

# Befehlsübersicht für PSTricks

Uwe Siart

Version 1.07 vom 27. April 2018

## 1 Grundeinstellungen

Bildumgebung

```
\begin{pspicture}[par](P1)(P2)
\end{pspicture}
```

Parameter für die Bildumgebung

```
showgrid = top/bottom/true/false
bgcolor = color
shift = dim oder *
```

Einheitslängen

```
xunit = dim    runit = dim
yunit = dim    unit = dim
```

Mit `unit` werden alle Werte gleichzeitig gesetzt.

Farbdefinition

```
\definecolor[class]{name}{model}{spec}
\colorlet{name}[model]{expr}
```

Dabei sind `spec` eine kommaseparierte Werteliste und `expr` ein Farbausdruck. Mit den Optionen `[dvipsnames]`, `[svgnames]` oder `[x11names]` werden vordefinierte Farbpaletten verfügbar. Weitere Informationen liefert die Dokumentation des Paketes `xcolor`.

Globale Parameter setzen

```
\psset{par}
\newpsobject{name}{par}
\newpsstyle{name}{par}
\addtopsstyle{name}{par}
```

Globale Parameter können lokal durch `[par]` oder `[style = name]` übersteuert werden.

Vollwinkelteilung

```
\radians    \degrees    \degrees[div]
```

Koordinatendarstellung umschalten

```
\NormalCoor    \SpecialCoor
```

Koordinatendarstellungen von (P) (`\SpecialCoor`)

```
(x,y)    (!code)    ([par]node)
(r;a)    (node)    (coor1|coor2)
```

## 2 Grafische Grundelemente

Linien, Polygone, Rahmen

```
\psline[par]{ends}(P0)(P1)...
\psPline[par]{ends}(P0)(P1)(P2)
\pspolygon[par](P0)(P1)...
\psframe[par](P0)(P1)
\psdiamond[par](P0)(P1)
\pstriangle[par](P0)(P1)
```

Bei `\psdiamond` und `\pstriangle` geben (P0) das Zentrum und (P1) die Breite und Höhe an. Eine Drehung um das Zentrum gibt der Parameter `gangle` an. Das Paket `pst-poly` bietet erweiterte Polygon-Makros.

Kreis- und Ellipsenbögen

```
\psarc[par]{ends}(P){rad}{ang}{ang}
\psarcn[par]{ends}(P){rad}{ang}{ang}
\psellipticarc[par]{ends}(P)(a,b){ang}{ang}
\psellipticarcn[par]{ends}(P)(a,b){ang}{ang}
\psarcAB[par]{ends}(P0)(PA)(PB)
\psarcnAB[par]{ends}(P0)(PA)(PB)

\pscircle[par](P){rad}
\pswedge[par](P){rad}{ang}{ang}
\psellipse[par](P)(a,b)
\psRing[par](P){rad1}{rad2}
```

Parameter für Linien

```
linewidth = dim
linecolor = color
linestyle = style
linearc = dim
linecap = 0/1/2
linejoin = 0/1/2
dash = dim1 dim2
dotsep = dim
border = dim
bordercolor = color
doubleline = true/false
doublesep = dim
doublecolor = color
arrows = ends
strokeopacity = num
```

### Mögliche Werte für linestyle

none      solid      dashed  
dotted    symbol

### Mögliche Werte für arrows

\*\*-\*\*      <<->>      <->      \*\*      (-)  
|\*-\*      >>-<<      >-<      o-o      )-(  
oo-oo      cc-cc      c-c      C-C      |-|  
<D-D>      <D<D-D>D>      [-]      ]-[      -

Linienenden können auch in der Form `{ends}` in Linienbefehlen angegeben werden.

### Parameter für Linienenden

arrowsize = *dim num*  
arrowlength = *num*  
arrowinset = *num*  
tbarsize = *dim num*  
bracketlength = *num*

### Punkt an jeder Koordinatenangabe

`\psdot*{par} (P0)`  
`\psdots*{par} (P0) (P1) ...`

### Parameter für Punkte ('var' bedeutet optional)

dotstyle = *style*  
dotsize = *dim 'num'*  
dotscale = *num1 'num2'*  
dotangle = *ang*

### Mögliche Werte für dotstyle (Auswahl)

\*      o      +      x  
asterisk    oplus      otimes      |  
square    diamond    triangle    pentagon  
square\*    diamond\*    triangle\*    pentagon\*

Weitere dotstyles siehe `pst-news06.pdf`.

