THE UNIVERSITY OF AKRON
Mathematics and Computer Science

Using LaTeX to Create Quality PDF Documents for the World Wide Web

Directory

Last Revision Date: August 11, 1999 Version 1.17

Using LaTeX to Create Quality PDF Documents

Table of Contents

1. Introduction

This article is devoted to methods of creating fine quality interactive PDF *documents using LaTeX*. For individuals who write technical material, TeX and LaTeX are the ideal authoring tools. Even though this article is written primarily for LaTeX users, people who prefer pure TeX may derive much from this article as well. I, myself, prefer $\mathcal{AMS}$-TeX; even so, there is no denying the power, convenience and utility of LaTeX.

Beyond the question of the *content* of your document (content being of premier importance), what elements go into making an attractive document suitable for the WWW?

Page Layout A document not meant to be printed but to be viewed over the Internet must be comfortable enough to the eyes to be read over long periods of time; therefore, making a good choice for page layout is certainly important.

Color *Emphasis* is a traditional way of attracting the attention of the reader to a particularly important point. We discuss the ways of

adding color to create a more eye pleasing document over a computer screen. For LaTeX users, that's you, David Carlisle's `color` package provides all the necessary macros.

Hypertext Of course, what would a WEB document be without cross-referencing using hypertext? LaTeX is known for its cross-referencing capabilities, but how can these be translated into hypertext links? Sebastian Rahtz has solved this problem with his `hyperref` package. A discussion of `hyperref` is given in section 6; the discussion is narrowly focused and is not a manual of use.

There are other issues that go into the mix of a quality WEB document:

- Choosing a TeX System
- Use Type 1 Fonts
- Creating PDF

These important points are covered but in less detail.

Finally, I hope this brief article will be of value to the LaTeX community. I certainly want to encourage people with technical knowledge to share a portion of it with the Internet community. DPS

2. Some Notes on TeX

This section is devoted to beginners, or newbies, in LaTeX. If you are interested in putting technical material on the WEB using LaTeX, you must begin by getting a TeX system (subsection 2.1), reading about LaTeX (subsection 2.2), and communicating with the LaTeX community whenever you have questions or problems you cannot otherwise answer yourself (subsection 2.3).

2.1. Choosing a TeX System

Y&Y, Inc., http://www.yandy.com, produces an exceptional commercial TeX system for Windows 95/98/NT. I used the Y&Y TeX system to produce this article.

Under the freeware category, a good choice for Windows 95/NT is MikTeX, http://www.inx.de/~cschenk/miktex/index.html by Christian Schenk cs@miktex.de

For the Mac, there is OzTeX, Textures and CMacTeX. All very

good, but all are either shareware or commercial products (Textures). Information about these and others can be obtained from Internet resources, see the next paragraph.

Sebastian Rahtz keeps a list of "Interesting TEX-Related URLs"

http://www.tug.org/interest.html

that represents a tremendous resource of information. Included is the collection of all TEX systems—freeware, shareware, and commercialware—available for use, as well as addresses for obtaining more information about these systems and where you can obtain them. By the way, the root address,

http://www.tug.org,

is that of the TEX User's Group, or TUG for short. Here is a quote from their WEB page referenced above:

> The TEX Users Group (TUG) was founded in 1980 to provide leadership for users of TeX, Donald Knuth's revolutionary typesetting system. It represents the interests of

TeX users worldwide: if you use TeX in any of its forms, please consider joining TUG.

2.2. Learning about LaTeX

Learning to use LaTeX is another matter. One of the advertised advantages of LaTeX is that it enables the new user to be composing simple documents in a very short time without a great knowledge of what goes on behind the scenes. (Obviously, this is very desirable; imagine not being able to use Microsoft Word without knowledge of the program code!)

TeX is a compiler written by Donald Knuth, just as any compiler for a computer language (Pascal, C, C++, etc.). When we purchase a compiler, we don't expect a complete manual on how to use the language that the compiler supports. Knowledge of the language must be acquired from other sources, perhaps through a college course, or through self-study. It is the latter case that all of us have learned about LaTeX.

TEX, being a compiler, compiles a source file consisting of the command language for which it was designed. (A C++ compiler compiles a source file consisting of commands from the C++ language.) What is the command language for TEX? Fundamentally, the command language consists of two distinct elements: (1) Words and phrases from a human language (such as English, German, French, etc.); and (2) macro commands.

Given that you know something about a human language, you need to learn about TEX's macros, for it is the macro that does much of the formatting and page layout. LATEX is a large collection of macros written by experts that perform certain routine and useful tasks; there are macros for creating cross-referencing, tables of contents, indices, and footnotes, for example.

To learn how to use the macros of LATEX, you must read a book or two. Included in the References Section is a list of generally well-received books on LATEX.

Leslie Lamport is the one who originally wrote LATEX, but many others have taken over the task of improving the quality and volume

of LaTeX. Reference [1] is his documentation of LaTeX.

The two books by Goossens *et al*, [2, 3], together cover the evolution of LaTeX beyond Lamport. The three make up a library of documentation on LaTeX.

The book by Kopka and Daly, [5], is a very nice tutorial and reference in one volume.

Finally, I would be remiss in my duties if I did not mention the book by Donald Knuth, [6]. Should you ever want to learn more about writing macros and the inner workings of TeX, *The TeXbook* is the place to start.

In terms of what is available over the Internet, check out

- "The Not So Short Introduction to LaTeX2e," by Tobias Oetiker Hubert Partl, Irene Hyna and Elisabeth Schlegl.

This is available in PDF format and can be found at

`CTAN:/tex-archive/info/lshort`

where CTAN refers the URL of any Comprehensive TeX Archive Network, see subsection 2.4 for details.

2.3. Newsgroups and Information Centers

When the TEX compiler or a LATEX macro package does not work as advertised, then you have a problem. Where to go? Colleagues with a knowledge of LATEX can be of help certainly. Alternately, you can seek help from newsgroups devoted to discussing specialized issues.

For all things TEX, including LATEX, subscribe to

comp.text.tex

Since you are planning to publish on the WWW, you may need help with PDF, the Portable Document Format. Subscribe to

comp.text.pdf

Here is a list of other information centers.

