

background: graphics

Tools for PostScript and pdf

Siep Kroonenberg
Kluwer, Dordrecht
siepo@cybercomm.nl

abstract

This paper explains why PostScript is interesting for TeX users, and describes various tools for working with PostScript, with special attention to Ghostscript. The paper concludes with a section on pdf, the derivative of PostScript which is destined to take over much of the role of PostScript in the prepress workflow.

keywords

PostScript, DSC comments, pdf, Ghostscript, graphics, conversions, eps inclusion, bounding box

At every turn, TeX users run into PostScript, the lynchpin of the graphics industry. Imagesetters and high-end desktop printers understand PostScript; professional illustration software is geared towards generating PostScript. On Unix platforms PostScript tends to be the only adequately supported printer language.

The PostScript imaging model

PostScript is both a graphics format and a printer language. Its primary purpose is to describe objects on a page. A low-level printer might only understand descriptions in terms of printer dots; PostScript understands all of the following:

- a variety of color models
- paths, which may consist of curved and straight segments; the curved segments are described by either arcs or Bézier curves.
- several font formats; in the first place its own scalable Type1 font format, in which characters are defined by outlines
- bitmaps of arbitrary resolution in various color models; the PostScript interpreter is capable of resampling and rasterizing the bitmap to something matching the printer hardware
- clipping paths: objects will be visible only insofar as they lie within the current clipping path.
- translating, scaling and rotating (affine transformations): it is possible to change the coordinate system on the fly. These transformations can also be applied to fonts.

PostScript-based illustration software

All these high-level features make it possible for graphics programs to adopt (a subset of) PostScript suitable as their native file format. Adobe Illustrator is the best-known of such programs, but all professional illustration programs (CorelDRAW, Macromedia FreeHand, Micrographix Designer and -Draw) are based on something close to the PostScript imaging model, and support Illustrator format almost as thoroughly as their own native format. Also worth mentioning in this context is Mayura Draw [11], a small inexpensive shareware PostScript-based Windows draw program.

Although traditionally Unix programs tended to support splines rather than Bézier curves, various draw programs for Linux/Unix (Sketch [12]; see figure 2, Killustrator [13], GYVE [14]) are now under development which do support Bézier curves. Of these, Sketch has Illustrator import- and export filters. Pstoedit [18], a program that attempts to convert PostScript to editable formats (see below), has export filters for Sketch and Killustrator since version 3.13.

Most office- and consumer-oriented Windows graphics programs ignore the special capabilities of PostScript: all printing and most graphics export seem to be mediated by the Windows GDI (Graphics Device Interface), which is rather low-level. If you manage to get such low-level PostScript code converted to an editable format, you'll see tiny line fragments rather than Bézier curves; areas hatched with lines or thin slices; words chopped up into single characters. With luck the PostScript code may print, but you won't be able to do much meaningful editing.

PostScript as a programming language

PostScript is a complete programming language, with variables, conditionals and procedures. It is stack-based: a line '1 2 add' puts first 1, then 2 on the stack, and finally removes those two and puts their sum 3 on the stack; see figure 1. PostScript-generating applications normally start PostScript files with a fixed collection of definitions, followed by some setup, followed by the actual page descriptions, which use the earlier definitions, and end with some cleanup. See [2] for a nice, although somewhat outdated introduction to PostScript programming.

Postprocessing and DSC comments

PostScript has been designed as both an input- and an output format: a program may generate PostScript to be sent to

```
/home/siepo/gs_ntg > gs
Aladdin Ghostscript 5.50 (1998-9-16)
Copyright (C) 1998 Aladdin Enterprises, Menlo P
This software comes with NO WARRANTY: see the f
GS>1
GS<1>2
GS<2>pstack
2
1
GS<2>add
GS<1>pstack
3
GS<1>
```

Figure 1 An interactive Ghostscript session. The pstack command prints the operand stack to the terminal.

a printer, but the code may also be stored in a file for subsequent postprocessing by another program. One example is incorporating a PostScript graphic into another document; another example is the psutils suite [10] which offers n-up printing and page imposition for PostScript files, i.e. arranging pages for printing in signatures.

