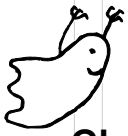


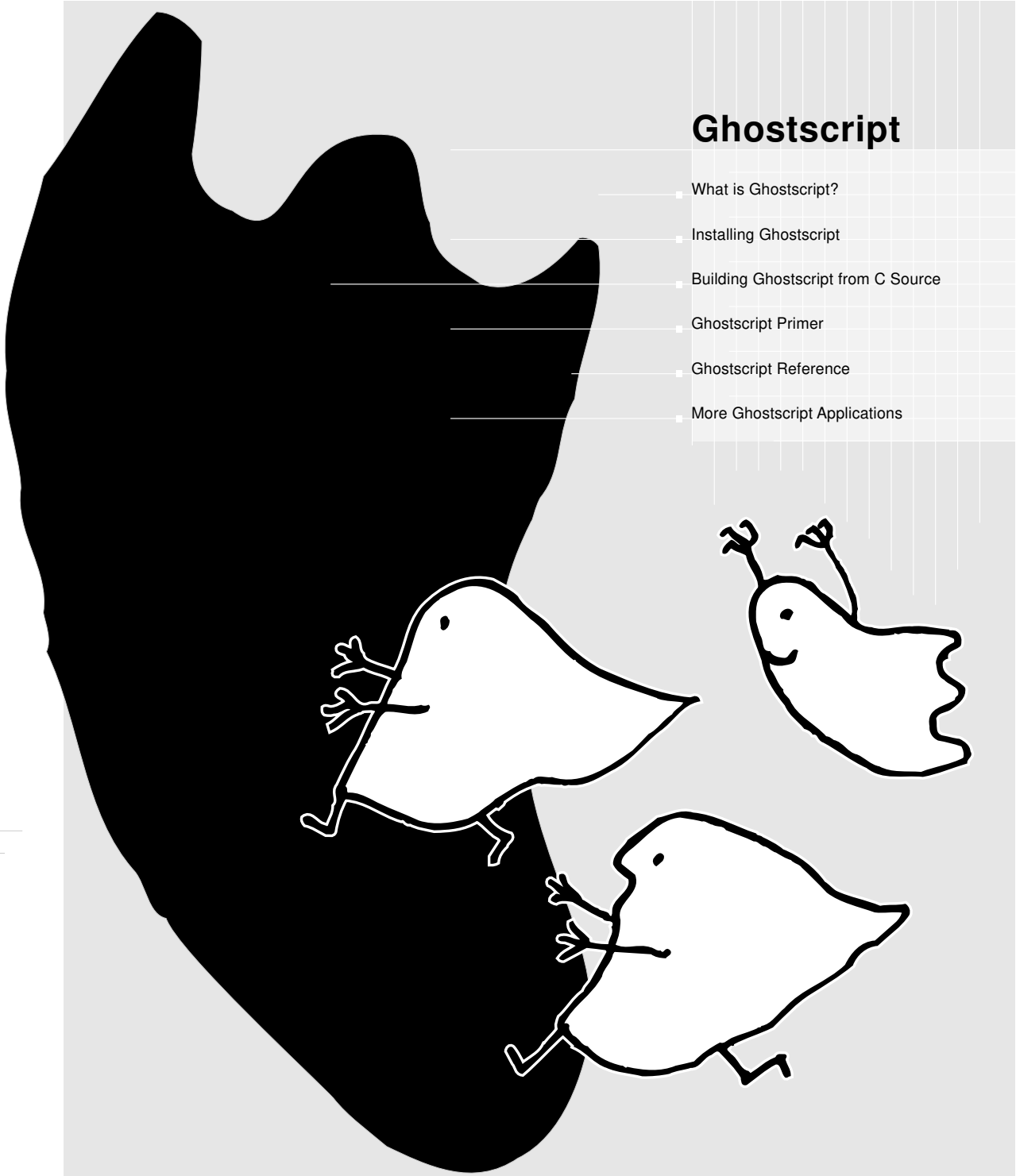
Ghostscript User Manual

© Thomas Merz 1996 (*tm@muc.de*)

This manual is adapted from appendix B of the following book: »PostScript and Acrobat/PDF. Applications, Troubleshooting, and Cross-Platform Publishing« by Thomas Merz, Springer Verlag Berlin Heidelberg New York 1997, ISBN 3-540-60854-0, 420 pages plus CD-ROM. Book and manual are also available in German (Thomas Merz Verlag München, ISBN 3-9804943-0-6).

