LaTeX for WPI CS Graduate Students

Andreas Köller
Worcester Polytechnic Institute
`koeller@wpi.edu`

Worcester, January 8, 1999

Contents

1 Introduction: Why should I use LaTeX—and what if I have to?

This document wants to provide help with some common problems to all who write documents in LaTeX. It also wants to establish some basic rules for LaTeX documents at WPI (such as using LaTeX2ε over old LaTeX2.09). It is **not** an introduction to LaTeX. References to introductions are given below. This document answers many questions that people have asked me about LaTeX in the past months.

If you're new to LaTeX, you're going to have to read one of the introduction documents available. A very short introduction (if you're *completely* new to LaTeX) is `essential2e.ps` in this directory. A somewhat larger document is "LaTeX in 87 Minutes" (`lshort2e.ps` in this directory). The latter also exists as a `pdf`-file for Acrobat Reader.

Most people first ask why they can't write documents in "MS Word". Well, "Word" is good for writing letters or memos, or even small term papers for International Studies classes, but it is largely unusable when working on scientific documents or documents over 10 pages, especially when they are in mathematics-related fields like Computer Science. Try to create footnotes, citations, tables of contents, figure captions, tables, formulae (!) in "Word" and you will find out soon that it's worth it to learn yet another programming language for the sole purpose of creating pretty documents without becoming a case for the psychiatrist. Look at this document and try to think how you would have created it in "Word"[1]. Also—out of experience—"Word" crashes about every two hours. LaTeX can't crash, and thus can't take your Master's thesis to document heaven. It's also impossible that LaTeX suddenly refuses to save your document. "Word" is notorious for doing just that.

LaTeX is a collection of macros on top of a programming language called TeX[2]. Like HTML, LaTeX is an instance of SGML. So you need to "program" your documents. But it's worth it. You can even define macros (they can be anything—text phrases, formulae, text formatting, even mathematical calculations), and then use

[1] If you know how to do this, ask the CCC for a teaching position or go and interview at Microsoft.
[2] It's written by Leslie Lamport, thus the "La".

those in your text. The best idea is to get a very simple introduction from the WWW and get started. I pointed out two documents that you can print or (preferably) view on the screen.

A good WWW-FAQ-List is `http://www-h.eng.cam.ac.uk/help/tpl/textprocessing/` which can help answering specific question. Many not-so-basic issues are handled in the document you are currently reading. If you still have questions, consult a LaTeX-book such as *Kopka, Daly: "A Guide to LaTeX2ε"* by Addison-Wesley which is available at least from Prof. Rundensteiner and me.

2 How to Include Graphics Into LaTeX-Documents?

LaTeX is able to include several kinds of graphics formats, but most of them are not very useful (like LaTeX's `picture`-Environment, which is a simple programming language for drawing). The reason for that is that LaTeX is older than most graphics formats (including `GIF` and PostScript). Therefore, the main way of including any type of graphics is through PostScript. Formats supported are `eps` and single-page, bounded `ps`. No format that LaTeX can include is supported by any Microsoft-Software, therefore the conversion from MS-Products to LaTeX is difficult (Section 5). Conversion of bitmaps is easier (Section 3). Under UNIX, there are several tools that generate `eps` or `ps` directly (`gnuplot` for graphs, `xfig` and `idraw` for general drawings. On `davis.wpi.edu`, only `xfig` is currently installed). This section describes how such `eps`-files are included into LaTeX-documents. For reasons of standardization, I will suggest a method which is different from what I have told people in the past.