### Kurven

`\psbezier[par] (P0) ... (P3)`  
`\pscurve[par] (P0) (P1) ...`  
`\psecurve[par] (P0) (P1) ...`  
`\psccurve[par] (P0) (P1) ...`

## 3 Textboxen

### Rahmenboxen

`\psframebox[par]{stuff}`  
`\psdblframebox[par]{stuff}`  
`\psshadowbox[par]{stuff}`  
`\pscirclebox[par]{stuff}`  
`\psovalbox[par]{stuff}`  
`\psdiabox[par]{stuff}`  
`\pstribox[par]{stuff}`  
`\psTextFrame[par] (P0) (P1){stuff}`

### Parameter für Rahmen und geschlossene Pfade

fillstyle = *style*  
fillcolor = *color*  
framearc = *num*  
framesep = *dim*  
hatchwidth = *dim*  
hatchcolor = *color*  
hatchangle = *ang*  
hatchsep = *dim*  
cornersize = relative/absolute  
dimen = inner/middle/outer  
opacity = *num*

### Mögliche Werte für fillstyle

none      solid      eofill      oefill  
hlines    vlines    crosshatch    penrose  
hlines\*    vlines\*    crosshatch\*    penrose\*  
dots

Die Verwendung der \*-Versionen bei Rahmen und Kurven wirkt wie `fillstyle = solid`. Die verwendete Füllfarbe ist in diesem Fall `linecolor`.

## 4 Platzierung von Objekten

### Skalierung

`\psscalebox{num1 num2}{stuff}`  
`\psscaleboxto(P){stuff}`

### Verschiebung und Drehung

`\rput[ref]{rot}(P){stuff}`  
`\multirput[ref]{rot}(P)(a,b){rep}{stuff}`  
`\psrotateright{stuff}`  
`\psrotateleft{stuff}`  
`\psrotatedown{stuff}`

### Drehung um beliebigen Punkt

`\rput{ang}(P){\rput(-P){stuff}}`

### Parameter für eine Nullpunktverschiebung

origin = {*coor*}  
swapaxes = true/false

### Mögliche Werte für ref

*horizontal*      *vertikal*  
l                    t  
r                    b  
                      B

### Beschriftung von Punkten

`\uput{sep}[ang]{rot}(P){stuff}`

### Parameter für Beschriftungen

labelsep = *dim*

## 5 Gitter

Gitterbefehl

`\psgrid(P0) (P1) (P2)`

Gitterparameter

```
gridwidth = dim
gridcolor = color
griddots = num
gridlabels = dim
gridlabelcolor = color
subgriddiv = int
subgridwidth = dim
subgridcolor = color
subgriddots = num
```

## 6 Verschiedenes

Zusammenhängende und geschlossene Pfade

`\pscustom[par]{paths}`

Innerhalb von `\pscustom` ist der letzte Punkt einer Kurve automatisch der erste Punkt der folgenden Kurve.

Parameter für zusammenhängende Pfade

```
linetype = int
liftpen = 0/1/2
```

Einige Befehle zur Verwendung innerhalb `\pscustom`

```
\closepath      \newpath
\reversepath    \translate(P)
\rineto(P)      \curveto(P1) ... (P3)
\fill           \rotate={ang}
\gsave         \scale={num1 ' num2'}
\grestore      \stroke
\lineto(P)     \swapaxes
\moveto(P)     \code{code}
```

Grafiken begrenzen (clipping)

```
\psclip{clipobjects}
...
\endpsclip
```

Schatten

```
shadow = true/false
shadowsize = dim
shadowangle = ang
shadowcolor = color
```

Einige POSTSCRIPT®-Operatoren

```
add      sub      mul      div
abs      neg      mod      dup
sin      cos      atan     sqrt
exp      ln       log      exch
ceiling  floor     round   truncate
```

Ausdrücke in algebraischer Form

`algebraic = true`

Arithmetische Funktionen

```
\pstFPadd{result}{num1}{num2}
\pstFPsub{result}{num1}{num2}
\pstFPMul{result}{num1}{num2}
\pstFPdiv{result}{num1}{num2}
\pstFPMul{result}{num1}{num2}
\pstFPDiv{result}{num1}{num2}
```

Wiederholungen

```
\psLoop{n}{stuff}
\psforeach{var}{list}{stuff}
```

`\the\psLoopIndex` gibt den Wert von  $n$  zurück.