The key to such postprocessing is use of DSC (Document Structuring Conventions) [5] comments. These comments in a PostScript file should give a postprocessing application the information it needs for doing its job without a need for parsing the actual PostScript code.

DSC comments can e.g. contain lists of required and included fonts and other resources or the number of pages, or they may mark beginning and end of sections such as prolog, setup, ‘ProcSets’ and font definitions.

EPS inclusion and the BoundingBox comment

Especially important for users is the boundingbox DSC comment, e.g.

```
%%BoundingBox: 12 32 400 300
```

This tells a text-processing application the dimensions of a PostScript graphic, so that it can incorporate the graphic in its own PostScript code. Above, we already saw that PostScript offers the means to position, scale and rotate the included graphic.

Such nesting only works if the PostScript file to be included satisfies some additional restrictions: it should contain only one page and it should not make certain global changes to its environment. See [6] for details. An eps- or Encapsulated PostScript file is a PostScript file satisfying these restrictions.

An eps file may also contain a screen preview. This is for the benefit of wysiwyg wordprocessing and graphics software that wants to make use of eps inclusion and needs

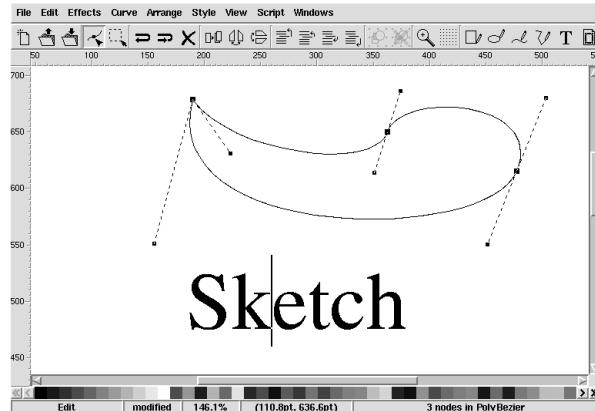


Figure 2 Sketch, a PostScript-based illustration program under Unix

to give the user a visual representation of the file.

There are two variants: screen previews in the form of ASCII comments, and a binary form, in which the PostScript code is preceded by a binary signature and a binary table of contents, and followed by a preview in tiff-, wmf- or Macintosh pict format. Recent versions of dvips and the LaTeX graphics package are no longer confused by preview headers. You may still want to remove them, since they are of no use to TeX and often take up a lot of space. You can do that with an editor, with a simple Perl script or, under MS Windows, with GSview, to be discussed below.

Ghostscript

Although one can do a lot with PostScript files by just reading DSC comments, it is still very useful to have a program such as GhostScript [8] that can parse PostScript:

- for printing to a non-PostScript printer
- for viewing PostScript on-screen
- for converting PostScript to other formats (pdf, bitmap formats, Adobe Illustrator)
- for creating valid eps files from generic PostScript
- for experimenting interactively with PostScript

In what follows I'll concentrate on the computer platforms I have access to, which are Win32 and Unix, or, to be specific, Windows 95 and Linux. Ghostscript is also available for VMS and for Apple Macintosh.

Printing with Ghostscript

On Unix systems, the role of Ghostscript in printing may be invisible to the user. For dvips, config.ps may contain a line `o | lpr` which tells dvips to send its output directly to the printer; lpr may use a filter:

```
#!/bin/sh
gs -dSAFER -dNOPAUSE -q -sPAPERSIZE=a4 \
-sDEVICE=cdj850 -sOutputFile=- -
```

if no real PostScript printer is installed. `cdj850` specifies the printer model; `gs -h` will display a list of available devices. Alternatively, one can print via a previewer; see below.

Under DOS and Windows, GhostScript doesn't work as transparently: generating a PostScript file and printing it with Ghostscript are two separate steps. Here is a possible batchfile for printing with GhostScript:

```
set GS_LIB=c:\gs5.50;c:\gs5.50\fonts
c:\gs5.50\gswin32c -dNOPAUSE -dBATCH
-sPAPERSIZE=a4 -sDEVICE=cdj850 %1
```

Here, too, a command-line option `-h` will give you a list of devices.