- (La)TeX navigator
  http://www.loria.fr/services/tex/english/index.html
- The LaTeX Catalogue Online, Home Edition
  http://www.cmis.csiro.au/Graham.Williams
  /TeX/catalogue.html

- Y&Y Inc, Resources
  `http://www.yandy.com/resources.htm`
- TUG Headquarters
  `http://www.tug.org/`
- Interesting TEX-Related URLs
  `http://www.tug.org/interest.html`

2.4. CTAN: Acquiring all things TeX

The Comprehensive TeX Archive Network, or CTAN, is a collection of sites that are repositories of TEX materials: systems, fonts, macros etc. There are many CTAN sites, the one to start with is

`http://ctan.tug.org/`

This site contains a listing of all CTAN sites. Choose the one nearest to you to minimize downloading time. The site can also be accessed through FTP:

`ftp://ctan.tug.org/`

3. Use Type 1 Fonts

The major point here is to use Type 1 fonts; they render better over the web than do traditional fonts, bitmap fonts (pk fonts), usually provided by a freeware TEX system.

If you have a `pdf` file that uses bitmap fonts produced by your LATEX system, you know how bad the fonts look on screen.

To check for the presence of bitmap fonts, open the suspect `pdf` file in the Acrobat Reader. Click on

`File > Document Info > Fonts ...`

In the **Font Info** window, you will see a listing of all fonts used in your document. The bitmap fonts are classified as **Type 3** fonts by Adobe.

Both systems recommended above, Y&Y and MikTEX, ones for which I have some experience, come with Type 1 fonts. The TEX systems for the Mac support Type 1 fonts as well.

The native font for Y&Y is the Type 1 font, so you need only install the Y&Y system off the CD, and you will be up and running

in 15 minutes.

The natural font for the MikTEX system is the bitmap font; however, the standard installation now comes with the Type 1 fonts provided by Y&Y, BlueSky Research, and the AMS (American Mathematical Society).

There are two approaches for making `dvips` use the Type 1 fonts (1) edit the `config.ps` file, and (2) use the `-P` command line switch.

Search for the `config.ps` file; for MikTEX 1.20 it is located in the folder

```
\texmf\dvips\config\config.ps
```

The following is a verbatim listing of a portion of that file, again, for MikTEX 1.20:

```
% Configuration of postscript type 1 fonts:
p psfonts.map

% This shows how to add your own map file.
```

```
% Remove the comment and adjust the name:
% p +myfonts.map
p +bsr.map
```

Here, I've added the line `p +bsr.map`. The above lines were taken from MikTEX 1.20b, later versions may differ slightly from this example.

This method will be useful if you are planning to create only PDF files and, hence, always need the Type 1 fonts. If, however, you want to use `pk` fonts (for non-PDF documents) as well as Type 1 fonts (for PDF documents), there is an alternate approach that is convenient: Use the switch `-P`:

```
dvips.exe -j0 -Pcmz -Pamz -o <filename>.ps <filename>
```

The switches `-Pcmz` and `-Pamz` tell `dvips` to read the configuration files `config.cmz` and `config.amz` for the CM fonts and AMS fonts mappings, respectively.

The above method causes `dvips` to embed the fonts within the postscript files, this makes for a large postscript files. Eliminating

the `-j0` turn back on partial font downloading. This sometimes cause problems when you create a PDF file using the Acrobat Distiller.

Another possibility is to use the configuration files `config.cm`and `config.ams`:

```
dvips.exe -Pcm -Pams -o <filename>.ps <filename>
```

These switches cause `dvips` *not* to embed the fonts in the postscript file. In this case, it is up to the Acrobat Distiller to find the fonts; see Section 8.1 on 55; in particular read the discussions in that section on Specifying Font Locations and Subsetting Fonts.

Beginning with version 1.20c, MikTEX provides a configuration file (config.pdf) designed to produced optimized PDF files:

```
dvips.exe -Ppdf -o <filename>.ps <filename>
```

For other systems, read very carefully the accompanying documentation.

4. Page Layout: Designing for the Web

When publishing a document on the web one is always faced with the question: Is this document meant to be printed, or simply read over the web?

If the document is meant to be printed and then read, then no special advice is needed. However, if the document is meant to be purely a creature of the Internet—not meant to be printed—then some additional considerations are important.

The three major points are (1) design the text region so that a single page fits on a 14 inch screen monitor (no scrolling should be necessary); (2) make the dimensions of your page roughly 3 by 2 (this is not an absolute necessity); (3) crop all pages to trim off all unnecessary white space around the margins (this will allow the user to magnify the page to help the eyes read a large amount of text on a computer monitor for long periods of time).

Comments on (1) For documents to be read on a computer screen, having the whole page fit on the screen is a real reading pleasure. I

found it rather annoying always having to scroll down, or up, to see the rest of the page. When reading large amounts of material, constantly scrolling the page vertically can be distracting and fatiguing as well. Rather than scrolling, it is much easier to simply paginate—go to the next page to continue reading.

Comments on (2) The dimensions of 3 by 2 (width by height) are not hard and fast. I personally prefer a narrower width. This document is closer to a 4 by 3 ratio (width by height). I've set

```
\setlength{\textwidth}{4.166in}
\setlength{\textheight}{3in}
```

(Don't ask for the history of the `.166` decimal part!)

These particular dimensions seem to work well, they create a large enough text area to present a substantial amount of material on it, yet small enough that it will magnify nicely (from within Acrobat Reader) to a large size font, comfortable to read.

Comments on (3) There are options within **`hyperref`**, discussed in section 6, that allow you to crop white space from around your page.

The cropping will manifest itself when you transform the document into PDF. This cropping of white space enables the user to magnify the page even more, which increases the size of the font and makes the document easier to read. See the comments on `pdfpagescrop` below.

5. Using Color

In this section, we discuss two types of color problems: Coloring text and boxes; and creating a background color.

- **Coloring Text and Boxes**

Color support is provided through David Carlisle's `color` package, which is included in your LaTeX document when you use `hyperref`. Documentation for this package is contained in the file `grfguide.dvi` and can be found in the `grapics` folder of the LaTeX files.