Wichtige Zahlenwerte (gerundet)

```
180°/π = 57,2958°    π/180° = 0,01745 rad/Grad
π = 3,14159          e = 2,71828
```

Die trigonometrischen Funktionen erwarten ihr Argument im Gradmaß. Weitere Operatoren und Konstanten werden vom Paket `pst-math` zur Verfügung gestellt.

## 7 Erweiterungen

**multido**

Wiederholungen

`\multido{variables}{rep}{stuff}`

Mögliche Variablentypen sind *Integer* (`\i`), *Dimension* (`\d`), *Number* (`\n`) und *Real* (`\x`). Ein Dekrement wird in der Form `\nx=5.30+-1.25` angegeben.

Festkommaaddition und -subtraktion

```
\FPadd{num1}{num2}{command}
\FPsub{num1}{num2}{command}
```

Durch diese Befehle wird das Kommando `command` definiert und das Ergebnis darin abgespeichert.

**pst-text**

Text entlang Kurven

`\pstextpath[justify](x,y){path}{text}`

Mögliche Werte für *justify*

l      c      r

Outline-Buchstaben

```
\pscharpath[par]{text}
\pscharclip[par]{text}... \endpscharclip
```

## pst-node

Knoten erzeugen

```
\rnode[ref]{node}{stuff}
\Rnode(P){node}{stuff}
\pnode[xoffset,yoffset](P){node}
\pnodes[xoffset,yoffset](P){node}(P){node}...
\pnodes{name}(P0)(P1)(P2)...
\fnode[par](P){node}
\cnode[par](P){rad}{node}
\Cnode[par](P){node}
\circnode[par]{node}{stuff}
\trinode[par]{node}{stuff}
\ovalnode[par]{node}{stuff}
\cnodeput[par]{ang}(P){node}{stuff}
\dotnode[par](P){node}
\psnode[par](P){node}{stuff}
\psLNode(P1)(P2){num}{node}
\psLCNode(P1){num1}(P2){num2}{node}
```

Knotenverbindungen (Segmentanzahl in Klammern)

```
\ncurve[par]{arrows}{nodeA}{nodeB} (0)
\ncline[par]{arrows}{nodeA}{nodeB} (1)
\ncarc[par]{arrows}{nodeA}{nodeB} (1)
\ncircle[par]{arrows}{node}{rad} (1)
\ncdiag[par]{arrows}{nodeA}{nodeB} (2)
\ncdiag[par]{arrows}{nodeA}{nodeB} (3)
\ncbar[par]{arrows}{nodeA}{nodeB} (3)
\ncangle[par]{arrows}{nodeA}{nodeB} (3)
\ncangles[par]{arrows}{nodeA}{nodeB} (4)
\ncloop[par]{arrows}{nodeA}{nodeB} (5)
\ncbarr[par]{arrows}{nodeA}{nodeB} (5)
```

Punktverbindungen (Segmentanzahl in Klammern)

```
\pccurve[par]{arrows}(P1)(P2) (0)
\pcline[par]{arrows}(P1)(P2) (1)
\pcarc[par]{arrows}(P1)(P2) (1)
\pcdiag[par]{arrows}(P1)(P2) (2)
\pcdiag[par]{arrows}(P1)(P2) (3)
\pcbar[par]{arrows}(P1)(P2) (3)
\pcangle[par]{arrows}(P1)(P2) (3)
\pcangles[par]{arrows}(P1)(P2) (4)
\pcloop[par]{arrows}(P1)(P2) (5)
\pcbarr[par]{arrows}(P1)(P2) (5)
```

Parameter für Knoten und Verbindungen

```
ncurv = num      arcangle = ang
offset = dim     loopsize = dim
arm = dim       [XY]nodesep = dim
angle = ang     radius = dim
lineAngle = ang
```

Die Parameter [XY]nodesep, offset, arm und angle können in den Varianten *parA* und *parB* auch für beide Endknoten separat gesetzt werden. Zusätzlich gelten alle Liniensparameter. Die Verschiebung offset zählt in runits positiv zur linken Seite des Pfades. radius dient als globaler

Parameter für \Cnode. lineAngle bestimmt die Steigung des schrägen Liniensegments von \ncdiag und \ncdiagg.

