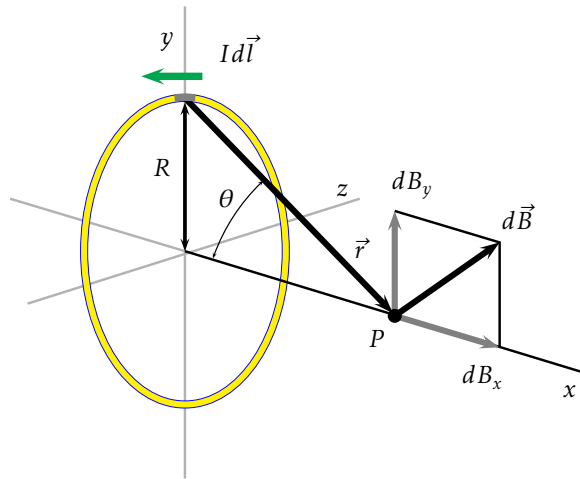
04-07-1



```
\usepackage{pstricks}

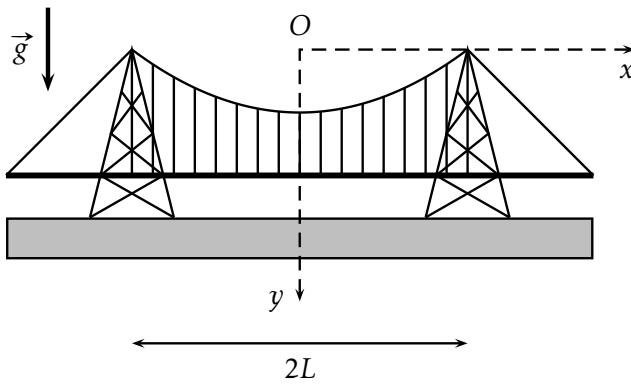
\begin{pspicture}[showgrid](3,2.4)
  \pstriangle[linecolor=blue](1.5,0)(3,2.4)
  \pstriangle*[linecolor=red](1.5,0.5)(1,1)
  \pstriangle[fillstyle=solid,fillcolor=cyan,
    gangle=45](1.5,1.5)(1,0.5)
\end{pspicture}
```

4.8 Beispiele



04-08-1

(Uwe Ziegenhagen)



04-08-2

(François Vandenbrouck)