To include graphics, you have to use a certain LaTeX package. Thus, the following command has to be included immediately after the `\documentclass{}`-declaration (for LaTeX 2_ε):

```
\usepackage[dvips]{graphicx}
```

It is useful to collect all pictures in a sub-directory of the document directory, for instance called `figures`. If you then want LaTeX to find your figures, you have to define a path for the graphics, with the following command:

```
\graphicspath{{figures/}}
```

This command has to be given before the `\begin{document}` statement. Note the double braces and the `/` at the end of the path. The inclusion of a picture then works as follows (if the `eps`- or `ps`-File is available):

```
\begin{figure}[htbp]
  \begin{center}
     \includegraphics[width=0.35\textwidth]{picture.eps}
  \end{center}
  \caption{This is my \emph{first} picture.}
  \label{pic:my_first_picture}
\end{figure}
```

Comments:

- The *placement* parameters (`[htb]`) specify that the figure may appear **h**ere, at the **t**op or **b**ottom of the current page, or at a separate **p**age by itself. If you omit any of these placement permissions, the figure will not be placed in that way. This is a matter of writing style—for most purposes, `[htb]` should be fine. If figures are slightly too large (usually wider than the page), they are placed at the end of the chapter or document, on separate pages. Since the order of figures is never changed, all following figures appear after the initial large one. When this happens, you should decrease the figure in size (i.e., change the `width`-parameter). You can also try to add a `!` to the placement parameter (e.g., `[!htb]`). This relaxes some constraints that LaTeX puts on the placement of figures and makes things easier sometimes.
- The width of the picture can be specified *absolutely* (`5cm, 2.8in`, etc.) or *relatively* as in the example. You can also specify the *height* using `height`, but only one of the two parameters should be used unless you want to change the aspect ratio.

- You can give a caption to the figure which will appear below the figure. You can format the caption just like any LaTeX-text (e.g., by including `\emph{}`-commands).
- The label is useful for referring to the picture number with, e.g., `~\ref{pic:my_first_picture}` or the page on which the picture was placed with, e.g., `~\pageref{pic:my_first_picture}`.
- If you add the `draft` parameter to the package (`\usepackage[draft,dvips]{graphicx}`), only the bounding boxes of figures will be displayed, which saves a lot of time when viewing documents with `xdvi`.

The above example generates the following output (Figure 1, Page 3)

Figure 1: This is my *first* picture.

The full syntax of the `\includegraphics`-command is as follows:

```
\includegraphics[bb=llx lly urx ury,angle=angle,
width=h_length,height=v_length,scale=factor,clip=true/false,
draft=true/false]{file_name}
```

`bb` sets the bounding box of the file, i.e. you can clip parts of the file (you are unlikely to need this feature). `draft` overrides the global setting of the `draft`-parameter (see above). `clip` clips the picture to the (specified or implicit) bounding box of the figure, which is very useful for including a Microsoft-generated figure (cf. Section 5). The other parameters should be self-explanatory. If you don't need any of the parameters, you can use the simple `\includegraphics{filename.eps}` syntax.

The file `epslatex.ps`, provided in this directory, goes into great detail about the usage of graphics in LaTeX. It handles dozens of complicated cases of the inclusion of figures, such as the customization of captions, the placing of tables next to figures, etc. Please refer to that documentation if you have greater plans with figures, but please view it on the screen (`ghostview`) and don't print it, since it has 86 pages.

2.1 Placing Two Figures Next To Each Other

If you have little space in your document, you may want to place two figures next to each other in the text. This is accomplished by enclosing each `\includegraphics`-command into a "minipage". You have to set the widths of the minipages to values that together do not exceed the page width (i.e., if you use `\textwidth`, the factors have to add up to less than 1). You can scale the figures inside each minipage relatively to the minipage (i.e., the `\textwidth`-function *inside* a minipage-environment returns the width of the minipage, not the whole page). If you need extra space between the two figures, decrease their combined width to something not larger than 90% of the text width and include a `\hfill` between the minipages. The two Figures 2 and 3 on page 4 were included with the following source code [3]:

```
\begin{figure}[htb]
  \begin{minipage}{0.45\textwidth}
```

