

# Importing Graphics in $\LaTeX$ 2 $\epsilon$

## Workshop #6

David Arnold

Friday, April 23, 1999

# Encapsulated Postscript Graphics

When Donald Knuth first wrote T<sub>E</sub>X, there wasn't the plethora of graphics types that are common today. No Gif's, no Jpegs, etc. So, T<sub>E</sub>X was written to recognize encapsulated postscript.

Actually, it is a bit of a lie to say that T<sub>E</sub>X "recognizes" an encapsulated postscript graphic. What T<sub>E</sub>X really "sees" is the *bounding box* of the encapsulated postscript image.

\* What is a bounding box? \*

Perform the following experiment.

- ① Open the file `clown.eps` in GSView.
- ② On the Options menu in GSView, uncheck EPS Clip and check Show Bounding Box. This will display the bounding box of the clown image as a dotted rectangle. Note that it surrounds the image.
- ③ In GSView, move the mouse and note that the coordinates of the cursor are given in bp (72 bp per

inch) on the status bar. Move the mouse cursor to the lower left-hand corner of the bounding box and write down coordinates. Move the cursor to the upper right-hand corner of the bounding box and write down the coordinates.

- ④ Open `clown.eps` in WinEdt. Note these lines near the top of the file.

```
%!PS-Adobe-3.0 EPSF-3.0
%%Creator: MATLAB, The Mathworks, Inc.
%%Title: F:\StaffDevelopment\tips\clown.eps
%%CreationDate: 04/01/99 21:31:04
%%DocumentNeededFonts: Helvetica
%%DocumentProcessColors: Cyan Magenta Yellow Black
%%LanguageLevel: 2
%%Pages: 1
%%BoundingBox: 185 320 425 470
%%EndComments
```

The bounding box measurements in the file `clown.eps` should match those found with the mouse in GSView.

The first two numbers, 185 and 320, are the coordinates of the lower left-hand corner of the image

in bp, as measured from the bottom left-hand corner of the physical page. the second two numbers, 425 and 470, are the coordinates of the upper right-hand corner of the image in bp, again measured from the bottom left-hand corner of the physical page.

$\text{T}_{\text{E}}\text{X}$  is trained to “spot” the coordinates of the bounding box.  $\text{T}_{\text{E}}\text{X}$  then uses the bounding box information to reserve the appropriate amount of space for the image in the DVI file.

$\text{T}_{\text{E}}\text{X}$  doesn’t actually “draw” the picture on the page. That is done by the DVI processor. Some previewers, through the use of “special” commands, can display other types of graphic images. For example, YAP can display bitmapped images (BMP). However,  $\text{T}_{\text{E}}\text{X}$  cannot “spot” the size of the bitmap image as there is no bounding box. You, the user, must calculate the size of the image and manually reserve enough space. This is a real inconvenience and is one of the main reasons that  $\text{T}_{\text{E}}\text{X}$  and  $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$  users prefer to use encapsulated postscript images.

Finally, even if your previewer supports the inclusion

of other types of images, it is probably unlikely that your previewer will be able to do anything with these images when converting the DVI file to postscript, an essential step in the DVI→PDF conversion process.

So, that closes the final door. The image type of choice is encapsulated postscript.

That said, there are all kinds of software packages that will save images in encapsulated postscript form: PhotoShop, PaintShop Pro, Adobe Illustrator, Matlab, etc. You can also set up the Adobe Postscript Printer Driver that is included on the Acrobat CD. Once this postscript driver is set up, then you can print from any application to this printer driver to create a postscript image file. GSView can then be used to change the postscript file into encapsulated postscript. GSView can also be used to change a variety of image types to encapsulated postscript.

We make the assumption that you possess a piece of software that will save your graphic as an encapsulated postscript image.

# The Graphicx Package

Once you obtain your encapsulated postscript image, including it in your  $\text{\LaTeX}$  file is simple and straightforward if you use the `graphicx` package. The following code will include the graphics file `clown.eps` in your document in a “centered” environment. You will need to  $\text{\LaTeX}$  this file, then use `dvips`, then view the final result in `GSView`.

```
\documentclass{article}
\usepackage{graphicx}
\begin{document}
A couple of words of text.
\begin{center}
\includegraphics{clown.eps}
\end{center}
A few more words of text.
\end{document}
```

Note how enclosing the graphic in `\begin{center}... \end{center}` centers the graphic. But also note that the centering environment puts a little space both before and after the centered graphic.

