Including Graphics Figures in Documents With PCTeX and Linux LATeX

James D Emery

Edited: 12/19/2014

Contents

0.1 Installation of PCTeX

The first time PCTex is started, one must run INITex from the menu (to choose between Plain TeX, Latex, or AMSTeX). Choose Latex. When INITex is run, Version 4 will complain that the source is more than 1 year old. Ignore this. Go to settings, select Default settings, select advanced. Under method for rendering DVI, choose Postscript. Go back. Select Postscript. For color depth, select full color.

0.2 The PostScript Mode in PCTeX

Later versions of PCTeX have a PostScript mode (PCTeX32 V4). This is not the default mode. To change the mode to PostScript we select settings, then advanced, then PostScript. When in PostScript mode, and when the program dvips is selected for viewing, a PostScript file is created from the dvi file. This occurs when one views the dvi file, or when one creates the dvi file by choosing typeset. The PostScript file is viewed using a built in PostScript viewer. This all happens automatically when the typeset button is selected. The default PostScript viewer setting is black and white. So a figure in gray scale, or in color, will usually not look good. The setting can be changed to gray scale, or to color, by selecting Settings, Default Settings, and then PostScript.

A LaTeX file that contains embedded Postscript figures, may be processed with the macro **psfig**. For **psfig** to work in PCTeX, the PostScript file for each figure must have certain characteristics that may differ from those required in older UNIX TeX and **psfig** versions. PCTeX is less forgiving concerning the characteristics of the PostScript file than are the Unix programs that I used in the past. The first character of the PostScript file must not be a space. The file should start with

```
%!PS
```

A bounding box statement must be present in the PostScript file that contains the figures, and should look something like

```
%%BoundingBox: 0 50 450 500
```

If one finds that the figure is not centered properly, then it can be adjusted by changing the bounding box parameters. So in this example 0 50 will be

the lower left corner of the PostScript figure, and 450 500 will be the upper right corner. The scale is 72 points to the inch, unless there is a scale change in the file. This all corresponds to how the PostScript file appears on the printed page before it is inserted into the document. So the lower left corner of the PostScript page is 0 0. The figure will be scaled to fit into the space allocated in the document by the psfig call, say in a 3 inch vertical space.

If one has a PostScript file with no bounding box, then a crude way of finding a bounding box would be to print the page and measure a rectangle that encloses the figure on the page, using the scale 72 points to the inch.

0.3 Saving A Typeset Document as PostScript

When in PostScript mode, and when viewing the typeset output, we will see a PostScript file. This screen display comes from the PCTeX built in PostScript viewer. When we choose to view the dvi file, it will be first converted to PostScript. We can choose **saveas** from the PCTeX menu and save the output as either PostScript or dvi (the dvi will always be saved). Alternately, we can select **Print Setup** and choose a PostScript printer driver and then select the **print to file** option. And then choose **Print**. This will cause the selected PostScript printer driver to be used to create the PostScript file. So the two methods will probably produce somewhat different files. One might try both methods and compare the result. The print method will work for earlier versions of PCTeX that did not have a PostScript mode. And in fact is a method of getting PostScript output from any Windows application.

0.4 Including a PostScript Figure

The following PostScript file **nestedrepeats.ps** is included in a figure in the following way. This Postcript algorithm draws a cubic Bezier curve defined with control points (0,0) (72,72) (72,-72) (288,0). It rotates and repeats this sixteen times. Then it does a reflection and then does the 16 rotations again. This creates a flower like PostScript graphic.

```
%!PS
%%BoundingBox: 0 50 450 500
%%EndComments
```

```
%Nested repeats 6-8, from "Learning PostScript"
306 396 translate
2{
16{
0 0 moveto
72 72 72 -72 288 0 curveto
360 16 div rotate
}repeat
-1 1 scale
}repeat
stroke
%eofill
1 setgray
0 0 10 0 360 arc fill
showpage
```

The figure is included with the macro psfig. The following code is embedded in the Latex file.

```
\begin{figure}
%\psfig{figure=/tex/nestedrepeats.ps,height=3in}
\psfig{figure=nestedrepeats.ps,height=5in}
\caption{ A PostScript Graphic.}
\end{figure}
```

The psfig macro must be included in the Latex document. This is done with the following command, which appears as the second line of the tex file.

```
\input{psfig.sty}
```

0.5 Including A Plot of a function Using psfig Under PCTeX

The PostScript plot file of the function was produced from the programs **plotf.cpp** and **eg2ps.c**. **plotf.cpp** is a program to plot a function and to create an output file in the eg format. **eg2ps.c** converts the eg file to a PostScript file. I might mention that there is a program called **pltax.c** that

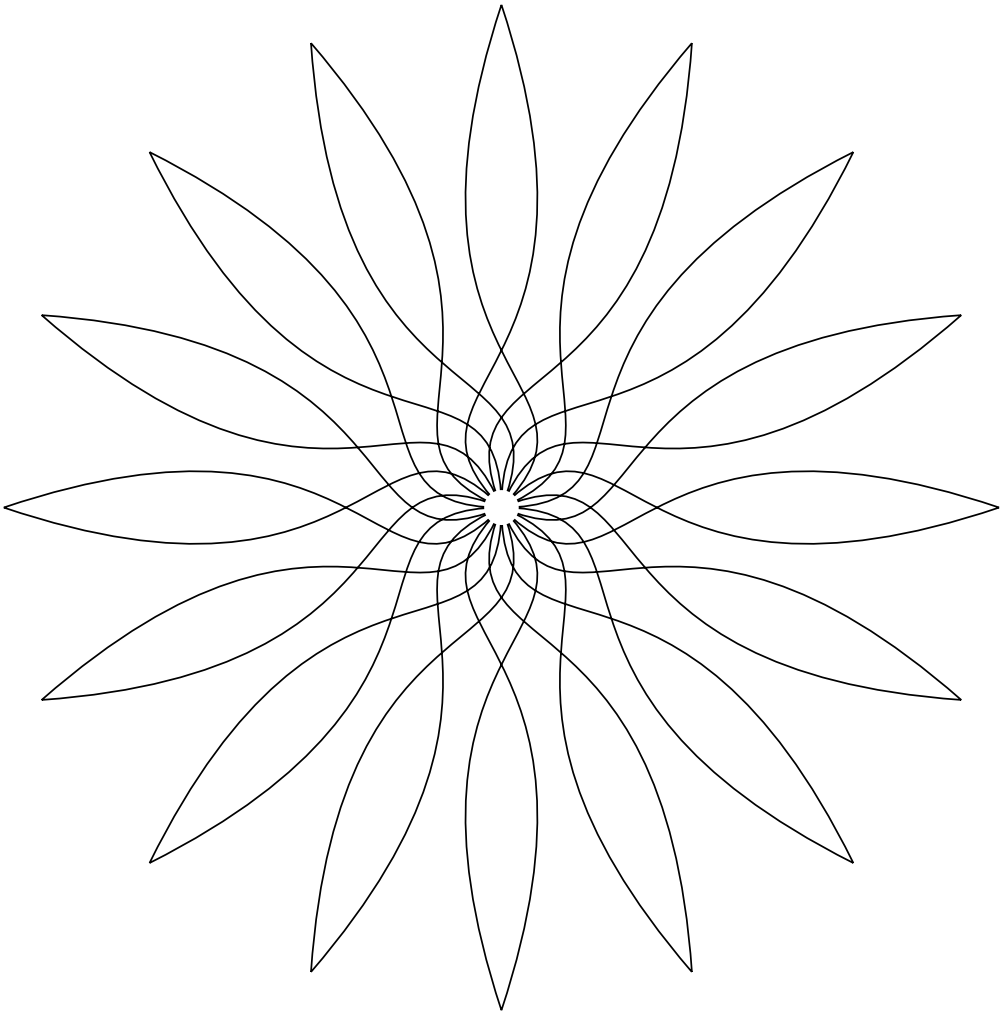


Figure 1: A PostScript Graphic.

adds axes and labels to xy data to create an EG function plot. A previous version of the program **eg2ps.c** would output the BoundingBox line without a double percentage sign, and without an ending ":". The Unix version of dvips that I used on an Apollo workstation accepted this. When PCTeX did not, I realized that **eg2ps** was generating incorrect PostScript. This has been corrected so that the start of the PostScript file, is created by eg2ps, is something like:

```
%!PS
%%BoundingBox: 0 50 450 500
%%Creator: eg2ps.c by Jim Emery
%%EndComments
72 300 div 72 300 div scale
100 100 translate
3 setlinewidth
newpath
186 790 moveto
200 846 lineto
214 900 lineto
228 954 lineto
241 1006 lineto
255 1057 lineto
```

NOTE. A PostScript error can occur in some systems if the (control)z left by the editor Kedit, which is an editor that I sometimes use, is at the end of the PostScript file for the figure. In particular, if an included PostScript figure contains a (cntrl)z, then when the LaTeX document, which is typeset as PostScript, is sent to the Hewlett Packard LaserJet 4mp, the printer will stop at the figure because of the (cntrl)z. In the first versions of DOS a (cntrl)z indicated end of file.

Recall that the placement of the figure is controlled by the "Bounding-Box" line of the PostScript file.

