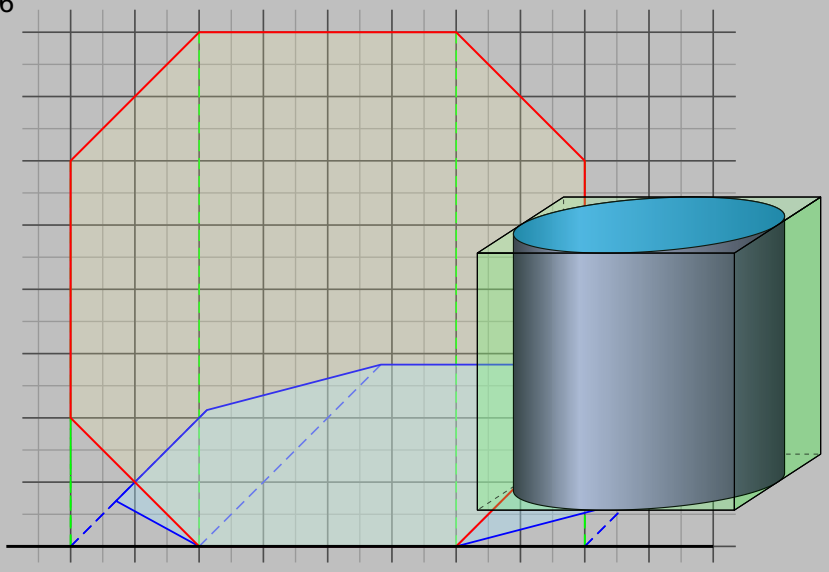
PSTricks

pst-perspective

Plotting the perspective view of a point; v.1.05

February 4, 2016



Package author(s):
Thomas Söll

Contents

pst-perspective loads by default the following packages: pst-xkey, and, of course pstricks. All should be already part of your local TEX installation. If not, or in case of having older versions, go to http://www.CTAN.org/ and load the latest version.

Thanks to:
Jürgen Gilg
Herbert Voß

1 Introduction

On the research to draw perspectives of geometrical objects for my classes with PSTricks, I couldn't find a package that fitted all my needs.
For my needs, it should be possible to draw the perspective (an orthogonal parallel projection) with an arbitrarily chosen angle and a variable shortening factor. The setup for the points should alternatively be possible with cartesian or polar coordinates in two dimensions.
The package `pst-solides3d` wasn't useful, cause it uses central projection.
The package `pst-3dplot` allows parallel projection, and a shortening factor in one of the dimensions can eventually be chosen, e. g. with xThreeDunit, but all points need to be setup with three coordinates. I am not sure about an easy handling of setting up polar coordinates or giving an independent angle for the projection.
With the macro `\ThreeDput` from the package `pst-3d`, only planes and lines in three dimensions are realizable—a third coordinate is needed. For my needs, this didn't seem to be too practicable.
I think that with some of the already existing PSTricks packages a realisation of perspectives in the wanted form could have surely been managed. I couldn't see this however and wanted some simple macros, without loading mighty packages.
To make a long story short: I decided to write a small PSTricks package, that will fit all my needs. This package contains only four small macros, so unwanted crashes with other PSTricks packages are not awaited.

2 General

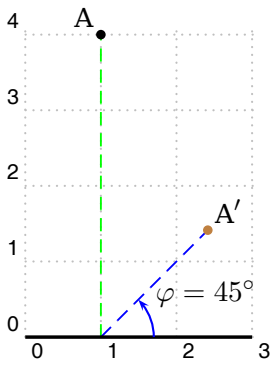
The described macros should help you to easily draw a perspective of a geometrical object like often used in school.
There are three macros, the first two called `\pstransTS` and `\pstransTSX` are used to draw a perspective of a geometrical object laying in the x, y-plane and transforms every vertex. The points first get projected orthogonally to the base line (parameter `base`), parallel to the x-axis, and therefrom they get transformed with an arbitrarily chosen angle measured from the positive x-axis (parameter `phi`) and shortened (multiplied with the shortening factor `vkf`). You then can use these transformed points with its given node names. It is similar with `\pstransTSX`, but the difference is, that the points are projected orthogonally onto a base line (Parameter `base`) parallel to the y-axis, and therefrom they get transformed with an arbitrarily chosen angle (parameter `phi`) and shortened (multiplied with the shortening factor `vkf`). These two cases correspond to a projection of an object of the drawing plane into the x-y-plane or x-z-plane.
The third macro `\pstransTSK` only shifts the points under a given angle `phi` and shortens the initial distance to the base line with the factor `vkf`. This macro is perfect, if the edges that need to be transformed, lay in the x, y-plane and the transformed edges lead to the back, orthogonal to the x, y-plane. Typical examples are the perspectives of a cuboid, square or an upright prism, where the base is in the x, y-plane.
Both macros only generate the nodes for the transformed points, but the points itselves are neither drawn nor labeled. This must be done e. g. with `\psdot` and `\uput`. With

booleans we can visualize the subsidiary lines, which show the orthogonal projection onto the base line and from there to the calculated transformed point.
The subsidiary lines can be drawn in different colors, line styles and line widths with some additional optional parameters.

3 The macro \pstransTS

\pstransTS [Options] $(x_\mathrm{A}, y_\mathrm{A})$ {A}{A'}

The macro \pstransTS [Options] $(x_\mathrm{A}, y_\mathrm{A})$ {A}{A'} uses parentheses for the coordinates of the point. These coordinates can be setup in the usual PSTricks ways. The second argument in curly braces is the node name that is now given for this point. The new calculated point gets the node name to be entered as third argument and is needed to be enclosed in curly braces as well.
In the following example the point $\mathrm{A}(1|4)$ is firstly mapped onto the x-axis (visualized by the green dashed line) and from there a node named A' is generated with a projection angle $\varphi = 45°$ and with half of its initial length (vkf=0.5).

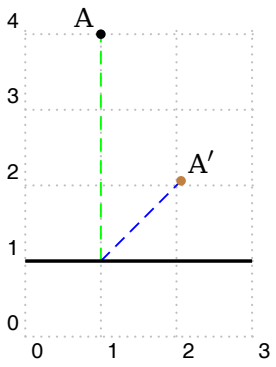



```
\begin{pspicture}[showgrid=true,shift=-4.9](0.5,-0.5)(3,4.4)
{\psset{translineA=true,translineB=true}
\pnode(0,\ba){O}%              \ba gives the y-value of the base line
\pstransTS(1,4){A}{A'}
}
\pcline[linewidth=1.3pt](0,0|O)(3,0|O)% draw base line
\psdot(A)%                     draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\pstMarkAngle[LabelSep=1.5,MarkAngleRadius=0.7,linecolor=blue,arrows
    =->]{3,\ba}{A|O}{A'}{$\varphi=45^{\circ}$}% draw and label the angle
\end{pspicture}
```

3.1 Choice of the base line

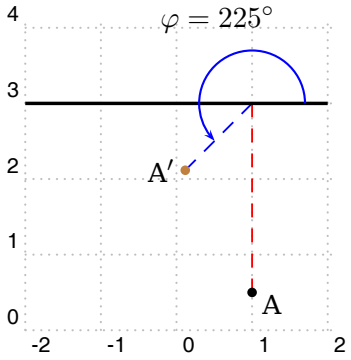
`\pstransTS[ base=... ]`

We can define the y-value of the base line with `\pstransTS[ base=... ]`. The default value is `base=0` . Using `base=1` we achieve, that the point is projected to a parallel to the x-axis with the equation $y = 1$. This case is shown in the following example.




```
\begin{pspicture}[showgrid=true,shift=-4.9](0.5,-0.5)(3,4.4)
{\psset{translineA=true,translineB=true,base=1}
\pstransTS[base=1](1,4){A}{A'}
\pnode(0,\ba){O}
}
\pcline[linewidth=1.3pt](0,0|O)(3,0|O)% draw base line
\psdot(A)%                          draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\end{pspicture}
```

Is the point to be projected below the base line, as shown in the example below ($\mathrm{A}(1|0.5)$ with `base=3` ; $3 > 0.5$), it is not projected with the angle φ but with the angle $180° + \varphi$. If the angle $\varphi = 45°$ is chosen, then we get a resulting angle of $225°$.

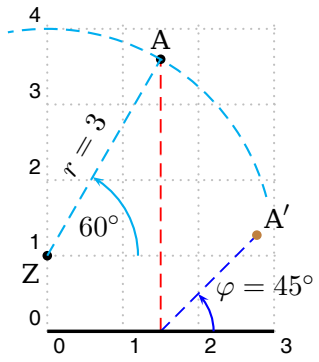



```
\begin{pspicture}[showgrid=true,shift=-4.9](-1.5,-0.5)(2,4.4)
{\psset{base=3,translineA=true,translineB=true,linestyle=dashed,
    transAcolor=red,transBcolor=blue,dash=5pt 4pt}
\pnode(0,\ba){O}
\pstransTS(1,0.5){A}{A'}
}
\pcline[linewidth=1.3pt](-2,0|O)(2,0|O)
\psdot(A)
\uput{4pt}[-30]{0}(A){$\text{A}$}
\psdot[linecolor=brown](A')
\uput{4pt}[180]{0}(A'){$\text{A}'$}
\pstMarkAngle[LabelSep=1.2,MarkAngleRadius=0.7,linecolor=blue,arrows
    =->]{3,\ba}{A|O}{A'}{$\varphi=225^{\circ}$}
\end{pspicture}
```

3.2 Shifting the origin

`\pstransTS[ originT={point} ]`

If we like to give points relative to a certain point, we can setup this point with `originT={Z}` . The syntax of this point is the usual PSTricks syntax, but it needs to be enclosed in curly braces and the parentheses are omitted. Especially with the usage of polar coordinates this can be profitable. A typical example is the drawing of a regular polygon. The vertices are mostly setup in polar coordinates so that we can rotate the polygon quite easily. For radius and angle we choose Z as reference point. As shown in later example, it is no big deal to generate an upright prism as well as an oblique prism.
Note, that the base line is shifted as well when shifting of the origin. With `originT={2,3}` we get the base line $y = 3$. With the additional setting of `base=-2` we additionally shift the base line two units down and finally get $y = 1$.
In the following example, we start from Z$(0|1)$ with a radius of 3 and an angle of $60°$ and a node named A will be generated. The point A will now be mapped onto the x-axis (`base=-1` , due to the base was already shifted with the choice of Z which itself was shifted one unit upwards, see red line) and from there a node named A′ is generated with the angle $\varphi = 45°$ and with half of its initial length (`vkf=0.5`).



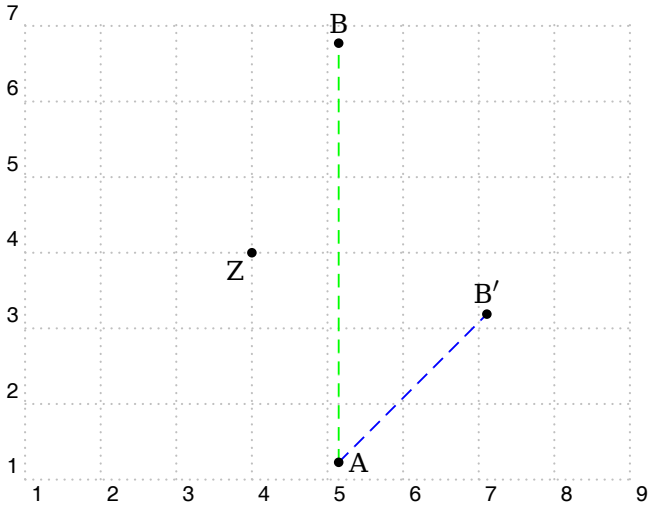

```
\begin{pspicture}[showgrid=true,shift=-4.9](0.5,-0.5)(3,4.4)
{\psset{base=-1,translineA=true,translineB=true,transAcolor=red,
    transBcolor=blue}
\pnode(0,0){O}%              \ba gives the y-value of the base line
\pnode(0,1){Z}%
\pnode(4,0|Z){W1}%
\pstransTS[originT={Z}](3;60){A}{A'}}
\pcline[linewidth=1.3pt](0,0|O)(3,0|O)% draw base line
\psdot(A)%                    draw point A
\uput{4pt}[90]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\psdot(Z)%                    draw point Z
\uput{4pt}[225]{0}(Z){$\text{Z}$}% label point Z
\psarc[linestyle=dashed,linecolor=cyan](Z){3}{15}{100}
\pcline[linecolor=cyan,linestyle=dashed](Z)(A)
\naput[nrot=:U]{$r=3$}
\pstMarkAngle[LabelSep=0.8,MarkAngleRadius=1.2,linecolor=cyan,arrows=->]{
    W1}{Z}{A}{$60^{\circ}$}% draw and label angle
\pstMarkAngle[LabelSep=1.5,MarkAngleRadius=0.7,linecolor=blue,arrows
    =->]{3,0}{A|O}{A'}{$\varphi=45^{\circ}$}% draw and label angle
\end{pspicture}
```

3.3 The base given by a point

```
\pstransTS[LowPoint=true ,LowP={point} ]
```

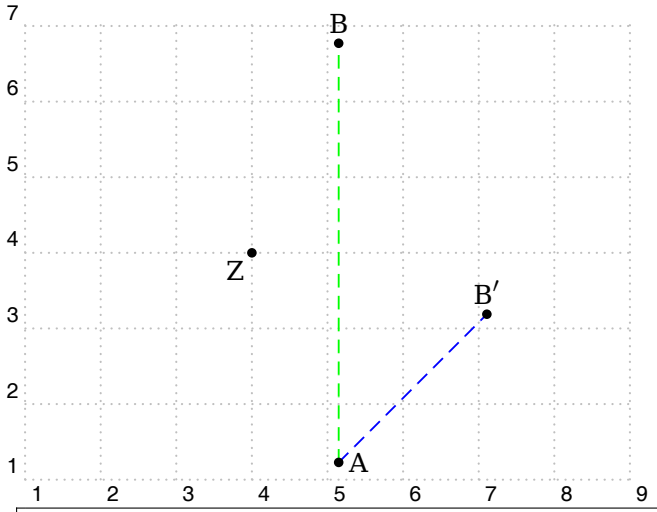
There might arise some difficulties in polar coordinates concerning the y-value of the point through which the base line should go, cause it is not explicitly known. This value for the base however can be calculated easily in most cases using RPN.
For more complex points, the described way might be quite complicated. With the options `LowPoint=true` and `LowP=point` the base is setup through this point and an explicitly given base is ignored.
Some examples will be visualized in the following examples. Starting from point Z$(4|4)$ we setup a point relatively to Z in polar coordinates. Should the base now go through this point (the transformed point matches with this point), some trigonometrical relations might help. We get however the same result when using the options `LowPoint=true` and `LowP=point` .

```
base=292.5 360 sub sin 3 mul
```




```
\begin{pspicture}[showgrid=true](1,0.8)(8.5,7)
%------ Vertices of the octagon -------------
{\pnode(4,4){Z}
\psset{originT=Z,base=292.5 360 sub sin 3 mul,translineA=true,translineB=true}
\pstransTS(3;292.5){A}{A'}
\pstransTS(3;67.5){B}{B'}
\psdot(A)%                      draw point A
\uput{4pt}[0]{0}(A){$\text{A}$} % label point A
\psdot(B)%                      draw point B
\uput{4pt}[90]{0}(B){$\text{B}$} % label point B
\psdot(B')%                     draw point B'
\uput{4pt}[90]{0}(B'){$\text{B}'$} % label point B'
\psdot(Z)%                      draw point Z
\uput{4pt}[225]{0}(Z){$\text{Z}$}% label point Z
}
\end{pspicture}
```

