Making Presentations with LaTeX and Prosper

by Rajarshi Guha, in Tutorials - Sat, Dec 28th 2002 00:00 PDT

A number of dedicated presentation programs have been written for Unix systems, but they may not serve your needs if you have special requirements, especially the need to display mathematical formulas. The Prosper package can help you create attractive presentations while letting you use the full power of LaTeX.

If you write a lot of technical documents, especially those containing formulas, you've probably used LaTeX. LaTeX is, basically, a set of macros for TeX. TeX, in turn, is a powerful typesetting system first developed by Donald Knuth. It has become an important tool for people who prefer to look at a document as series of logical units, leaving the actual presentation or layout to the software. LaTeX was developed by Leslie Lamport to aid in the writing of classes of documents such as journal articles, book chapters, and even letters. LaTeX abstracts many of the nitty gritty details of TeX, such as margin widths, line offsets, etc., allowing the user to simply decide on a document class and leave the style and format to the macros.

Numerous people have written macro packages that can be used with LaTeX. These packages provide an enormous range of functions, from formatting of citations to drawing Feynman diagrams. Together with features such as automatic index generation and bibliographies (using the BibTeX package), they provide the technical writer with an extremely powerful tool to create beautiful documents, concentrating on the logical flow, rather than having to worry about the underlying details of formatting and layout.

However, documents are not the only the things that need to be written; many times, a presentation must be made. Under Linux, tools such as KPresent and MagicPoint exist, and, of course, Windows users have MS PowerPoint. These are the traditional GUI tools. However, when you have to make a presentation containing formulas, they seem a little clunky, and you're stuck with whatever the package provides. Furthermore, if your documents are written using LaTeX, it would be nice if you could use those documents to generate slides for a presentation.

TeX and LaTeX being the all-powerful pieces of software they are, this is indeed possible. However, the problem with making presentations in LaTeX is the large number of packages available to do so. I've listed a few of the packages available, but there are quite a few more

which I haven't mentioned.

The *slides* class
: Part of the LaTeX distribution, it defines the page sizes, font sizes, etc. suitable for printing transparencies. Though the resultant DVI file can be converted to a PDF, there is no support for the various features of PDFs such as slide transitions and hyperlinks. Also, the package provides no defined slide styles (i.e., backgrounds, frames, etc.).

The Seminar package
: Developed by Timothy van Zandt, this is an extremely powerful set of macros with which you can develop presentations that take full advantage of the PostScript and PDF specifications. There are an extremely large number of options and commands available for this package, so the learning curve is a little steep.

The PDFLatex package
: This package is specifically designed for converting LaTeX source files to the PDF format without having to go through the intermediate DVI stage. Using this package along with the FoilTeX, pdfslide, and PPower4 packages allows you to generate presentations as well.

Prosper
: This is a set of macros which allows you to generate PostScript or PDF presentations. There are certain advantages of this package over the others. First, though it has a simple structure, it provides enough options to generate good-looking slides. All the features of a PDF document (such as transitions, overlays, etc.) are available. In addition, it is easy to generate different slide styles, a la PowerPoint. Of course, you still have access to the full power of TeX, so you are free to extend your documents if you have the knowhow. For LaTeX beginners, however, Prosper encapsulates a lot of the details in an easy-to-use manner.

In this article, I'll be discussing the Prosper package in some detail. You can find a good review of presentation tools for both PDF and HTML formats here.

Prosper

All LaTeX documents have a common basic structure. The first line always defines the document type -- article, letter, chapter, or, in this case, slides. After that comes the preamble. In the case of Prosper, this is where you specify the title slide. The next section is the document proper. When using Prosper, this is where you define the contents of successive slides. I'll cover the individual sections of a document written with Prosper in detail, but the first step is to install the package.

Installation:

As I mentioned above, the Prosper package provides a set of macros which define functional elements of a presentation -- the slides, how slides should transition, etc. To use the package, you will require the *seminar*, *pstricks*, and *hyperref* packages (which come with the standard TeX distribution on Red Hat). To generate the final output, you'll also need *dvips*, *GhostScript*, and *ps2pdf*. After downloading the tarball, extract it into a directory. To make use of the package and associated style files, you can place the required files (prosper.cls, the style file that you are using, and any associated images, such as for bullets) into the directory that contains your LaTeX document. However, a neater method is to put the Prosper directory into your TEXINPUTS environment variable:

```
~: export TEXINPUTS=~/src/tex/Prosper:$TEXINPUTS
```

(Where ~/src/tex/Prosper is the directory into which you extracted the Prosper files.) That completes your installation.