Knoten beschriften

```
\nput[par]{ang}{node}{stuff}
```

Knoten- und Punktverbindungen beschriften

```
\ncput[par]{stuff}
\naput[par]{stuff}
\nbput[par]{stuff}
```

Die Befehle \lput, \aput, \bput, \Aput, \Bput, \Lput, \Mput und \Rput sind obsolet, werden aber weiter unterstützt.

Parameter für Beschriftungen

```
nrot = rot
npos = num
```

Mit der Angabe nrot=: ang erfolgt die Drehung relativ zur Richtung der Verbindungslinie (häufig nrot=:Ü).

Rahmen um Knoten herum

```
\ncbox[par]{nodeA}{nodeB}
\ncarcbox[par]{nodeA}{nodeB}
```

Knotenkoordinaten auslesen

```
\psGetNodeCenter{node}
```

Ermöglicht die Verwendung von *node.x* und *node.y* innerhalb von POSTSCRIPT®-Code.

## pst-grad

Parameter für Gradientenfüllungen

```
fillstyle = gradient
gradbegin = color
gradend = color
gradlines = int
gradmidpoint = num
gradangle = ang
gradientHSB = true/false
GradientCircle = true/false
GradientScale = num
GradientPos = coor
```

## pst-math

Zusätzliche POSTSCRIPT®-Funktionen

|         |      |       |           |
|---------|------|-------|-----------|
| GAMMA   | SIN  | ASIN  | GAUSS     |
| GAMMALN | COS  | ACOS  | BESSEL_J0 |
| EXP     | TAN  | ATAN  | BESSEL_J1 |
| SINC    | SINH | ASINH | BESSEL_Y0 |
| SEC     | COSH | ACOSH | BESSEL_Y1 |
| COSEC   | TANH | ATANH | BESSEL_Yn |
| COTAN   | ASEC | ACSC  | SIMPSON   |

Die trigonometrischen Funktionen von `pst-math` erwarten ihr Argument im Bogenmaß (rad).

### pst-plot

Achsenkreuz

```
\psaxes[par](P0)(P1)(P2)
  [xlabel,ang][ylabel,ang]
```

Parameter für Achsenkreuze (Auswahl)

```
axesstyle = none/axes/frame/polar
ticks = x/y/all/none
labels = x/y/all/none
comma = true/false
[xy]trigLabels = true/false
trigLabelBase = int
tickstyle = full/top/bottom/inner
ticklinestyle = style
[xy]ticksize = dim1 'dim2'
tickwidth = dim
tickcolor = color
[xy]subticks = int
[xy]subticksize = num
[xy]subtickwidth = dim
[xy]subtickcolor = color
showorigin = true/false
labelsep = dim
logLines = x/y/all/none
[xy]Decimals = int
[xy]logBase = int
[xy]labelFactor = stuff
xyAxes = true/false
xAxis = true/false
yAxis = true/false
```

```
Ox = num Dx = num dx = num
Oy = num Dy = num dy = num
```

`Ox` und `Oy` sind die Startwerte der Nummerierungen im Ursprung. `Dx` und `Dy` sind die Inkremente der Nummerierungen. `dx` und `dy` sind die Abstände der Achsenmarken. Der Parameter `subticks` gibt genau die Anzahl der Teilstriche an, das heißt, dass im Fall einer logarithmischen Achsenteilung (`logLines`) zur Basis 10 eine Zehnerteilung durch `subticks=9` erreicht wird. Durch wiederholten Aufruf von `\psaxes` können mehrere Teilstrichgruppen mit verschiedenen Abständen und Längen erzeugt werden.