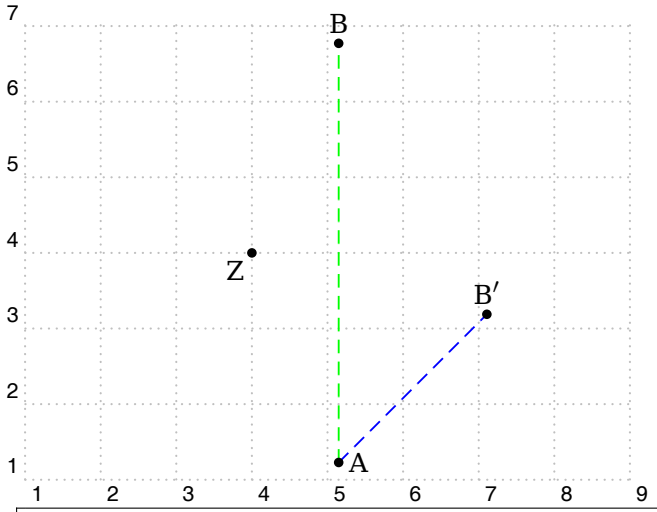
LowPoint=true ,LowP={3;292.5}




```
\begin{pspicture}[showgrid=true](1,0.8)(8.5,7)
%------ Vertices of the octagon -------------
{\pnode(4,4){Z}
\psset{originT=Z,LowPoint=true,LowP={3;292.5},translineA=true,translineB=true}
\pstransTS(3;292.5){A}{A'}
\pstransTS(3;67.5){B}{B'}
\psdot(A)%                      draw point A
\uput{4pt}[0]{0}(A){$\text{A}$} % label point A
\psdot(B)%                      draw point B
\uput{4pt}[90]{0}(B){$\text{B}$} % label point B
\psdot(B')%                     draw point B'
\uput{4pt}[90]{0}(B'){$\text{B}'$} % label point B'
\psdot(Z)%                      draw point Z
\uput{4pt}[225]{0}(Z){$\text{Z}$}% label point Z
}
\end{pspicture}
```

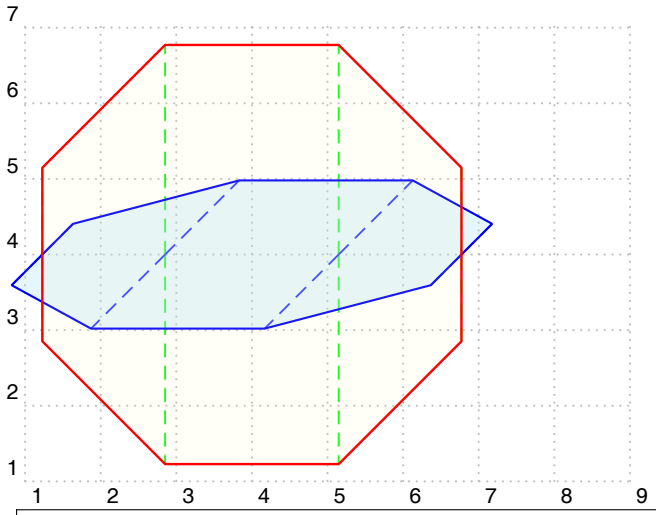
It must be pointed out, that with the definition of `\pnode(r;phi){point}` followed by `LowP=point` , the base is not shifted, which however only becomes noticeable, when the origin was shifted. In this case we write `\rput(origin){\pnode(r;phi){point}}`. Referring to the previous example, this looks as follows.

`\pnode(4,4)Z\rput(Z){\pnode(3;292.5){LP}}` `originT=Z ,LowPoint=true ,LowP={LP}`



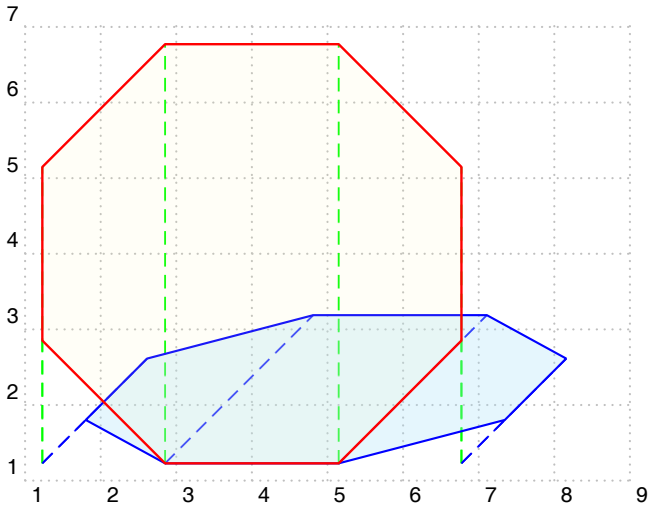

```
\begin{pspicture}[showgrid=true](1,0.8)(8.5,7)
%------ Vertices of the octagon -------------
{\pnode(4,4){Z}
\rput(Z){\pnode(3;292.5){LP}}
\psset{originT=Z,LowPoint=true,LowP={LP},translineA=true,translineB=true}
\pstransTS(3;292.5){A}{A'}
\pstransTS(3;67.5){B}{B'}
\psdot(A)%                    draw point A
\uput{4pt}[0]{0}(A){$\text{A}$} % label point A
\psdot(B)%                    draw point B
\uput{4pt}[90]{0}(B){$\text{B}$} % label point B
\psdot(B')%                   draw point B'
\uput{4pt}[90]{0}(B'){$\text{B}'$} % label point B'
\psdot(Z)%                    draw point Z
\uput{4pt}[225]{0}(Z){$\text{Z}$}% label point Z
}
\end{pspicture}
```

A typical example is the regular octagon, where the vertices can easily be given in polar coordinates. Without changing the base explicitly, it goes through the center.




```
\begin{pspicture}[showgrid=true](1,0.8)(8.5,7)
\psset{linejoin=2}
%------ Vertices of the octagon -------------
{\pnode(4,4){Z}
\psset{originT=Z,translineA=true,translineB=true}
\pstransTS(3;22.5){A}{A'}
\pstransTS(3;67.5){B}{B'}
\pstransTS(3;112.5){C}{C'}
\pstransTS(3;157.5){D}{D'}
\pstransTS(3;202.5){E}{E'}
\pstransTS(3;247.5){F}{F'}
\pstransTS(3;292.5){G}{G'}
\pstransTS(3;337.5){H}{H'}
}
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue](A')(B')(C')(D')(E
    ')(F')(G')(H')
\pspolygon[fillstyle=solid,fillcolor=yellow!40,opacity=0.1,linewidth=0.9pt,linecolor=red](A
    )(B)(C)(D)(E)(F)(G)(H)
\end{pspicture}
```

If we want to have the base through the lower points, we can calculate the base or use a suitable option.

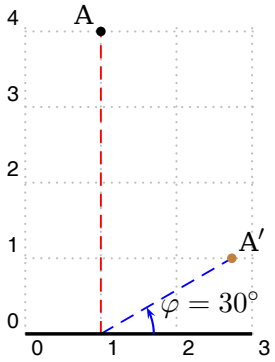



```
\begin{pspicture}[showgrid=true](1,0.8)(8.5,7)
\psset{linejoin=2}
%------ Vertices of the octagon -------------
{\pnode(4,4){Z}
\rput(Z){\pnode(3;292.5){LP}}
\psset{originT=Z,LowPoint=true,LowP={LP},translineA=true,translineB=true}
\pstransTS(3;22.5){A}{A'}
\pstransTS(3;67.5){B}{B'}
\pstransTS(3;112.5){C}{C'}
\pstransTS(3;157.5){D}{D'}
\pstransTS(3;202.5){E}{E'}
\pstransTS(3;247.5){F}{F'}
\pstransTS(3;292.5){G}{G'}
\pstransTS(3;337.5){H}{H'}
}
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue](A')(B')(C')(D')(E
    ')(F')(G')(H')
\pspolygon[fillstyle=solid,fillcolor=yellow!40,opacity=0.1,linewidth=0.9pt,linecolor=red](A
    )(B)(C)(D)(E)(F)(G)(H)
\end{pspicture}
```

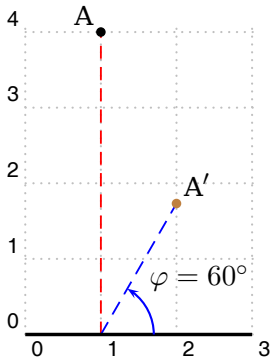
3.4 Angle of projection

`\pstransTS[ phi=30 ].`

The angle of projection is $45°$ by default. If you like another angle e. g. $30°$, use the optional argument `\pstransTS[ phi=30 ]`.




```
\begin{pspicture}[showgrid=true,shift=-4.6](0.5,-0.5)(3,4.1)
{\psset{phi=30,base=0,translineA=true,translineB=true,transAcolor=red,
    transBcolor=blue}
\pnode(0,\ba){O}%              \ba gives the y-value of the base line
\pstransTS(1,4){A}{A'}
}
\pcline[linewidth=1.3pt](0,0|O)(3,0|O)% draw base line
\psdot(A)%                     draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\pstMarkAngle[LabelSep=1.5,MarkAngleRadius=0.7,linecolor=blue,arrows
    =->]{3,\ba}{A|O}{A'}{$\varphi=30^{\circ}$}% draw and label angle
\end{pspicture}
```

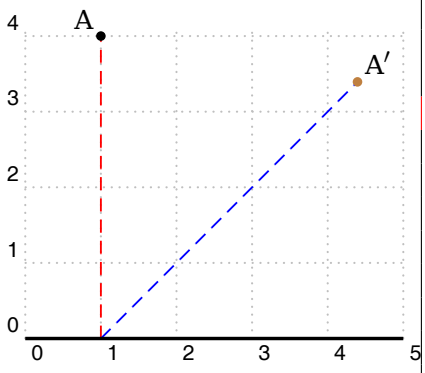



```
\begin{pspicture}[showgrid=true,shift=-4.6](0.5,-0.5)(3,4.1)
{\psset{phi=60,base=0,translineA=true,translineB=true,transAcolor=red,
    transBcolor=blue}
\pnode(0,\ba){O}%              \ba gives the y-value of the base line
\pstransTS(1,4){A}{A'}
}
\pcline[linewidth=1.3pt](0,0|O)(3,0|O)% draw base line
\psdot(A)%                     draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\pstMarkAngle[LabelSep=1.5,MarkAngleRadius=0.7,linecolor=blue,arrows
    =->]{3,\ba}{A|O}{A'}{$\varphi=60^{\circ}$}% draw and label angle
\end{pspicture}
```

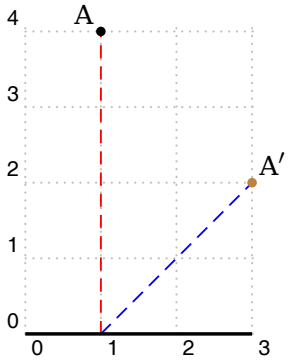
3.5 Shortening factor

`\pstransTS[ vkf=0.5 ]`.

The shortening factor is `vkf=0.5` by default. This means that the distance from the orthogal projected point to the calculated point only has half the length. A possible example is `\pstransTS[ vkf=1.2 ]`. We are allowed to calculate the shortening factor in RPN as in the following example:
`\pstransTS[ vkf=2 sqrt 2 div ]`. The shortening factor is then approximately 0.707.



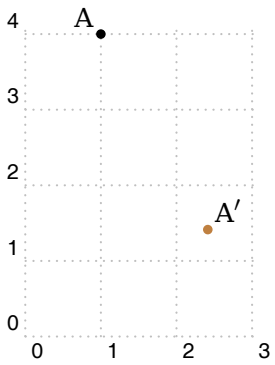

```
\begin{pspicture}[showgrid=true,shift=-4.9](0.5,-0.5)(5,4.4)
{\psset{vkf=1.2,translineA=true,translineB=true,transAcolor=red,
    transBcolor=blue}
\pnode(0,\ba){O}%              \ba gives the y-value of the base
    line
\pstransTS(1,4){A}{A'}
}
\pcline[linewidth=1.3pt](0,0|O)(5,0|O)% draw base line
\psdot(A)%                     draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\end{pspicture}
```



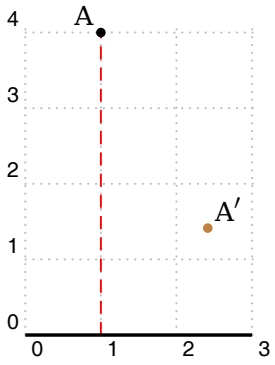

```
\begin{pspicture}[showgrid=true,shift=-4.9](0.5,-0.5)(3,4.4)
{\psset{vkf=0.5 sqrt,base=0,translineA=true,translineB=true,transAcolor=
    red,transBcolor=blue}
\pnode(0,\ba){O}%              \ba gives the y-value of the base line
\pstransTS(1,4){A}{A'}
}
\pcline[linewidth=1.3pt](0,0|O)(3,0|O)% draw base line
\psdot(A)%                     draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\end{pspicture}
```

3.6 Subsidiary lines and their options

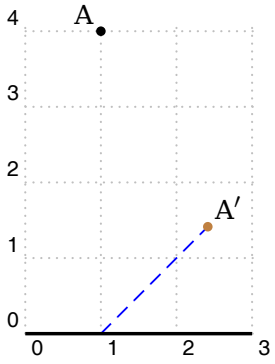
With the two booleans `translineA=true/false` and `translineB=true/false` the subsidiary lines can be shown or not. Their defaults are `translineA=false` and `translineB=false` .




```
\begin{pspicture}[showgrid=true,shift=-4.6](0.5,-0.5)(3,4.1)
\pstransTS(1,4){A}{A'}
\psdot(A)%                          draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\end{pspicture}
```



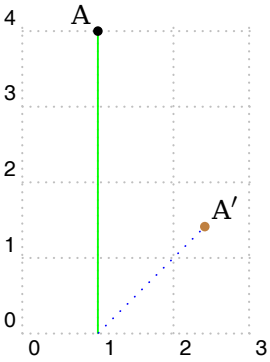

```
\begin{pspicture}[showgrid=true,shift=-4.6](0.5,-0.5)(3,4.1)
{\psset{translineA=true,translineB=false,transAcolor=red,transBcolor=
    blue}
\pnode(0,\ba){O}%               \ba gives the y-value of the base line
\pstransTS(1,4){A}{A'}
}
\pcline[linewidth=1.3pt](0,0|O)(3,0|O)% draw base line
\psdot(A)%                          draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\end{pspicture}
```



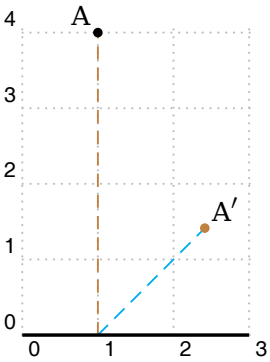

```
\begin{pspicture}[showgrid=true,shift=-4.6](0.5,-0.5)(3,4.1)
{\psset{translineA=false,translineB=true,transAcolor=red,transBcolor=
    blue}
\pnode(0,\ba){O}%               \ba gives the y-value of the base line
\pstransTS(1,4){A}{A'}
}
\pcline[linewidth=1.3pt](0,0|O)(3,0|O)% draw base line
\psdot(A)%                          draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\end{pspicture}
```

Every of the subsidiary lines can individually be setup with three properties: line width, line color and line style. Therefore we define the following options with its defaults.