To enhance the look of your document, you might occasionally include a little color. Carlisle's `color` package predefines a few colors

for you: black, white, red, green, blue, cyan, magenta and yellow. These are available for use; you can also define your own colors.

The macro `\textcolor` is used to add color to a particular word or phrase. For example: Green. This was obtained with the code

```
\textcolor{green}{Green}.
```

The first argument is the color, the second is the word or phrase to be colored.

This is a very unpleasant green, so I have defined my own personal color green. In the preamble, and after `hyperref` is included into the file, define a color

```
\definecolor{webgreen}{rgb}{0,.5,0}
```

then use it: Green! (`\textcolor{webgreen}{Green!}`).

Sometimes you might want to box around an important point.

```
\colorbox{yellow}{point.}
```

The first argument is the color and the second the text. Or you might want to put a frame around your important point.

```
\fcolorbox{blue}{yellow}{point.}
```

The first argument is the color of the border, the second the background, and the third is the text. A useful application for this is for displaying special equations:

$$\fcolorbox{blue}{yellow}{$x^2+y^2=1$} \tag{1}$$

The code for this is

```
\begin{equation}\setlength{\fboxrule}{1.5pt}
              \fcolorbox{blue}{yellow}{$x^2+y^2=1$}
               \label{eq:xxyyeq1}
\end{equation}
```

Notice how to change the thickness of the bounding line of the frame box.

Study the style files of this document, `webplain.sty`, to see how to color the section titles.

- **Coloring the Background**

It is possible to paint the background a color other than the usual white by using the `\pagecolor` macro from `color` package (See the manual `grfguide.dvi` for the correct usage.) I generally prefer, however, to use raw postscript commands to create a background effect.

The simplest case is to paint the same background on all pages of the document. In the preamble, insert the lines:

```
% For dvipsone
\special{headertext=/bphook{gsave clippath
   0.98 0.92 0.73 setrgbcolor fill grestore} def}%
% For dvips
\special{! userdict begin /bop-hook{gsave clippath
   0.98 0.92 0.73 setrgbcolor fill grestore} def end}%
```

Here is a more complicated variation; it is the one that appears in this document. In the preamble of this document I have placed the following code:

```
% For dvipsone - this document
\special{headertext=/bphook{
   PhysicalPage ColorPageNumber eq {gsave clippath
   0.98 0.92 0.73 setrgbcolor fill grestore} if} def}%
```

Alternately, someone who is using `dvips` would use the following lines of code:

```
% For dvips
\special{! userdict begin /bop-hook{2 copy
   /LogicalPage exch def /PhysicalPage exch def
   PhysicalPage ColorPageNumber eq {gsave clippath
   0.98 0.92 0.73 setrgbcolor fill grestore} if} def
   end}%
```

`PhysicalPage` is the page number in TEX's numbering scheme; this is pre-defined for Y&Y, but is passed to the `bop-hook` procedure as a parameter and must be defined, as above.

`ColorPageNumber` is a postscript definition made at the beginning

of this section—the page I wanted to have a different background color. Here is the code from the source file:

```
\subsubsection{Coloring the Background}
% For dvipsone - this document
\special{! /ColorPageNumber{\arabic{page}}def}
% For dvips
%\special{! userdict begin
%  /ColorPageNumber{\arabic{page}} def end}%
```

The beginning of page procedure (`bophook` for DVIPSONE, and `bop-hook` for `dvips`) executes the code. If the current page number at the beginning of the page equals `ColorPageNumber`, we paint the page a nice yellow color; otherwise, do nothing.

The third and final example is more complicated still. See if you can figure out what this code does, or try it out yourself in a document of your own.

In the preamble, put

```
% For dvipsone
```

```
\special{headertext=/bphook{gsave clippath
   PhysicalPage 1 eq {0.98 0.92 0.73}
      {PhysicalPage ColorPageNumber ge
      {0.98 0.92 0.73}{1 0.8 0.6} ifelse}
  ifelse setrgbcolor fill grestore} def}%
% For dvips
\special{! userdict begin /bop-hook{2 copy
  /LogicalPage exch def /PhysicalPage exch def
  gsave clippath PhysicalPage 1 eq {0.98 0.92 0.73}
  {PhysicalPage ColorPageNumberer ge
  {0.98 0.92 0.73}{1 0.8 0.6} ifelse}
  ifelse setrgbcolor fill grestore} def end}%
```

You are not restricted to merely painting the background a different color, you can also include some "watermark" graphics. Examples of this are in my calculus and algebra tutorials. See

http://www.math.uakron.edu/~dpstory/tutorial/c1/c1_menu.pdf

http://www.math.uakron.edu/~dpstory/tutorial/mptii/lesson01.pdf

6. Hyperref

For users of LaTeX, perhaps the easiest way of acquiring hypertext links (and form features) in their Web publications is to use the `hyperref` package written by Sebastian Rahtz. The package can be obtained from any CTAN site, and may be found in the subdirectory:

`CTAN:/tex-archive/macros/latex/contrib/supported/hyperref`

If you already have `hyperref`, you might download the latest release.

6.1. Package Options

There are numerous package options that can be selected, we list only the ones used for this article. A complete listing of all options is contained in the `hyperref` manual (in PDF form) that accompanies the package. The manual is dated June, 1998, and is a little out of date.

A description of `hyperref` and all its options may also be found in [4], *The LaTeX Web Companion*, written by Michel Goossens and

Sebastian Rahtz with Eitan Gurari, Ross Moore, and Robert Sutor.

```
\usepackage
  [dvipsone,                          % or dvips
%------------- Backref Switch -------------------------
  pagebackref,                        % or backref
%------------- Cropping -------------------------------
  pdfpagescrop={53 436 389 704},   % dvipsone; 19 lines
% pdfpagescrop={53 486 389 754},   % dvips; 19 lines
%------------- Color Links ----------------------------
  colorlinks=true,
  linkcolor=webgreen,                  % defined below
  filecolor=webbrown,                  % defined below
  citecolor=webgreen,                  % defined below
%------------- Doc Info -------------------------------
  pdftitle={Using LaTeX to Create Quality PDF Documents},
  pdfauthor={D. P. Story},
  pdfsubject={How to use hyperref, color packages},
  pdfkeywords={LaTeX hyperref pdf},
%------------ Doc View -------------------------------
```

```
  bookmarksopen=false,
  pdfpagemode=None]{hyperref}
%
% Define some eye-pleasing colors for this document
%
\definecolor{webgreen}{rgb}{0,.5,0}
\definecolor{webbrown}{rgb}{.6,0,0}
```

The interested reader needs to consult the manual that comes with the `hyperref` package for a full listing of all the large number of options. We will be satisfied by commenting on the ones chosen for this document.