Stil der Achsenbezeichnungen

```
\renewcommand{\psxlabel}[1]{commands #1}
\renewcommand{\psylabel}[1]{commands #1}
```

Zusätzliche Achsenmarken

```
\psxTick[par][num]{stuff}
\psyTick[par][num]{stuff}
```

Daten einlesen

```
\readdata{object}{filename}
\savedata{object}{filename}
```

Daten plotten

```
\psfileplot[par]{filename}
\psdataplot[par]{object}
\pslistplot[par]{object}
```

Funktionen plotten

```
\psplot{x1}{x2}{y(x)}
\psparametricplot{t1}{t2}{x(t) y(t)}
```

Allgemeine Plotparameter

```
algebraic = true/false
polarplot = true/false
plotstyle = style
plotpoints = int
showpoints = true/false
yMaxValue = num
yMinValue = num
```

Parameter für `\readdata`

```
nStep = int ignoreLines = int
```

Daten nachverarbeiten (vor `\pslistplot`)

```
\pstScalePoints(num,num){code}{code}
```

Plotparameter für `\pslistplot`

```
nStep = int nStart = int nEnd = int
xStep = int xStart = int xEnd = int
yStep = int yStart = int yEnd = int

plotNoX = int plotNo = int
plotNoMax = int
```

Mögliche Werte für `plotstyle`

|       |         |         |             |
|-------|---------|---------|-------------|
| dots  | line    | polygon | LineToXAxis |
| curve | ecurve  | ccurve  | LineToYAxis |
| bar   | ybar    | values  | xvalues     |
| LSM   | cspline |         |             |

### pst-coil

Spiral- und Zick-Zack-Linien

```
\psCoil[par]{ang1}{ang2}
\pscoil[par]{arrows}(P1)(P2)
\pssin[par]{arrows}(P1)(P2)
\pszigzag[par]{arrows}(P1)(P2)
```

Knotenverbindungen

```
\nccoil[par]{arrows}{nodeA}{nodeB}
\ncsin[par]{arrows}{nodeA}{nodeB}
\nczigzag[par]{arrows}{nodeA}{nodeB}
```

Punktverbindungen

```
\pccoil[par]{arrows}(P1)(P2)
\pcsin[par]{arrows}(P1)(P2)
\pczigzag[par]{arrows}(P1)(P2)
```

Parameter für Spiral- und Zick-Zack-Linien

```
coilwidth = dim      coilinc = ang
coilheight = num     periods = dim|num
coilarm = dim        amplitude = num
coilaspect = ang     ppoints = num
function = code
```

Der Parameter coilarm kann in den Varianten coilarmA und coilarmB auch für beide Endknoten separat gesetzt werden.

**pst-func**

Makros zum Plotten spezieller Funktionen

```
\psBessel[par]{ord}{x1}{x2}
\psPolynomial[par]{x1}{x2}
\psBernstein[par](t1,t2)(i,n)
\psFourier[par]{x1}{x2}
\psSi[par]{x1}{x2}
\psCi[par]{x1}{x2}
```

Makros für numerische Berechnungen

```
\psIntegral[par]{x1}{x2}(a,b){y(x)}
\psConv[par]{x1}{x2}(a,b){f(x)}{g(x)}
\psCumIntegral[par]{x1}{x2}{y(x)}
```

Makros zum Plotten von Wahrscheinlichkeitsdichten

```
\psGauss[par]{x1}{x2}
\psGaussI[par]{x1}{x2}
\psBinomial[par]{N}{p}
\psBinomial[par]{m,N}{p}
\psBinomial[par]{m,n,N}{p}
\psBinomialN[par]{N}{p}
\psPoisson[par]{N}{lambda}
\psPoisson[par]{M,N}{lambda}
\psGammaDist[par]{x1}{x2}
\psChiIIDist[par]{x1}{x2}
\psTDist[par]{x1}{x2}
\psFDist[par]{x1}{x2}
\psBetaDist[par]{x1}{x2}
```

Parameter für \psBessel

```
constI = expr      constII = expr
```

Parameter für \psPolynomial

```
coeff = a0 a1 a2 ...
xShift = num
Derivation = ord
markZeros = true/false
```

Parameter für \psFourier

```
cosCoeff = a0 a1 a2 ...
sinCoeff = b1 b2 b3 ...
```

Parameter für \psIntegral, \psCumIntegral, \psConv und \psGaussI

```
Simpson = int
```

Parameter für \psGauss und \psGaussI

```
sigma = num      mue = num
```

Parameter für \psBetaDist und \psGammaDist

```
alpha = num      beta = num
```

Parameter für \psChiIIDist und \psTDist

```
nue = num
```

**pst-eucl**

Orthogonalprojektion(en) auf eine Gerade

```
\pstProjection{node}{node}{nodes}[nodes]
```

Die Optionen PointSymbol=none und PointName=none unterdrücken die automatische Beschriftung der Ergebnisknoten.