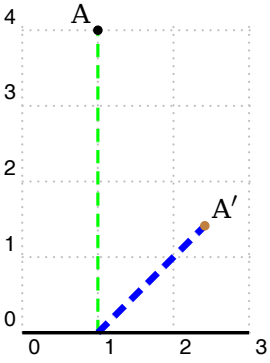
```
\pstransTS[transAlinewidth=0.7pt],
\pstransTS[transAcolor=green],
\pstransTS[transAlinestyle=dashed],
\pstransTS[transBlinewidth=0.7pt],
\pstransTS[transBcolor=blue],
\pstransTS[transBlinestyle=dashed]
```




```
\begin{pspicture}[showgrid=true,shift=-4.6](0.5,-0.5)(3,4.1)
\pstransTS[translineA=true,translineB=true,transAlinestyle=solid,
    transBlinestyle=dotted](1,4){A}{A'}
\psdot(A)%                          draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\end{pspicture}
```




```
\begin{pspicture}[showgrid=true,shift=-4.6](0.5,-0.5)(3,4.1)
{\psset{translineA=true,translineB=true,transAcolor=brown,transBcolor=
    cyan}
\pnode(0,\ba){O}%                   \ba gives the y-value of the base line
\pstransTS(1,4){A}{A'}
}
\pcline[linewidth=1.3pt](0,0|O)(3,0|O)% draw base line
\psdot(A)%                          draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\end{pspicture}
```

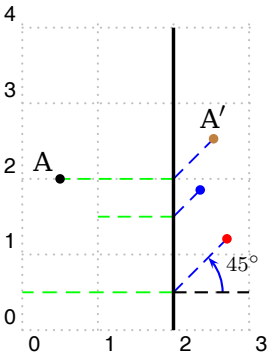



```
\begin{pspicture}[showgrid=true,shift=-4.6](0.5,-0.5)(3,4.1)
{\psset{translineA=true,translineB=true,transAlinewidth=1.2pt,
    transBlinewidth=2.5pt}
\pnode(0,\ba){O}%                   \ba gives the y-value of the base line
\pstransTS(1,4){A}{A'}
}
\pcline[linewidth=1.3pt](0,0|O)(3,0|O)% draw base line
\psdot(A)%                          draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\uput{4pt}[45]{0}(A'){$\text{A}'$}% label point A'
\end{pspicture}
```

4 The macro \pstransTSX

\pstransTSX [Options] (x_A, y_A) {A}{A'}

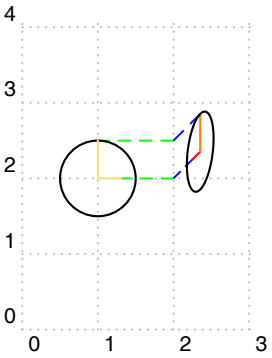
The macro \pstransTSX [Options] (x_A, y_A) {A}{A'} is used like \pstransTS. Also the parameters have the same names. The only difference is, that the points are projected on a line parallel to the y-axis.
This case we use, when we want to transform points on the lateral face of the perspective. Therefore, the lateral face needs to spread orthogonally backwards related to the x, y-plane.
In the following example, the point $A(0.5|2)$ is projected orthogonally on the line $x = 2$ (see green dashed line) und from there with an angle of $\varphi = 45°$ and with a shortening factor of a (vkf=0.5) a node named A′ is generated.




```
\begin{pspicture}[showgrid=true](0.5,-0.5)(3,4.4)
{\psset{translineA=true,translineB=true,base=2,symline=0,symX=false}
\pnode(2,0.5){O}%
\pstransTSX(0.5,2){A}{A'}
\pstransTSX(1,1.5){B}{B'}
\pstransTSX(0,0.5){C}{C'}
}%
\pcline[linewidth=1.3pt](2,0)(2,4)% draw base line
\psdot(A)%                        draw point A
\uput{4pt}[135]{0}(A){$\text{A}$} % label point A
\psdot[linecolor=brown](A')%  draw point A'
\psdot[linecolor=blue](B')%   draw point B'
\psdot[linecolor=red](C')%    draw point C'
\uput{4pt}[90]{0}(A'){$\text{A}'$}% label point A'
\pnode(3,0.5){P}%
\pstMarkAngle[LabelSep=1.0,MarkAngleRadius=0.65,linecolor=blue,arrows
    =->]{P}{O}{C'}{$\scriptstyle 45^{\circ}$}% draw and label angle
\pcline[linestyle=dashed](O)(P)
\end{pspicture}
```

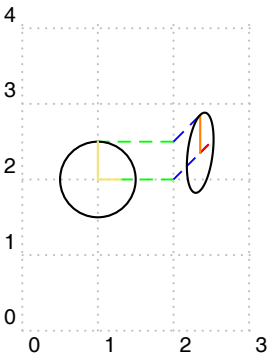
4.1 Symmetry of the transformation

As shown in the example above the points laying in the object left to the base are transformed further away than those laying right. This might disturb, if we want to transform points onto the lateral face. The following example will discuss this problem. It is three o'clock, after the transformation onto the lateral face it is nine o'clock. (Note: base=1 means, that the base is $x = 2$, cause originT={1,2} shifts the origin, from which the base is counted positive to the right.)



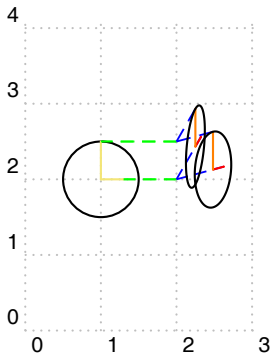

```
\begin{pspicture}[showgrid=true](0.5,-0.5)(3,4.4)
\pnode(1,2){UZ}
{\psset{originT={UZ},base=1,symX=false}
\pstransTSX(0,0){U1}{U1'}
\pstransTSX[translineA=true,translineB=true](0.3;0){zg1}{zg1'}
\pstransTSX[translineA=true,translineB=true](0.5;90){zg2}{zg2'}
%-----------------------------------------------------
\pscircle(UZ){0.5}
\psline[linecolor=yellow!90!black!70]{c-c}(U1)(zg1)
\psline[linecolor=yellow!90!black!70]{c-c}(U1)(zg2)
%-----------------------------------------------------
\psline[linecolor=red]{c-c}(U1')(zg1')
\psline[linecolor=orange]{c-c}(U1')(zg2')
\multido{\i=0+5,\n=5+5}{72}{%
\pstransTSX(0.5;\i){A\i}{A'\i}
\pstransTSX(0.5;\n){B\n}{B'\n}
\psline(A'\i)(B'\n)}
}
\end{pspicture}
```

A corresponding symmetrical transformation we get with `symX=true,symline=0` which is the default. (Note as well, that `symline=0` gives the symmetry axis relatively to `originT={UZ}` . Thus, the symmetry axis is $x = 1$.)




```
\begin{pspicture}[showgrid=true](0.5,-0.5)(3,4.4)
\pnode(1,2){UZ}
{\psset{originT={UZ},base=1,symX=true,symline=0}
\pstransTSX(0,0){U1}{U1'}
\pstransTSX[translineA=true,translineB=true](0.3;0){zg1}{zg1'}
\pstransTSX[translineA=true,translineB=true](0.5;90){zg2}{zg2'}
%-----------------------------------------------------
\pscircle(UZ){0.5}
\psline[linecolor=yellow!90!black!70]{c-c}(U1)(zg1)
\psline[linecolor=yellow!90!black!70]{c-c}(U1)(zg2)
%-----------------------------------------------------
\psline[linecolor=red]{c-c}(U1')(zg1')
\psline[linecolor=orange]{c-c}(U1')(zg2')
\multido{\i=0+5,\n=5+5}{72}{%
\pstransTSX(0.5;\i){A\i}{A'\i}
\pstransTSX(0.5;\n){B\n}{B'\n}
\psline(A'\i)(B'\n)}
}
\end{pspicture}
```

Another effect with the macro `\pstransTSX` can be achieved by the parameter `deltaphi=...` with which we can increase or decrease the projection angle for some objects locally and leaving the setting of `phi=...` globally. `deltaphi=...` adds/subtracts the given projection angle by that value. The result looks like an object swinging out the lateral face.

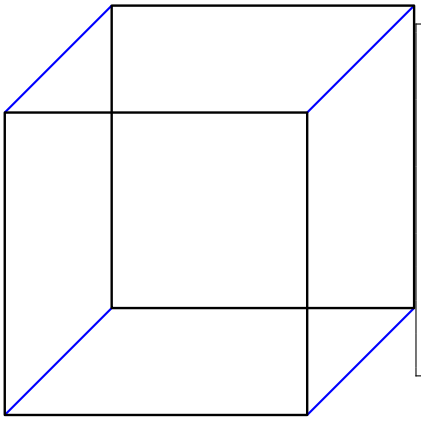



```
\begin{pspicture}[showgrid=true](0.5,-0.5)(3,4.4)
\pnode(1,2){UZ}
{\psset{originT={UZ},base=1,phi=15,symX=true,symline=0}
\pstransTSX(0,0){U1}{U1'}
\pstransTSX[translineA=true,translineB=true](0.3;0){zg1}{zg1'}
\pstransTSX[translineA=true,translineB=true](0.5;90){zg2}{zg2'}
%-------------------------------------------------------
\pscircle(UZ){0.5}
\psline[linecolor=yellow!90!black!70]{c-c}(U1)(zg1)
\psline[linecolor=yellow!90!black!70]{c-c}(U1)(zg2)
%-------------------------------------------------------
\psline[linecolor=red]{c-c}(U1')(zg1')
\psline[linecolor=orange]{c-c}(U1')(zg2')
\multido{\i=0+5,\n=5+5}{72}{%
\pstransTSX(0.5;\i){A\i}{A'\i}
\pstransTSX(0.5;\n){B\n}{B'\n}
\psline(A'\i)(B'\n)}
{\psset{deltaphi=45}
\pstransTSX(0,0){U1}{U1'}
\pstransTSX[translineA=true,translineB=true](0.3;0){zg1}{zg1'}
\pstransTSX[translineA=true,translineB=true](0.5;90){zg2}{zg2'}
%-------------------------------------------------------
\psline[linecolor=red]{c-c}(U1')(zg1')
\psline[linecolor=orange]{c-c}(U1')(zg2')
\multido{\i=0+5,\n=5+5}{72}{%
\pstransTSX(0.5;\i){A\i}{A'\i}
\pstransTSX(0.5;\n){B\n}{B'\n}
\psline(A'\i)(B'\n)}
}
}
\end{pspicture}
```

5 The macro \pstransTSK

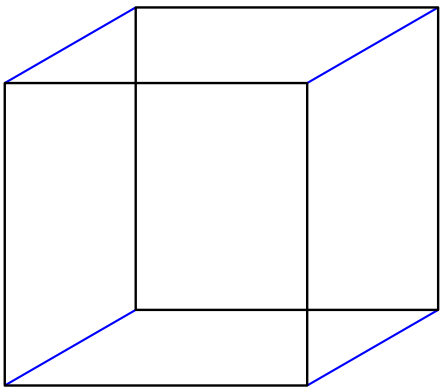
\pstransTSK [Options] (x_A, y_A) {length}{A}{A'}

If the edges that should be transformed run vertically in the x, y-plane, they are projected orthogonally to the base line and from there transformed by a chosen angle and eventually by a shortening factor. The orthogonal projection of all points of a vertical edge end in the intersection point of this edge with the base line. Choosing a proper base, the intersection point of the vertical edge with the base line is a vertex of the geometrical object. In that special case it is sufficient to only project one of the vertices of the vertical line—such as doing it, constructing a perspective of the edges vertical to the x, y-plane (leading to the back) of a three-dimensional body with a given projection angle and a choden shortening factor. Thus, the projection can be reduced, cause only half of the vertices need to be transformed by a simple shifting.
The macro \pstransTSK shifts a point to be setup in parentheses. The length of the shift is calculated by the value given after it in curly braces multiplied with the shortening factor that is vkf=0.5 by default. The angle of the shifting in relation to the horizontal (x-axis), $45°$ by default, can be varied by phi=... . The subsidiary lines can be shown or hidden. The options of the subsidiary lines can be varied.
The following set of examples will make this clearer.
First we setup the points of a square that lays in x, y-plane. These points we give the node names A, B, C and D. The transformed points are shifted by two units (the length 4 is multiplied by the shortening factor 0.5) and the transformed nodes names A′, B′, C′ and D′ are calculated and stored under that names.