Viewing PostScript

Although Ghostscript is capable of displaying PostScript directly, you'll get better functionality and a nicer interface with a separate frontend. Under Unix there are Ghostview and its more modern derivative gv; under all Windows platforms there is GSview. All these can be obtained from the Ghostscript site.

These previewers read DSC comments themselves and pass PostScript code to Ghostscript. From the DSC comments they get pagination information, which enables them to jump backwards and forwards arbitrarily within a document. It also enables them to print and save ranges of pages, something which Ghostscript by itself cannot do. If DSC comments are missing, they can still display the document, but only page by page in forward direction.

Gv and GSview can optionally display the bounding-box area rather than the page area: gv has a button *BBox/Automatic/Letter/...*, GSview has a menu option *Option/EPS Clip*. GSview can add or modify a boundingbox comment (File/PS to EPS); see figure 3 although that may not be enough to turn a PostScript file into a valid eps file; see above. It can also add a preview header (*Edit/Add EPS Preview*) or remove it (*Edit/Extract EPS*).

Converting Postscript to other formats

Ghostscript supports a large number of 'devices', many of which are really file formats, such as:

- ❑ pcxmono, pcxgray and pcx24b, ppx bitmap format at various color depths
- ❑ nullpage, the null device
- ❑ pswrite, regularized PostScript
- ❑ epswrite, regularized eps
- ❑ pdfwrite, pdf(!)

A command

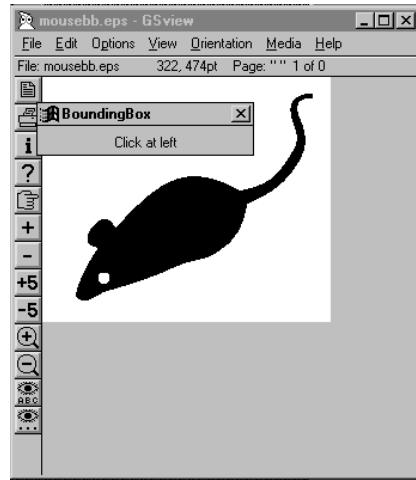


Figure 3 GSview PS to EPS: telling GSview about the boundingbox

```
gs -dBATCH -dNOPAUSE -q -sDEVICE=pcx24b \
-sOutputFile=test.pcx test.ps
```

would convert a one-page PostScript file `test.ps` to an RGB bitmap `test.pcx`. If you want, you can add a resolution parameter, e.g. `'-r50'`. If the source is an eps graphic, you might want to run it through a script such as `epstopdf` by Sebastian Rahtz to make page dimensions equal to the boundingbox; see [16] and figure 6. This will ensure that Ghostscript will convert the right area. Without such a script, you can still trim the result afterwards in an image manipulation program.

Texutil from the ConTeXt package [15] contains similar functionality, based on the same source.

The `pswrite` and `epswrite` output formats are interesting because they 'draw' any text present in the file: references to characters from a font are replaced by curves drawing those characters; see figure 5. If the file is an eps graphic with just a few words of text, especially from an exotic font, this is a good way to remove font dependence.

The `pdfwrite` output format makes Ghostscript into an alternative for Acrobat Distiller. At the time of writing, proper support of general Type 1 fonts is still only available in beta releases, and Ghostscript will probably never have Distiller's range of prepress options. Even so, Ghostscript will in more and more cases be a perfectly adequate alternative.

Regularizing PostScript

There are various utilities, all using Ghostscript, which attempt to regularize the structure of PostScript files. The idea is to redefine primitive operators to write PostScript instructions to a file rather than to draw pixels on a device.

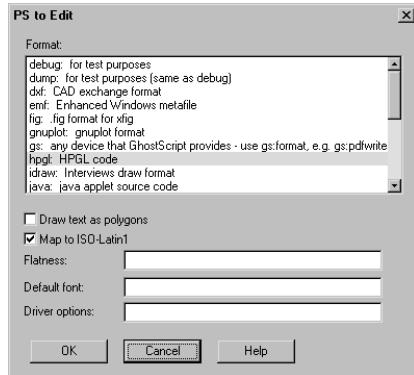


Figure 4 ps2edit as a GSView plugin; selection of output format

Ghostscript's high-level output formats mentioned above fall more or less into this category, but there are others.