The figure is handled by a "figure" macro. This TeX macro is called psfig. A figure is placed as a floating element. It will not be placed by the program in a definite place. This can be a bit frustrating, especially if the figure is placed in the middle of some verbatim text list. There are TeX commands to control this placement. One should consult the TeX manual for information. To use psfig, the beginning of the tex file must contain a line that brings the

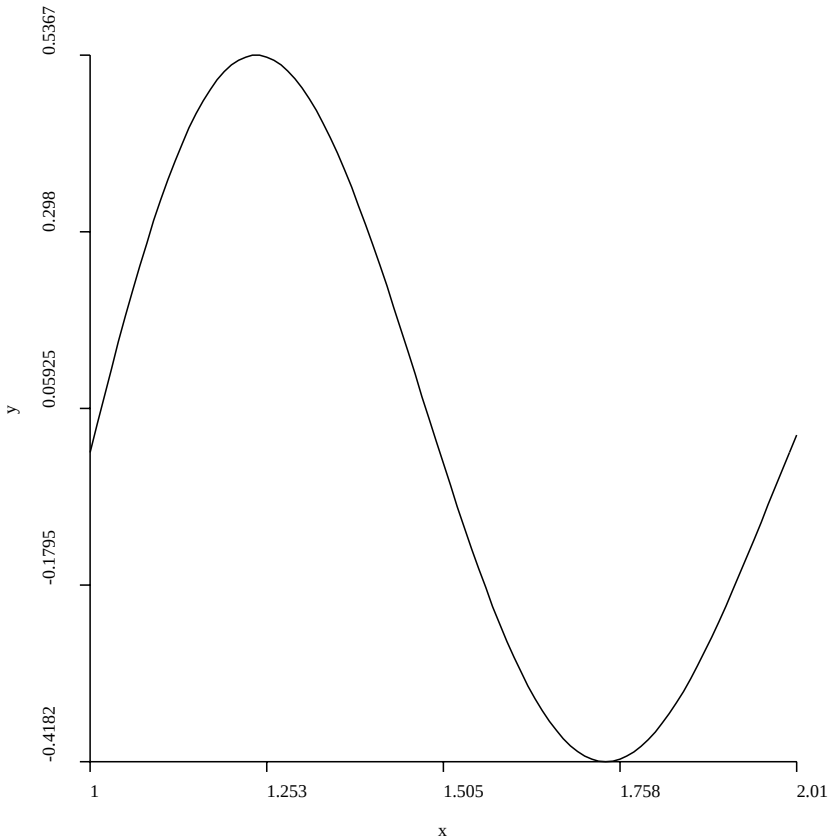


Figure 2: A plot of a function.

psfig macro into the TeX document. Here are the first few lines of a file that uses psfig.

```
\documentclass{report}
\input{psfig.sty}
\title{Figures in PCTeX}
\author{Jim Emery}
\date{November 21, 2000}
\begin{document}
\maketitle
```

The figure itself is produced with the code:

```
\begin{figure}
\psfig{figure=/tex/fun1.ps,height=3in}
\caption{ A plot of a function.}
\end{figure}
```

Notice that the forward slash is used to delimit directories. This is the convention for Unix systems. The PC uses the back slash. In order to be consistent the forward slash is used for both systems by psfig. If the file fun1.ps is in the same directory as the source file pcldst.tex, then PCTEX should find the source file without the directory specification.

0.6 Using Acrobat Distiller to Make a PDF

To print a file containing PostScript figures on a non postscript printer we can use Acrobat Distiller. To do this make the default windows printer some PostScript printer such as the HP Laserjet 6mp. Then select print, which first does a file open, and then gives a printer dialog box, in which the "print to file" option should be checked. The default print file name will have a .prn extension, which should be changed to .ps so that distiller can recognize it as a PostScript file. So suppose a print file called myfile.ps is generated. Then from windows double click this file, or from DOS type start myfile.ps, and Distiller should go to work producing a file called myfile.pdf. Then double click the file myfile.pdf and Acrobat Reader or Acrobat exchange will display the file, from which the file can be printed to any printer.

0.7 Including a Windows Bitmap Figure With PCTeX

The disadvantage to using a windows bitmap file instead of a PostScript file is usually lower resolution unless the bitmap size is set very high. A bitmap will be very large as compared to a PostScript file, which just contains a program to draw the figure. Sometimes people think that PostScript files are bigger than other files. But this is a misunderstanding of PostScript. Some graphics programs have the ability to output a bitmap file to a PostScript file. But this takes advantage of the ability of PostScript to include a bitmap inside of itself. In fact when this is done, the PostScript file is bigger than the original, because a PostScript file is a text file, and each pixel is represented as two hexidecimal characters. Here is code for a windows bitmap figure example:

```
\begin{figure}
\vskip 6in
\special{bmp:/je/tex/front24.bmp y=6in}
\caption{Front view of gem.}
\end{figure}
```

It does not need an include file. The TEX command "special" allows placing some content external to TEX in a dvi document. Actually the psfig macro works by using the TeX "special" command. The bmp figures, depending on how they are saved and compressed in Paint, can be rendered as black and white. To make Acrobat convert a bmp to shaded colors, we should save the bmp file as 24 bit color.

0.8 Another Method of Placement: An Optics Figure

This figure is from the document **optics.tex**. It is placed as described in the PCTeX manual, were the description is very inadequate. One might have much trouble in making **psfig** work with the description given in the PCTeX manual. Notice that there is no begin figure and no end figure. Also there is no caption. Sometimes when using this method there is not enough space to place the figure. I do not recommend this method.

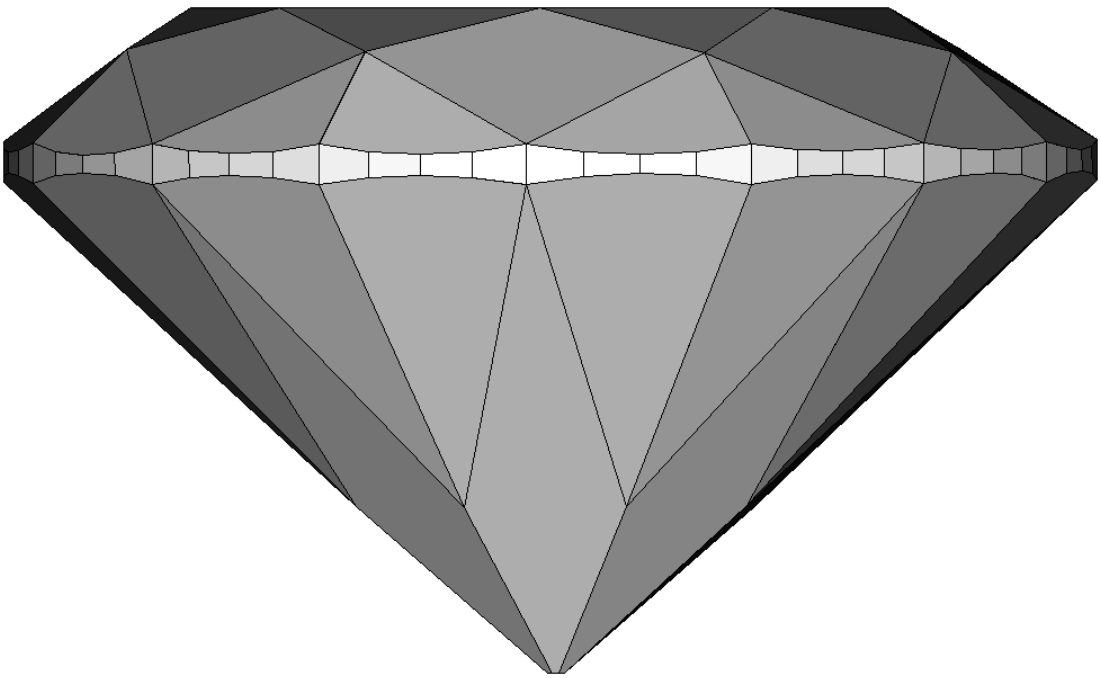
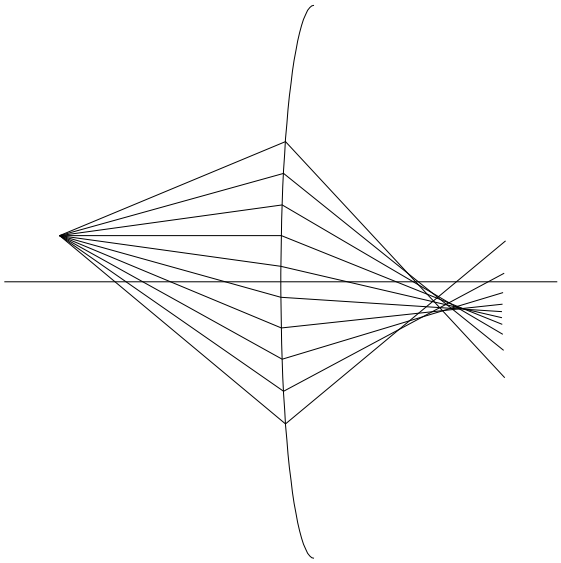


Figure 3: Front view of gem.