The *prosper* Document Class

To make a presentation using the Prosper package, you need to specify it in your \documentclass (you can also specify it in a \usepackage command in the preamble). Thus, the first line in the LaTeX file should be of the form:

```
\documentclass[ OPTIONS ]{prosper}
```

There are several options that can be specified to the package. You can read about all the options in detail in the documentation that comes with Prosper. I'll just give a brief overview of some of the common and useful ones:

draft
: Compiles a draft version of the presentation, with figures replaced by bounding boxes.

final
: Compiles a complete version of the presentation with figures and captions in their proper places.

ps
: Compiles the LaTeX file to PostScript for printing purposes.

pdf
: Compile the LaTeX file to a PDF format suitable for projectors.

Another important option to specify is which presentation style to use. Prosper comes with several styles, and new styles can easily be made with a little knowledge of the *pstricks* package.

There are also options to specify slide background colors, slide numbers, etc. In general, unless you require black and white slides (e.g., for printing purposes), you won't need to set any color options in the \documentclass; the style files will manage them for you.

The Preamble

The next section is the preamble, the part between \documentclass and \begin{document}. In this section, you should specify the contents of the title page and some options (such as logos and slide captions) that can be applied to all the slides. The normal LaTeX macros have been redefined to generate the title and associated text with proper font sizes, etc. Some of the macros available for designing the title slide include:

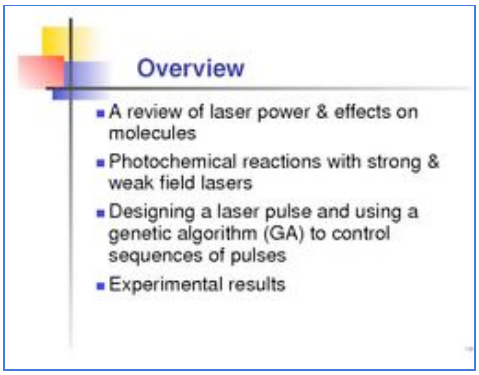
- \title
- \subtitle
- \author
- \email
- \slideCaption (You can use this macro to put a caption at the bottom of each slide.)
- \Logo (This allows you to place a logo on each slide at a specified position.)
- \DefaultTransition (This defines the type of transition that should occur between slides.)

Since the *hyperref* package is included by Prosper, you can use the \href command to include mailto: links or direct hyperlinks to Web pages in the above commands (and, of course, in the rest of your document). As in standard LaTeX, the title slide is generated by the \maketitle command in the document body.

The *slide* Environment

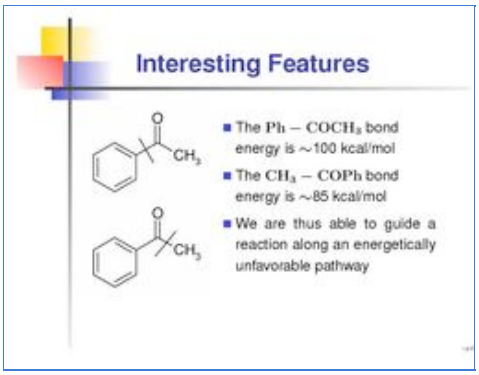
The Prosper package defines the *slide* environment. This represents the basic unit of a presentation (a single slide) and is placed in the document body (i.e., after the \begin{document} command). Within a slide environment, all the usual LaTeX commands may be used. Images, formulas, tables, footnotes, page structure commands, etc. can all be used. The Prosper package does redefine the *itemize* environment so that the text is no longer justified. It also supplies images for the bullets. Thus, a single slide containing a bulleted list can be represented by the following LaTeX source (alongside, you can see how the final PDF output for this slide would look):

```
\begin{slide}{The Title of the Slide}
\begin{itemize}
\item Item 1
\item Item 2
\item Item 3
\end{itemize}
\end{slide}
```