**pstricks-add**

Mehrfachplatzierung

```
\rmultiput[par]{stuff}(P1)(P2)...
```

Drehung um beliebigen Punkt

```
\psrotate[par](P){ang}{stuff}
```

Schnittpunkt

```
\psIntersectionPoint(P1)(P2)(P3)(P4){node}
```

Freihandlinie

```
\pslineByHand[par](P0)(P1)...
```

Tangenten von Punkt P1 an Kreis

```
\psCircleTangents(P1)(P0){rad}
```

Die berechneten Tangentenpunkte sind in den Knoten CircleT1 und CircleT2 abgelegt.

Gemeinsame Tangenten zweier Kreise

```
\psCircleTangents(P1){rad1}(P2){rad2}
```

Tangenten von Punkt P1 an Ellipse

```
\psEllipseTangents(P0)(a,b)(P1)
```

Die berechneten Tangentenpunkte sind in den Knoten EllipseT1 und EllipseT2 abgelegt.



Mediation und Drehung

`\psRelNode[par](P1)(P2){factor}{name}`

Linie relativ zu  $\overline{P_0, P_1}$

`\psRelLine[par](P0)(P1){factor}{name}`

Als *[par]* kann bei `\psRelNode` und `\psRelLine` ein zusätzlicher Drehwinkel angegeben werden. Mit der Option `trueAngle` erscheint genau dieser Winkel, auch dann, wenn `xunit` und `yunit` nicht betragsgleich sind.

Füllung mit zufällig verteilten Punkten

`\psRandom[par](P0)(P1){clippath}`

Parameter für `\psRandom`

`randomPoints = int`    `color = true/false`

### pst-3dplot

Globale Parameter

`SphericalCoor = true/false`

Mit `SphericalCoor = true` wird jedes Koordinatentripel interpretiert als Radius, Längengrad (bezüglich der *x*-Richtung) und Breitengrad.

Achsenkreuz

`\pstThreeDCoor[par]`

Parameter für Achsenkreuze

`xMin = num`    `zMin = num`  
`xMax = num`    `zMax = num`  
`yMin = num`    `Alpha = ang`  
`yMax = num`    `Beta = ang`

Gitter in den Koordinatenebenen

`\pstThreeDPlaneGrid[par](a0,b0)(a1,b1)`

Parameter für Gitter in den Koordinatenebenen

`planeGrid = xy/xz/Yz`  
`subticks = int`  
`planeGridOffset = num`

Platzierung von Objekten

`\pstThreeDPut[par](x,y,z){stuff}`

Linien, Polygone, Rahmen, Quader

`\pstThreeDLine[par]{ends}(P0)(P1)...`  
`\pstThreeDTriangle[par](P0)(P1)(P2)`  
`\pstThreeDSquare[par](P0)(P1)(P2)`  
`\pstThreeDBox[par](P0)(P1)(P2)(P3)`

(*P<sub>n</sub>*) steht für eine 3D-Koordinatenangabe in der Form (*x<sub>n</sub>*, *y<sub>n</sub>*, *z<sub>n</sub>*). Bei `\pstThreeDSquare` und `\pstThreeDBox` bedeuten (P0) eine Ecke und (P1) bis (P3) die Vektoren, die das Rechteck oder den Quader aufspannen.

Kreise und Ellipsen

`\pstThreeDCircle[par](P0)(P1)(P2)`  
`\pstThreeDEllipse[par](P0)(P1)(P2)`

Parameter für Kreise und Ellipsen

`beginAngle = ang`  
`endAngle = ang`

Punkt an der Koordinatenangabe

`\pstThreeDDot[par](x0,y0,z0)`

Funktionen plotten

`\pstplotThreeD[par](x1,x2)(y1,y2){z(x,y)}`

Daten plotten

`\fileplotThreeD[par]{filename}`  
`\dataplotThreeD[par]{object}`  
`\listplotThreeD[par]{object}`

Knoten festlegen

`\pstThreeDNode(x,y,z){node}`

Parameter für Drehungen

`RotX = ang`  
`RotY = ang`  
`RotZ = ang`  
`RotSequence = xyz/xzy/.../quaternion`