- `dvipsone` indicates the "backend driver". There are many possibilities here. A common alternative is `dvips` or `pdftex`.
- `backpageref` adds in the page numbers at the end of each bibliographic entry for each instance that entry was cited in the document. There is a `backref` option as well; this options inserts the section numbers rather than the page numbers. See

Section 7 for more details.

- `pdfpagescrop={53 436 389 704}` tells `hyperref` to leave cropping commands in the `dvi` file. Cropping will appear when you convert to a PDF file. For a detailed description of how to get the "cropping values", see Using Acrobat to Crop the Document on page 60.
- `colorlinks=true` tells `hyperref` to color the text of the links. The default is to draw a colored rectangle around links; this is not very attractive.
- `linkcolor=webgreen` defines the color as `webgreen` for within document jumps.
- `filecolor=webbrown` defines the color as `webbrown` for jumps to another document.
- `citecolor=webgreen` defines the color as `webgreen` for jumps to citations in the References Section; these jumps are within document, so color them the same as `linkcolor`.
- The values of `pdftitle,pdfauthor,pdfsubject,pdfkeywords` will appear under the

`File > Document Info > General...`

menu from within Acrobat Reader or Exchange.

- `bookmarksopen=false` means the bookmarks (within `Acrobat Reader` are *not* expanded to all levels. Only the highest levels are shown. (See the bookmarks of this document.)
- `pdfpagemode=None` determines how the page will look when you open the document in Acrobat. `None` here means to just show the page. `pdfpagemode=UseOutlines` is the default, which means to open the document with the bookmarks (outlines) window open.

Again, all options—and there are a lot of them—are documented in the `hyperref` manual. This set of options is suitable for most `Web` documents.

6.2. Creating Marks or Anchors

There are two ways of creating a marker (a marker being a target of a hypertext jump).

1. `\label{name}`: This is the built in labeling system of LaTeX. A label can appear after a section, subsection, equation, figure, or an enumerated item, such as this one.

2. `\hypertarget{name}{text}`: For marks that do not correspond to any situation `\label` would be used, as discussed in point #1 above, the `\hypertarget` marker can be used. Perhaps it can be used to mark a particularly important passage that needs to be brought to the reader's attention. The syntax for this marker is `\hypertarget{name}{text}`. See below for an example.

6.3. Managing References

In this section the basic cross-referencing capabilities of LaTeX are discussed.

You can use a label, `\label{name}`, to give section headings, figures, equations, etc., symbolic `names`. These labels can be referenced in the document using the macros `\ref{name}` and `\pageref{name}` macros.

When using the `hyperref` package, the standard labels are turned into destination pdfmarks and the cross-references are made into hypertext links understood by the Acrobat Reader.

For example, let's jump to a test area, Section 8.2 on page 64. As you can see, this makes the section number and the page number into a hypertext link. Below is a listing of the source code:

```
For example, let's jump to a test area,
Section~\ref{s:destpage} on page~\pageref{s:destpage}.
As you can see, this makes the section number and
the page number into a hypertext link.
```

Now let's jump to equation (2) also on page 64.

The `hyperref` package melds seamlessly into LaTeX; however, the method of referencing is a bit puny. See Section 6.4.1 on page 33 for

more details on making more attractive references—such as the one just made.

6.4. Jumping within the Document

Sebastian makes a distinction between marks created by `\label` and marks created by `\hypertarget`. In the next several subsections explore the differences.

- **Using label/hyperref**

Jumping to a `\label` mark can be accomplished by using the seamless methods of subsection 6.3—the `\ref{name}` and/or `\pageref{name}` macros—, or by using the `\hyperref` macro.

Suppose we want to jump to Section 6.3 on page 31 in a nice hypertext way. The label for that section is `\label{s:latexlabel}`; use the syntax

```
\hyperref[s:latexlabel]{Section~\ref*{s:latexlabel}}
```

to get a link like this: Section 6.3.

Notice that we write

`\ref*{s:latexlabel}`,

this turns off the creation of a hypertext link. (The same is true for `\pageref*{s:latexlabel}` as well.) We can say, without creating hypertext links, 'Section 6.3' (`Section~\ref*{s:latexlabel}`) on 'page 31' (`page~\pageref*{s:latexlabel}`).

- **Using label/autoref**

The macro `\autoref{name}` can be also be used to refer to a `\label`. Associated with each `\label` is a name (or category) and a number. The macro `\autoref` puts these two together for form a reference phrase. Here are some examples from around this document.

1. subsection 6.3: `\autoref{s:latexlabel}`.
2. Equation 2: `\autoref{eq:xxyyzz}`.
3. item 1: `\autoref{enum:label}`.

This is a quick and dirty way of setting hypertext links. Sometimes the text given by `\autoref` is not adequate for you needs, For exam-

ple, in item 3 above, perhaps you want the word 'item' capitalized. In this case, you must use the `\hyperref` macro:

Item 3: `\hyperref[it:myitem]{Item~\ref*{it:myitem}}`

as illustrated in subsubsection 6.4.1.

- **Using the nameref package**

This might be a good opportunity to mention a macro from Sebastian Rahtz's `nameref` package. (The `nameref` package is automatically included when you use `hyperref`.)

Suppose we wanted to refer to the title of Section 6.3 (a title that may change with revisions), in this case, we can refer to it as follows: "See Section 6.3, entitled Managing References, on page 31."

```
See Section~\ref*{s:latexlabel}, entitled
\nameref{s:latexlabel}, on page~\pageref*{s:latexlabel}.
```

Here we use the `\nameref` macro from the `nameref` package. (Confusing isn't it?)