The environment does not provide any means to divide the slide area into columns or rows; it simply provides a rectangular display area (the dimensions of which may vary from style to style). However, using the *minipage* environment, it is very easy to make a two-column slide. For example, the following would create a slide with a picture in one column and a bulleted list in the other:

```
\begin{slide}{Another Example Slide}
\begin{minipage}{4cm}
\epsfig{file=./picture.eps}
\end{minpage}
\begin{minipage}{7cm}
\begin{itemize}
\item Item 1
\item Item 2
\item Item 3
\end{itemize}
\end{minipage}
\end{slide}
```



Prosper also defines some commands which are allowed to appear in a *slide* environment. Examples include:

\FontTitle
: Defines the font to be used in the slide title

\FontText
: Defines the font to be used in the slide text

\fontTitle
: Writes its argument as the slide title

\fontText
: Writes its argument as the slide text

In general, the above macros are not used when writing a presentation. They are, however, useful when you create slide styles of your own.

Page Transitions

An important command is \PDFtransition, which can be used to specify how the current slide should appear. However, the usual way to specify a slide transition for a specific slide is to put the transition mode into the \begin{slide} command as:

```
\begin{slide}[Glitter]{Slide Title}
```

The Prosper package supports several types of transitions:

- Split
- Blinds
- Box
- Wipe
- Dissolve
- Glitter
- Replace (the default)

The above transition modes provide you with ample opportunity to make flashy presentations (if that's what you're into :). You can see a PDF which displays each of the transitions here.

Overlays

A very useful feature of computer-based presentations is the ability to make overlay slides so parts of the same slide will appear at different times. Prosper provides commands to implement this in a very simple fashion. The \overlay command is used to specify that a given \slide environment will consist of a sequence of overlays. You must specify the number of overlays that make up the slide. There are several commands that can be used to specify exactly what material should appear on which slide within an overlay:

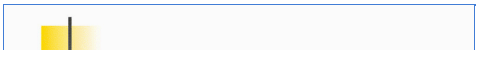
\fromSlide{p}{*material*}
: Puts *material* on slides p to the end of the overlay.

\onlySlide{p}{*material*}
: Puts *material* only on slide p.

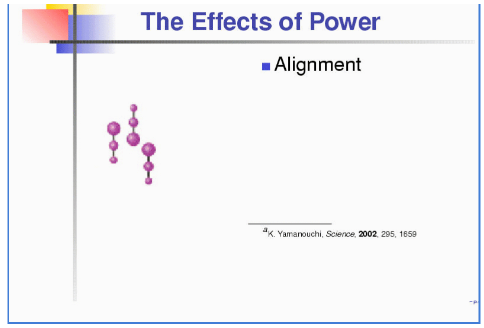
\untilSlide{p}{*material*}
: Puts *material* on all slides from the first to the p^{th}.

There are three macros analogous to the above (obtained by capitalizing the first letter) which cause all material after the occurrence of the macro to be included (rather than specifically defining *material*). The macros in the above list also have starred counterparts (i.e., \fromSlide*, etc.). These versions are useful when the successive overlays need to replace previous overlays. Below, I've provided an example of a slide that consists of several overlays and uses the *itemstep* environment to allow an itemized list to progress through successive overlays. Alongside is an animation of how the PDF version of the slide would look:

```
\overlays{5}{
\begin{slide}{The Effects of Power}
\begin{tabular}{rc}
\begin{minipage}{4cm}
\onlySlide*{1}{\epsfig{file=stage1.eps}}
\onlySlide*{2}{\epsfig{file=./stage2.eps}}
```



```
\onlySlide*{3}{\epsfig{file=./stage3.eps}}
\onlySlide*{4}{\epsfig{file=./stage4.eps}}
\onlySlide*{5}{\epsfig{file=./stage5.eps}}
\end{minipage} &
\begin{minipage}{6cm}
\begin{itemstep}
\item Alignment
\item Deformation
\item Coulomb explosion
\item X-ray emission
\item Nuclear reaction
\end{itemstep}
\end{minipage}
\end{tabular}
\end{slide}}
```



An important point to note about the overlay commands is that they are only valid when the Prosper package is used with the *pdf* option. However, the package does provide a set of macros:

- \PDForPS{*ifpdf*}{*ifps*}
- \onlyInPS{*material*}
- \onlyInPDF{*material*}

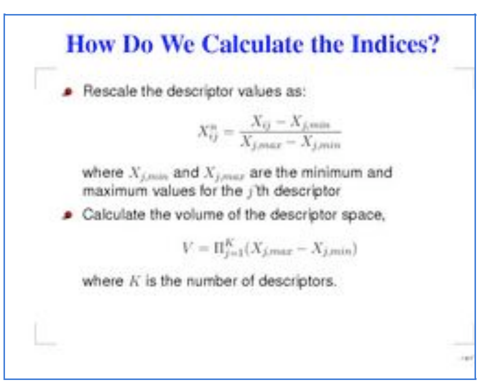
which allow you to include different material depending on whether the LaTeX document is compiled in PS or PDF mode. An example of the use of these macros would be:

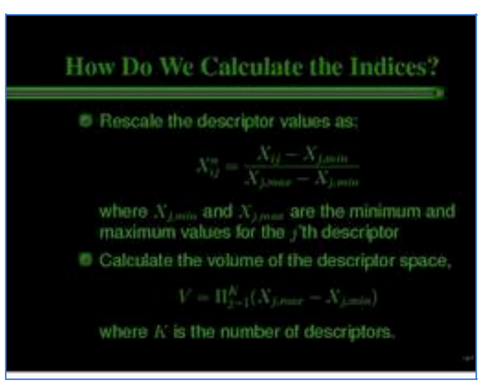
```
\overlays{3}{
\begin{slide}{An Example Slide}
\onlySlide*{1}{\epsfig=./pic1.eps}
\onlySlide*{2}{\epsfig=./pic2.eps}
\onlySlide*{3}{\epsfig=./pic3.eps}
\onlyInPS{\epsfig=./epspic.eps}
\end{slide}}
```

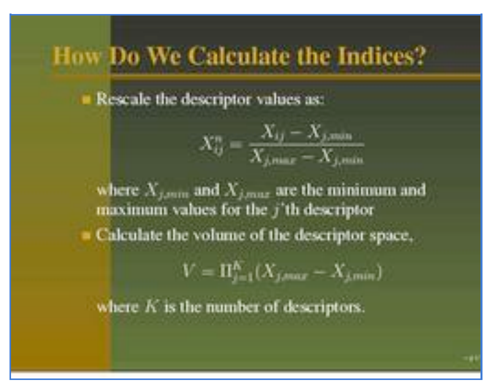
If the snippet were converted to a PDF, we would get a slide which would successively display pic1.eps, pic2.eps, and pic3.eps. If it were compiled to PS format, the slide would only contain the image epspic.eps.

Presentation Styles

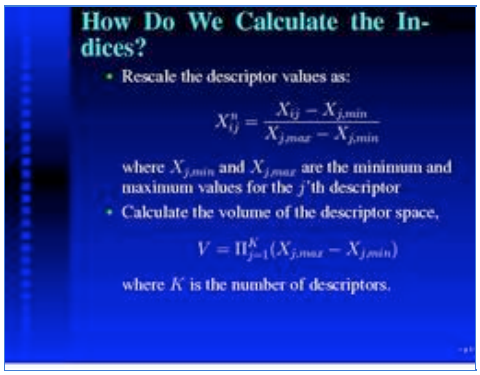
The Prosper package comes with several style files. Essentially, these provide predefined background colors and patterns, title fonts, bullet styles, etc. You can easily change the look of your presentation by including a different style file. Which style to use is specified in the \documentclass. Below, you can see slides generated using the different slide styles.