The Ghostscript manual may be freely copied and redistributed in printed or digital form if no payment is involved. Commercial reproduction is prohibited. However, the author agrees to any redistribution along with the Ghostscript software, provided the distributor complies with the applicable Ghostscript license terms.

The Ghostscript manual is available from *http://www.muc.de/~tm*.



Ghostscript

- What is Ghostscript?
- Installing Ghostscript
- Building Ghostscript from C Source
- Ghostscript Primer
- Ghostscript Reference
- More Ghostscript Applications

Ghostscript User Manual

© Thomas Merz 1996

Notes on the page layout. The design of the Ghostscript manual was chosen to meet several objectives:

- It uses fonts from the PostScript core font set in order to reduce file size. This also avoids font embedding issues since the original book layout uses fonts from the commercial Thesis family.
- Pages can be printed on both letter and A4 format paper. The double-page design also makes it easy to read the manual on screen.
- A two-page spread was slightly reduced and printed on one sheet of paper in order to economically make use of paper and screen estate.

Production notes. I created a PostScript file from the (marginally adapted) FrameMaker book file via the PostScript printer driver. The resulting PostScript file was converted to PDF using Acrobat Distiller. Bookmarks, article threads and general document information were automatically generated by FrameMaker (with a little manual help from pdfmarks).

Translation. If you are interested in translating the Ghostscript manual to other languages, please contact the author (*tm@muc.de*). The manual is currently available in English and German.

This is version 1.2 of the manual. It covers Ghostscript 4.03 and GSview 2.1.

1 What is Ghostscript?

L. Peter Deutsch, founder of Aladdin Enterprises, Menlo Park, California, wrote the PostScript Level 2 and PDF interpreter *Ghostscript* in the C programming language. The program runs on most operating systems, including MS-DOS, Windows 3.x, Windows 95, Windows NT, OS/2, Macintosh, Unix, and VAX/VMS and has been available free of charge ever since its introduction 1988. With the help of many users and programmers on the Internet Ghostscript has become a high-quality and versatile PostScript interpreter. Peter Deutsch also distributes a commercial version with customer-specific enhancements and support. Ghostscript's capabilities include:

Screen output. Ghostscript displays PostScript data on the screen. This is useful to examine PostScript graphics or for saving a few trees if you want to browse some product documentation which is available in PostScript format only. Ghostscript checks PostScript files before you transfer them (e.g., to a service bureau): Are all the necessary fonts there? Are the graphics okay? Do the files contain all pages?

Ghostscript also helps with PostScript trouble-shooting: A faulty page can be rendered on screen revealing which graphics element yields an error message. Ghostscript provides the usual PostScript error messages. A separate frontend to the interpreter, called GSview (for Windows and OS/2) or Ghostview (for the X Window System), simplifies the handling of PostScript files with a user-friendly GUI interface: with these frontends the user can access random pages in the document. Without them, Ghostscript displays the pages one after another, from the beginning to the end of the file.

Printer output. Another important task of a PostScript RIP is to render PostScript data for output on a graphics-capable printer. The Ghostscript distribution contains a wealth of drivers for a wide range of printer models, from the more popular to the more esoteric. A list of all drivers is found in section 3.3, »Configuration Options and Drivers«. These drivers are an integral part of Ghostscript and are not related to Macintosh or Windows system drivers.