[3] Here, the width of the two minipages together is $(0.45 + 0.45) * $ `\textwidth`; the space between the figures inserted by `\hfill` therefore amounts to 10% of the width of the page.

```
      \begin{center}
        \includegraphics[width=0.9\textwidth,angle=270]{pixelpic.eps}
        \caption{A Bitmap Picture.}
        \label{pic_bitmap}
      \end{center}
    \end{minipage}
  \hfill
    \begin{minipage}{0.45\textwidth}
      \begin{center}
        \includegraphics[width=0.9\textwidth,angle=270]{vectorpic.eps}
        \caption{The Same Picture as a Vector Graphic.}
        \label{pic_vector}
      \end{center}
    \end{minipage}
  \end{figure}
```

3 Bitmaps vs. Vector Graphics

A vector graphic picture can be scaled without loss of quality and its file is much smaller. So unless you *need* a bitmap (screenshots), you should always convert your original picture into a vector graphic (Section 5). The difference between the same picture inserted as a bitmap and a vector graphic, respectively, is shown in Figures 2 and 3. Also, the `eps`-file of the bitmap picture is 35KB large, vs. 13KB for the vector graphic file. As an *uncompressed* PostScript-File, the size of the bitmap picture exceeds 4MB!

Figure 2: A Bitmap Picture.

Figure 3: The Same Picture as a Vector Graphic.

4 Including Bitmaps into LaTeX-Documents

A bitmap is a picture whose elements are *pixels*. That means, anything you can possibly produce in Excel, Powerpoint, Corel Draw, etc., is **not** a bitmap. The most common type of bitmap is probably the *screen shot*.

Conversion from bitmaps to something that LaTeX can include (i.e., PostScript) can be done in different ways. The two most common ones are `xv` and `convert`.

- Conversion with `xv`
  Save your bitmap as `gif` if it is a screen shot with few colors, and as `jpg` if it is a photo. Open it on any UNIX-machine with `xv` and save it as `PostScript`. If you plan to print in black-and-white, save the file

in Greyscale, that saves space in the file system. Use the Compress option if possible (may not work on all printers). Bitmap files turned PostScript are very large (>4MB for an average screenshot), so think how many you really need.

- Conversion with `convert`
  `convert` is a command-line converter for image formats. If you use `convert`, you can convert bitmaps on-the-fly and don't need to store the converted `ps`-files. That can save megabytes of memory in your file system. `convert` is working on at least `owl`, `wpi`, and `davis`.

  The procedure:

  1. Extract the bounding box declaration for the file. The UNIX command for that is:
     `convert` *`picture.gif`* `eps:- | grep BoundingBox >` *`picture.gif`*`.bb`
  2. In your LaTeX main file, include the following definition:
     `\DeclareGraphicsRule{.gif}{eps}{.gif.bb}{`convert -colorspace GRAY #1 'eps:-'}`
     Replace `gif` with `tif` or `jpg` for other bitmap formats. For a color picture, omit the `-colorspace GRAY` command.
  3. To include the picture, use the standard method (e.g., `\includegraphics{picture.gif}`; Section 2).

  This method saves a lot of space, but is slower when displaying `dvi`-Files using `xdvi`. To save time while writing, you can temporarily display only the bounding boxes of pictures. Set the `draft`-parameter of the `graphicx`-package (`\usepackage[draft]{graphicx}`) to display only the bounding box of the picture. Remember to remove the parameter before creating the final `ps`-File.

4.1 How to Obtain a Screenshot?

- UNIX
  Display the image you want to shoot on the screen, then start `xv` and click on `Grab`. Does not work through `PCXWare`.
- Windows
  Display the image you want to shoot, and make its window the active window. Then, press `Alt` + `PrintScrn`. The bitmap of the active window is now on the Clipboard. From there, you can `paste` the bitmap into any useful image processor. Unfortunately, the two programs that are included in Windows (Paint and Wang Imaging) support neither `gif` nor `jpg`. The latter one is completely unusable for our purpose. I recommend PaintShop Pro which is shareware. If you have MS Office installed, you can use MS Photo Editor. (Except that this one always dithers your image, so saving in `GIF` will look awful.) After pasting the image into a image processor, save the file as `jpg` or `gif` (see above). If you don't have any image software besides Paint, save the file as a "24-color-bitmap" (it's then VERY big) and convert it using `convert` or `xv` on UNIX.