The code for this figure is:

```
\par
\psfig{figure=/je/tex/opticsf1.ps,height=3in}
\par
```



0.9 Making an Acrobat PDF

To make a PDF we need a program that converts PostScript to Acrobat PDF. One such program from Adobe is called Distiller. It comes with Acrobat 4.0. First we must save the typset document in PostScript. We can do this in two ways. In the first way, while we are viewing the dvi file in PostScript mode, we can do a saveas, and choose the PostScript file format.

For earlier versions of PCTeX that did not have a PostScript mode, we can do the following. We do a print to file in PC tex. We select a PostScript

printer such as the HP 6MP/PS. We save the print file as p.ps. We do the command "start p.ps", which will call Acrobat Distiller, which will convert to PDF. A PDF file has a built in compression, so it will compress a large bitmap into a much smaller file.

0.10 A Figure Showing Planck's Blackbody Radiation Curves

See the report **The Sea Ice Project**, file name: **nic.tex**. Here is the code for this included PostScript figure:

```
\begin{figure}
\psfig{figure=/je//tex/planck.ps,height=5in}
\caption{ Blackbody radiation curves, energy density per unit wavelength.}
\end{figure}
```

0.11 Using psfig Under Linux and Unix

The use of psfig under Linux and Unix is quite similar to its use under PCTeX. The required include line may be slightly different.

Under some versions of LaTeX on a Unix system, one does not include the sty file extension in the input psfig line. Some Unix systems may still require no file extension. The TeX "input" command and the "include" command both insert text into the current TeX document. The "include" command apparently will not recognize the sty file extension. So that when LaTeX reads

```
\include{psfig.sty}
```

it searches for the file psfig.sty.tex, which it will not find. That is, "include" always tags on the tex extension. This behavior may have something to do with the new version of LaTeX.

Here is an example of the psfig input line from the document called **plcdst.tex**. Note that under the new version of LaTeX one replaces documentstyle with documentclass.

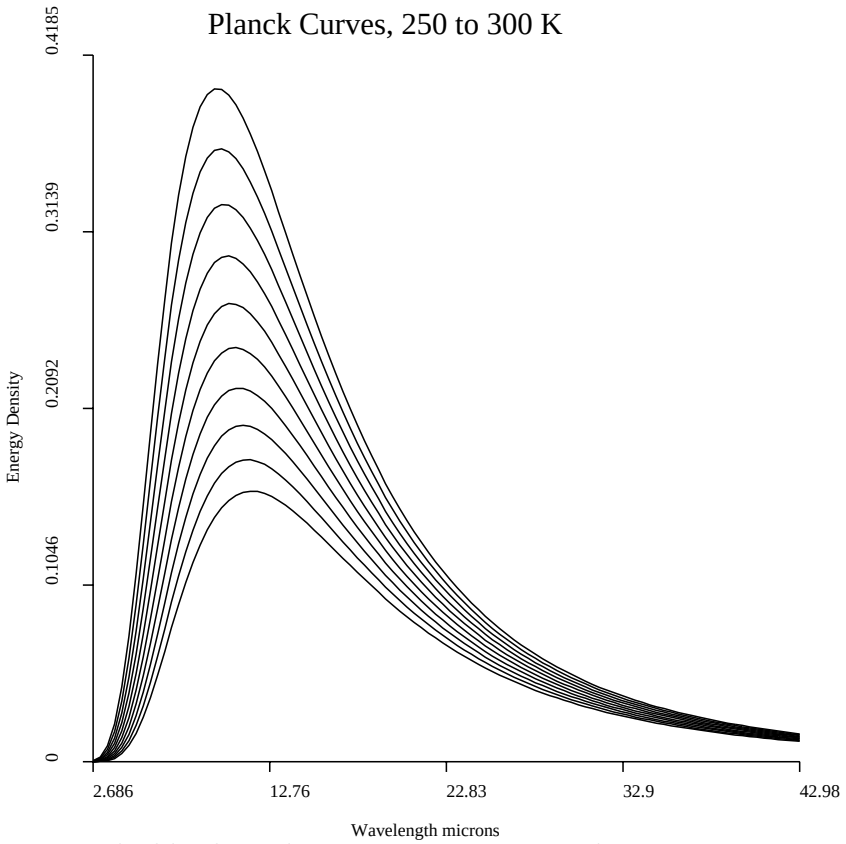


Figure 4: Blackbody radiation curves, energy density per unit wavelength.

```
\documentstyle[12pt]{report}
\input{psfig.sty}
\title{Calculating Distance Between Plane Curves}
\author{Jim Emery}
\date{Revised April 28, 1992}
\begin{document}
\maketitle
\tableofcontents
\section{Introduction}
.....................
.....................
\begin{figure}
%vspace{3.0in}
\psfig{figure=plcdstf2.ps,height=3.in}
\caption{The extreme point lies on a segment bisector.}
\end{figure}
.....................
.....................
\end{document}
```

Recall that in the PCTeX case the include line could have been

```
\input{psfig.sty}
```

and so the psfig line could have specified the directory using the Unix forward slash delimiter.

```
\psfig{figure=/je/tex/plcdstf2.ps,height=3.in}
```

The Linux version of psfig will use the psfig.sty form also, but some older Unix versions may not, and may only accept the file name without the sty extension.

The user of TeX, as well as the user of most software, must be an empiricist. He must experiment, he must probe the black box to divine its behavior. That is why they call it Computer Science rather than Computer Logic. The best way to teach students about gravity is to throw them from a cliff. The best way to find high caliber students is to take them to the circus and fire them from a cannon.

0.12 Generating A PostScript File in Linux and Unix

To generate a PostScript output file from the dvi file generated by Latex, we can use the program **dvips**. Under Linux we type

```
dvips -f myfile.dvi > myfile.ps
```

0.13 Viewers

Document viewing must be done in X-Windows on a Linux system., First we must start X-Windows with the command: **startx**. A dvi viewer comes on the Linux system that is called **xdvi**. To run it we should type:

```
xdvi myfile.dvi
```

We shall see no figures using **xdvi**, but only blank space where each figure should go. To view the document with figures, we must use a PostScript viewer such as **GhostScript**. To run **GhostScript** on a Linux system, from the command line, we type: **ghostview**. Alternately, we may choose the run menu in the Gnome graphics manager. We type ghostview in the run box. Or we may double click the Ghostscript icon, if we can find it. The PhotoShop clone, which is called Gimp, and which comes with Linux, will also view a PostScript file.

Another method of viewing is to convert the PostScript file to PDF using Acrobat distiller. Then the PDF file may be viewed with Acrobat reader.

A third method is to view the file with a graphics program that supports PostScript. The graphics program CorelDraw8 will display a PostScript file. The PCTeX program itself has a built in PostScript viewer, so files with the ps extension can be opened and viewed in PCTeX. For that to work with a single figure, the PostScript file may need a bounding box declaration. Recall that in a PostScript file there are 72 points to the inch, and the bounding box, which is defined by the lower left corner coordinates and the upper right corner coordinates, should bound the PostScript graphic. PostScript uses a conventional coordinate system with the origin at the lower left corner of the page. PostScript always has a current transformation matrix, and it is applied to all coordinates. This transformation is a combined scaling,

rotating, and translating. Initally it will be the identity transformation. But, for example, if the scale factors have been changed, then the 72 points to the inch may no longer apply.

The default PostScript viewing setting is black and white. This can be changed to gray scale or to color, by using the default setting menu.

0.14 A Figure Created in a Drawing Program

This geodetic coordinates figure was created in CorelDraw. It was exported from CorelDraw as an encapsulated PostScript file (eps). It was saved with the name **ellgen.ps**. When saved from CorelDraw as a PostScript print file, there is some binary data placed at the beginning of the file, and at the end of the file. Although some PostScript viewers, including Ghostscript on Linux, displayed the geodetic figure correctly, PCTeX had a problem with the file. But when the file is exported as an eps file, the exported eps file worked fine.

CorelDraw creates elaborate PostScript dictionaries, so the output PostScript is quite unreadable. Early versions of Adobe Illustrator made more readable PostScript output files. The button "outputtextascurves" was checked in this eps export. This is probably not necessary, but if one does this no external fonts will be needed by the printer or viewer. This PostScript file **ellgen.ps**, which should be about 2k in length, became about 38k in size. This is because of all of the excess general garbage that CorelDraw adds to an eps file.

0.15 A Figure Created Using EG Graphics

The following illustration of an electromechanical device was created using the EG graphics commands. It was created by running the programs winedit, arcs.cpp, lines.cpp, and wineg.cpp, simultaneously. This constitutes a poor man's CAD system. I used to use the program AutoSketch, save as DXF, and use a program I wrote to convert from DXF to EG. But AutoSketch is a pretty weak program. I suppose I should write my own CAD system. Commercial systems never seem to do exactly what one wants. The file is called **magdevf1.eg**. It was then converted to PostScript using **eg2ps**. This produced the file **magdevf1.ps**. It is quite laborious to create a graphic by manually assembling EG (Emery Graphics) commands. But it is possible.