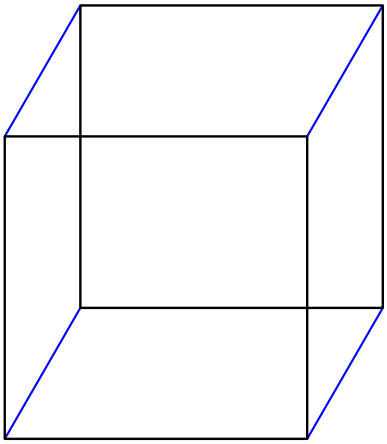


```
\begin{pspicture}[showgrid=false,shift=-5.0](0.7,-0.4)
    (7,5.8)
%---- Vertices of the square ---
\pstransTSK(1,0){4}{A}{A'}
\pstransTSK(5,0){4}{B}{B'}
\pstransTSK(5,4){4}{C}{C'}
\pstransTSK(1,4){4}{D}{D'}
\pspolygon[linewidth=0.9pt,linecolor=black](A)(B)(C)(D)
\pspolygon[linewidth=0.9pt,linecolor=black](A')(B')(C')(D')
\end{pspicture}
```

With some other angles this looks as follows:

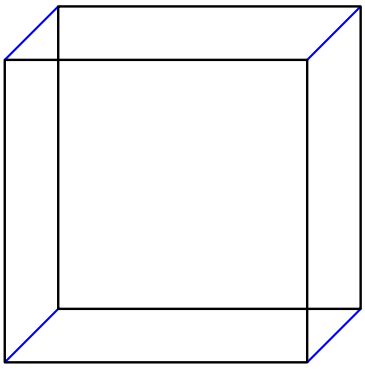


```
\begin{pspicture}[showgrid=false,shift=-5.0](0.7,-0.4)
    (7,5.8)
%---- Vertices of the square ---
\pstransTSK[phi=30](1,0){4}{A}{A'}
\pstransTSK[phi=30](5,0){4}{B}{B'}
\pstransTSK[phi=30](5,4){4}{C}{C'}
\pstransTSK[phi=30](1,4){4}{D}{D'}
\pspolygon[linewidth=0.9pt,linecolor=black](A)(B)(C)(D)
\pspolygon[linewidth=0.9pt,linecolor=black](A')(B')(C')
    (D')
\end{pspicture}
```

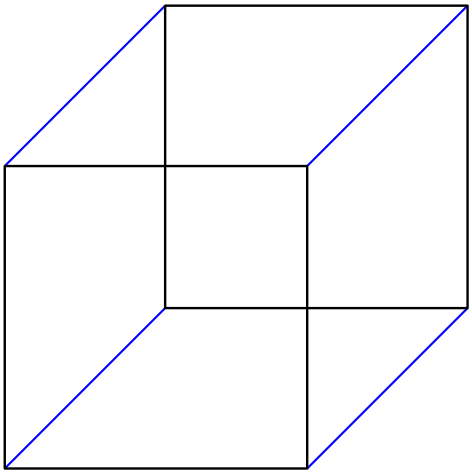


```
\begin{pspicture}[showgrid=false,shift=-5.0](0.7,-0.4)
    (7,5.8)
%---- Vertices of the square ---
\psset{phi=60}
\pstransTSK(1,0){4}{A}{A'}
\pstransTSK(5,0){4}{B}{B'}
\pstransTSK(5,4){4}{C}{C'}
\pstransTSK(1,4){4}{D}{D'}
\pspolygon[linewidth=0.9pt,linecolor=black](A)(B)(C)(D)
\pspolygon[linewidth=0.9pt,linecolor=black](A')(B')(C')(D')
\end{pspicture}
```

A change of the shortening factor leads to the following designs.



```
\begin{pspicture}[showgrid=false,shift=-5.0](0.7,-0.4)
    (7,5.8)
%---- Vertices of the square ---
\psset{vkf=0.25}
\pstransTSK(1,0){4}{A}{A'}
\pstransTSK(5,0){4}{B}{B'}
\pstransTSK(5,4){4}{C}{C'}
\pstransTSK(1,4){4}{D}{D'}
\pspolygon[linewidth=0.9pt,linecolor=black](A)(B)(C)(D)
\pspolygon[linewidth=0.9pt,linecolor=black](A')(B')(C')(D')
\end{pspicture}
```

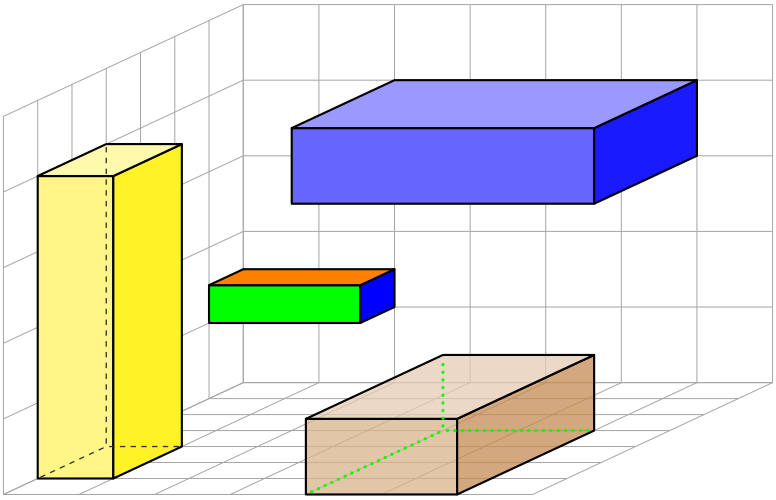


```
\begin{pspicture}[showgrid=false,shift=-5.0](0.7,-0.4)
    (7,5.8)
%---- Vertices of the square ---
\psset{vkf=0.75}
\pstransTSK(1,0){4}{A}{A'}
\pstransTSK(5,0){4}{B}{B'}
\pstransTSK(5,4){4}{C}{C'}
\pstransTSK(1,4){4}{D}{D'}
\pspolygon[linewidth=0.9pt,linecolor=black](A)(B)(C)(D)
\pspolygon[linewidth=0.9pt,linecolor=black](A')(B')(C')
    (D')
\end{pspicture}
```

6 The macro \psboxTS

\psboxTS [Options] (x, y, z) {length in x}{width in y}{height in z}{color}

With the macro \psboxTS we can easily draw cuboids. For its three coordinates we use the left, bottom vertex in the back and put them in parentheses separated by commas, followed by its length, width and height and last by its color—all these four arguments are put in curly braces.
With the option hideline=true the hidden lines are shown with the keys hidelinewidth= , hidelinestyle= and hidecolor= .
With the option differentcol=true we can give the three visible faces different colors. The color of the right side can changed with facecolorR= , the color of the top can changed with facecolorT= . The frontcolor is the color in curly braces.
This macro and the others as well can be used in combination with the pst-3dplot package with the settings: coorType=1 , xThreeDunit=vkf and phi = 180 Alpha sub .

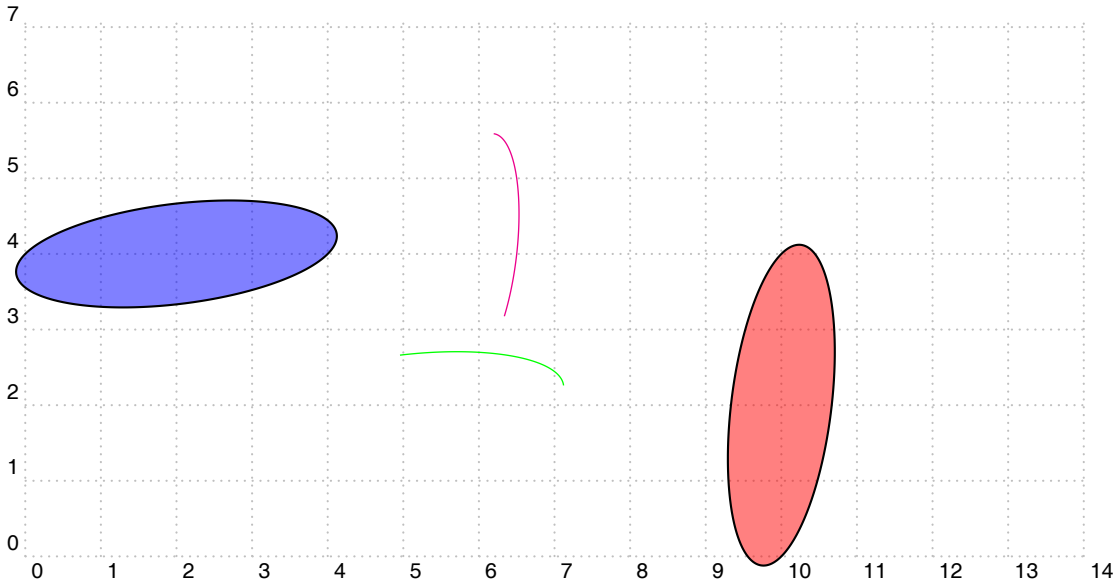


```
\begin{pspicture}[showgrid=false](-2,-2.5)(6,6)
\psset{xMin=0,yMin=0,zMin=0,xMax=11,yMax=11,zMax=4,Alpha=155,Beta=20,Dx=1,Dy=1,
    Dz=1,arrowsize=.2,arrowinset=0.1,coorType=1,xThreeDunit=0.5,phi=180 155 sub}%
\pstThreeDPlaneGrid[planeGrid=xy,linewidth=0.3pt,linecolor=gray!70,xsubticks=7,
    ysubticks=7](0,0)(7,7)%
\pstThreeDPlaneGrid[planeGrid=xz,linewidth=0.3pt,linecolor=gray!70,xsubticks=7,
    ysubticks=5](0,0)(7,5)%
\pstThreeDPlaneGrid[planeGrid=yz,linewidth=0.3pt,linecolor=gray!70,xsubticks=7,
    ysubticks=5](0,0)(7,5)%
%--------------------------------------
\psboxTS(0,2,3){3}{4}{1}{blue}
\psboxTS[hideline=true,dash=2pt 2pt,hidelinewidth=0.5pt](4,0,0){2}{1}{4}{yellow}
\psboxTS[opacity=0.75,hideline=true,hidelinewidth=1.2pt,hidelinestyle=dotted,
    hidecolor=green,dotsep=1.5pt](3,4,0){4}{2}{1}{brown}
\psboxTS[differentcol=true,facecolorR=blue,facecolorT=orange](0,0,1){1}{2}{0.5}{
    green}
%--------------------------------------
\end{pspicture}
```

7 Macros for circles and arcs

`\psCircleTS [Options] {Radius},`
`\psCircleTSX [Options] {Radius}`

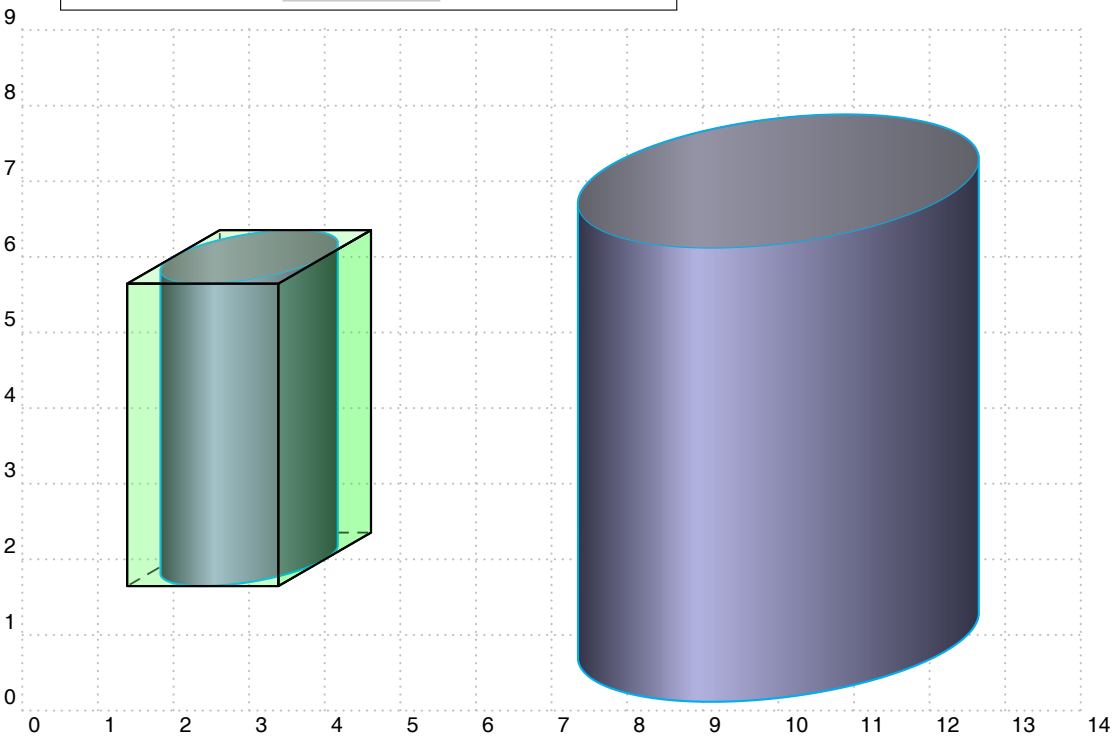
`\psArcTS [Options] {Radius}{Startwinkel}{überstrichener Winkel},`
`\psArcTSX [Options] {Radius}{Startwinkel}{überstrichener Winkel}`




```
\begin{pspicture}[showgrid=true,shift=-4.9](0.5,-0.5)(14,7.3)
\rput(2,4){\psCircleTS[fillstyle=solid,fillcolor=blue,opacity=0.5]{2}}
\rput(10,2){\psCircleTSX[fillstyle=solid,fillcolor=red,opacity=0.5]{2}}
\rput(5,2){\psArcTS[linecolor=green,linewidth=0.5pt]{2}{0}{90}}
\psArcTSX[linecolor=magenta,linewidth=0.5pt,originT={6,4},symX=false
    ]{1.5}{0}{120}
\end{pspicture}
```

8 Macro for cylinder

\psZylinderTS [Options] {Radius}{Höhe}

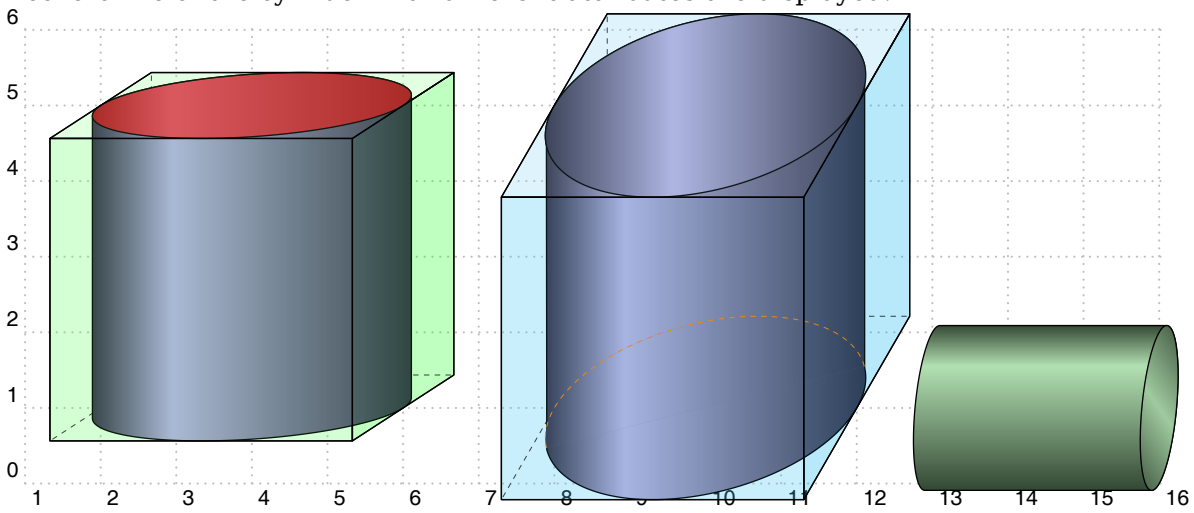



```
\begin{pspicture}[showgrid=true,shift=-4.9](0.5,-0.5)(14,8.8)
\psset{toplinewidth=0.3pt,toplinecolor=cyan}
\rput(10,1){%
\psZylinderTS[opacity=0.6,fillstyle=gradient,gradbegin=black!90!blue!80,gradend=
    black!40!blue!30,gradangle=90,%
gradmidpoint=0.3,linecolor=cyan,linewidth=0.8pt]{2.5}{6}}
{\psset{phi=30,vkf=2 sqrt 2 div,opacity=0.2}
\psboxTS[hideline=true](-1,2,2){2}{2}{4}{green}
\rput(3,2){%
\psZylinderTS[opacity=0.6,fillstyle=gradient,gradbegin=black!90!blue!80,gradend=
    black!40!blue!30,gradangle=90,%
gradmidpoint=0.3,linecolor=cyan,linewidth=0.8pt]{1}{4}}
\psboxTS[hideline=false](-1,2,2){2}{2}{4}{green}
}
\end{pspicture}
```

The following additional options are possible for the cylinder so that the top surface of the cylinder can be designed independently.