Sketch

Figure 5 Characters drawn as outlines, as displayed in Sketch

ps2ai.ps attempts to convert to Adobe Illustrator. It is written entirely in PostScript and is part of the GhostScript distribution. Use it as follows:

```
gs -q -dNODISPLAY -dBATCH ps2ai.ps inputfile \
>outputfile
```

Since nowadays Illustrator can read almost arbitrary PostScript, ps2ai is more likely to be useful if you want to read a file into another PostScript-based drawing program.

ps.conv [17] also tries to make generic PostScript Illustrator-compatible and works in a similar way, except that a script or batch program has to add a header part and a tail. Contrary to ps2ai, it draws characters rather than leave them in text form; see figure 5 for an example of characters drawn as outlines.

pstoedit [18] converts to a variety of vector formats, including regularized PostScript, xfig, gnuplot, hpgl and nowadays also Sketch and Killustrator. Other formats, such as wmf, can be supported via plug-ins, and it will run ps2ai for you if you want.

Pstoedit also works by redefining primitives. It is a binary executable and calls Ghostscript – or another PostScript interpreter – to read the PostScript code. On the windows platform, it can be installed as a plug-in to GSview version

```
$bbneeded=1;
$bbpatt="0-9\.\-]*";
while (<>) {
    if (
        /%&BoundingBox:(\s$bbpatt+)\s($bbpatt+)\s($bbpatt+)/
    ) {
        if ($bbneeded) {
            $width = $3 - $1;
            $height = $4 - $2;
            $xoffset = 0 - $1;
            $yoffset = 0 - $2;
            print "%&BoundingBox: 0 0 $width $height\n";
            print "<< /PageSize [$width $height] >> setpagedevice\n";
            print "gsave $xoffset $yoffset translate\n";
            $bbneeded=0;
        }
    }
    else
    {
        print;
    }
}
print "grestore\n";
```

Figure 6 Perl code by Sebastian Rahtz giving an eps file a page definition matching its bounding box; a more complete version can be downloaded from [16].

2.7I or later; see figure 4.

All these conversion options are less of an overkill than you might think: none of them does a perfect job, and sometimes it helps to apply two of them in succession.

Pdf in prepress

Over the last few years, Adobe has pushed pdf (Portable Document Format, see [7]) as a ‘final-form’ format for distributing documents. Pdf is a subset of PostScript level 3 without programming functionality, and with page independence. Pdfmarks are unique to pdf. The restrictions of page independence and lack of programmability should make pdf more tractable than generic PostScript. The graphics industry has enthusiastically adopted pdf as a workflow format.

Designers, desktop publishers and printers have been very reluctant to hand off or receive printjobs in the form of PostScript dumps, and usually preferred source files. Of course, this created lots of problems with missing files and version differences and text reflow, but apparently these problems were preferable over dealing directly with PostScript files. One has to admit that there was a point in handoff of source files: it certainly facilitated last-minute fixes, and printer/imagesetter settings could be selected by the person running the printer/imagesetter.

Professional graphics software increasingly supports pdf in terms of direct pdf generation and in terms of reading pdf. Various modern TeX implementations (Web2c, VTeX) include a pdf backend, which supports pdf inclusion as a replacement for eps inclusion (VTeX can parse PostScript and is therefore capable to include eps files directly).

Software for pdf

As to generating pdf from PostScript: we already saw that Ghostscript can do that. For best results, however, Acrobat Distiller is still to be preferred. Distiller is available for the Windows, Macintosh and some Unix platforms. Linux users may be interested to know that Distiller 3 runs well under Linux and Wine ([19]).