`\Nameref` is a more expansive version of `\nameref`, expanding to both title and page number.

`\Nameref{s:latexlabel}` yields: 'Managing References' on page 31.

Unlike `\ref` and `\pageref`, there is no '`*`' version for `\nameref` or `\Nameref`. Should you want to turn off hypertext linking, you can use the `NoHyper` environment: The title of Section 6.3 is Managing References. This was obtained by

```
The title of \hyperref[s:latexlabel]
{Section~\ref*{s:latexlabel}} is
\begin{NoHyper}\nameref{s:latexlabel}\end{NoHyper}.
```

- **Using hypertarget/hyperlink**

When `\hypertarget` is used to mark a passage in the document, the macro `\hyperlink` is used to jump to the defined mark. The syntax for `\hyperlink` is

```
\hyperlink{name}{text}
```

At the end of the document, the mark `\hypertarget{mymark}` has been placed. To jump to that mark we can say Jump There!

You cannot use `\hyperlink` to jump to marks defined by `\label`'s.

6.5. Jumping to another Document

As in the case of internal jumping, a distinction is made between jumping to another document where the target mark is defined using a `\label` and target marks defined by `\hypertarget`.

- **Using label/hyperref**

For this to work, you must use David Carlisle's xr package. In the preamble have the statement

```
\usepackage{xr}
```

Important: `hyperref` comes with its own `xr` package. You will have to disable the version of `xr` that comes with the standard LaTeX distribution. You can find it in the `tools` folder of the LaTeX macros.

To successfully jump to the target file `hytarg` in the same directory as this file, again, in the preamble, you must have the statement:

```
\externaldocument[hytarg,]{hytarg}
```

The name that is inside the brackets is used to reference the file; it can be anything, such as `\externaldocument[A-]{hytarg}`. The file can appear in a subfolder, in this case use

`\externaldocument[mytarg,]{myfolder/hytarg}`.

In the file `hytarg`, there is a section entitled 'Main Section' that has been labeled `\label{s:main}`. Let's jump there. We can use the seamless method: "See Section 2 for more details."

Text: `See Section~\ref{hytarg,s:main} for more details.`

Let's see if `\autoref` will work: See section 2 on page 3 of the file `hytarg`.

Text:
```
See \autoref{hytarg,s:main} on
    page~\pageref{hytarg,s:main}
```

Let's reference Equation 1 as well.

As explained earlier, you can use `\hyperref` to reference labels as well, *even when they are in another file*: "Go to Equation (1)." This is accomplished by the code:

```
\hyperref[hytarg,eq:xyeq1]
          {Equation~(\ref*{hytarg,eq:xyeq1})}
```

Notice that the reference to the file occurs each time you reference the label. Notice also the '`*`' form of the `\ref` to suppress the creation of a redundant hypertext link.

Finally, we note that `\nameref` works correctly with `xr`. We simply type `\nameref{hytarg,s:main}` to get Main Section, a nice hypertext link on the title of the section in another file.

Section Remarks: The use of the brackets in the

```
\externaldocument[hytarg,]{hytarg}
```

is optional. In this document I could have said

```
\externaldocument{hytarg}
```

and then later just typed

```
\hyperref[eq:xyeq1]{Equation~(\ref*{eq:xyeq1})}
```

This technique will work provided there is not another label by the same name as `\label{eq:xyeq1}`, anywhere in the current document, or in any other external document you have referenced with the `\externaldocument` command. Of course, this goes uniformly for all exterior labels referenced.

If there is a possibly of conflict between labels it is perhaps best to bring the exterior tags using the method discussed above.

By the way, the use of the comma in

```
\externaldocument[hytarg,]{hytarg}
```

is a personal preference. As mentioned above, you can define the name in the brackets anyway you wish:

`\externaldocument[A-]{hytarg}` or,

`\externaldocument[hytarg]{hytarg}` or,

`\externaldocument[XYZ]{hytarg}`.

In each case you would change the referencing accordingly, e.g.,

```
\hyperref[XYZeq:xyeq1]{Equation~(\ref*{XYZeq:xyeq1})}.
```

- **Using hypertarget/href**

If a mark is created in another file using `\hypertarget`, jumping to it is accomplished using the `\href` macro. The syntax is

```
\href{URL}{text}
```

Jumping to a target uses the same syntax as `html`, for example: Let's try to jump to `\hypertarget{mytarget}` in the target file `hytarg`: Jump Here! The code for this is

```
\href{hytarg#mytarget}{Jump Here!}
```

Notice the use of the '`#`' symbol here to separate the target file from the target mark in that file.

Finally, it is important to note that the macro `\href` does not use the `xr` package.

6.6. Jumping to a URL on the WWW

The `\href` macro is also used to jump to full URL's as well. Here are some examples

1. Go to e-Calculus
   ```
   \href{http://www.math.uakron.edu/~dpstory/
         e-calculus.html}{e-Calculus}
   ```

2. Comments? E-mail D. P. Story
   ```
   \href{mailto:dpstory@uakron.edu}{D. P. Story}
   ```

3. To simultaneously construct an URL and show the Web address use `\url`:
   http://www.math.uakron.edu/~dpstory/e-calculus.html
   Type
   ```
   \url{http://www.math.uakron.edu/~dpstory/%
        e-calculus.html}
   ```

4. Mail: dpstory@uakron.edu. (`\url{dpstory@uakron.edu}`)

The default color for url links is *cyan*; the colors for URL links can be changed, however, to any user defined color, say brown, by inserting

```
urlcolor=webbrown
```

in the options of `hyperref`, see Section 6.1, page 26. Below is the definition of `webbrown`:

```
\definecolor{webbrown}{rgb}{.6,0,0}
```

This is placed in the preamble of the document, after the `hyperref` package has been read in. (The macros for color are defined by the `color` package of David Carlisle, which is automatically read in by `hyperref`.)

6.7. Voilà: Bookmarks

Bookmarks (or outline annotations) are a navigational aid used by the Acrobat Reader. The `hyperref` package automatically adds bookmark code to an auxiliary file for `\sections`, `\subsections`, etc. For

example, the above section title appears in the source code as you might expect it. The code

```
\subsection{Voil\‘a: Bookmarks}
```

yields the correct results; i.e., open the bookmarks window of this file in the Acrobat Reader and see the entry "Voilà: Bookmarks", including the accent. The process of converting accents and other special symbols to the encoding of the PDFEncoding character set is quite complicated.