## Adding Options to Includegraphics

A more advanced use of the `\includegraphics` command takes the form

```
\includegraphics[options]{filename},
```

where you can place a variety of different instructions in place of the word options. For example, the following code will set the graphic at a width of exactly 1.5 inches.

```
\documentclass{article}
\usepackage{graphicx}
\begin{document}
A couple of words of text.
\begin{center}
\includegraphics[width=1.5in]{clown.eps}
\end{center}
A few more words of text.
\end{document}
```

The vertical height of the graphic is automatically scaled to maintain the original *aspect ratio* of the graphic. That is, if the ratio of width to height of the original figure is 8 in to 6 in, then setting the width with

```
\includegraphics [width=4in] {filename}
```

yields an image in the document with width 4 inches and height 3 inches.

It is a simple task to rotate a graphic through a prescribed angle.

```
\documentclass{article}
\usepackage{graphicx}
\begin{document}
A couple of words of text.
\begin{center}
\includegraphics [angle=45] {clown.eps}
\end{center}
A few more words of text.
\end{document}
```

Of course, you can combine options. For example, change the width, then rotate the result by changing the `\includegraphics` command to

```
\includegraphics [width=1.5in,angle=45] {clown.eps}
```

Question: What if we rotated first, then changed the width? Would that be different? Try it by changing the `\includegraphics` command to

```
\includegraphics[angle=45,width=1.5in]{clown.eps}
```

You can also *scale* a graphic. The following code will uniformly scale the graphic to 75% of its original size.

```
\documentclass{article}
\usepackage{graphicx}
\begin{document}
A couple of words of text.
\begin{center}
\includegraphics{clown.eps}
\includegraphics[scale=0.75]{clown.eps}
\end{center}
A few more words of text.
\end{document}
```

Scaling, changing the width, and rotating can sometimes give unexpected results. For a thorough presentation of these ideas, read *Using Imported Graphics in L<sup>A</sup>T<sub>E</sub>X<sub>2</sub>e*, pages 17, 20–26. This file is available at

<http://online.redwoods.cc.ca.us/instruct/darnold/StaffDev/epslatex.pdf>

# The Figure Environment

If you include a graphic in a figure environment, that is, if you include a figure between `\begin{figure} ... \end{figure}`, then  $\text{\LaTeX}$  allows the graphic to “float” to a position on the page that minimizes white space and maximizes the amount of text present on the page.

This concept of “floating” figures is particularly troubling to former WYSIWYG users, who are used to constantly repositioning graphics as they craft their document. Two words of advice are in order.<sup>1</sup>

☞ **Don’t compose text which is dependent of figure placement.** Using the phrase “This figure ... ” or “The following figure ... ” requires the figure to be in a certain location. Using the phrase “Figure 14 ... ” allows the figure to be positioned anywhere.

---

<sup>1</sup>Taken directly from *Using Imported Graphics in  $\text{\LaTeX}2e$* , page 41.

☞ **Relax.** Some users get quite worried when a figure isn't placed exactly where they want it. Figure placement is L<sup>A</sup>T<sub>E</sub>X's job; users should generally not worry about it.

With these thoughts in mind, let's put a floating figure in a document.

☞ First, create the following source file.

```
\documentclass{article}
\usepackage{graphicx}
\begin{document}
Now is the time for all good men to come to the aid
of their country. Now is the time for all good men
to come to the aid of their country. Now is the time
for all good men to come to the aid of their country.

Now is the time for all good men to come to the aid
of their country. Now is the time for all good men
to come to the aid of their country. Now is the time
for all good men to come to the aid of their country.
\end{document}
```

☞ Next, copy and paste the two existing paragraphs

in the source file about 50 times. This will give an output file that is several pages long, perfect for examining the various options of the floating figure environment.

