The pstool package

Will Robertson and Zebb Prime

v0.3 2008/08/06

Contents

Part I
Documentation

1 Introduction

While pdfLaTeX is a great improvement in many ways over the 'old method' of DVI→PS→PDF, it loses the ability to interface with a generic PostScript workflow, used to great effect in numerous packages, most notably PSTricks and psfrag.

Until now, the best way to use these packages while running pdfLaTeX has been to use the pst-pdf package, which processes the entire document through a filter, sending each relevant PostScript environment through DVI→PS→PDF. The resulting PDF versions of each image are then included into the pdfLaTeX document. The auto-pst-pdf package provided a wrapper to execute this separate process automatically.

The disadvantage in this route is that for every document compilation, *every* graphic must be re-processed. The pstool package uses a different approach to allow each graphic to be processed only as-needed, speeding up and simplifying the typesetting of the main document.

2 Processing modes

The main command provided by this package is

`\psfig[`⟨*graphicx options*⟩`]{`⟨*filename*⟩`}{`⟨*graphic data*⟩`}`.

By default it can be used in the following modes:

`\psfig` Process the graphic ⟨*filename*⟩ if no PDF of the same name exists, or the source EPS file is *newer* than the PDF;

`\psfig*` Always process this figure; and,

`\psfig!` Never process this figure.

The following package options override the above: `[process=all]`, `[process=none]` (the default is `[process=auto]`).

3 Cropping graphics

Graphics are cropped to the appropriate size with the preview package. Sometimes, however, this will not be good enough when an inserted label protrudes from the natural bounding box of the figure. A good way to solve this problem is to use the pdfcrop program (requires a Perl installation under Windows). This can be activated in pstool with the `[pdfcrop]` package option.

4 Todo

1. Use `dir /h & call echo ^%ERRORLEVEL^%`.
2. Higher commands (`\psfragfig`, `\matlabfig`, `\mathfig`)
3. Generalise "olding" code for multiple files.
4. Basic EPS→PDF processing (no need to read in the document preamble).
5. Check for correct behaviour in shells other than bash.

6. More flexible usage (support things like \begin{postscript} in pst-pdf).
7. mylatex integration

Part II
Implementation

```
\ProvidesPackage{%
      pstool}[2008/08/06 v0.3 Wrapper for processing PostScript/psfrag figures]
```

External packages

```
\RequirePackage{%
      catchfile,color,ifpdf,ifplatform,inversepath,graphicx,suffix,xkeyval}
```

Initialisations

```
\if@pstool@always@
 \if@pstool@never@
\if@pstool@pdfcrop@
        \pstool@out
\newif\if@pstool@always@
\newif\if@pstool@never@
\newif\if@pstool@pdfcrop@
\newwrite\pstool@out
```

Package options

```
      pdfcrop
\DeclareOptionX{pdfcrop}{\@pstool@pdfcrop@true}

      process
\define@choicekey*{pstool.sty}{process}[\@tempa\@tempb]{%
      all,none,auto}{%
  \ifcase\@tempb\relax
    \@pstool@always@true
  \or
    \@pstool@never@true
  \or
  \fi}

\ifshellescape\else
  \ExecuteOptionsX{process=none}
  \PackageWarning{pstool}{^^J\space\space%
    Package option [process=none] activated^^J\space\space
    because -shell-escape is not enabled.^^J%
    This warning occurred}
\fi

\ProcessOptionsX
```

These are cute:

\OnlyIfFileExists
\NotIfFileExists

```
\providecommand\OnlyIfFileExists[2]{\IfFileExists{#1}{#2}{}}
\providecommand\NotIfFileExists[2]{\IfFileExists{#1}{}{#2}}
```

Command line abstractions between platforms:

```
\edef\pstool@cmdsep{\ifwindows\string&\else\string;\fi\space}
\edef\pstool@rm{\ifwindows␣del␣\else␣rm␣--␣\fi}
```

\pstool@try@rm

```
\newcommand\pstool@try@rm[1]{%
  \begingroup
    \@for\@tempa:=#1\do{%
      \OnlyIfFileExists{\ip@directpath\@tempa}{%
        \immediate\write18{%
          cd␣"\ip@directpath"\pstool@cmdsep
          \pstool@rm␣"\@tempa"}}}%
  \endgroup}
```

Generic function to execute a command on the shell and pass its exit status back into LaTeX. Any number of `\pstool@exe` statements can be made consecutively followed by `\pstool@endprocess`, which also takes an argument. If *any* of the shell calls failed, then the execution immediately skips to the end and expands `\pstool@error` instead of the argument to `\pstool@endprocess`.