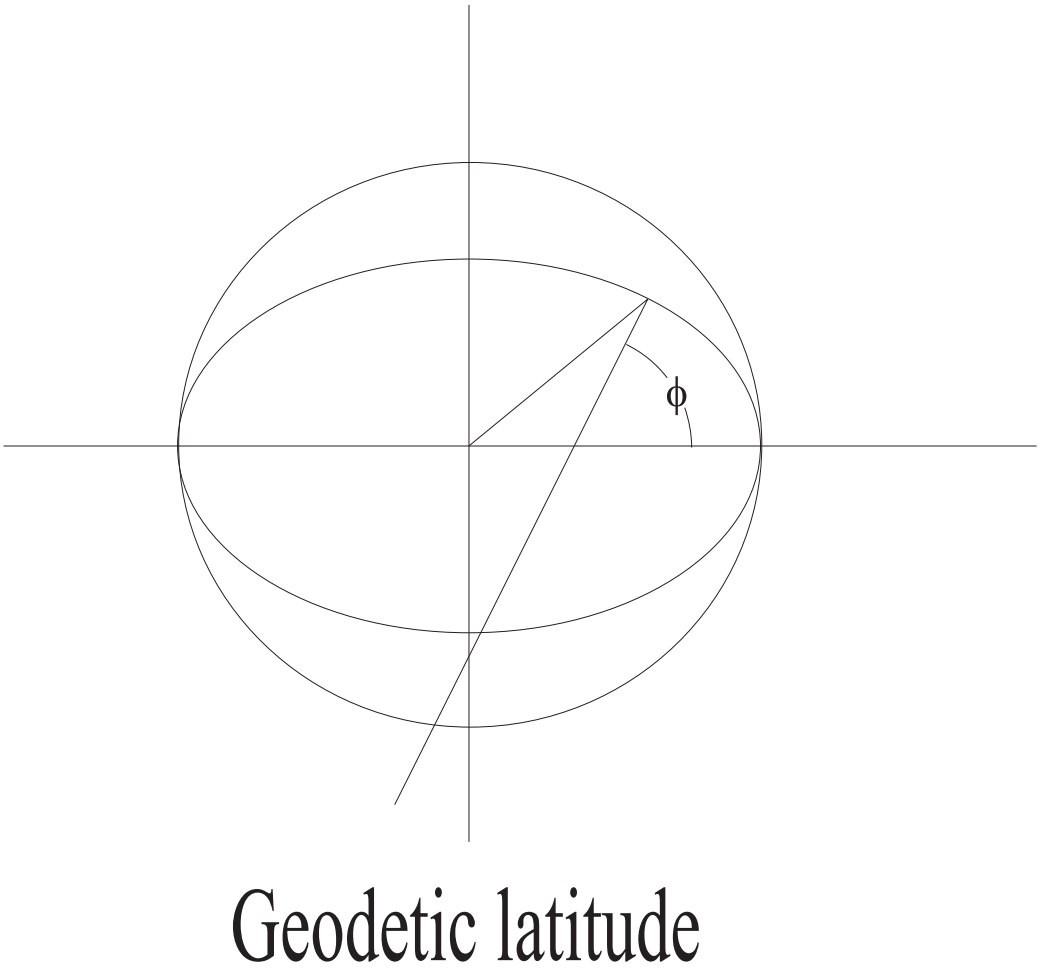


Figure 5: The Geodetic latitude ϕ differs from the spherical coordinate latitude as the sphere is deformed to the prolate ellipsoid.

Here is the file **magdevf1.eg**:

```
v-1 1 -1 1
w0 8 3 11
a 4.0614067 7.3728812 1.604951 -3.97 .65 100
a 4.0614067 7.3728812 1.604951  1.118 1.54 100

%a 4.0572182 7.373701 2.0830612 -3.1415926 3.1415926 100
a 4.0572182 7.373701 2.0830612 -3.66 .49 100
a 4.0572182 7.373701 2.0830612 .645 .91 100

a 2.38432 8.74626 0.35345971 1.8058527 5.1559178 100
m 2.302 9.09
d 3.975 9.557
m 2.536 8.427
d 3.731 8.767
a 2.38432 8.74626 0.083000015 -3.14159 3.14159 100
a 3.7660643 8.6591609 0.11339655 -0.52831593 1.885167 100
m 3.864 8.602
d 3.598 7.991
d 3.983 7.821
a 4.2167804 7.6665492 0.28019335 2.5577556 -0.20686316 100
m 4.491 7.609
d 4.806 7.466
d 5.179 8.295
%filet
a 5.2595341 8.2440431 0.09530133 1.2343097 2.5774492 100
m5.251 8.334
d             5.7           8.302

%slot
a 5.6876776 8.923197 0.62131922 -1.5509624 1.5300292 100
a 5.6876776 8.923197 0.3 -1.5707 1.5707 100
a 4.422 8.923197 0.3 1.5707 4.7123889 100
m 4.422 8.623197
d 5.6876776 8.623197
m 4.422 9.223197
d 5.6876776 9.223197
```

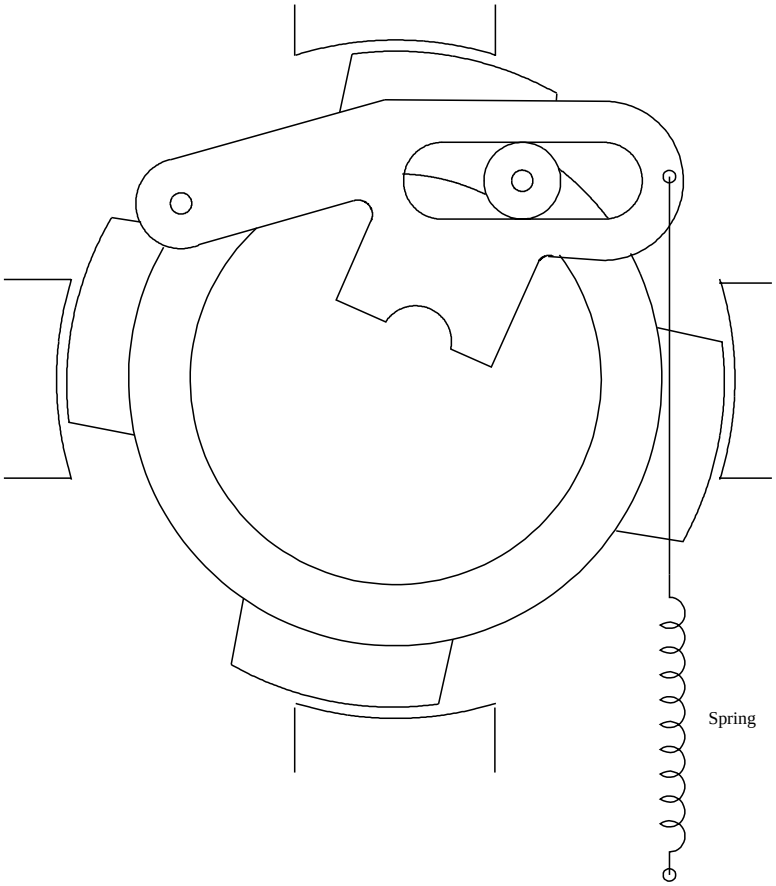


Figure 6: An Electromechanical Device.

```
a 5.05 8.923197 0.3 -3.1415926 3.145926 100
a 5.05 8.923197 0.083 -3.1415926 3.145926 100

%tab1
m 2.87315.8 5.6615.8
d 2.77515.8 5.14415.8
m 4.50815.8 5.3415.8
d 4.39715.8 4.83615.8
a 4.0392853 7.3771063 2.5661607 -2.0859583 -1.430944 100
m           5.713           9.544
d 3.975 9.557

%tab2
m 5.786 6.188
d           6.303           6.102
a 4.0577535 7.3542586 2.5708527 -0.50876444 0.12038355 100

m            6.61           7.663
d 6.106 7.774


%tab3
a 4.061476 7.3727854 2.5629308 1.0579509 1.7040366 100
m           5.319           9.606
d 5.316 9.547
m           3.721           9.913
d 3.623 9.46

%tab4

m           1.841           8.634
d           2.073 8.6
a 4.0547086 7.3405925 2.563866 2.6128183 3.2606759 100

m           1.509           7.036
d 2.025 6.938
```

```
%stator 1
m       6.5863503        6.6
d       7        6.6
%a 4.0547086 7.3405925 2.65 -.3 .3 100
a 4.0614067 7.3728812 2.65 -.3 .3 100
m       6.5863503        8.123721
d       7        8.123721
%stator2
m        4.838081       9.9
d        4.838081       10.3
%a 4.0547086 7.3405925 2.65 1.2707 1.87079  100
a 4.0614067 7.3728812 2.65 1.2707 1.87079  100
m       3.2715961       9.9
d       3.2715961       10.3
%stator3
m        1.523069       8.15
d        1.       8.15
%a 4.0547086 7.3405925 2.65 2.84159 3.44159 100
a 4.0614067 7.3728812 2.65 2.84159 3.44159 100
m       1.5230648       6.6
d       1.       6.6
%stator4
m       3.2715961       4.8089458
d       3.2715961       4.3
%a 4.0547086 7.3405925 2.65 -1.87079 -1.27079 100
a 4.0614067 7.3728812 2.65 -1.87079 -1.27079 100
m       4.8378532       4.8089558
d       4.8378532       4.3


%spring



z .03
a  6.2000   8.956 .05 -3.14159 3.14159 20
a  6.2000   3.5   .05 -3.14159 3.14159 20
```

```
m  6.2000       5.8490
d  6.2000       5.6687
a  6.2000       5.5475       .12124       1.5708       -2.2143      20
a  6.2000       5.3535       .12124       2.2143       -2.2143      20
a  6.2000       5.1595       .12124       2.2143       -2.2143      20
a  6.2000       4.9655       .12124       2.2143       -2.2143      20
a  6.2000       4.7715       .12124       2.2143       -2.2143      20
a  6.2000       4.5775       .12124       2.2143       -2.2143      20
a  6.2000       4.3835       .12124       2.2143       -2.2143      20
a  6.2000       4.1895       .12124       2.2143       -2.2143      20
a  6.2000       3.9955       .12124       2.2143       -2.2143      20
a  6.2000       3.8015       .12124       2.2143       -1.5708      20
d  6.2000       3.5000
t  6.5031       4.6745       6 Spring
m  6.2000       8.9560
d  6.2000       5.8490
```

0.16 EG Graphics in Programming.

The best use of EG graphics is to store graphical objects that are created in a programming language. For example, we have several programs to generate surfaces as sets of polygons. The program stores the polygons in EG format. One such program is called **dph.c**. Many such graphics programs can be written. One can create a program to draw an electrical circuit, or to draw other special purpose diagrams. One such program is called **elecschm.ftn**. Creating a general mathematical surface in a commercial drawing program such as Adobe Illustrator is nearly impossible. However, some of the mathematical programs such as Maple, Mathematica, and Matlab can create multidimensional function plots and can export them as PostScript files.