5 Microsoft Generated Figures and LaTeX

Many documents require drawings or pictures and those are often done using a Microsoft product like Excel or Powerpoint. Microsoft does not support any vector image format that LaTeX can use, so it is difficult to include a Microsoft figure into a LaTeX document. The easiest way is to convert the figure to an `eps`-File.

1. Print the figure through a PS printer driver (HP LaserJet 4/4M PS works for me) to a file.

   If you use Powerpoint, make sure that you print only one page. Only one-page documents can be converted to `EPS`. Make sure that you set the printer to color if you want a color picture in LaTeX. The Powerpoint printer dialog is quite buggy, so you have to change to color in the "Properties" and then uncheck the "Black & White" checkbox again.

2. With a text-editor on either Windows (not Notepad or Word; WordPad is fine) or UNIX, remove the command containing "`setpagedevice`". The command looks similar to this:

```
[{
%%BeginFeature: *PageSize Letter

    1 dict
    dup /Policies 2 dict dup /PageSize 2 put dup /MediaType 0 put put
setpagedevice
2 dict
    dup /PageSize [612 792] put
    dup /ImagingBBox null put
    setpagedevice
%%EndFeature
} stopped cleartomark
```

This code has to be removed[4]. Search for the string `setpagedevice` and then remove the text from "`[{`" until the closing "`} stopped cleartomark`". Save the `ps`-file as text.

3a. With GSView on Windows, convert to EPS:

Open PS-file in GSView, choose PStoEPS from the file menu and save the file as *filename*.`eps`.

3b. If you use UNIX, process the file with `ps2epsi` *`filename.ps`* which gives you a file ending on `.epsi`. Rename the file to `.eps`. Don't forget to remove the `setpagedevice` command **before** converting to `eps`! Unfortunately, this method doesn't work on some printers, so the method using GSView is strongly preferred.

4. Copy the file to your LaTeX figures directory and include into LaTeX-file using `\includegraphics` according to Section 2. The result of this should look similar to Figure 4.

If the figure has the wrong orientation (Powerpoint slide), add `angle=270` as a parameter to the include command. If the figure overwrites part of the text (e.g., the figure caption), add `clip=true`.

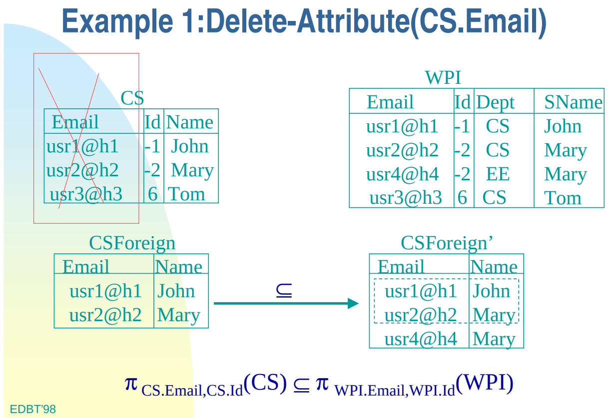

Figure 4: A Powerpoint slide in LaTeX

6 Conclusion

LaTeX IS USEFUL. It's hard, I know. But Word is harder. And it doesn't even look good. And you guys are all CS-people—learning programming languages shouldn't be a problem for you. LaTeX exists on all platforms including DOS and Windows. If you want LaTeX for your PC, search for `miktex` or `emtex` on the Web. This document will grow as (i.e., if) I have more time to write stuff. For now, this is all.

Andreas Köller

[4]If you would like to write a Perl-script for this task, please feel free to contact me