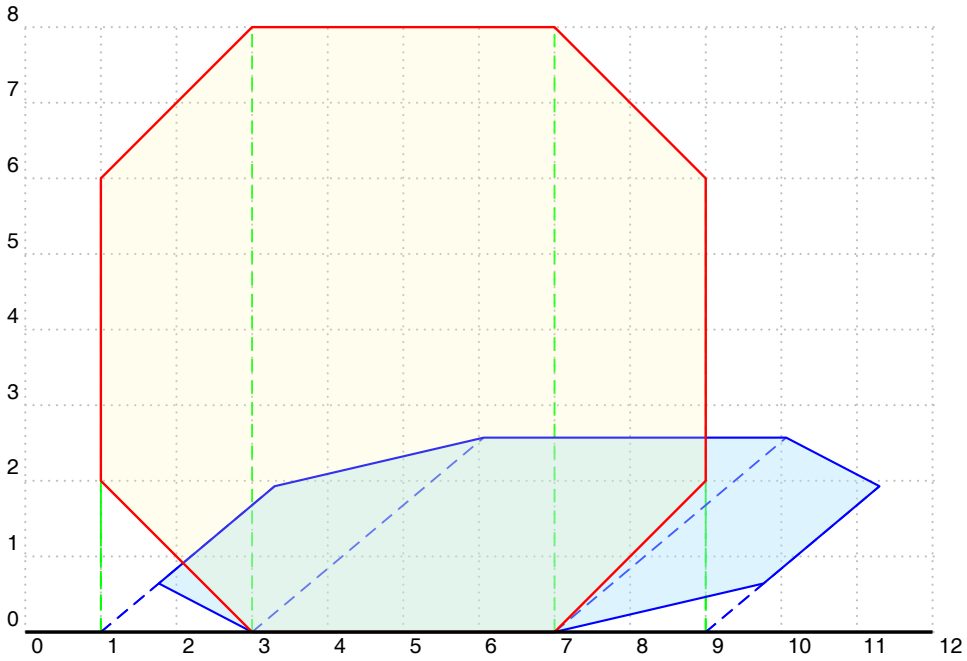
```
\pstransTS[topfillstyle],
\pstransTS[topmidpoint],
\pstransTS[topangle],
\pstransTS[toplinecolor],
\pstransTS[topfillcolor],
\pstransTS[toplinewidth]
```

In addition, with `hideline=true` , `hidelinewidth=` , `hidelinestyle=` and `hidecolor=` covert line of the cylinder with different attributes are displayed.

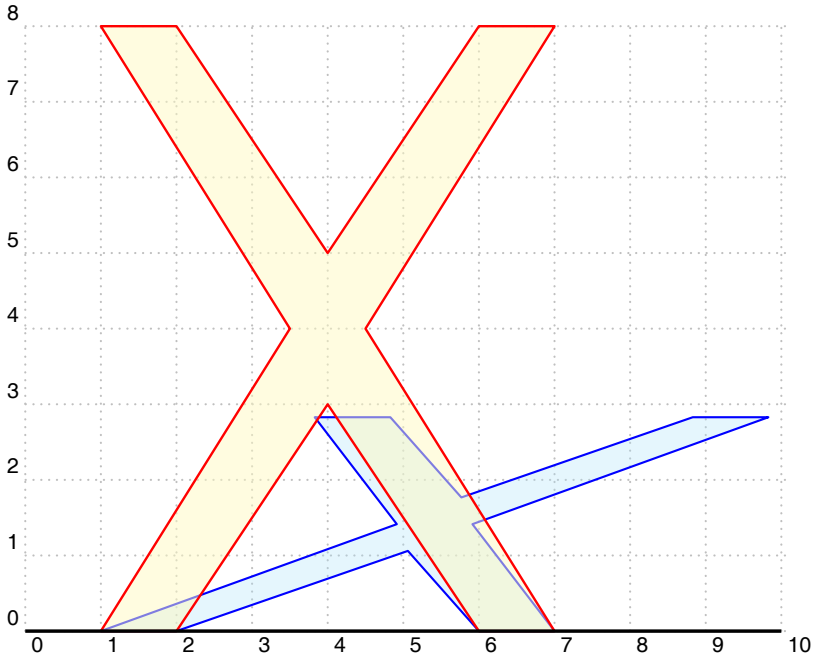



```
\begin{pspicture}[showgrid=true](1,-0.3)(16,6)
\psset{toplinewidth=0.5pt,opacity=0.3}
{\psset{vkf=0.4,phi=33,topfillcolor=red}
\psboxTS[linewidth=0.5pt,opacity=0.2,linejoin=1,hideline=true,dash=2pt 2pt,
    hidelinewidth=0.3pt](-2,2,1){4}{4}{4}{green}
\rput(4,1){\psZylinderTS[opacity=0.6,linewidth=0.5pt,fillstyle=gradient,
    gradbegin=black!90!blue!80,%
gradend=black!40!blue!30,gradangle=90,gradmidpoint=0.25]{2}{4}}
\psboxTS[linewidth=0.5pt,opacity=0.1,linejoin=1](-2,2,1){4}{4}{4}{green}}
{\psset{vkf=0.7,phi=60,topfillstyle=gradient,topmidpoint=0.4,topangle=90}
\psboxTS[linewidth=0.5pt,opacity=0.2,linejoin=1,hideline=true,dash=2pt 2pt,
    hidelinewidth=0.3pt](-2,8,1){4}{4}{4}{cyan}
\rput(10,1){\psZylinderTS[hideline=true,hidecolor=orange,dash=2pt 2pt,%
hidelinewidth=0.3pt,opacity=0.6,linewidth=0.5pt,fillstyle=gradient,gradbegin=
    black!90!blue!80,%
gradend=black!40!blue!30,gradangle=90,gradmidpoint=0.25]{2}{4}}
\psboxTS[linewidth=0.5pt,opacity=0.1,linejoin=1](-2,8,1){4}{4}{4}{cyan}}
\rput{-90}(13,1){\psZylinderTS[phi=-30,linewidth=0.5pt,fillstyle=gradient,
    gradbegin=black!90!green!80,%
gradend=black!40!green!30,gradangle=90,gradmidpoint=0.25,topfillstyle=gradient,
    topmidpoint=0.5,topangle=35]{1}{3}}
\end{pspicture}
```

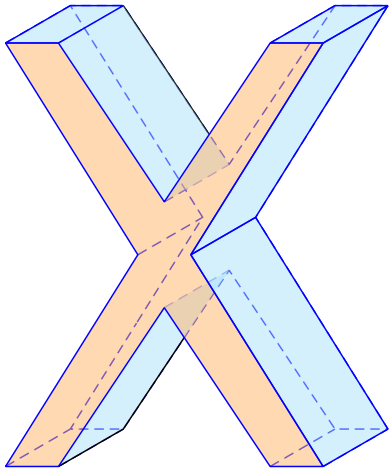
9 Examples



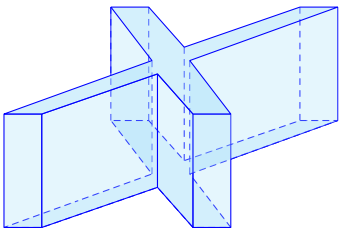

```
\begin{pspicture}[showgrid=true](0.5,-0.5)(11.5,8.0)
{\psset{phi=40,translineA=true,translineB=true}
%------ Vertices of the octagon -------------
\pstransTS(3,0){A}{A'}
\pstransTS(7,0){B}{B'}
\pstransTS(9,2){C}{C'}
\pstransTS(9,6){D}{D'}
\pstransTS(7,8){E}{E'}
\pstransTS(3,8){F}{F'}
\pstransTS(1,6){G}{G'}
\pstransTS(1,2){H}{H'}
%-------------------------------------------
}
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.4,linecolor=blue](A')(B')
    (C')(D')(E')(F')(G')(H')
\pspolygon[fillstyle=solid,fillcolor=yellow!40,opacity=0.2,linewidth=0.9pt,
    linecolor=red](A)(B)(C)(D)(E)(F)(G)(H)
\pcline[linewidth=1.3pt](0,0|O)(12,0|O)
\end{pspicture}
```



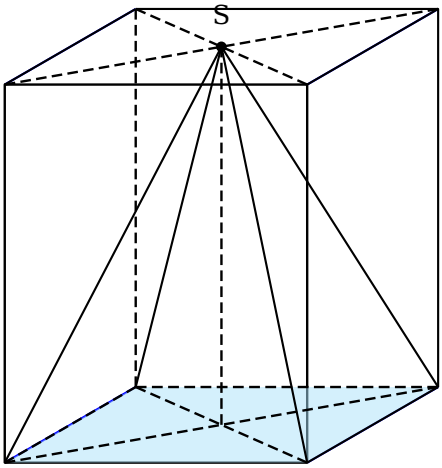

```
\begin{pspicture}[showgrid=true](0.5,-0.5)(10,8.4)
{\psset{base=0,translineA,translineB,linecolor=blue,linestyle=dashed,transAcolor
    =blue,transBcolor=orange,dash=5pt 5pt}
\pnode(0,\ba){O}% \ba gives the y-value of the base line
%------ Vertices of the "X" -------------
\pstransTS(1,0){A}{A'}
\pstransTS(2,0){B}{B'}
\pstransTS(4,3){C}{C'}
\pstransTS(6,0){D}{D'}
\pstransTS(7,0){E}{E'}
\pstransTS[transAlinestyle=solid,transAcolor=red,transAlinewidth=2pt](4.5,4){F}{
    F'}
\pstransTS[linestyle=solid,linecolor=green](7,8){G}{G'}
\pstransTS[transAcolor=red,transBcolor=black,transBlinewidth=1.4pt](6,8){H}{H'}
\pstransTS(4,5){I}{I'}
\pstransTS(2,8){J}{J'}
\pstransTS[linecolor=red](1,8){K}{K'}
\pstransTS(3.5,4){L}{L'}
%-------------------------------------
}
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue](A')(B')
    (C')(D')(E')(F')(G')(H')(I')(J')(K')(L')
\pspolygon[fillstyle=solid,fillcolor=yellow!30,opacity=0.5,linewidth=0.9pt,
    linecolor=red](A)(B)(C)(D)(E)(F)(G)(H)(I)(J)(K)(L)
\pcline[linewidth=1.3pt](0,0|O)(10,0|O)
\end{pspicture}
```



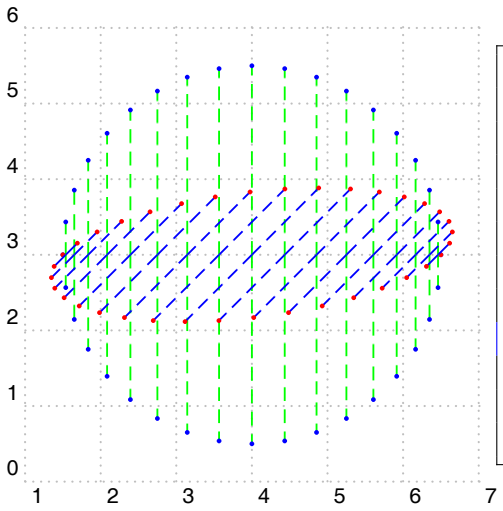
```
\psscalebox{0.7}{%
\begin{pspicture}[showgrid=false](0.5,-0.5)(10,9.4)
\def\lange{2 sqrt 2 mul}
{\psset{phi=30,base=0,translineK,translinestyle=dashed,linecolor=blue,linejoin
    =2,fillstyle=solid,opacity=0.5}
%------ Vertices of the "X" -------------
\pstransTSK(1,0){\lange}{A}{A'}
\pstransTSK(2,0){\lange}{B}{B'}
\pstransTSK(4,3){\lange}{C}{C'}
\pstransTSK(6,0){\lange}{D}{D'}
\pstransTSK(7,0){\lange}{E}{E'}
\pstransTSK(4.5,4){\lange}{F}{F'}
\pstransTSK(7,8){\lange}{G}{G'}
\pstransTSK(6,8){\lange}{H}{H'}
\pstransTSK(4,5){\lange}{I}{I'}
\pstransTSK(2,8){\lange}{J}{J'}
\pstransTSK(1,8){\lange}{K}{K'}
\pstransTSK(3.5,4){\lange}{L}{L'}
%-------------------------------------
\pspolygon[linestyle=dashed](A')(B')(C')(D')(E')(F')(G')(H')(I')(J')(K')(L')
\pspolygon[fillcolor=cyan!30,linestyle=none](B)(B')(C')(C)
}
\psIntersectionPoint(C')(B')(C)(D){S1}
\psline(B)(B')(S1)
\psIntersectionPoint(H)(I)(I')(J'){S2}
\psline(J')(S2)
{\psset{phi=30,base=0,translineK,translinestyle=dashed,linecolor=blue,linejoin
    =2,fillstyle=solid,opacity=0.5}
\pspolygon[fillcolor=cyan!30](E)(E')(F')(F)
\pspolygon[fillcolor=cyan!30](F)(F')(G')(G)
\pspolygon[fillcolor=cyan!30](G)(G')(H')(H)
\pspolygon[fillcolor=cyan!30,linestyle=none](I)(I')(J')(J)
\pspolygon[fillcolor=cyan!30](J)(J')(K')(K)
\pspolygon[fillcolor=orange!60](A)(B)(C)(D)(E)(F)(G)(H)(I)(J)(K)(L)}
\end{pspicture}}
```