As an example of an EG file written by a program, we present a figure consisting of a triangulated spherical surface. This was created with the

program **decomp.ftn**. The function of the program is to find a good way to distribute points uniformly on a sphere. As is well known it is impossible to position n points on a sphere exactly uniformly, except for special cases, namely the vertex positions of the Platonic solids. A report on this is called **subdpoly.tex** The actual drawing is done with one of the programs with a name like **drwt.ftn**, which draws triangles to an EG file.

0.17 A CorelDraw Figure

This cross section figure was created in CorelDraw and exported as an eps file. The eps extension was changed to ps.

```
\begin{figure}
\psfig{figure=magdevf2.ps,height=5in}
\caption{ Cross section of rotor and stator showing windings.}
\end{figure}
```

0.18 A Scanned Figure

The figure titled "Derivation of spring torque" was scanned as a tiff file at a low resolution (150 pixels per inch). Then it was saved as an EPS file in a graphics program. First I tried to do this with PaintShopPro. But PCTeX did not like the EPS file from PaintShopPro. The reason is, I think, that the bitmap in PostScript is stored as a giant string of hexidecimal pairs. This creates a line of huge length. I speculate that PCTex can not handle such a long line. According to the PostScript documentation, newline characters can be embedded in this giant bitmap string without affecting it. The newlines will be ignored. So the string can be broken into normal size lines. When PhotoShop saves a bitmap as an EPS file, it does this line breaking. I imported the tiff file in CorelDraw, but the resolution was poor, so I moved on. Then I tried Adobe PhotoShop. As I saved the file as an EPS file, I set the bitmap header to "none." After I saved the file as EPS, I changed the file extension to ps. This PostScript file, which I saved from PhotoShop, worked fine in PCTeX.

The bad news is that saving the tiff file in EPS format doubles the size of the file. The good news is that it can be imported with **psfig** as a PostScript file. The advantage of this is that both PCTeX and Linux can handle the

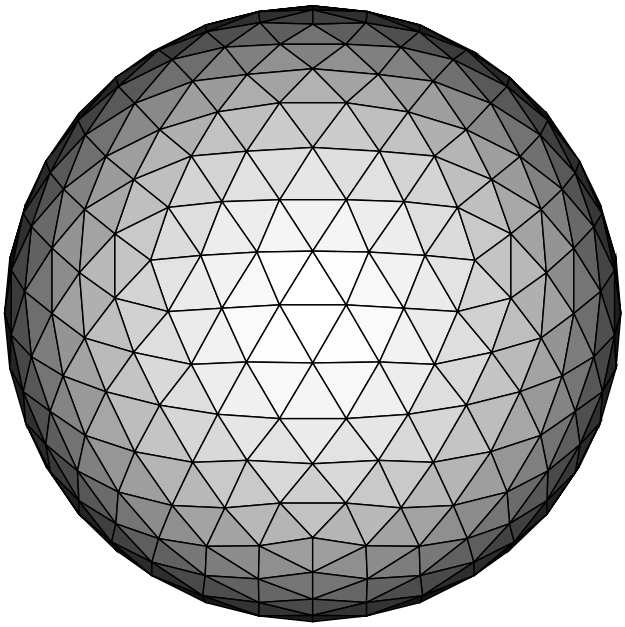


Figure 7: Nearly Uniformly Spaced Points On a Sphere.

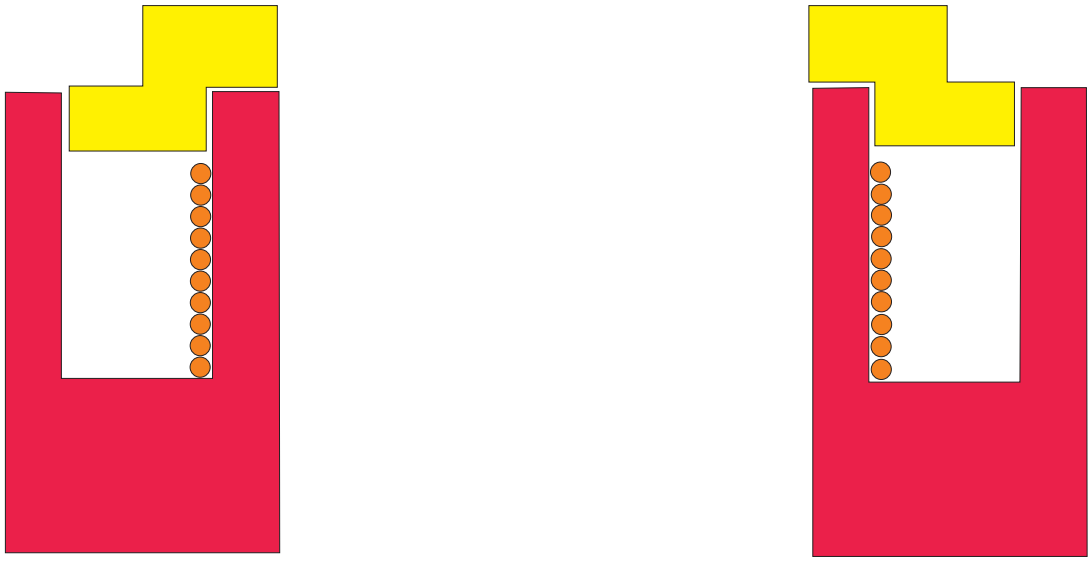


Figure 8: Cross section of rotor and stator showing windings.

figure in the same way. We could have imported the scanned file as a bmp file in PCTeX, but then our Latex file would have not worked in Unix and Linux. Actually Linux dvips will still work, it will simply ignore the bitmap figure. Linux version of **dvips** would have ignored the figure because it only handles PostScript figures. Art historians will think that the figure could use a little cleaning and restoring, and I agree. However, one must be careful. Sometimes it is best to leave antiques alone.

```
\begin{figure}
\psfig{figure=magdevf3.ps,height=5in}
\caption{ Derivation of spring torque.}
\end{figure}
```

0.19 A Projective Geometry Figure

A projective figure showing the representation of a circle as a cone of lines was created with a Fortran program called **conef.ftn**. This program generates a set of polygons and curves. The curves are sets of line segments in 3-space. Program **drwpgnl.ftn** converts the polyhedron and the lines to EG graphics using a perspective projection. The EG file is converted to Postscript with **eg2ps.c**. The final file is called **quadricf1.ps**, being a figure in the document called **Conics, Quadrics, and Projective Space**. The source file for the document is **quadric.tex**. The plane with the hole consists of an outer curve and an inner curve. The only real trick is to make sure that the inner and outer curves have opposite orientations, so that the PostScript fill command will determine inside and outside correctly. PostScript uses a winding number calculation for this. **drwpgnl.ftn** only adds PostScript fill commands for polygons, so a setgray, a stroke, and a fill command were added to the EG file by hand. Line segments read by **drwpgnl.ftn** do not necessarily form closed curves and are not necessarily to be filled, so projected curves are not followed by a PostScript **fill** command.

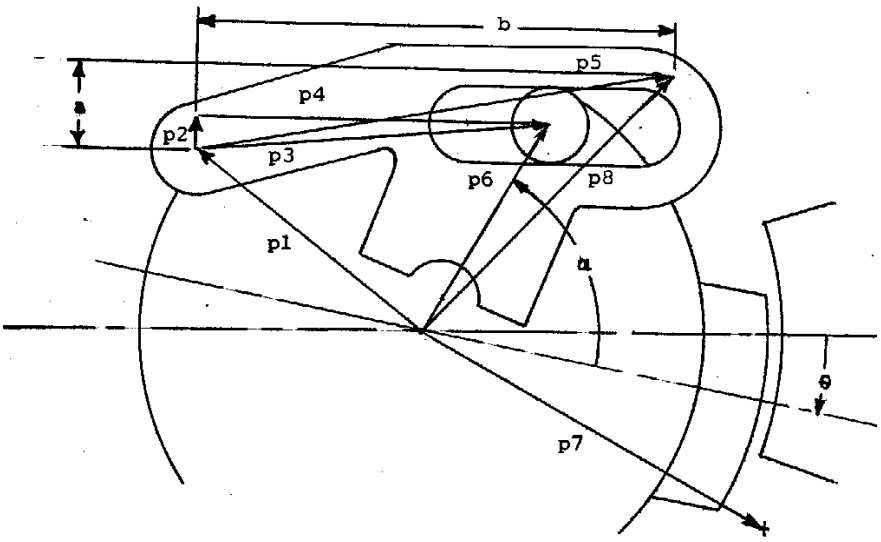


Fig. 2

Figure 9: Derivation of spring torque.