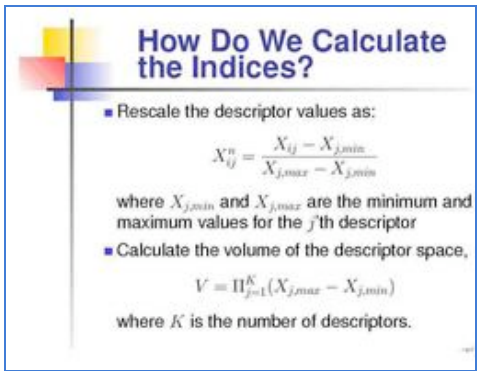




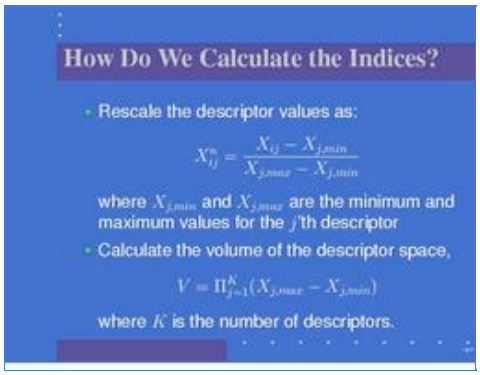
Default



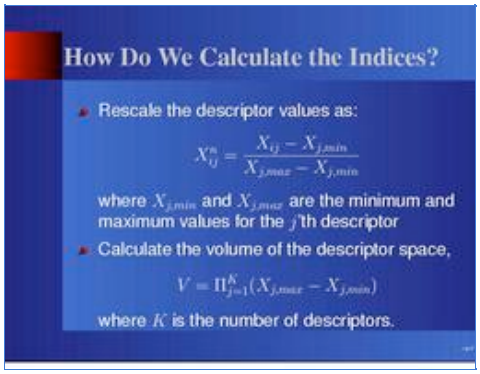
Alienglow



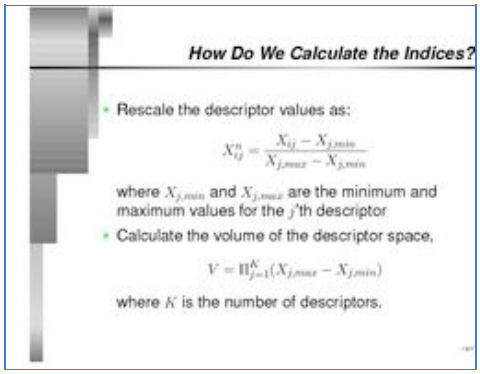
Autumn



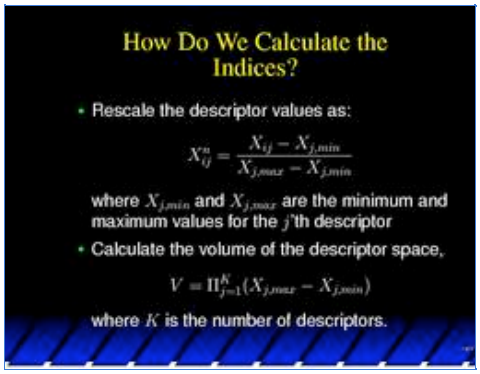
Azure



Blends



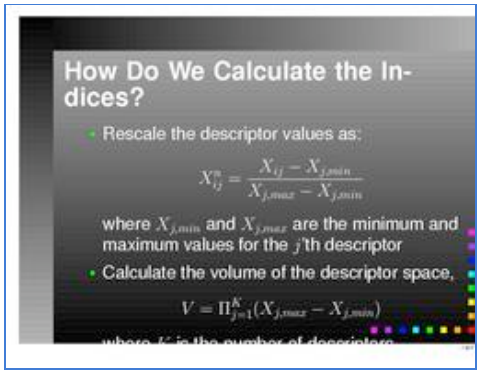
Contemporain



Dark Blue

Frames

Lignes Bleues



Nuance trois

It should be noted that all the styles do not provide the same display area for the actual slide material. You can see this in some of the slide examples above. If you decide to change the slide style of your presentation, you might need to tweak things such as spacing (\hspace, \vspace, etc.) or line lengths, etc. Furthermore, if a given style does not really suit your taste, it is possible to make modifications such as font type, colors, etc. using the Prosper macros, rather than digging into the source of the style in question.

Assuming you're comfortable with the *pstricks* package, designing a new slide is made easier by a number of macros defined by Prosper. You have access to a number of boolean macros which allow you to include features depending on the current environment (PDF or PS, color or black & white, etc.). The main macro that Prosper provides to design a new style is the \NewSlideStyle command. After designing the style, you need to tell Prosper the details, such as how much display area you are providing, where it should be located, etc., using this macro.