Ghostscript can even help optimize the output of a PostScript-capable printer: if the computer's CPU is significantly faster than the printer's, Ghostscript can in many cases speed up PostScript output. PostScript printers with too little RAM sometimes cause trouble. Ghostscript can remedy this by making use of your computer's main memory (and a swap file or swap partition). Ghostscript has proven to be a robust and reliable PostScript-RIP that is superior to many commercial PostScript clones.

PDF on every platform. Beginning with version 3.33, Ghostscript also contains an interpreter for the *Portable Document Format* (PDF), the foundation of Adobe Acrobat. Large parts of Ghostscript's PDF interpreter are written in PostScript. It displays and prints PDF files and even converts them back to PostScript. However, Ghostscript only interprets layout-related information and currently ignores hypertext links or annotations.

Starting with Ghostscript 4.0, the program is also capable of converting PostScript files to PDF, i.e., it offers Distiller functionality. Though this feature, called the pdfwrite device, still has some shortcomings, it is certainly an important milestone since Ghostscript is the first and still the only free program that converts PostScript to PDF.

Utilities and converters. A complete PostScript interpreter together with suitable drivers and utilities makes it possible to carry out many of the operations covered in this book. These include displaying graphics files in the formats GIF, JPEG, or PBM; extracting textual data from PostScript or PDF files; rasterizing PostScript to raster graphics formats such as TIFF, PBM, PNG; converting EPS graphics to the editable Illustrator format, and many other useful features.

License conditions for the end-user. Although Ghostscript is available free of charge, it is still subject to certain license conditions. These are always contained in the Ghostscript program package. Until 1994, Ghostscript was subject to the GNU Public License (GPL). Under the terms of this license, the originator retains the copyright for his work. The use and further distribution of the program, however, are not restricted significantly. Starting with version 3.0 in 1994, Peter Deutsch replaced the GPL with the more restrictive Aladdin Ghostscript Free Public License (AGFPL). Under the terms of the AGFPL, no payment is required for private and commercial use of the program. The sale of Ghostscript is explicitly prohibited, however. Exempted from these conditions are BBSs, or servers for which users pay access fees independent of the downloaded software, and CD-ROMs whose contents may be reproduced and distributed without any payment involved. Anyone interested in commercially licensing Ghostscript should contact Aladdin Enterprises or one of its distribution partners. The complete text of the AGFPL is found in the file named *public* which is part of the Ghostscript distribution fileset.

2 Installing Ghostscript

2.1 Installation on MS-DOS, Windows, and OS/2

Requirements and versions. The Ghostscript installation (not including fonts) uses 3 MB hard disk space. There are several flavors of Ghostscript for PC systems:

System	File name	Notes
MS-DOS	*gs.exe*	*Limited to 640 KB; little support*
MS-DOS, 80386 and higher	*gs386.exe plus dos4gw.exe*	*Version with DOS extender; very fast.*
Windows 3.x	*gswin16.exe plus gsdll16.dll*	*16-bit version; it consists of a DLL and a (small) EXE program.*
Windows 3.x with Win32s; Windows 95 and NT	*gswin32.exe plus gsdll32.dll; gs16spl.exe*	*32-bit version with enhancements for Windows 95 and NT; it consists of a DLL and a (small) EXE program.*
OS/2	*gsos2.exe plus gsdll2.dll*	*Uses gspmdrv.exe as display driver for the presentation manager.*

Setup program for Ghostscript and GSview. Under Windows and OS/2 the user frontend GSview facilitates using Ghostscript by supplying a handy interface to the interpreter. With the GSview setup program you can install both Ghostscript and GSview comfortably. After uncompressing the zip file of GSview, launch the program *setup.exe* (for Windows) or os2setup.exe (for OS/2). The file readme.txt lists the required ZIP files. After displaying some information, the setup programs asks for the name of the directory to which the software