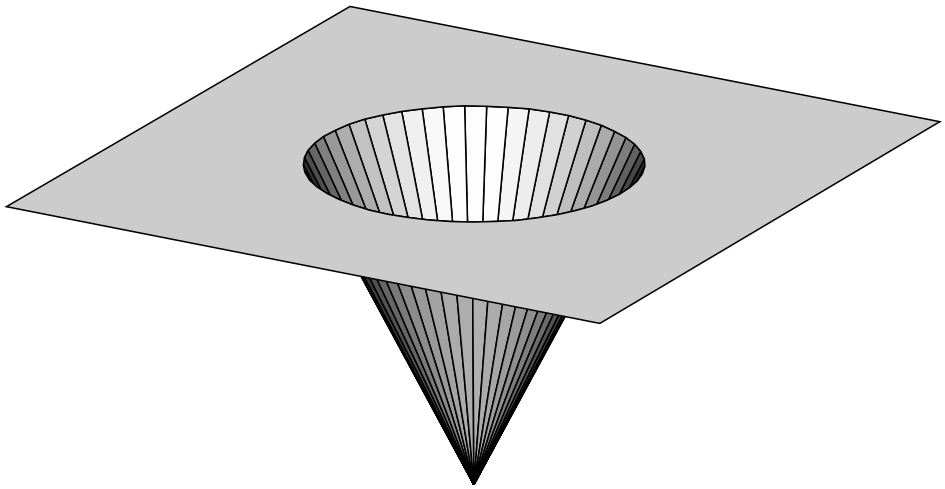


Figure 10: Points of projective 2-space are lines in 3-space. A 3-d linear transformation is a 2-d projective transformation. A rotation of the cone can project the circle to an ellipse, a parabola, or a hyperbola. In the same way the lines through any figure in the plane are a projective space representation of the figure. Any sequence of perspective projections of a figure onto a set of planes can be represented as a sequence of linear transformations in 3-space. The plane shown in the figure is known as the affine plane.

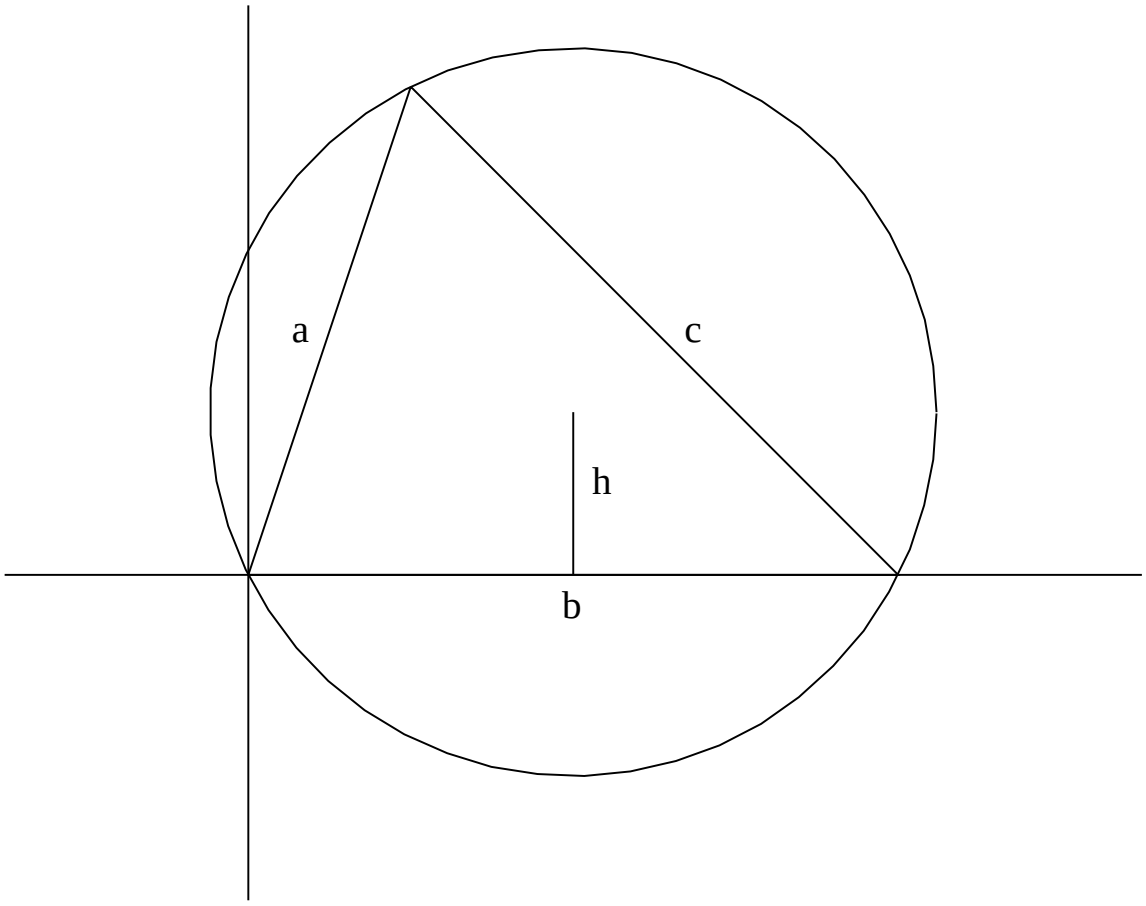


Figure 11: A circle circumscribing A triangle. This is a postscipt figure with text created with **addpstxt.bat**.

0.20 Pasting a Figure Created in DesignCAD into MathCAD

DesignCAD Express Version 12, can read AutoCAD files, and can import and export IGES, which later versions of AutoCAD cannot. To past a drawing from DesignCAD to MathCAD, select the drawing with a bounding box using the mouse. Select copy, this puts it in the clipboard. Click somewhere on the MathCAD sheet, and select paste. Also DesignCAD can export a drawing as a windows meta file wmf. This can be put into a microsoft word document then it can be selected and put in the clipboard and pasted to MathCAD. Also a bitmap file may be placed in MathCAD by selecting picture, and somehow specifying the filename? According to the manual anyway. MathCAD documents may be transferred to microsoft word via the clipboard.

0.21 Creating a Postscript Figure Using AutoCad or DesignCad

To create a postscript figure, follow these steps.

(1) Create the figure in AutoCad, or DesignCad. If created in AutoCad, save as an AutoCad file and read it into DesignCad. Export it as an IGES file.

(2) Use program **iges2eg** to convert the iges file to an eg file.

(3) Use program **pltmerge** to add window and viewport commands to the eg file.

(4) Use program **eg2ps** to convert the eg file to postscript file say p.ps.

(5) Start the postscript file as in s p.ps calling ghostview. Move the curser to a location where one wishes to add letter labels, annotation, et cetera. Note the cursor position. Write down the coordinates in a notebook. Continue for additional text.

(6) Create a text file called tmptxt.txt containing the coordinates of the text from the notebook as in

```
35 245 A
156 346 Note
.........
```

(7) Rename the postscript file to **tmp.ps**. Above the last stroke command add

```
:include tmptxt.ps
```

(8) Run the bat file **addpstxt.bat** specifying the output file as say **myfigure.ps**

(9) Load myfigure.ps in Ghostview, and also in a text editor like notepad.

(9) If the symbol font is used, then using notepad, uncomment the symbol font command in the file **myfigure.ps**, and locate it above the first symbol text.

(10) If the text is still not located properly, then find where the text is to be moved using ghostview, and adjust tmp.txt, which also should be in a notepad window for convenience. Rerun addpstxt, click in the Ghostview window again, and the new figure, myfigure.ps will appear. Repeat untill the figure is satisfactory.

See the next section for more details about the programs that are being used in this process.