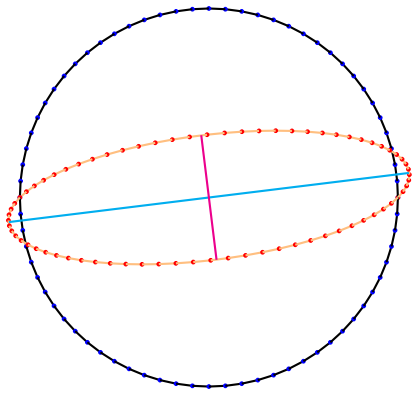
```
\psscalebox{0.5}{%
\begin{pspicture}[showgrid=false](0.5,0)(10,5.7)
{\psset{base=0,linecolor=blue,linestyle=dashed,dash=5pt 4pt}
%------ Vertices of the lower "X" -------------
\pstransTS(1,0){A}{A'}
\pstransTS(2,0){B}{B'}
\pstransTS(4,3){C}{C'}
\pstransTS(6,0){D}{D'}
\pstransTS(7,0){E}{E'}
\pstransTS(4.5,4){F}{F'}
\pstransTS(7,8){G}{G'}
\pstransTS(6,8){H}{H'}
\pstransTS(4,5){I}{I'}
\pstransTS(2,8){J}{J'}
\pstransTS(1,8){K}{K'}
\pstransTS(3.5,4){L}{L'}
%------ Vertices of the upper "X" -------------
\rput(0,3){% The same "X" shifted 3 units upwards
\pstransTS(1,0){A1}{A1'}
\pstransTS(2,0){B1}{B1'}
\pstransTS(4,3){C1}{C1'}
\pstransTS(6,0){D1}{D1'}
\pstransTS(7,0){E1}{E1'}
\pstransTS(4.5,4){F1}{F1'}
\pstransTS(7,8){G1}{G1'}
\pstransTS(6,8){H1}{H1'}
\pstransTS(4,5){I1}{I1'}
\pstransTS(2,8){J1}{J1'}
\pstransTS(1,8){K1}{K1'}
\pstransTS(3.5,4){L1}{L1'}}
%-------------------------------------
\pcline(F')(F1')
\pcline(H')(H1')
\pcline(I')(I1')
\pcline(J')(J1')
\pcline(K')(K1')
\pcline(L')(L1')
}
\psIntersectionPoint(K')(K1')(A1')(L1'){S1}
\psline[linestyle=solid,linecolor=blue](K1')(S1)
\psIntersectionPoint(E')(E1')(F')(G'){S2}
\psline[linestyle=solid,linecolor=blue](S2)(G')(G1')%(F1')(E1')(E')
{%
\psset{linejoin=2,fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue}
\pspolygon[linestyle=none](K')(L')(L1')(K1')
\pspolygon[linestyle=dashed](A')(B')(C')(D')(E')(F')(G')(H')(I')(J')(K')(L')
\pspolygon(A')(B')(B1')(A1')
\pspolygon(B')(C')(C1')(B1')
\pspolygon(C')(D')(D1')(C1')
\pspolygon(D')(E')(E1')(D1')
\pspolygon[linestyle=none](F')(G')(G1')(F1')
\pspolygon[opacity=0.5](A1')(B1')(C1')(D1')(E1')(F1')(G1')(H1')(I1')(J1')(K1')(L1')
}
\end{pspicture}}
```



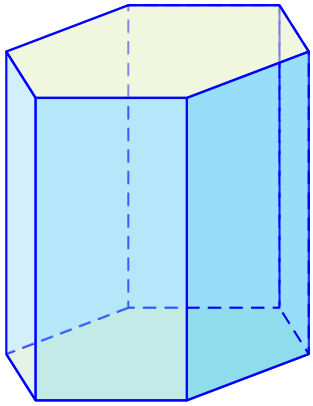

```
\begin{pspicture}[showgrid=false,shift=-5.0](0.7,-0.4)(7,6)
{\psset{phi=30}
%------ Eckpunkte vom Aufriss --------------
\pstransTSK(1,0){4}{A}{A'}
\pstransTSK(5,0){4}{B}{B'}
\pstransTSK(5,5){4}{C}{C'}
\pstransTSK(1,5){4}{D}{D'}
%-----------------------------------------
}
\psIntersectionPoint(A)(B')(B)(A'){S1}
\psIntersectionPoint(C)(D')(D)(C'){S2}
\pspolygon[linewidth=0.9pt,linecolor=black](A)(B)(C)(D)
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.5,linestyle=none](A)(B)(B')(A')
{\psset{linestyle=dashed,dash=4pt 2pt,linewidth=0.9pt,linecolor=black}
\pcline(A')(B')
\pcline(A')(D')
\pcline(A)(A')
\pcline(A)(B')
\pcline(B)(A')
\pcline(D)(C')
\pcline(C)(D')
\pcline(S1)(S2)
}
\pcline(B)(B')
\pcline(C)(C')
\pcline(D)(D')
\pcline(B')(C')
\pcline(C')(D')
\pcline(A)(S2)
\pcline(B)(S2)
\pcline(B')(S2)
\pcline(A')(S2)
\qdisk(S2){2pt}\uput{0.3}[90](S2){S}
\end{pspicture}
```



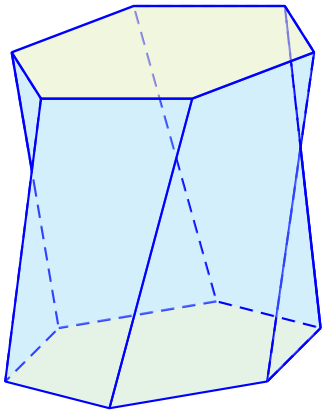

```
\begin{pspicture}[showgrid=true](1,-0.3)(7,6)
\pnode(4,3){M}
{\psset{originT={M}}
\multido{\i=0+10}{36}{\psset{phi=45,vkf=0.5,
    translineA=true,translineB=true}
\pstransTS[linecolor=blue,linewidth=0.5pt,
    linestyle=dashed](2.5;\i){A\i}{A'\i}
\psdot[dotsize=1.8pt,linecolor=blue](A\i)\psdot
    [dotsize=1.8pt,linecolor=red](A'\i)
}
}
\end{pspicture}
```



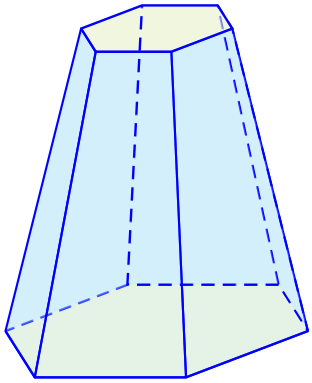
```
\begin{pspicture}[showgrid=false](1,-0.3)(7,6)
\pnode(4,3){M}
{\psset{originT={M}}
\multido{\i=0+5,\n=5+5}{72}{%
\pstransTS[linecolor=blue,linewidth=0.5pt,
    linestyle=dashed](2.5;\i){A\i}{A'\i}
\pstransTS(2.5;\n){B\n}{B'\n}
\psdot[dotsize=1.8pt,linecolor=blue](A\i)\psdot
    [dotsize=1.8pt,linecolor=red](A'\i)\psline(
    A\i)(B\n)\psline[linecolor=orange!50](A'\i)
    (B'\n)
}
}
\rput(M){\pnode(2.665;7){C'}}
\rput(M){\pnode(2.665;187){D'}}
\pcline[linecolor=cyan](C')(D')
\rput(M){\pnode(0.83;97){E'}}
\rput(M){\pnode(0.83;277){F'}}
\pcline[linecolor=magenta](E')(F')
\end{pspicture}
```



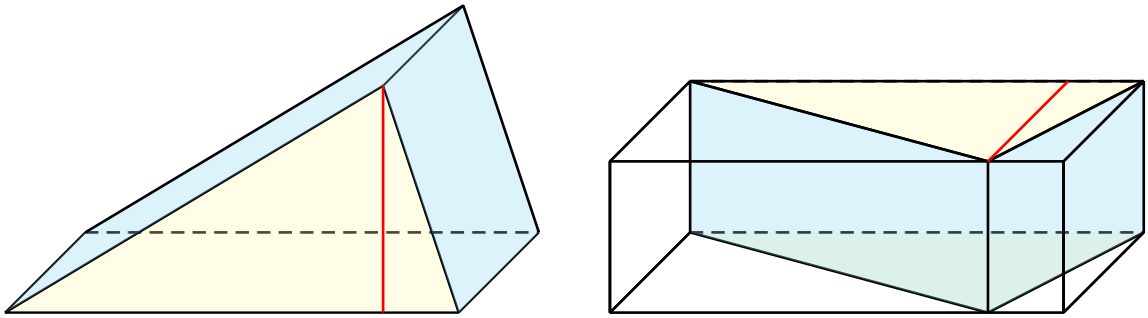
```
\begin{pspicture}[showgrid=false](-2,0)(8,6)
\pnode(4.5,1){Z}\psset{originT={Z}}
\pstransTS(2;-60){A'}{A}
\pstransTS(2;0){B'}{B}
\pstransTS(2;60){C'}{C}
\pstransTS(2;120){D'}{D}
\pstransTS(2;180){E'}{E}
\pstransTS(2;240){F'}{F}
\pspolygon[fillstyle=solid,fillcolor=yellow!30,opacity=0.5,linewidth=0.9pt,
    linecolor=blue,linestyle=none](A)(B)(C)(D)(E)(F)%(G)(H)(I)(J)(K)(L)
%\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue](A')(B
    ')(C')(D')(E')(F')%(G')(H')
\psline[linewidth=0.9pt,linecolor=blue](E)(F)(A)(B)
\psline[linewidth=0.9pt,linecolor=blue,linestyle=dashed](D)(E)

\pnode(4.5,5){Z}\psset{originT={Z}}
\pstransTS(2;-60){I'}{I}
\pstransTS(2;0){J'}{J}
\pstransTS(2;60){K'}{K}
\pstransTS(2;120){L'}{L}
\pstransTS(2;180){M'}{M}
\pstransTS(2;240){N'}{N}
\pspolygon[fillstyle=solid,fillcolor=cyan!60,opacity=0.6,linecolor=blue](A)(B)(J
    )(I)%(E')(F')%(G')(H')
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue,
    linestyle=dashed](B)(C)(K)(J)%
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue,
    linestyle=dashed](C)(D)(L)(K)%
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue,
    linestyle=none](D)(E)(M)(L)%
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue](E)(F)(N
    )(M)%
\pspolygon[fillstyle=solid,fillcolor=cyan!50,opacity=0.4,linecolor=blue](F)(A)(I
    )(N)%
\pspolygon[fillstyle=solid,fillcolor=yellow!30,opacity=0.5,linewidth=0.9pt,
    linecolor=blue](I)(J)(K)(L)(M)(N)%(G)(H)(I)(J)(K)(L)
\end{pspicture}
```



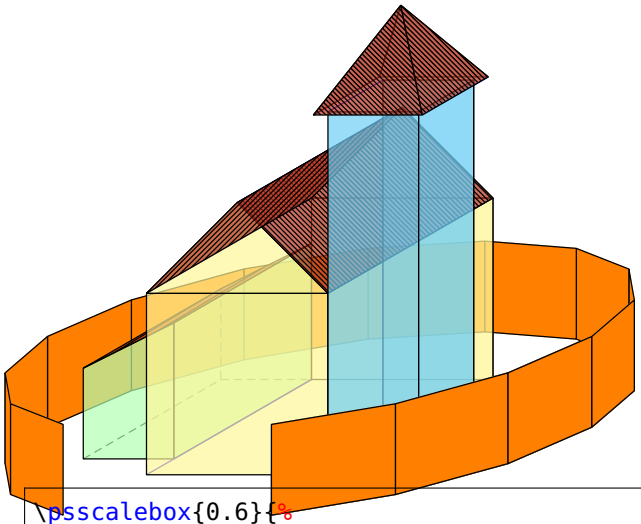
```
\begin{pspicture}[showgrid=false](-0.5,0.2)(7,5.6)
\psset{linejoin=2}
\pnode(2.5,1){Z}\psset{originT={Z}}
\pstransTS(2;-90){A'}{A}
\pstransTS(2;-30){B'}{B}
\pstransTS(2;30){C'}{C}
\pstransTS(2;90){D'}{D}
\pstransTS(2;150){E'}{E}
\pstransTS(2;210){F'}{F}
\pspolygon[fillstyle=solid,fillcolor=yellow!30,opacity=0.5,linewidth=0.9pt,linecolor=blue,
    linestyle=none](A)(B)(C)(D)(E)(F)%(G)(H)(I)(J)(K)(L)
\psline[linewidth=0.9pt,linecolor=blue](B)(C)
\psline[linewidth=0.9pt,linecolor=blue,linestyle=dashed](D)(E)
\pnode(2.5,5){Z}\psset{originT={Z}}
\pstransTS(2;-60){I'}{I}
\pstransTS(2;0){J'}{J}
\pstransTS(2;60){K'}{K}
\pstransTS(2;120){L'}{L}
\pstransTS(2;180){M'}{M}
\pstransTS(2;240){N'}{N}
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue](A)(B)(J)(I)%(E')(F
    ')%(G')(H')
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue,linestyle=dashed](B
    )(C)(K)(J)%
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue,linestyle=dashed](C
    )(D)(L)(K)%
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue,linestyle=none](D)(
    E)(M)(L)%
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue,linestyle=dashed](E
    )(F)(N)(M)%
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue](F)(A)(I)(N)%
\pspolygon[fillstyle=solid,fillcolor=yellow!30,opacity=0.5,linewidth=0.9pt,linecolor=blue](
    I)(J)(K)(L)(M)(N)%(G)(H)(I)(J)(K)(L)
\psIntersectionPoint(C)(K)(B)(J){SBJ}
\psIntersectionPoint(E)(M)(F)(N){SFN}
\psline[linewidth=0.9pt,linecolor=blue](C)(SBJ)
\psline[linewidth=0.9pt,linecolor=blue](SFN)(M)
\end{pspicture}
```