Heiko Oberdiek wrote an encoding scheme, called PD1, which takes the bookmark entries—the titles of each section, subsection, etc.,— and writes them to the file `\jobname.out` using PDFDocEncoding, the encoding that the Acrobat Reader expects the strings to use.

- **Using `\texorpdfstring`: The e^x Function**

Sometimes it is desired to have mathematics in the title of a section. Such a title does not translate to PDFDocEncoding automatically.

In this case, use `\texorpdfstring` to offer an alternative for your bookmarks.

```
\subsubsection{Using
\texorpdfstring{\cs{texorpdfstring}}{\\texorpdfstring}:
The \texorpdfstring{$e^x$}{EXP} Function}
```

Again, look at bookmarks window to see the results of these commands.

As the name of the command suggests, `\texorpdfstring`, the first argument is use in the TEX document, the second for the PDF bookmarks.

- **An Example:** • $a^2 + b^2 = c^2$

You can access individual characters in the PDFDocEncoding character set through an octal escape code (for example, `\200` is the 'bullet') or, access can be gained symbolically through predefined text commands (for the 'bullet' symbol the text command is `\textbullet`).

Below is the section title for this section

```
\subsubsection{An Example:
    \texorpdfstring{$\bullet\ a^2+b^2=c^2$}%
    {\textbullet\ a\texttwosuperior\
     + b\texttwosuperior\ = c\texttwosuperior}}
```

Here, `\textbullet` and `\texttwosuperior` are the the bullet symbol and the exponent of 2, respectively.

A complete listing of these text commands can be obtained by LaTeXing the file `hyperref.dtx`.

6.8. Icon Buttons

In this section we demonstrate two methods of creating direction icons, one using rules and the other using `.eps` images.

- **Icons using Rules**

You can use `\colorbox` from the `color` package to create nice looking buttons. For example,

```
\colorbox{webgray}{\textcolor{webblue}{$\bigl\Uparrow$}}
```

produces a very nice up arrow, shown below on the left. We can expand the space around the arrow, if desired, by using `\fboxsep`, for example, we can `\setlength{\fboxsep}{6pt}` to obtain the icon shown below on the right.

You can then combine this with hypertext commands to create a link, here are a couple of macros for doing so:

```
\newcommand\ArrowUp[1]{\setlength{\fboxsep}{6pt}\normalsize
\raisebox{-\depth}[0pt][0pt]{#1{\colorbox{webgray}%
{\textcolor{webblue}{$\bigl\Uparrow$}}}}}

\newcommand\ArrowDown[1]{\setlength{\fboxsep}{6pt}\normalsize
\raisebox{-\depth}[0pt][0pt]{#1{\colorbox{webgray}%
{\textcolor{webblue}{$\bigl\Downarrow$}}}}}
```

Use these macros by typing `\ArrowUp{\hyperlink{intro}}`, for example, to get the icon shown to the left. The command `\ArrowUp` takes one argument, use any hypertext command. For this example, `\hyperref` was used: the first argument of `\hyperref` is the destination '`intro`'; the second argument is the material to be typeset, which becomes the hot button to initiate the jump, here, it is the icon defined using the `\colorbox` command.

As another example, to jump to the AcroTEX Home Page, simply use as the argument of `\ArrowUp` the following:

```
\href{http://www.math.uakron.edu/~dpstory/acrotex.html}
```

- **Icons using EPS**

The standard LaTeX graphics packages can be used to include `eps` pictures. These pictures can be linked to a within document jump, exterior document jump, or to a full-fledged absolute `url` jump. To illustrate ...

The following icon button jumps to Section 1, the 'Introduction' to this article: 

The source code that created this button follows:

```
The following icon button jumps to Section~\ref*{intro}, the
‘\begin{NoHyper}\nameref{intro}\end{NoHyper}’ to this article:
\hyperref[intro]{\mbox{\includegraphics[scale=.5]{home.eps}}}
```

We earlier typed `\section{Introduction}\label{intro}`, and in the preamble we have `\usepackage{graphicx}` to bring in the `graphicx` package into the document.

You can, I'm sure, think of interesting applications to the ways of using `eps` files as hypertext buttons. Here's another illustration.

Figure 1: College of the Redwoods

I downloaded the picture `campus1.gif`, as seen in Figure 1, from the web site of the College of the Redwoods. Loading this file into Photoshop Lite, I saved it as an `eps` file. I took note of the size of the image, which was 160 points wide by 106 points high. Photoshop Lite gave me a choice of saving as **ASCII** or **binary**, For DVIPSONE: I saved in **binary** with **TIFF** preview (which previews in DVIWindo); For DVIPS: I chose **ASCII** with *no* **TIFF** Preview. (Important: I ran `dvips` with the `-K` switch as well.)

In the preamble, I inserted `\usepackage{graphicx}`, and to include the graphics with a hypertext link, I typed:

```
\href{http://online.redwoods.cc.ca.us/}
   {\includegraphics{campus1.eps}}
```

Dvips is notorious for issuing the much feared error message:

```
!  premature end of file in binary section
```

It appears that this can be overcome by avoiding binary data in the `EPS` file, and using the `-K` switch.

The difficulties encountered with `dvips` can be traced to the use of Adobe PhotoShop Lite. Using other applications or utilities to convert from `.gif` to `.eps` may not cause as much problems.

7. Using Backref

The `backref` package is a set of biographical back referencing macros written by Sebastian Rahtz; `backref` comes with `hyperref` and it can be controlled through the `hyperref` option field.

What is back referencing? Back referencing refers to the process of inserting page numbers (or section numbers) at the end of each

bibliographic entry for each instance that entry was cited in the document.

For example, let's reference Donald Knuth's *The TEXbook* (see [6]). Now click on the hypertext link to inspect the bibliographic reference. Following Professor Knuth's entry is a list of page numbers `backref` has inserted. These page numbers are themselves hypertext links that point the locations in the document where citations of the book were made. This reference is the one on page 52.