0.22 Adding Annotation to A Postscript File

The batch file that adds text to a postscript file is called **addpstxt.bat**. It contains the following commands:

```
@echo off
rem addpstxt.bat, Version 2/21/03, Jim Emery
rem adds postscript code to a postscript file for text
if "%1"=="" goto help
if "%2"=="" goto nofontinfo
pstext tmp.txt tmptxt.ps %2
includef tmp.ps %1
goto end
:nofontinfo
pstext tmp.txt tmptxt.ps
includef tmp.ps %1
goto end
:help
echo addpstxt.bat, Version 2/21/2003 Adds postscript code for text annotation.
```

```
echo The text is contained in a file called tmp.txt, type pstext for the data
echo structure. The postscript file, to which annotation is to be added, is
echo called tmp.ps, which must exist. File tmp.txt must also exist.
echo The first parameter is the file name of the new postscript file.
echo The second parameter is the fontsize in points, say 12 pts.
echo If the second parameter is not present, the font definition is assumed
echo to be already in the initial file. Also in order for the new postcript
echo code to be added, there must be a :include tmptxt.ps statement in tmp.ps
echo If a second text file is to be added rename the first tmptxt.ps
echo to say tmptxt1.ps and add a :include tmptxt1.ps in tmp.ps
echo Usage: addpstxt newpostscriptfile fontsize
:end
```

The batch file assumes that there is a figure source file called tmp.ps. This represents a figure to which annotation is to be added. This file would typically be created as an eg file and then converted to postscript with eg2ps. The file tmp.txt contains the definition of the text to be added. Each line of tmp.txt would have the postscript x and y coordinates of the text in pts, followed by the text itself. The coordinates would have been found using the curser position displayed when the postscript file is viewed in Ghostscript. The program pstext.c is called to read tmp.txt and to produce a postscript file, which is called tmptxt.ps and which has the new postscript code for the text annotation. In the file tmp.ps one must put a line

```
:include tmptxt.ps
```

so that the new postscript code is added at that point.

The program includef.c is called in the batch file to merge the new code into the original postscript file tmp.ps. An example of a tmp.txt file is

```
160 230 a
260 128 b
305 230 c
271 174 h
```

A finished file displayed here is a figure for a circle circumscribing a triangle:

```
%!PS
%%BoundingBox: 47 17 476 371
%%Creator: eg2ps.c by Jim Emery
%%EndComments
72 300 div 72 300 div scale
100 100 translate
3 setlinewidth
newpath
125 500 moveto
1875 500 lineto
500 -374 moveto
500 0 moveto
500 1375 lineto
stroke
750 1250 moveto
500 500 lineto
1500 500 lineto
750 1250 lineto
stroke
1559 750 moveto
1554 821 lineto
1541 892 lineto
1518 960 lineto
1487 1024 lineto
1448 1084 lineto
1402 1139 lineto
1349 1187 lineto
1290 1228 lineto
1227 1261 lineto
1159 1286 lineto
1090 1302 lineto
1018 1309 lineto
947 1306 lineto
876 1295 lineto
807 1275 lineto
742 1246 lineto
681 1209 lineto
625 1164 lineto
```

```
575 1113 lineto
532 1056 lineto
497 993 lineto
470 927 lineto
451 858 lineto
442 787 lineto
442 715 lineto
451 644 lineto
469 575 lineto
496 508 lineto
531 446 lineto
574 388 lineto
623 337 lineto
679 292 lineto
740 255 lineto
806 226 lineto
874 205 lineto
945 194 lineto
1017 191 lineto
1088 198 lineto
1158 214 lineto
1225 238 lineto
1289 271 lineto
1347 312 lineto
1400 360 lineto
1447 414 lineto
1486 474 lineto
1518 539 lineto
1540 607 lineto
1554 677 lineto
1559 748 lineto
stroke
1000 500 moveto
1000 750 lineto
/Times-Roman findfont
%/Symbol findfont
60 scalefont setfont
 567 858 moveto
```

```
 567 858 translate
(a) show
 -567 -858 translate
 983 433 moveto
 983 433 translate
(b) show
 -983 -433 translate
 1171 858 moveto
 1171 858 translate
(c) show
 -1171 -858 translate
 1029 625 moveto
 1029 625 translate
(h) show
 -1029 -625 translate
stroke
showpage
```

The line

```
:include tmptxt.ps
```

in the file tmp.ps becomes

```
/Times-Roman findfont
%/Symbol findfont
60 scalefont setfont
 567 858 moveto
 567 858 translate
(a) show
 -567 -858 translate
 983 433 moveto
 983 433 translate
(b) show
 -983 -433 translate
 1171 858 moveto
 1171 858 translate
(c) show
```

```
 -1171 -858 translate
 1029 625 moveto
 1029 625 translate
(h) show
 -1029 -625 translate
```

in the new figure file. The figure generated by this file is captioned **A Circle Circumscribing A Triangle.**

0.23 Creating Figures With AutoCad and DesignCad

A figure can be created in AutoCad and saved in some format that can be exported. A good format for saving is the IGES format. However, later versions of AutoCad no longer support the IGESOut command. DesignCad can read the AutoCad file and then export an IGES file. One then may use my program **iges2eg** to convert to an eg file. Then one may convert this file to PostScript using my program eg2ps. Annotation may be added as shown in the previous section.

0.24 Importing A Figure Generated In A Program Language Into DesigCAD of AutoCAD

Program plthp creates an hpgl (Hewlett-Packard Pen Plotter Language) file from an eg file. DesignCad will import an hpgl file. Perhaps AutoCAD will also.

0.25 References

The book **The Latex Graphics Companion** by Michel Goossens, Sebastian Rahtz, and Frank Mittelbach, Addison-Wesley, 1997, explains the use of various graphics programs with Latex and PostScript. It shows how to use several programs that create figures for Latex. Unfortunately it does not say much about psfig, but concentrates on a similar macro package called

PSTricks. There may be documentation for psfig that can be obtained from the internet. Only later versions of PCTeX support the use of psfig and PostScript. To view the output of PCTeX that contains psfig pictures, PCTeX must be operated in PostScript mode. The disadvantage of this mode is that it gives somewhat slower typesetting. See the PCTeX manual for details of using embedding graphics figures.. A version of PCTeX that supports PostScript mode is: PCTeX32 version 4.0. The manual for this version of PCTeX gives information on the PostScript mode.

There are many books on PostScript, including the Red, Green, and Blue books from Adobe. The Red book is now in its third edition. There is a document by Jim Emery containing material about EG graphics. It is called "Computer Graphics" (graphics.tex). There are several programs for viewing and printing EG graphics files. One for Windows 95-98 is called wineg.cpp. Utility programs mentioned in this document can be obtained from the author.

0.26 An Example of Some of the EG Commands: example.eg

A listing of the file **example.eg**:

```
v-1 1 -1 1
w0. 10. 0. 10.
m1.0 1.0
d9 1
d 9 9.0
d1 9
d1 1
c3
$ the a command draws an arc as short line segments
$ Here center (5,5), radius 3, from angle 0 to 2pi, with 50 points
a5 5 3 0 6.28 50
z .1
c1
$ the #tc command causes the following text command to produce centered text
#tc 5. 5. 1
t2.5 4.8 9 EGRAPHICS
c2
$ a p command specifies a postscript command that is written directly to the output file,
$ so the next command simply writes 0 setgray to the output.
p 0 setgray
$ The following commands create a gray shaded polygon
m 4 4.5
d 6 4.5
d 7 3
```

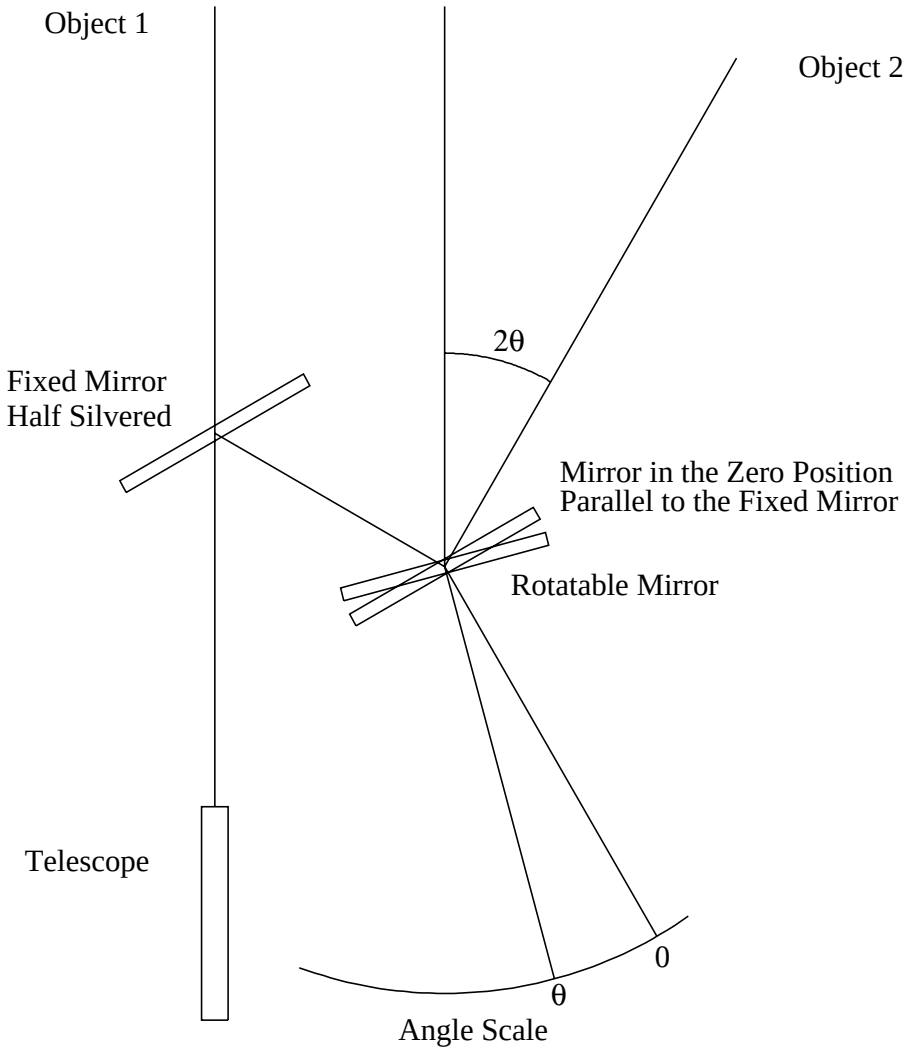


Figure 12: **The Principle of the Sextant**. The use is described in the section called **The Sextant: Principle and Use**.