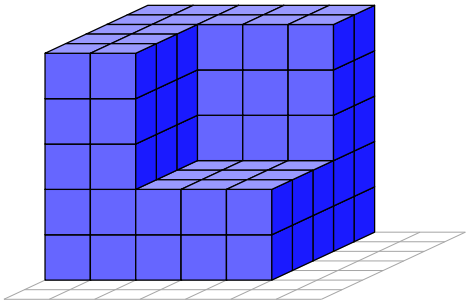
```
\begin{pspicture}[showgrid=false](0,0)(6,6)
\pnode(3.5,1){Z}\psset{originT={Z}}
\pstransTS(2;-60){A'}{A}
\pstransTS(2;0){B'}{B}
\pstransTS(2;60){C'}{C}
\pstransTS(2;120){D'}{D}
\pstransTS(2;180){E'}{E}
\pstransTS(2;240){F'}{F}
\pspolygon[fillstyle=solid,fillcolor=yellow!30,opacity=0.5,linewidth=0.9pt,linecolor=blue,
    linestyle=none](A)(B)(C)(D)(E)(F)%(G)(H)(I)(J)(K)(L)
\psline[linewidth=0.9pt,linecolor=blue](E)(F)(A)(B)
\psline[linewidth=0.9pt,linecolor=blue,linestyle=dashed](D)(E)
\pnode(3.5,5){Z}\psset{originT={Z}}
\pstransTS(1;-60){I'}{I}
\pstransTS(1;0){J'}{J}
\pstransTS(1;60){K'}{K}
\pstransTS(1;120){L'}{L}
\pstransTS(1;180){M'}{M}
\pstransTS(1;240){N'}{N}
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue](A)(B)(J)(I)%(E')(F
    ')%(G')(H')
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue,linestyle=dashed](B
    )(C)(K)(J)%
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue,linestyle=none](C)(
    D)(L)(K)%
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue,linestyle=none](D)(
    E)(M)(L)%
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue](E)(F)(N)(M)%
\pspolygon[fillstyle=solid,fillcolor=cyan!30,opacity=0.3,linecolor=blue](F)(A)(I)(N)%
\pspolygon[fillstyle=solid,fillcolor=yellow!30,opacity=0.5,linewidth=0.9pt,linecolor=blue](
    I)(J)(K)(L)(M)(N)%(G)(H)(I)(J)(K)(L)
\psline[linewidth=0.9pt,linecolor=blue,linestyle=dashed](C)(D)(L)
\end{pspicture}
```



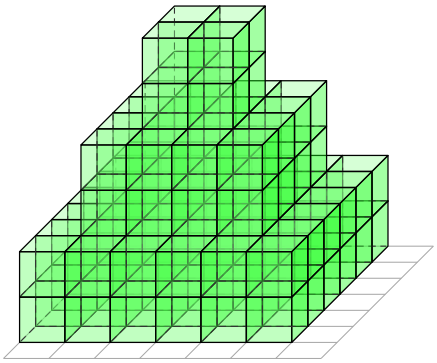
```
\begin{pspicture}[showgrid=false](0,0)(13,3.5)
\psset{linejoin=2,linewidth=1pt}
\psset{transcolor=black,translinestyle=solid}%
\pstransTSK(0,0){3}{A}{A'}%
\pstransTSK(6,0){3}{B}{B'}%
\pstransTSK(5,3){3}{C}{C'}%
\psline[linestyle=dashed](A')(B')%
\pspolygon[fillstyle=solid,fillcolor=yellow!40,opacity=0.3,linewidth=0.9pt,linecolor=black
    ](A)(B)(C)%
\pspolygon[fillstyle=solid,fillcolor=cyan!40,opacity=0.3,linestyle=none](A')(A)(C)(C')
\pspolygon[fillstyle=solid,fillcolor=cyan!40,opacity=0.3,linestyle=none](B)(B')(C')(C)
\psline(A')(C')(B')%
\pcline[linecolor=red](C)(C|A)
\pstransTS(8,3){A}{A'}
\pstransTS(14,3){B}{B'}
\pstransTS(13,0){C}{C'}
\pspolygon[fillstyle=solid,fillcolor=yellow!40,opacity=0.3,linestyle=none](A')(B')(C')
\psline[linestyle=dashed](A')(B')
\psline(A')(C')(B')
{\psset{base=0,originT={0,2}}
\pstransTS(8,3){D}{D'}
\pstransTS(14,3){E}{E'}
\pstransTS(13,0){F}{F'}
\pstransTS(F|D){K}{K'}
\pspolygon[fillstyle=solid,fillcolor=yellow!40,opacity=0.3](D')(E')(F')
\pspolygon[fillstyle=solid,fillcolor=cyan!40,opacity=0.3,linestyle=none](A')(C')(F')(D')
\pspolygon[fillstyle=solid,fillcolor=cyan!40,opacity=0.3,linestyle=none](C')(B')(E')(F')
\psline[linestyle=dashed](D')(E')
\psline(D')(F')(E')
\psline(C')(F')
}
{\psset{phi=-135,transcolor=black}
\pstransTSK(A'){3}{G'}{G}
\pstransTSK(B'){3}{H'}{H}
\pstransTSK(D'){3}{I'}{I}
\pstransTSK(E'){3}{J'}{J}
\pspolygon(G)(H)(J)(I)
}
\pspolygon(B')(H)(J)(E')
\pspolygon(A')(G)(I)(D')
\pcline[linecolor=red](F')(K')
%}
\end{pspicture}
```



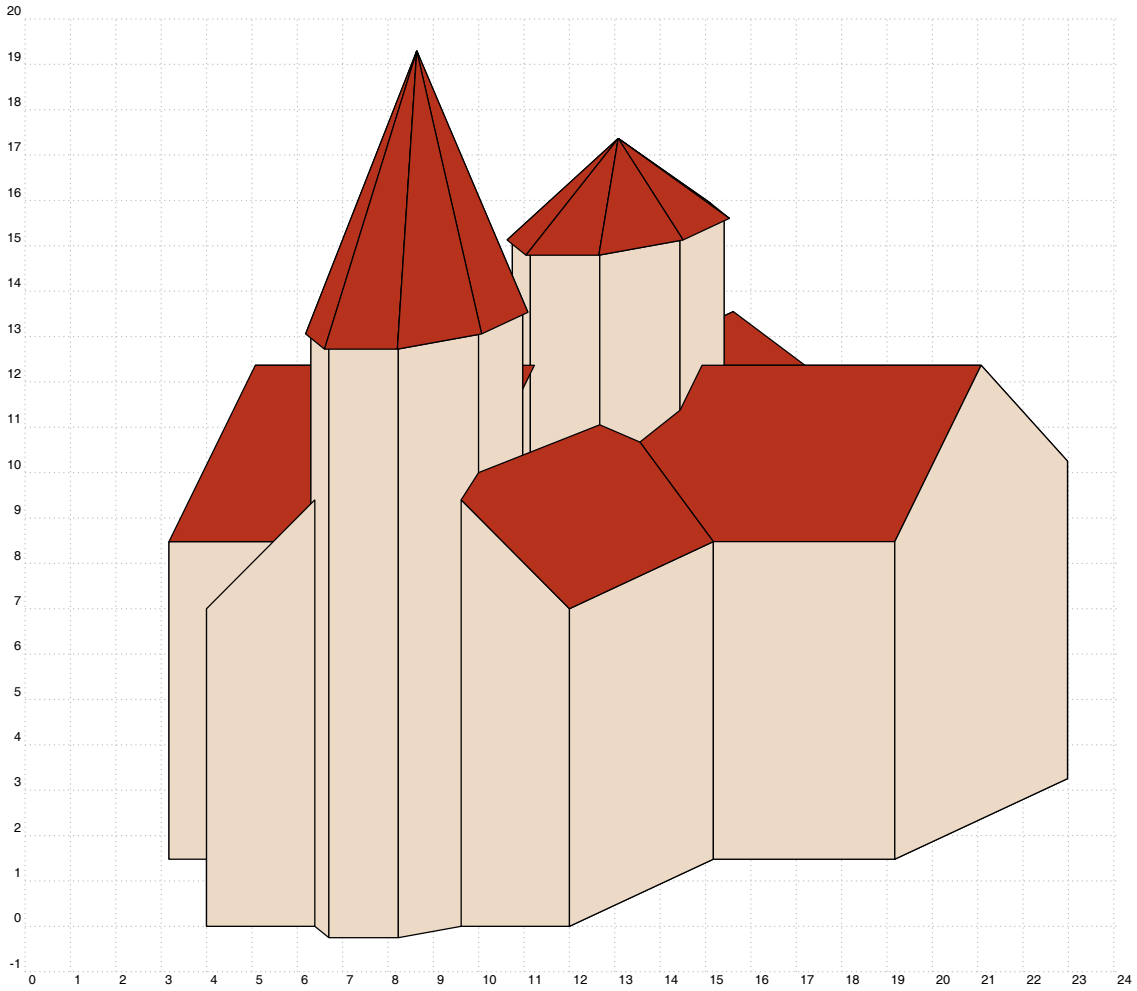
```
\psscalebox{0.6}{%
\begin{pspicture}[showgrid=false](-0.5,0)(17,10.4)
\psset{linejoin=2,phi=30,vkf=0.7}

{\psset{base=-3}
\multido{\i=0+1,\n=1+1,\ra=-67.5+22.5,\rb=-45+22.5}{14}{%
\pstransTS[originT={4,3}](6;\ra){D\i}{E\i}
\pstransTS[originT={4,3}](6;\rb){F\n}{G\n}
\pstransTS[originT={4,5}](6;\ra){H\i}{I\i}
\pstransTS[originT={4,5}](6;\rb){J\n}{K\n}
\psline[linecolor=orange!50](E\i)(G\n)
\psline[linecolor=orange!50](I\i)(K\n)
\pspolygon[fillstyle=solid,fillcolor=orange,opacity=1](E\i)(G\n)(K\n)(I\i)
}}

{\psset{translineK=false}%
\pstransTSK(0,0){1}{A1}{B1}
\pstransTSK(2,0){1}{A2}{B2}
\pstransTSK(2,3){1}{A3}{B3}
\pstransTSK(0,2){1}{A4}{B4}
%----------------------------
\pstransTSK(0,0){6}{A1}{C1}
\pstransTSK(2,0){6}{A2}{C2}
\pstransTSK(2,3){6}{A3}{C3}
\pstransTSK(0,2){6}{A4}{C4}
}
\psline[linestyle=dashed](C1)(C4)
\psline[linestyle=dashed](B1)(C1)(C2)
\pspolygon[fillstyle=solid,fillcolor=green!30,opacity=0.7](B1)(B2)(B3)(B4)
\pspolygon[fillstyle=solid,fillcolor=green!30,opacity=0.7](B2)(C2)(C3)(B3)
\pspolygon[fillstyle=vlines*,fillcolor=BrickRed,opacity=0.7,hatchangle=120,hatchsep=1.5pt](
    B4)(B3)(C3)(C4)
%----------------------------
\pstransTSK(2,0){6}{A5}{C5}
\pstransTSK(6,0){6}{A6}{C6}
\pstransTSK(6,4){6}{A7}{C7}
\pstransTSK(4,6){6}{A8}{C8}
\pstransTSK(2,4){6}{A9}{C9}
%----------------------------
\pspolygon[fillstyle=solid,fillcolor=yellow!50,opacity=0.2](C5)(C6)(C7)(C9)
\pspolygon[fillstyle=solid,fillcolor=yellow!50,opacity=0.2](C7)(C8)(C9)
\pspolygon[fillstyle=solid,fillcolor=yellow!50,opacity=0.7](A5)(A6)(A7)(A9)
\pspolygon[fillstyle=solid,fillcolor=yellow!50,opacity=0.7](A7)(A8)(A9)
\pspolygon[fillstyle=solid,fillcolor=yellow!50,opacity=0.7](A6)(C6)(C7)(A7)
\pspolygon[fillstyle=vlines*,fillcolor=BrickRed,opacity=0.7,hatchangle=45,hatchsep=1.5pt](A
    7)(C7)(C8)(A8)
\pspolygon[fillstyle=vlines*,fillcolor=BrickRed,opacity=0.7,hatchangle=135,hatchsep=1.5pt](
```