This feature is obviously very useful for quickly locating all references to a given book or article made in the document.

This document specifies the `backpageref` option; hence, page numbers appear following the bibliographic entries. The other option is `backref`; this option inserts the section numbers of each citation.

The bibliography can be entered using `thebibliography` environment within the document itself (see [2, page 372]), or can be drawn in from a bibliographic database processed by the *BibTEX* utility ([2, page 375]).

Important Point: When preparing the listing it is important to

leave a blank line or a `\par` between bibliographic entries; otherwise, `backref` may not function correctly.

Below is an abbreviated listing of the References section of this document. Note the blank lines between entries *and after the last entry as well.*

```
\begin{thebibliography}{[1]}
\bibitem{book:Lamport}
   Leslie Lamport, \textsl{\LaTeX: A Document Preparation}
   System (2nd ed.), Addison-Wesley Publishing Company,
   1994, ISBN 0-201-52983-1.

\bibitem{book:Goossens1}
   Michel Goossens, Frank Mittelbach and Alexander Samarin,
   \textsl{The \LaTeX{} Companion}, Addison-Wesley
   Publishing Company, 1994, ISBN 0-201-54199-8.

\end{thebibliography}
```

The citation linking in `hyperref` does not work with all biblio-

graphic packages. The "`natbib`" package is recommended. See CHAPTER 10 of [2] for a discussion of *BibTEX*.

The `backref` package can also be used without `hyperref` to prepare paper documents.

8. Creating PDF

Now that we have discussed TEX related issues, let us turn now to questions involving PDF. For a complete discussion of issues relating to putting PDF on the WWW, see Thomas Merz's book *Web Publishing with Acrobat/PDF*, [7].

There are several methods for producing interactive PDF documents for the Internet when the source document is written in (La)TEX: The Adobe Acrobat Distiller, PDFTEX and one of several 'backend' drivers—`dvipdfm`, `dvipdf` and the commercial TEX System, VTEX, for example. Figure 2 illustrates the ways of creating a PDF document.

PDFTEX bypasses the `.dvi` and `.ps` stages of the process and writes the `.pdf` file directly from the TEX source; `dvipdfm`, for ex-

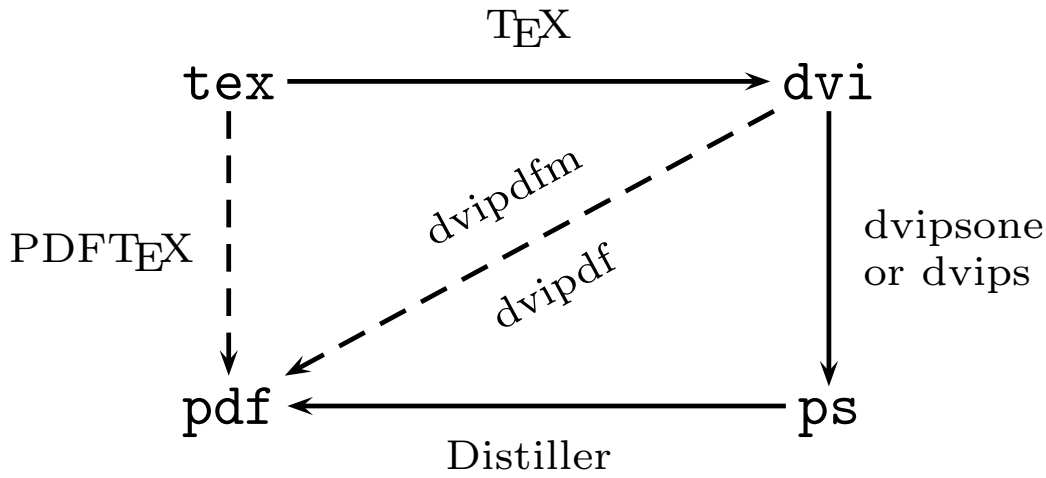

Figure 2: From TEX to PDF

ample, bypasses just the `.ps` stage, going from `.dvi` to `.pdf`. Both PDFTEX and `dvipdfm` come with the standard MikTEX distribution.

8.1. Distiller

The DISTILLER is a commercial product from the Adobe Corporation that comes with the Acrobat suite of software. Acrobat carries a hefty pricetag, but for educators, there is a significant discount. At this writing, someone in academics can buy Adobe Acrobat for around

$50.

- **Specifying Font Locations**

Open up the file menu

`Distiller > Font Locations...`

Use the '`Add Folder`' button to include any folders that contain the `postscript` fonts you normally use. Typically, on Wintel machines, `postscript` fonts installed by the Adobe Type Manager are located in the folder `C:\psfonts`.

- **Subsetting Fonts**

It is an important to subset all licensed fonts that you use to 99%. Start the DISTILLER and click on the menu item labeled `Distiller`, then click on `Job Options ...`. Now choose the `Font Embedding` tab.

Now check the `Subset Fonts below` `35` `%`, but change the default value of 35% to 99%. This line should like this:

☑ `Subset Fonts below` `99` `%`

menu. Now use the dropdown menu just below, locate the folder in which the Computer Modern Fonts or any postscript licensed fonts reside. Highlight all fonts you normally use, and move them using the double arrow button to the `Always Embed List:`. Click on the '`OK`' button and exit from the DISTILLER.

- **Distilling**

After you have LaTeXed your source document to create a `dvi` file, use `dvips` (or `dvipsone`) to create a postscript file, be sure to **print to a file**.

Now start the DISTILLER and open the postscript file you have just created. The DISTILLER goes to work transforming the postscript file into a file having the Portable Document Format (PDF).

There are a couple of ways you can speed up this process: by using *watched folders* or by using the DISTILLER ASSISTANT (Wintel machines).

You define your watched folders under the

`Distiller > Watched Folders...`

In this case, when a postscript file is left in one of the `Watched Folders`, the DISTILLER (if already started) detects the presence of the file and automatically distills it. You can also drag and drop your `ps` file into the watched folders.

This method is useful for processing a large number of files. For example, I use this method to update all the files in my e-Calculus tutorials. A `make` utility is used to detect any changes in the source file, those files are TEXed, then the `dvi-to-postscript` driver (DVIPSONE in my case) converts all these new `dvi` files to `ps` files; the output files are placed in a watched folder. The DISTILLER then converts all the `ps` files to `pdf` files.