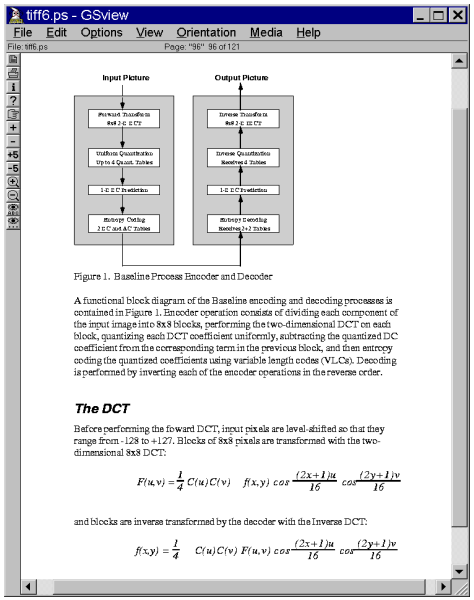


Fig. 1. GSview simplifies Ghostscript usage under Windows and OS/2. It also offers additional possibilities, e.g. ,dealing with EPS files.

is to be installed, and decompresses the ZIP archives. Then it creates a program group (Windows 3.x, NT 4.0) or a start menu entry (Windows 95).

When GSview is first run, it asks whether you want to create a file association between GSview and files with the extension .ps, .eps, and .pdf. This means that double-clicking a file of the respective type launches GSview automatically. You may always want to have the .ps and .eps associations created; however, create the .pdf association only if you don't want to use Acrobat Reader.

If you didn't install Ghostscript into the default directory, you have to choose "Options", "Configure Ghostscript..." after launching GSview for the first time to the enter the path name of the Ghostscript DLL, the include path, and possibly some Ghostscript options (see figure 2).
Note that *GSview* requires the 32-bit version of Ghostscript. With "File", "Open" you can open PostScript files. GSview then passes these files on to Ghostscript for rendering. Using the menu sequence "Media", "Display Settings" you can adjust the

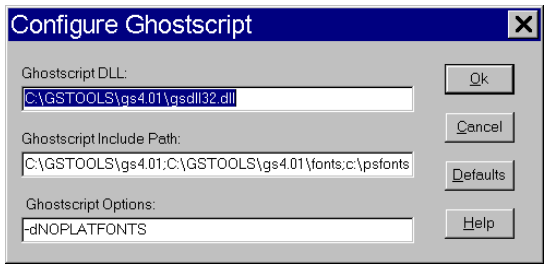


Fig. 2. Configuring Ghostscript path names in GSview.

screen size: larger resolution values yield a larger screen representation. Use smaller resolution if you want to see the entire page at a time without scrollbars. With the "Media" menu you can adjust the page size (media format).

Installing Ghostscript manually. If you don't use Windows, don't need GSview, or can't stand setup programs, you can install Ghostscript manually. All of the files contained in the archive *gs403ini.zip* plus one of the executables listed in the table above are necessary for using Ghostscript. However, the CD-ROM already contains complete file sets (excluding fonts) in the directories *dos, win, win32*, and *os2*. To install Ghostscript, simply copy the files from the suitable directory on the CD-ROM to a directory on your hard disk (e.g., *C:\gs*). To use the 32-bit version for Windows 3.x, you need the *win32s* system extension in addition to the Ghostscript files. Win32s can be found on the CD-ROM in the *tools\win32s\windows* directory and has to be installed separately. (Note: many software packages use win32s, so it may already be installed on your system.)

If you're short on disk space, you can run Ghostscript directly from CD-ROM. The MS-DOS version of Ghostscript uses the environment variable GS_LIB to locate the initialization files if they cannot be found in the current directory or the standard directory *c:\gs*. It's best to include the following statements in your *autoexec.bat* file (assuming you installed Ghostscript in *d:\progs\gs*):

```
set GS_LIB=d:\progs\gs
set PATH=...other path entries...;d:\progs\gs
```

Alternatively, the command line option *-I* can be used to tell Ghostscript where to find its files:

```
gs -ID:\progs\gs
```

Launching and testing Ghostscript. On MS-DOS, you start Ghostscript by typing the program name as usual. With Windows, it's easier to double-click on Ghostscript's icon in the file manager or to create an icon in the program manager. The file *tiger.ps* (part of the Ghostscript distribution) is perfectly suited to test your installation. Start Ghostscript and type in the following command at Ghostscript's prompt:

```
GS>(tiger.ps) run
```

Ghostscript should now display the test file on screen. Having finished the page, Ghostscript asks you to

```
>>showpage, press <return> to continue<<
```

If the file contains more than one page, Ghostscript renders the next page after the return key is pressed.