Adobe's free *Acrobat reader* is available for the Windows- and Macintosh platforms, and for a variety of Unix platforms, including Linux. Ghostscript and its frontends also support viewing pdf.

xpdf [20] is a non-commercial specialized pdf viewer, available under Unix and VMS. It is mainly of interest for some of the accompanying utilities – also available on the Win32 platform: *pdftotext* extracts text from a pdf and *pdfimages* the bitmaps. Of course, you can let Ghostscript convert PostScript or pdf to a bitmap, but original bitmapped data will usually get resampled, resulting in loss of quality. *pdfimages* will recreate the original bitmaps pixel for pixel, although it won't respect the original color depth. Since Ghostscript can convert eps to pdf, *it is finally possible to convert bitmapped eps files without resampling back into an editable format!*

For the Windows- and Macintosh platforms, reasonably-priced commercial utilities have arrived to manipulate pdf, e.g. a visual editor Enfocus Pitstop [21], or Quite a Box of Tricks [22] which can do some very useful conversions, such as converting pages to grayscale. Color separation-and imposition software now becomes available as plugins for Exchange. Check out www.pdfzone.com for more. Final printer settings now can be made when printing from Exchange; the settings made when the original PostScript file was generated can usually be overruled¹. The Crackerjack Exchange [23] plug-in gives access to high-end pre-press functions.

Pdf is also interesting because of its interactive possibilities, but these fall outside the context of this paper.

References

- [1] PostScript Language Reference Manual, Adobe Systems Incorporated, Addison Wesley, 1999
- [2] PostScript Language Tutorial and Cookbook, Adobe Systems Incorporated, Addison Wesley, 1986.
- [3] Adobe web site www.adobe.com
- [4] Adobe developer information, partners.adobe.com/asn/developer/main.html
- [5] PostScript Language Document Structuring Conventions Specification Version 3.0, available from [4]
- [6] Encapsulated PostScript File Format Specification Version 3.0, available from [4]

- [7] Portable Document Format Reference Manual Version 1.3, available from [4]
- [8] Ghostscript, www.cs.wisc.edu/~ghost. Viewer frontends can also be obtained via this url.
- [9] Ghostscript User Manual, Thomas Merz, 1997, www.ifconnection.de/~tm
- [10] Psutils by Angus Duggan, [ftp://ftp.dcs.ed.ac.uk/pub/ajcd](http://ftp.dcs.ed.ac.uk/pub/ajcd) and [ftp://ftp.tardis.ed.ac.uk/users/ajcd](http://ftp.tardis.ed.ac.uk/users/ajcd)
- [11] Mayura Draw, www.mayura.com
- [12] Sketch, www.online.de/home/sketch
- [13] Killustrator, wwwiti.cs.uni-magdeburg.de/~sattler/killustrator.html
- [14] GYVE, the GNU Yellow Vector Editor, www.maru.cs.ritsumei.ac.jp/~sanchan/gyve
- [15] ConTeXt, www.pragma-ade.nl
- [16] www.tug.org/applications/pdftex/epstopdf, a Perl script by Sebastian Rahtz to set the page dimensions of an epsfile to the bounding box.
- [17] Various PostScript tools by Bogusław Jackowski, Piotr Pianowski and Piotr Strzelczyk, [ftp://ftp.GUST.org.pl/pub/TeX/GUST/contrib/PS](http://ftp.GUST.org.pl/pub/TeX/GUST/contrib/PS); also available from CTAN
- [18] Pstoedit, www.geocities.com/SiliconValley/Network/1958/pstoedit
- [19] Wine Is Not an Emulator, www.winehq.com
- [20] Xpdf, [www.foolabs.com/xpdf](http://www foolabs.com/xpdf)
- [21] Enfocus Pitstop, www.enfocus.com
- [22] Quite a Box of Tricks, www.quite.com
- [23] Crackerjack, www.lantanarips.com
- [24] Pdfzone for information on pdf software, both free and commercial, www.pdfzone.com

1. This is not true for the resolution of bitmap fonts generated by dvips. Acrobat handles Type1 fonts much better than bitmapped fonts so the latter should be avoided if at all possible. It is still possible to print from Acrobat with bitmapped fonts at the wrong resolution, but the results won't be optimal.