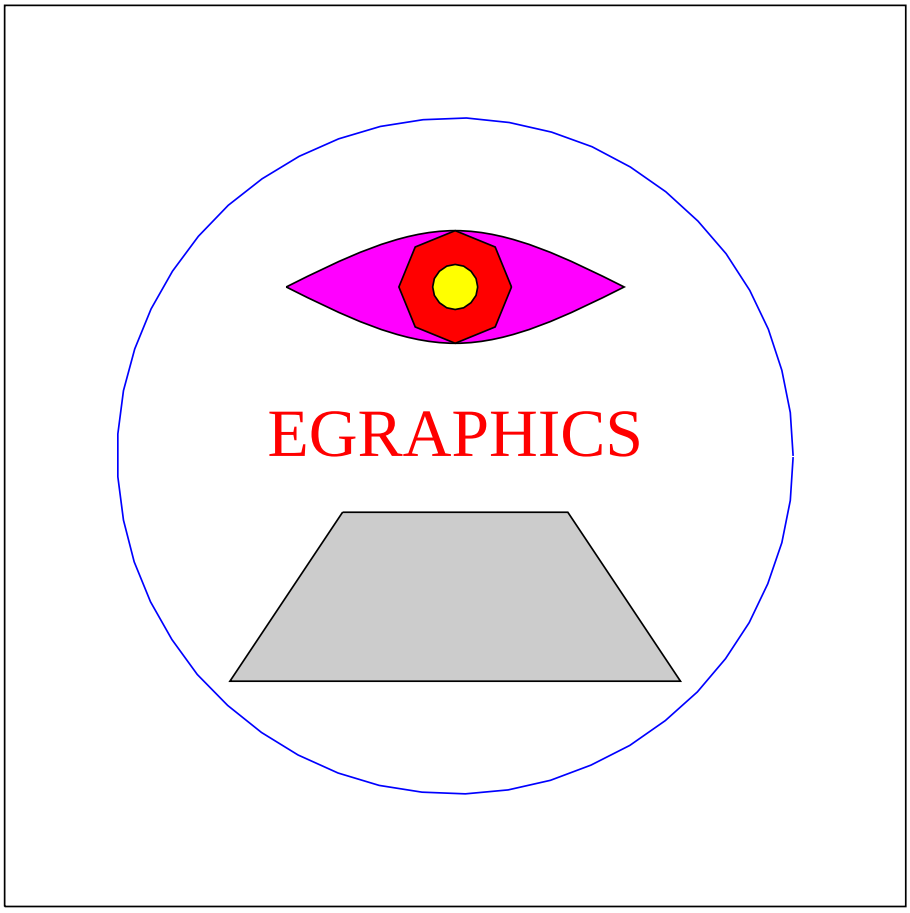


Figure 13: **Example Postscript File: called example.eg** This file demonstrates uses of several EG commands, and shows how to fill and shade/color drawn polygons.

```
d 3 3
d 4 4.5
p gsave
p .8 setgray
p fill
p grestore
p stroke
$
$ draw eye with bezier cubics
$ we are using the 3 control point version here.
$ a 4 control point version makes a moveto and 50 drawtos.
$ in the 4 point case, the fill may not be done correctly,
$ because of the movetos in the middle of the path.
m 3.5 6.5
b 4 6.75 4.5 7 5 7
b 5.5 7 6 6.75 6.5 6.5
b 6 6.25 5.5 6 5 6
b 4.5 6  4 6.25 3.5 6.5
$ fill the region bounded by the bezier path
p gsave
p 1 0 1 setrgbcolor
p fill
p grestore
p stroke
$ draw and shade iris
$ A stroke draws the boundary of the path and ends the path. A newpath just ends the path.
p stroke
p 0 setgray
m 5.5 6.5
d 5.3535534 6.8535534
d 5 7
d 4.6464466 6.8535534
d 4.5 6.5
d 4.6464466 6.1464466
d 5 6
d 5.3535534 6.1464466
d 5.5 6.5
p gsave
p 1 0 0 setrgbcolor
p fill
p grestore
p stroke
$ shade pupil
m 5.2 6.5
d 5.1847759 6.5765367
d 5.1414214 6.6414214
d 5.0765367 6.6847759
d 5 6.7
d 4.9234633 6.6847759
d 4.8585786 6.6414214
d 4.8152241 6.5765367
d 4.8 6.5
d 4.8152241 6.4234633
d 4.8585786 6.3585786
d 4.9234633 6.3152241
d 5 6.3
d 5.0765367 6.3152241
```

```
d 5.1414214 6.3585786
d 5.1847759 6.4234633
d 5.2 6.5
p gsave
p 1 1 0 setrgbcolor
p fill
p grestore
p stroke
```

A listing of the postscript file **example.ps** produced by program **eg2ps.c** :

```
%!PS
%%BoundingBox: 0 50 450 500
%%Creator: eg2ps.c by Jim Emery
%%EndComments
72 300 div 72 300 div scale
100 100 translate
3 setlinewidth
newpath
200 200 moveto
1800 200 lineto
1800 1800 lineto
200 1800 lineto
200 200 lineto
stroke 0 0 1 setrgbcolor
1600 1000 moveto
1595 1077 lineto
1580 1152 lineto
1556 1225 lineto
1523 1294 lineto
1481 1359 lineto
1431 1417 lineto
1374 1469 lineto
1311 1513 lineto
1243 1549 lineto
1171 1575 lineto
1096 1592 lineto
1020 1600 lineto
943 1597 lineto
867 1585 lineto
793 1563 lineto
723 1532 lineto
657 1492 lineto
597 1445 lineto
544 1390 lineto
498 1328 lineto
460 1261 lineto
431 1190 lineto
411 1116 lineto
401 1039 lineto
401 963 lineto
411 886 lineto
430 812 lineto
459 741 lineto
497 674 lineto
542 612 lineto
```

```
596 557 lineto
656 509 lineto
721 469 lineto
792 437 lineto
865 415 lineto
941 403 lineto
1018 400 lineto
1094 407 lineto
1169 424 lineto
1241 451 lineto
1310 486 lineto
1373 530 lineto
1430 581 lineto
1480 640 lineto
1522 704 lineto
1555 773 lineto
1580 846 lineto
1595 921 lineto
1600 998 lineto
stroke 1 0 0 setrgbcolor
/Times-Roman findfont
 120 scalefont setfont
1000 1000 moveto
1000 1000 translate
(EGRAPHICS) stringwidth pop -.5 mul 0 rmoveto
(EGRAPHICS) show
-1000 -1000 translate
stroke 0 1 0 setrgbcolor
 0 setgray
800 900 moveto
1200 900 lineto
1400 600 lineto
600 600 lineto
800 900 lineto
 gsave
 .8 setgray
 fill
 grestore
 stroke
700 1300 moveto
 800 1350  900 1400  1000 1400  curveto
 1100 1400  1200 1350  1300 1300  curveto
 1200 1250  1100 1200  1000 1200  curveto
 900 1200  800 1250  700 1300  curveto
 gsave
 1 0 1 setrgbcolor
 fill
 grestore
 stroke
 stroke
 0 setgray
1100 1300 moveto
1071 1371 lineto
1000 1400 lineto
929 1371 lineto
900 1300 lineto
929 1229 lineto
```

```
1000 1200 lineto
1071 1229 lineto
1100 1300 lineto
 gsave
 1 0 0 setrgbcolor
 fill
 grestore
 stroke
1040 1300 moveto
1037 1315 lineto
1028 1328 lineto
1015 1337 lineto
1000 1340 lineto
985 1337 lineto
972 1328 lineto
963 1315 lineto
960 1300 lineto
963 1285 lineto
972 1272 lineto
985 1263 lineto
1000 1260 lineto
1015 1263 lineto
1028 1272 lineto
1037 1285 lineto
1040 1300 lineto
 gsave
 1 1 0 setrgbcolor
 fill
 grestore
 stroke
stroke
showpage
```

Note. When shading a curve or path consisting of several curves written in succession, make sure there are no moves in between the subcurves, because this will result in improper shading.

0.27 Bibliography

The following may have some material on postscript, graphics, and Emery graphics.

[1] Emery James, **Including Graphics Figures in Documents With PCTeX and Linux LATeX**, (Figures.tex).

[2] Emery James, **Emery Graphics**, (eg.tex, currently not much in the file)

[3] Emery James, **Bitmap Graphics**, bitmap.tex.

[4] Emery James, **Computer Graphics and Geometry**, graphics.tex.

[5] Emery James, **Computer Graphics**, graphics-.tex.

[6] Emery James, **Creating Graphics and Figures with the Postscript Language**, postscript.tex

[7] Emery James, **Scientific Calculating, Programming, and Writing**, (sciprog.tex).

[8] Emery James, **Geometric Calculations**, (geometry.tex).