\pstool@exe

```
\def\pstool@exe#1{%
  \immediate\write18{%
    cd␣"\ip@directpath"\pstool@cmdsep
    \pstool@writestatus{#1}%
  }%
  \pstool@retrievestatus\@tempa
  \ifnum\@tempa␣>␣0
    \PackageWarning{pstool}{%
         Execution␣failed␣during␣process:^^J␣␣#1^^J}%
    \expandafter\pstool@abort
  \fi}
```

Edit this definition to print something else when graphic processing fails.

\pstool@error

```
\def\pstool@error#1{\fbox{\color{red}%
      \ttfamily␣An␣error␣occured␣processing␣this␣graphic.}}
```

\pstool@abort

```
\def\pstool@abort#1\pstool@endprocess{\pstool@error}
\let\pstool@endprocess\@firstofone
```

It is necessary while executing commands on the shell to write the exit status to a temporary file to test for failures in processing. `#1 & echo %ERRORLEVEL%`

doesn't return the correct value inside a `\write18` in Windows, so we have to do something different there.

`\pstool@statusfile`
`\pstool@writestatus`

```
\def\pstool@statusfile{status-deleteme.txt}
\def\pstool@writestatus#1{%
  \ifwindows
    echo␣0␣>␣\pstool@statusfile\pstool@cmdsep
    #1␣\detokenize{||}␣echo␣1␣>␣\pstool@statusfile
  \else
    #1\pstool@cmdsep␣echo␣$?␣>␣\pstool@statusfile
  \fi
}
```

Read the exit status from the temporary file and delete it.

`\pstool@retrievestatus`

```
\def\pstool@retrievestatus#1{%
  \pstool@touchstatus
  \CatchFileEdef{#1}{\ip@directpath\pstool@statusfile}{}%
  \pstool@try@rm{\pstool@statusfile}%
}
```

This ensures the file is written to disk properly (allowing it to be read by `\CatchFileEdef`). Not necessary on Windows, whose file writing is evidently more crude/immediate.

`\pstool@touchstatus`

```
\def\pstool@touchstatus{%
  \ifwindows\else
    \immediate\write18{%
      cd␣"\ip@directpath"\pstool@cmdsep
      touch␣\pstool@statusfile}%
  \fi
}
```

4.1 File age detection

Use `ls` (or `dir`) to detect if the EPS is newer than the PDF:

`\pstool@datefiles`

```
\def\pstool@datefiles{%
  \edef\pstool@filenames{\ip@lastelement.eps\space␣%
       \ip@lastelement.pdf\space}%
  \immediate\write18{%
    cd␣"\ip@directpath"\pstool@cmdsep
    \ifwindows
      dir␣/T:W␣/B␣/O-D␣"\ip@lastelement.eps"␣"%
           \ip@lastelement.pdf"␣>␣\pstool@statusfile
    \else
```

```
      ls␣-t␣"\ip@lastelement.eps"␣"\ip@lastelement.pdf"␣>␣%
           \pstool@statusfile
    \fi
  }%
  \pstool@retrievestatus\@tempa
  \ifx\@tempa\pstool@filenames
    \@tempswatrue
  \else
    \@tempswafalse
  \fi
}
```

5 Command parsing

User input is \psfig (with optional * or ! suffix) which turns into one of the following three macros depending on the mode.

\pstool@alwaysprocess

```
\newcommand\pstool@alwaysprocess[3][]{%
  \inversepath*{#2}% calculate filename, path & inverse path
  \pstool@process[#1]{#2}{#3}}
```

\pstool@neverprocess

```
\newcommand\pstool@neverprocess[3][]{%
  \includegraphics[#1]{#2}}
```

For regular operation, which processes the figure only if the command is starred, or the PDF doesn't exist.