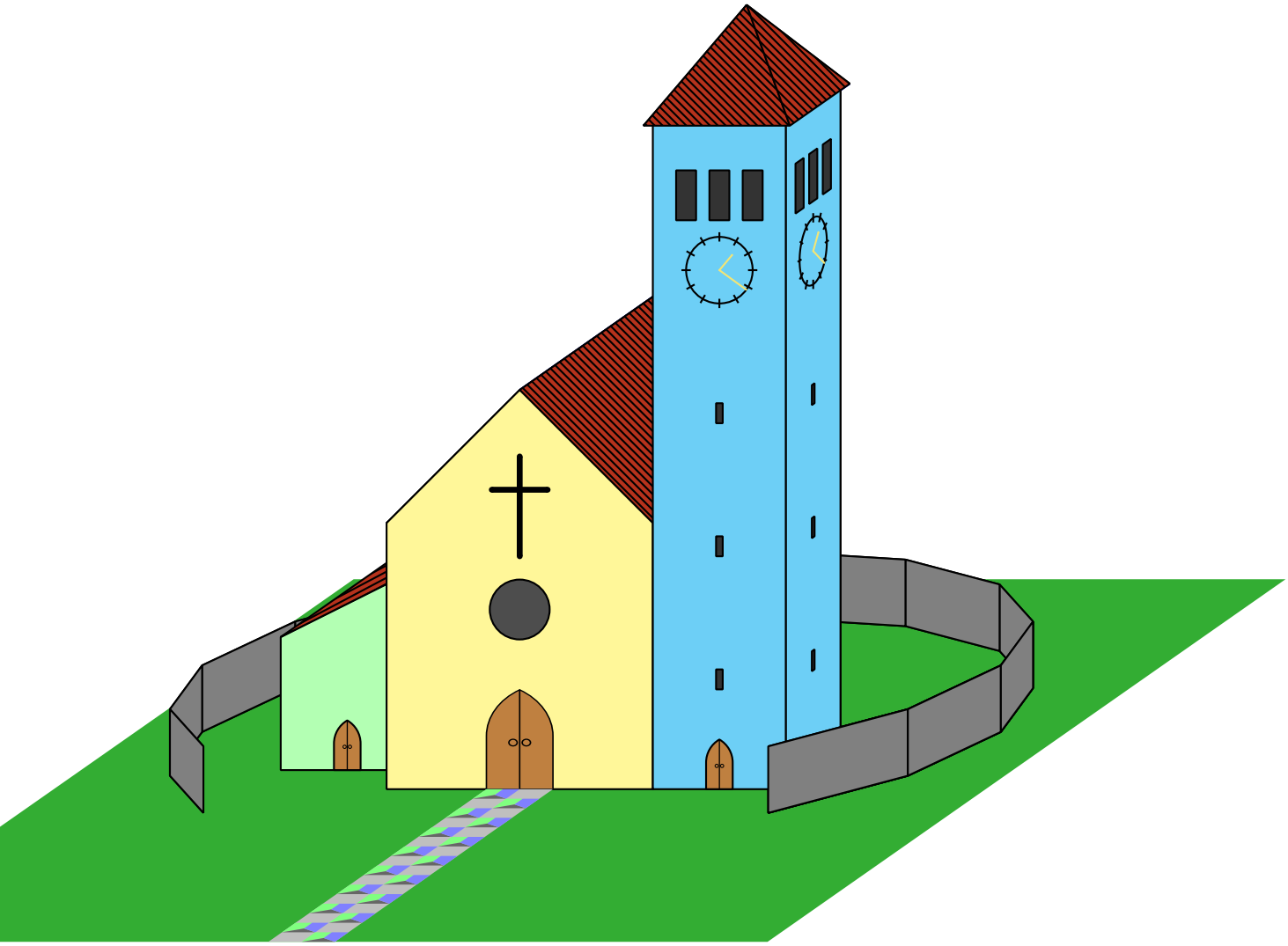


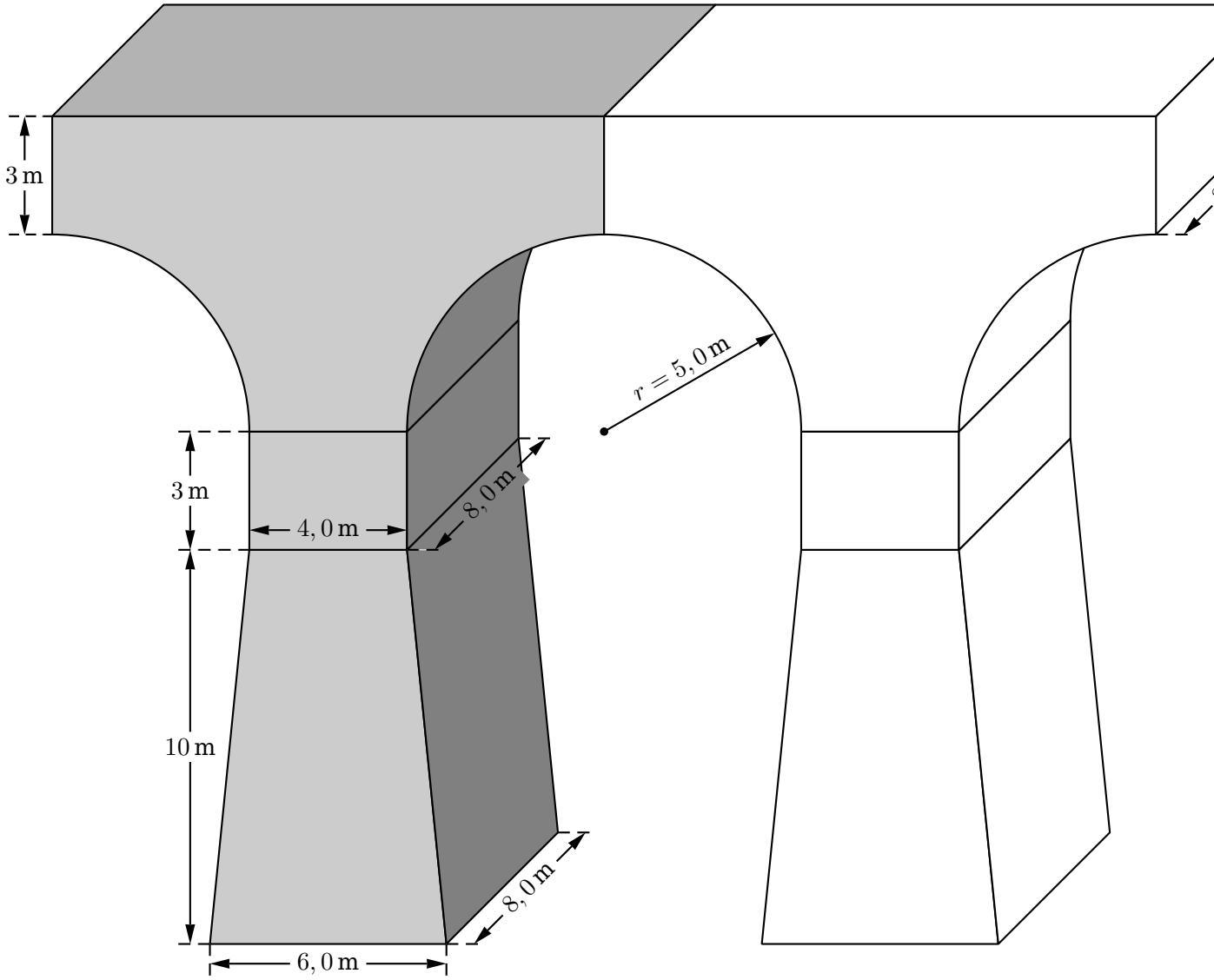
```
\psscalebox{0.6}{%
\begin{pspicture}[showgrid=false](-2,-2.5)(6,6)
\psset{xMin=0,yMin=0,zMin=0,xMax=11,yMax=11,zMax=4,Alpha=155,Beta=20,Dx=1,Dy=1,Dz=1,
    arrowsize=.2,arrowinset=0.1,coorType=1,xThreeDunit=0.5,phi=180 155 sub}%
\pstThreeDPlaneGrid[planeGrid=xy,linewidth=0.3pt,linecolor=gray!70,xsubticks=7,ysubticks
    =7](0,0)(7,7)%
%---------------------------------------
\multido{\i=0+1}{5}{%
\multido{\n=0+1}{5}{%
\multido{\r=0+1}{2}{%
\psboxTS(\i,\n,\r){1}{1}{1}{blue}}}}
%---------------------------------------
\multido{\i=0+1}{2}{%
\multido{\n=0+1}{5}{%
\multido{\r=2+1}{3}{%
\psboxTS(\i,\n,\r){1}{1}{1}{blue}}}}
%---------------------------------------
\multido{\i=2+1}{3}{%
\multido{\n=0+1}{2}{%
\multido{\r=2+1}{3}{%
\psboxTS(\i,\n,\r){1}{1}{1}{blue}}}}
%\pstPlanePut[plane=xy](6,1,0){\fbox{\Huge\red xy plane}}
\end{pspicture}
}
```

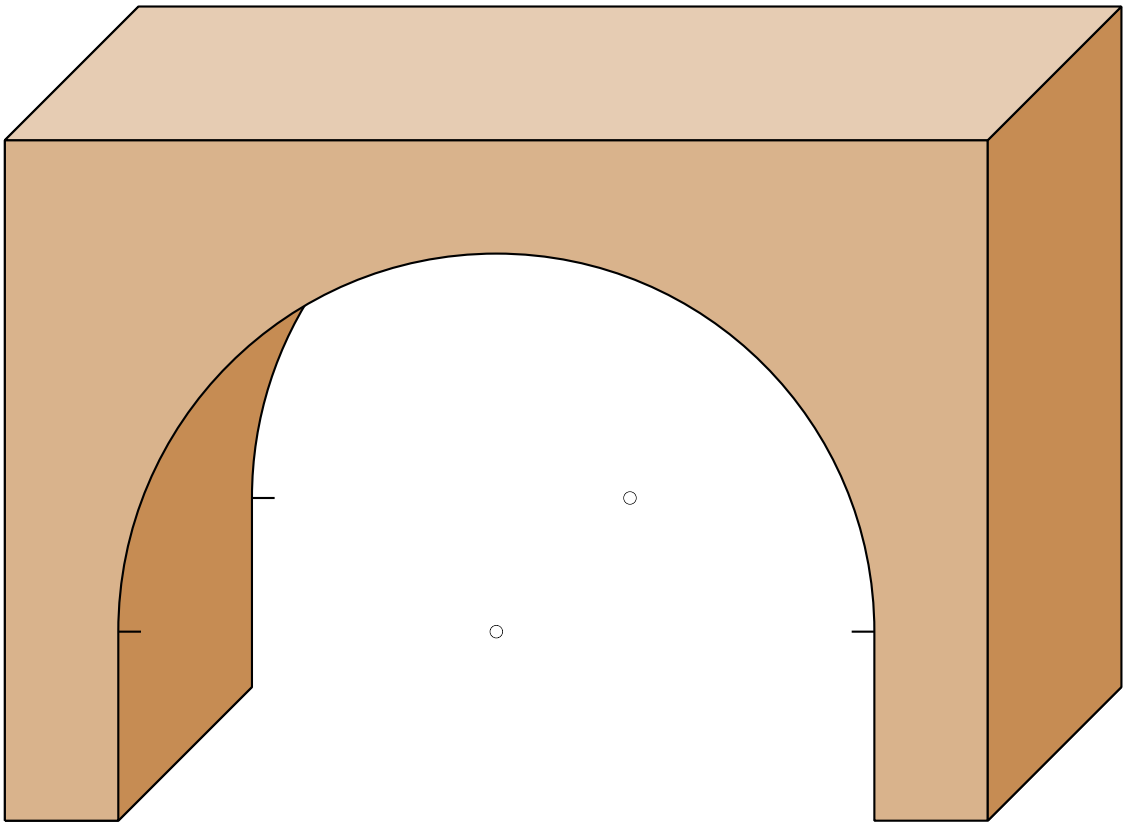


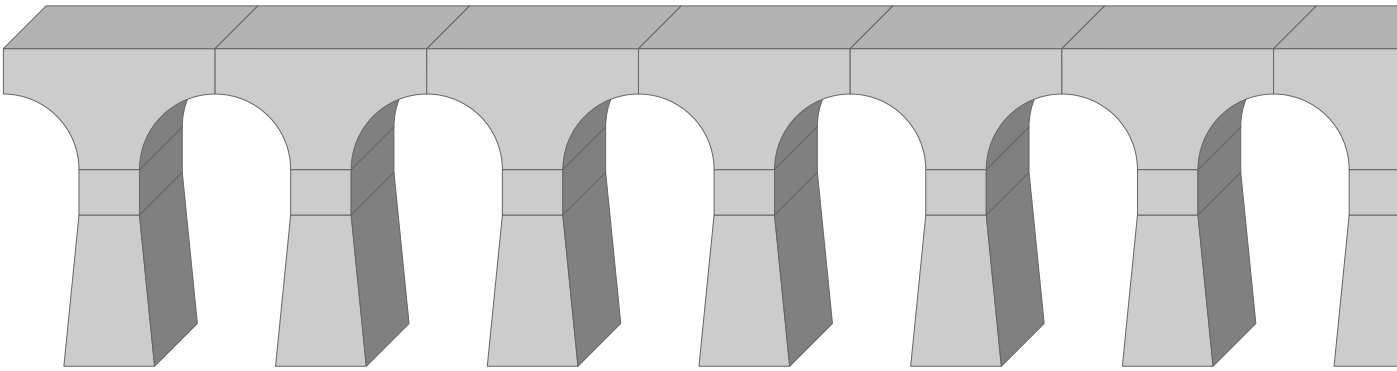
```
\psscalebox{0.6}{%
\begin{pspicture}[showgrid=false](-2,-2.5)(6,6)
\psset{xMin=0,yMin=0,zMin=0,xMax=11,yMax=11,zMax=4,Alpha=135,Beta=20,Dx=1,Dy=1,Dz=1,
    arrowsize=.2,arrowinset=0.1,coorType=1,xThreeDunit=0.5,opacity=0.4,hideline=true}%\
\pstThreeDPlaneGrid[planeGrid=xy,linewidth=0.3pt,linecolor=gray!70,xsubticks=7,ysubticks
    =7](0,0)(7,7)%
%---------------------------------------
\multido{\i=0+1}{6}{%
\multido{\n=0+1}{6}{%
\multido{\r=0+1}{2}{%
\psboxTS(\i,\n,\r){1}{1}{1}{green}}}}
%---------------------------------------
\multido{\i=1+1}{4}{%
\multido{\n=1+1}{4}{%
\multido{\r=2+1}{2}{%
\psboxTS(\i,\n,\r){1}{1}{1}{green}}}}
%---------------------------------------
\multido{\i=2+1}{2}{%
\multido{\n=2+1}{2}{%
\multido{\r=4+1}{2}{%
\psboxTS(\i,\n,\r){1}{1}{1}{green}}}}
\end{pspicture}
}
```


20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
-1
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24




3 m
3 m
4,0 m
8,0 m
r = 5,0 m
10 m
8,0 m
6,0 m





10 List of all optional arguments for pst-perspective

Key	Type	Default
translineA	boolean	false
translineB	boolean	false
translineK	boolean	true
hideline	boolean	false
differentcol	boolean	false
LowPoint	boolean	false
symX	boolean	true
topfillstyle	ordinary	solid
topangle	ordinary	45
topmidpoint	ordinary	0.5
toplinecolor	ordinary	black
topfillcolor	ordinary	gray
toplinewidth	ordinary	0.8pt
hidecolor	ordinary	black!80
facecolorT	ordinary	yellow
facecolorR	ordinary	red
hidelinestyle	ordinary	solid
hidelinewidth	ordinary	0.7pt
transcolor	ordinary	blue
transAcolor	ordinary	green
transBcolor	ordinary	blue
translinestyle	ordinary	solid
transAlinestyle	ordinary	dashed
transBlinestyle	ordinary	dashed
translinewidth	ordinary	0.8pt
transAlinewidth	ordinary	0.7pt
transBlinewidth	ordinary	0.7pt
LowP	ordinary	0|0
originT	ordinary	0|0
base	ordinary	0
symline	ordinary	0
phi	ordinary	45
deltaphi	ordinary	0
vkf	ordinary	0.5

References

[1] Denis Girou. Présentation de PSTricks. *Cahier GUTenberg*, 16:21–70, April 1994.

[2] Michel Goosens, Frank Mittelbach, Sebastian Rahtz, Denis Roegel, and Herbert Voß. *The LaTeX Graphics Companion*. Addison-Wesley Publishing Company, Reading, Mass., 2007.

[3] Laura E. Jackson and Herbert Voß. Die Plot-Funktionen von `pst-plot`. *Die TEXnische Komödie*, 2/02:27–34, June 2002.

[4] Nikolai G. Kollock. *PostScript richtig eingesetzt: vom Konzept zum praktischen Einsatz*. IWT, Vaterstetten, 1989.

[5] Herbert Voß. *Chaos und Fraktale selbst programmieren: von Mandelbrotmengen über Farbmanipulationen zur perfekten Darstellung*. Franzis Verlag, Poing, 1994.

[6] Herbert Voß. Die mathematischen Funktionen von PostScript. *Die TEXnische Komödie*, 1/02, March 2002.

[7] Herbert Voß. *`PSTricks` – Grafik für TEX und LATEX*. DANTE – Lehmanns, Heidelberg/Berlin, 6. edition, 2010.

[8] Herbert Voß. *Typesetting mathematics with LATEX*. UIT, Cambridge, 2010.

[9] Herbert Voß. *PSTricks – Graphics for TEX and LATEX*. UIT, Cambridge, 2011.

[10] Herbert Voß. *LATEX quick reference*. UIT, Cambridge, 2012.

[11] Eric Weisstein. *Wolfram MathWorld*. http://mathworld.wolfram.com, 2007.

[12] Timothy van Zandt. *PSTricks - `PostScript` macros for generic TEX*. http://www.tug.org/application/PSTricks, 1993.

[13] Timothy van Zandt. *`multido.tex` - a loop macro, that supports fixed-point addition*. CTAN:/graphics/pstricks/generic/multido.tex, 1997.

[14] Timothy van Zandt. *`pst-plot`: Plotting two dimensional functions and data*. CTAN:graphics/pstricks/generic/pst-plot.tex, 1999.

[15] Timothy van Zandt and Denis Girou. Inside PSTricks. *TUGboat*, 15:239–246, September 1994.