The other method of conveniently creating a `pdf` file is to use the DISTILLER ASSISTANT

If you manage to get the DISTILLER ASSISTANT properly installed, you can print to a file `\distasst.ps` in the root directory. The ASSISTANT detects the presence of this file, starts the DISTILLER and feeds this file to the DISTILLER to be distilled into a PDF file.

- **Optimizing the Document**

When a postscript file is distilled, a PDF file is created. This file is optimized for printing. If you want to distribute your PDF over the WEB, you will want to *optimize the file for the* WEB. Acrobat Exchange, another program in the Acrobat suite, is used to do this.

Exchange has many, many capabilities that this article will not go into. One thing it does do is to optimize files. You can optimize individual file, or do a batch optimize.

Optimizing a Single File: Open the desired file in Exchange, then do a

`File > Save As...`

Before you click on `Save`, make sure the `Optimize` box is checked:

☑ `Optimize`

The result is a PDF file optimized for the WEB.

Batch Optimization: If you have a number of PDF files, all in the same folder to be optimized, use batch optimization. Start EXCHANGE and open the menu sequence:

`File > Batch Optimize...`

Choose the folder in which your PDF files reside, then click on OK.

Important: Beginning with Acrobat 4.0, the Distiller can now do the optimization. Start Distiller 4.0, and click on the drop down menu on the panel. Choose "ScreenOptimized". Now, when you distill, the resulting `.pdf` file will be ready for the Web, no need to open Acrobat Exchange, which is not just called Acrobat. You can still optimize and batch optimize within the Acrobat application, as described above.

- **Using Acrobat to Crop the Document**

Cropping allows you to trim off white space from around your page; it is optional, but is advantageous if you are designing a Web document meant to be read on screen.

Why crop your page? When the pages are trimed of excess white space, the natural magnification factor of the Reader will be much higher when the user sets the view to 'Fit Page' or 'Fit Width'. (*Note*: The default view when using `hyperref` is 'Fit Page'.) All this is

aimed at giving the one reading the document, a document that is confortable to read on the computer screen for long periods of time.

After you have settled on a page layout for your LaTeX document, at some point you need to determine your cropping parameters—the ones needed for the `hyperref` option `pdfpagescrop`.

`pdfpagecrop` takes four numbers are its value, how do you get these values? The easiest way is with Exchange (now called Acrobat in Acrobat 4.0). Open the document you want to crop in Exchange. Open the menu

`Document > Crop Pages...`

Use the control arrows to adjust the cropping boundary, as desired. After you are satisfied, click on the radio button `All`, to crop all pages in the document, then click on `OK`.

Next save your document using

`File > Save As...`

and exit Exchange/Acrobat.

Use a good text editor, one that is not bothered by binary data,

to open up the PDF file itself. Search for the word `/Cropbox`. For this document, it looks like this

```
/Cropbox [ 53 436 389 704 ]
```

The numbers inside the brackets are the four numbers you need in that order. Copy these numbers and put them into the options part of `hyperref`:

```
\usepackage
 [dvipsone,          % or dvips
  : : :
  pdfpagescrop={53 436 389 704},
  : : : ]{hyperref}
```

From this point on, your PDF document will be automatically cropped to these dimensions when the postscript file is distilled.

Note: The cropping is not permanent, no data are lost. You can take the cropped document and change the crop boundary from within EXCHANGE/Acrobat.

8.2. PDFTeX

PDFTEX is a modified TEX compiler written by Han The Thanh. The interesting thing about PDFTEX is that you can set it to produce a PDF file as its output file instead of the traditional `dvi` file. In this way, the process of producing a PDF file is sped up tremendously. The program is still in alpha testing.

PDFTEX comes with the standard distribution of MikTEX. I found PDFTEX very easy to set up and use along with the installation of MikTEX.

The use of PDFTEX is beyond the goals of these notes. The program comes with a manual (in PDF format). Suffice it to say, `hyperref` works with PDFTEX. Just specify the `pdftex` option for the `hyperref` package:

```
\usepackage
[pdftex,
 : : : :,
 : : : :,
```

```
: : : : ]{hyperref}
```

Destination Page

This is a page used to illustrate hypertext jumping. By the way, the section heading reads:

```
\section*{Destination Page}\label{s:destpage}
```

Let's have an equation

$$\boxed{x^2 + y^2 = z^2} \tag{2}$$

```
\begin{equation}
                \boxed{x^2 + y^2 = z^2}  \label{eq:xxyyzz}
\end{equation}
```

Let's plant a mark here.

```
Let's plant a mark \hypertarget{mymark}{here.}
```

References

[1] Leslie Lamport, *LaTeX: A Document Preparation* System (2nd ed.), Addison-Wesley Publishing Company, 1994, ISBN 0-201-52983-1. 10

[2] Michel Goossens, Frank Mittelbach and Alexander Samarin, *The LaTeX Companion*, Addison-Wesley Publishing Company, 1994, ISBN 0-201-54199-8. 10, 52, 52, 54

[3] Michel Goossens, Sebastian Rahtz, and Frank Mittelbach, *The LaTeX Graphics Companion*, Addison-Wesley Publishing Company, 1997, ISBN 0-201-85469-4. 10

[4] Goossens, Michel, and Rahtz, Sebastian, with Gurari, Eitan, Moore, Ross, and Sutor, Robert, *The LaTeX Web Companion*, Addison Wesley Longman, Reading, Massachusetts, USA, 1999. ISBN: 0-201-43311-7. 26

[5] Helmut Kopka and Patrick W. Daly, *A Guide to LaTeX2e* (2nd ed.), Addison-Wesley Publishing Company, 1995, ISBN 0-201-43777-X. 10

[6] Donald E. Knuth, *The TeXbook*, Addison-Wesley Publishing Company, 1987, ISBN 0-201-13448-9. 10, 52

[7] Thomas Merz, *Web Publishing with Acrobat/PDF*, Springer-Verlag Berlin Heidelberg New York, 1998, ISBN 3-540-63762-1. 54