### A Voreingestellte Werte

|                                  |                                   |
|----------------------------------|-----------------------------------|
| <code>angle=0</code>             | <code>hatchsep=4pt</code>         |
| <code>arcangle=8</code>          | <code>hatchwidth=0.8pt</code>     |
| <code>arm=10pt</code>            | <code>labelsep=5pt</code>         |
| <code>arrowinset=0.4</code>      | <code>lineararc=0pt</code>        |
| <code>arrowlength=1.4</code>     | <code>linecap=0</code>            |
| <code>arrows=-</code>            | <code>linecolor=black</code>      |
| <code>arrowsize=1.5pt 2</code>   | <code>linestyle=solid</code>      |
| <code>border=0pt</code>          | <code>linewidth=0.8pt</code>      |
| <code>bordercolor=white</code>   | <code>loopsize=1cm</code>         |
| <code>cornersize=relative</code> | <code>ncurv=0.67</code>           |
| <code>dash=5pt 3pt</code>        | <code>nodesep=0pt</code>          |
| <code>dimen=outer</code>         | <code>nrot=0</code>               |
| <code>dotangle=0</code>          | <code>offset=0pt</code>           |
| <code>dotscale=1</code>          | <code>plotpoints=50</code>        |
| <code>dotsep=3pt</code>          | <code>plotstyle=line</code>       |
| <code>dotsize=2pt 2</code>       | <code>radius=0.25cm</code>        |
| <code>dotstyle=*</code>          | <code>runit=1cm</code>            |
| <code>doublecolor=white</code>   | <code>shadow=false</code>         |
| <code>doubleline=false</code>    | <code>shadowangle=-45</code>      |
| <code>fillcolor=white</code>     | <code>shadowcolor=darkgray</code> |

```
fillstyle=none      shadowsize=3pt
framearc=0          showgrid=false
framesep=3pt       subgridcolor=gray
gridcolor=black     subgriddiv=5
griddots=0          subgriddots=0
gridlabelcolor=black subgridwidth=0.4pt
gridlabels=10pt    tbarsize=2pt 5
gridwidth=0.8pt    unit=1cm
hatchangle=45      xunit=1cm
hatchcolor=black   yunit=1cm
```

## B Nomenklatur

|              |  |
|--------------|--|
| <i>(P)</i>   | Punkt in gültiger Koordinatendarstellung |
| <i>(a,b)</i> | Wertepaar (z. B. Halbachsenlängen)       |
| <i>ang</i>   | Winkelangabe                             |
| <i>color</i> | Farbausdruck                             |
| <i>dim</i>   | absolute oder relative Längenangabe      |
| <i>ends</i>  | Angaben zu den Linienenden               |
| <i>int</i>   | ganzzahliger Wert                        |
| <i>model</i> | Farbmodell (z. B. rgb)                   |
| <i>name</i>  | Objektname                               |
| <i>node</i>  | Knotenname                               |
| <i>num</i>   | numerischer Wert                         |
| <i>par</i>   | Optionenliste in Keyval-Syntax           |
| <i>paths</i> | Sequenz von Pfaden                       |
| <i>rad</i>   | Kreisradius                              |
| <i>ref</i>   | Bezugspunkt                              |
| <i>rep</i>   | Anzahl der Wiederholungen                |
| <i>rot</i>   | Drehwinkel                               |
| <i>stuff</i> | Text- oder Grafikobjekte                 |
| <i>style</i> | Name eines vordefinierten Stils          |
| <i>text</i>  | Text                                     |

---

Copyright © 2003–2018 by Uwe Siart <uwe@siart.de>

This material may be distributed only subject to the terms and conditions set forth in the *Open Publication License*, v1.0 or later (the latest version is presently available at <http://www.opencontent.org/openpub/>).

Diese Befehlsübersicht enthält eine Auswahl häufig verwendeter PSTricks-Befehle zur schnellen Referenz. Sie umfasst *nicht* den gesamten Befehls- und Parametervorrat von PSTricks oder seiner Erweiterungen und ersetzt vor allem nicht die Lektüre der ausführlichen PSTricks-Dokumentation. Für weitere Informationen siehe <http://PSTricks.tug.org/> und

Voß, H.: *PSTricks*. Grafik mit PostScript für T<sub>E</sub>X und L<sup>A</sup>T<sub>E</sub>X. 6. Aufl. Berlin : Dante e. V. und Lehmanns Fachbuchhandlung, 2011

PostSCRIPT<sup>®</sup> is a registered trademark of Adobe Systems Incorporated.