☞ Finally, place the following figure environment in the source between paragraph one and two.

```
\documentclass{article}
\usepackage{graphicx}
\begin{document}
Now is the time for all good men to come to the aid
of their country...
\begin{figure}[tbp]
  \centering
  \includegraphics{clown}
  \caption{My first figure.}
\end{figure}
Now is the time for all good men to come to the aid
of their country...
\end{document}
```

Note the use of the `\centering` command. The figure environment places a little space both before and after the included graphic. If we use

`\begin{center}...\end{center}`, that environment also adds space before and after the included graphic. That would be too much space. The `\centering` command centers the graphic without adding space before or after the graphic.

☞ `LATEX`, `dvips`, then view the result in `GSView`. Note how the figure “floats” to the top of the first page.

The options in `\begin{figure}[tbp]`, that is `[tbp]`, stand for “top,” “bottom,” or “page of floats.” Their order is irrelevant. That is, if you write `\begin{figure}[bpt]`, it makes absolutely no difference.

☞ Replace `\begin{figure}[tbp]` with `\begin{figure}[htbp]` in the previous source code. `LATEX`, `dvips`, then preview in `GSView`. How does this change affect the floating placement of the figure?

The “h” stands for “here.”

☞ Finally, sprinkle the code

```
\begin{figure}[tbp]
  \centering
  \includegraphics{clown}
  \caption{My first figure.}
\end{figure}
```

throughout your source code and note the resulting figure placement in GSView.

Did you get any pages with *two* figures placed at the top of a page? If not, try placing another figure environment immediately after the one following the first paragraph and compile again.

Now, how can you eliminate double figure environments at the top of a page?

☞ Place the command `\setcounter{topnumber}{1}` in the preamble of your document and compile again. This should eliminate double figure placement at the top of pages.

Do you find it curious that there are no floating figures on the bottom of the pages? That because the size of our figure exceeds the maximum allowable size for bottom figure placement. We could scale our graphic to a smaller size, but we will adjust the variable `\bottomfraction` instead.

☞ Place the command

```
\renewcommand{\bottomfraction}{0.4}
```

in the preamble and compile again.

The variable `\bottomfraction` is defined as the maximum fraction of a text page which can be occupied by floats at the bottom of a page. The default value of this variable is 0.3. Increasing this allows some of our graphics to “float” to the bottom of some pages.<sup>2</sup>

---

<sup>2</sup>For more information on how to tune floating parameters, see *Using Imported Graphics in L<sup>A</sup>T<sub>E</sub>X<sub>2</sub>e*, pages 41, 45–48.

## Floating Tables

Everything that has been said about floating figure environments applies equally to floating table environments. Simply surround your tabular environment with `\begin{table}...\end{table}` and your table will “float” in exactly the same manner as your graphics.

☞ Enter the code

```
\begin{table}[tbp]
\centering
\begin{tabular}{|l|c|c|c|}\hline
Player & Round 1 & Round 2 & Round 3\\ \hline
Dave & 89 & 91 & 88\\
Don & 102 & 105 & 99\\
Todd & 153 & 256 & 357\\\hline
\end{tabular}
\caption{Through three rounds at the Muni.}
\end{table}
```

☞ Make the following change and note the difference.

```
\begin{table}[htbp]
```

## One Final Caution

We end with a word of caution. Novice  $\LaTeX$  users are constantly fussing with figures, which is a monumental waste of time. Each time the document is edited, the figures are more than likely going to “float” to a brand new location.

Consequently, don't try to force figure placement with commands such as

```
\begin{figure}[h],
```

as this is only asking for trouble. Restricting options like this can actually “lock up”  $\LaTeX$ 's figure handling mechanism, causing untold hours of debugging.

Heed the advice on an earlier slide.

☞ **Relax.** Some users get quite worried when a figure isn't placed exactly where they want it. Figure placement is  $\LaTeX$ 's job; users should generally not worry about it.

Sound advice!

## Further Exploration

We've only touched the surface of graphic and tabular inclusion in a  $\LaTeX$  document. For a more thorough treatment, be sure to read *Using Imported Graphics in  $\LaTeX$ 2 $\epsilon$* .

# Congratulations!

Congratulations to all workshop participants! I am proud of you for making it this far. Please know that I have thoroughly enjoyed sharing the wonderful world of  $\text{\LaTeX}$  with such an enthusiastic group!

Now we await the visit of Dr. Story.