Processing the LaTeX File

At this point, you should be able to write your presentation. The last step is to convert the LaTeX source to a PDF file. The steps involved are pretty simple:

1. latex file.tex
2. dvips -Ppdf -G0 file.dvi -o file.ps
3. ps2pdf -dPDFsettings=/prepress file.ps file.pdf

Two points to note:

- The -G0 parameter passed to dvips is used to get around a bug in GhostScript which converts the "f" character to a pound sign in the final PDF.
- The -dPDFsettings parameter for ps2pdf is used to prevent downsampling of EPS images when they are converted to PDF. Without this switch, EPS graphics in the final PDF look very fuzzy, especially when viewed with a projector.

Miscellaneous Features

- Since Prosper includes the *hyperref* package by default, you can easily set links and targets within your presentation with the \hyperlink and \hypertarget commands to enable easy navigation.
- PowerPoint allows you to embed animations within a presentation. This is also possible when using Prosper, since it uses the *hyperref* package. To embed an MPEG movie, you can include the following code snippet:

  ```
  \href{run:movie.mpg}{Click here to view the movie}
  ```

  Two points to note:

  - Viewing the movie depends on Acrobat Reader being able to run the viewing program. This can be set by making sure you have an entry in your .mailcap file for the filetype you want to play.
  - The resultant movie plays in its own window; it is not possible to actually "embed" the movie in the presentation itself (at least under Linux).

  Using this technique, you could run any type of file (assuming you have a program to handle it) or even executables like shell scripts, etc.

- You may want to convert your PDF presentation to an HTML slideshow. This is possible using the program pdf2htmlpres.py. It can use the `convert` program from the ImageMagick suite or GhostScript directly to convert the PDF slides to a series of JPGs (or GIFs or PNGs) and generate HTML pages to form a slideshow.

Conclusion

I hope I've been able to convey some of the features and benefits that the Prosper package provides. Granted, for a person who doesn't use LaTeX, a GUI alternative would be easier. But for all the TeXnicians out there, the Prosper package allows you to generate well-designed and stylish slides efficiently, at the same time allowing the knowledgeable user to extend the package using predefined macros and pure TeX.

The Prosper community has a very useful mailing list which can be accessed at the Prosper Web site. The Prosper tarball contains comprehensive documentation explaining the available commands and macros provided by the package. It also includes a document displaying the capabilities of the package. The LaTeX sources of these documents are the best way to learn how to use the various features of Prosper.

I, for one, have finally been able to get rid of MS PowerPoint and use Prosper to develop all my presentations. Using this package, I'm able to create presentations which rival those produced by more popular GUI packages and which can be viewed with the very common Acrobat Reader (and converted to *clean* HTML when required!). You can take a look at presentations I've made using prosper on my Web site

For all LaTeX users, I strongly recommend taking a look at Prosper.

Author's bio:

Rajarshi Guha is a grad student working on computer-aided chemistry. When not studying and porting ancient (and twisted) F77 code, he fiddles around with Python programming and the Beowulf in his lab, and daydreams about marrying his fiancee.

T-Shirts and Fame!

We're eager to find people interested in writing articles on software-related topics. We're flexible on length, style, and topic, so long as you know what you're talking about and back up your opinions with facts. Anyone who writes an article gets a t-shirt from ThinkGeek in addition to 15 minutes of fame. If you think you'd like to try your hand at it, let jeff.covey@freshmeat.net know what you'd like to write about.

[add comment]

Referenced categories

Topic :: Text Processing :: Markup :: TeX/LaTeX

Referenced projects

ImageMagick - A comprehensive package supporting automated and interative manipulation of imag
KOffice - An integrated office suite for KDE.
LaTeX - A high-quality typesetting system based on TeX.
MagicPoint - A presentation tool.
PPower4 - PDF Presentation Post Processor.
Prosper - A LaTeX class for writing transparencies.
teTeX - A TeX distribution for Unix.

Comments

[»] Slide clipping in acroread when using prosper
by G. A. Gallup - Nov 19th 2004 08:21:46

I am very pleased with the way prosper works and plan to use it soon for the first time.

I did find that acroread clipped the right sides of the slide (in all modes) unless ps2pdf was used

with the option

ps2pdf -sPAPERSIZE=a4 <file>.ps <file>.pdf

I am using the standard texmf distribution with RH 9.1.

Regards,

[reply] [top]

[»] making bitmap graphics in PDF look good
by Holger Jakobs - Jan 22nd 2003 03:46:25

When converting the .ps output of dvips into .pdf using ps2pdf (thus ghostscript), all included bitmap graphics looked ugly, full of artefacts.