\pstool@maybeprocess

```
\newcommand\pstool@maybeprocess[3][]{%
  \inversepath*{#2}% calculate filename, path & inverse path
  \IfFileExists{#2.pdf}{%
    \pstool@datefiles
    \if@tempswa\expandafter\@firstoftwo
      \else\expandafter\@secondoftwo\fi{%
        \pstool@process[#1]{#2}{#3}%
      }{%
        \includegraphics[#1]{#2}}%
  }{%
    \pstool@process[#1]{#2}{#3}%
  }}
```

5.1 User commands

Finally, define \psfig as appropriate for the mode:

```
\ifpdf
```

```
    \if@pstool@always@
      \let\psfig\pstool@alwaysprocess
      \WithSuffix\def\psfig!{\pstool@alwaysprocess}
      \WithSuffix\def\psfig*{\pstool@alwaysprocess}
    \else\if@pstool@never@
      \let\psfig\pstool@neverprocess
      \WithSuffix\def\psfig!{\pstool@neverprocess}
      \WithSuffix\def\psfig*{\pstool@neverprocess}
    \else
      \let\psfig\pstool@maybeprocess
      \WithSuffix\def\psfig!{\pstool@neverprocess}
      \WithSuffix\def\psfig*{\pstool@alwaysprocess}
    \fi\fi
  \else
    \let\psfig\pstool@neverprocess
    \WithSuffix\def\psfig!{\pstool@neverprocess}
    \WithSuffix\def\psfig*{\pstool@neverprocess}
  \fi
```

6 The figure processing

\pstool@process

```
  \newcommand{\pstool@process}[3][]{%
    \pstool@write@processfile{#1}{#2}{#3}%
    \pstool@exe{latex␣-shell-escape␣-output-format=dvi
        -interaction=batchmode␣"\ip@lastelement-process.tex"}%
    \pstool@exe{dvips␣"\ip@lastelement-process.dvi"}%
    \if@pstool@pdfcrop@
      \pstool@exe{ps2pdf␣"\ip@lastelement-process.ps"␣"%
            \ip@lastelement-process.pdf"}%
      \pstool@exe{pdfcrop␣"\ip@lastelement-process.pdf"␣"%
            \ip@lastelement.pdf"}%
    \else
      \pstool@exe{ps2pdf␣"\ip@lastelement-process.ps"␣"%
            \ip@lastelement.pdf"}%
    \fi
    \pstool@endprocess{\includegraphics[#1]{#2}}}
```

The file that is written for processing is set up to read the preamble of the original document and set the graphic on an empty page (cropping to size is done either here with preview or later with pdfcrop).

\pstool@write@processfile

```
\def\pstool@write@processfile#1#2#3{%
    \immediate\openout\pstool@out␣#2-process.tex\relax
    \immediate\write\pstool@out{%
      \unexpanded{%
```

```
    \pdfoutput=0% force DVI mode if not already
```

Input the main document; redefine the document environment so only the preamble is read:

```
    \let\origdocument\document
\document  \def\document{\endgroup\endinput}%
  }\noexpand
    \input{\ip@inversepath\jobname}%
```

Now the preamble of the process file: (restoring document's original meaning)

```
  \if@pstool@pdfcrop@\else
    \noexpand\usepackage[active,tightpage]{preview}
  \fi
  \unexpanded{%
    \let\document\origdocument
    \pagestyle{empty}% remove the page number
    \begin{document}
```

And the document body to place the graphic on a page of its own:

```
    \centering\null\vfill}%
  \if@pstool@pdfcrop@\else
    \noexpand\begin{preview}%
  \fi
    \unexpanded{#3}% this is the "psfrag" material
    \noexpand\includegraphics[#1]{\ip@lastelement}%
  \if@pstool@pdfcrop@\else
    \noexpand\end{preview}%
  \fi
  \unexpanded{%
    \vfill\end{document}}%
  }%
\immediate\closeout\pstool@out}
```

⟨*eof*⟩