Even the hint "-dPDFsettings=/prepress" did not help. Funny enough, using ghostscript 5.50 produced good results, but lacking the transition effects between slides.

After a long search, I found that the options
-dAutoFilterColorImages=false -dColorImageFilter=/FlateEncode
together produced the intended result of good-looking bitmaps (which had to be converted into eps using ImageMagick's convert before). The .pdf files don't get too big.

Hope this helps. From now on, prosper is my choice in producing presentations! Thanks for this great LaTeX package!

--
-- Holger Jakobs, Bergisch Gladbach, Germany

[reply] [top]

[»] no font resize anymore
by pts - Dec 28th 2002 08:58:20

I use a self-hacked version of pdfscreen.sty with pdfLaTeX. It works great for me. Stability, consistent font sizes and colors, math formulas, scalable slides, portable PDF output. That's what I need and what I get. I know that its capability is limited when dealing with interactivity and animating slides, but for my presentations I don't need such features.

And LaTeX doesn't resize the characters on the whole slide if I type a new \item into a list.

[reply] [top]

[»] pdfscreen.sty does it as well
by iztok - Dec 28th 2002 08:45:41

Hi!

I am using pdfscreen.sty by C.V.Radhakrisnan
(found on usuall latex sites).

StarOffice was used for a while, but it is WYSIonlyWYG,
as bloated as Wintendo M$office. Two pdf files are generated, one for printer, one for screen.

Even Wintendo PCs, normally used on conferences, have acrobat reader installed. I am usually the only not using
powerpoint:, and I never had any problems with fonts,
colors etc usually seen there. Works great.

BR

Iztok

[reply] [top]

[»] Install for Debian users
by Alexandre Dulaunoy - Dec 28th 2002 07:04:30

apt-get install prosper

--
--- Alexandre Dulaunoy http://www.foo.be/

[reply] [top]

[»] other LaTeX slide tool: advi
by Basile Starynkevitch - Dec 28th 2002 06:30:55

There is also another interesting LaTeX tool: **advi** (or Active-DVI). See here for details. Of course, **advi** is open-source, and it is written in Ocaml, a very good functional programming language (with some object orientations abilities). The interesting features of **advi** are animations, colors, external fancy programs (your slide could include an xterm!). etc... As its name suggest, **advi** is an extension of dvi, with interesting LaTeX styles. (Advi requires LaTeX).

Another tool to make slides is the **Lout** formatter. See here for details. **Lout** is a simple text formatter (much smaller than LaTeX) and offer some slides style. But the result is just a printable PostScript file, without any interactivity.

--
Basile STARYNKEVITCH

[reply] [top]

[»] Thanks
by boris - Dec 28th 2002 03:30:49

My PHd supervisor showed me this the other day when I mentioned I was giving a presentation on my work to the other research students. This isn't a big thing but I needed a means of displaying slides, I was going to write my own system - I am not a great fan of existing packages, I dont have the funds to get OfficeX and it seemed the easiest solution.

Anyway, I put it to one side [by that I mean I book marked it, and left it and forgot]. But I am very impressed with the article and I am definatly going to go play with it. Just need to get TeX and friends installed again on my iBook to play.

Thanks for a good article!

--
Chris Ross - boris [chris at darkrock.co.uk, http://www.darkrock.co.uk] [http://www.ferite.org]

[reply] [top]

[»] yeeh..
by Roman Joost - Dec 28th 2002 01:34:50

good work!! Iam using LaTeX for one year now and its really amazing. I used the OpenOffice Powerpoint for presentations and LaTeX for my text documents. I never recognize, that slide handling is so easy with LaTeX. Allright, for now Powerpoint is the past ;)

[reply] [top]

[»] err... powerpoint?
by waldemar - Jan 4th 2004 15:36:17

```
> good work!! Iam using LaTeX for one year
> now and its really amazing. I used the
> OpenOffice Powerpoint for presentations
> and LaTeX for my text documents. I never
> recognize, that slide handling is so
> easy with LaTeX. Allright, for now
> Powerpoint is the past ;)
```

i think what you're saying here is impress not powerpoint.
by the way, good article. although i'm still in the newbie stage when it comes to latex.

[reply] [top]