Caution: The installation is not yet complete! To complete the installation, font access for Ghostscript needs to be configured. This is covered in section 2.4, »Font Configuration«. If you want Ghostscript to use the same command line options every time, you can use the environment variable GS_OPTIONS. Ghostscript evaluates this variable before checking the "real" command line options. Especially with Windows 3x this is much more convenient than the rather clumsy call by way of the program manager and "File", "Run..."

Additional features of GSview. GSview offers many additional Ghostscript functions which are described in several chapters of this book. This includes dealing with EPS files, randomly accessing the pages of DSC (*Document Structuring Conventions)* conforming PostScript files, following hypertext links in PDF files, extracting selected pages to print parts of a document only, copying the bitmapped page contents to the clipboard, or selecting printer drivers in a convenient manner.

GSview uses temporary files for printing and for extracting text. These temporary files are normally created in the directory specified in the TEMP environment variable. A warning will be given if the TEMP variable is not set. The TEMP variable must therefore point to a writable directory on disk, if GSview is run directly from CD-ROM.

2.2 Installation on the Macintosh

Requirements and versions. Ghostscript for the Macintosh is called *Mac GS Viewer*. It's available in three versions, one for Macs with Motorola CPU 68020 and higher, a native version for PowerPC's, and a version for Mac Classics with 68000 CPU (min. 3MB available memory). The installation needs approximately 6MB disk space (including fonts). Mac GS viewer is based on Ghostscript 3.33. Contrary to Windows and OS/2, there is no equivalent of GSview for the Mac. Fortunately, this deficiency is compensated for by several extensions in Mac GS Viewer and its user interface. You can find the following compressed archive files on the CD-ROM:

gs_files.hqx	*init files and documentation*
gs_68k.hqx	*executable for 68020 CPU and higher*
gs_ppc.hqx	*executable for PowerPC*
gs_class.hqx	*executable for Mac Classic*
gs_fonts.hqx	*standard fonts for Ghostscript*
gsmacsrc.hqx	*Mac specific C source files for Ghostscript*

To install Mac GS Viewer you need the program Stuffit Expander which is not included on the CD-ROM. Decompress the archive *gs-files.hqx*. This places the contents of this archive in the

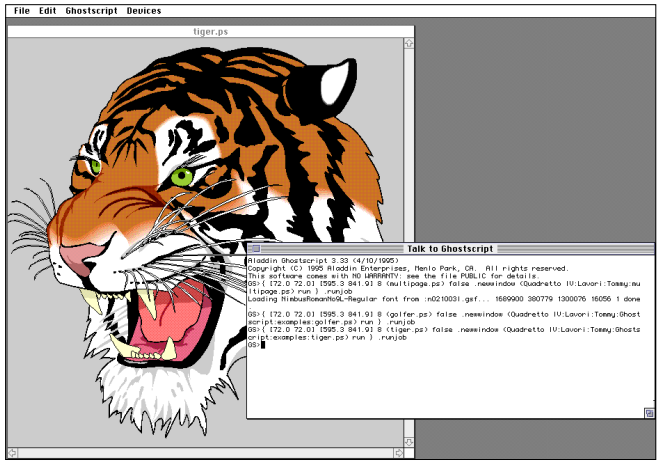


Fig. 3
Mac GS Viewer displays the tiger test page. You can enter PostScript commands in the console window and open files via "File".

Ghostscript folder. Decompress one of the executable files (*gs_68k.hqx*, *gs_ppc.hqx* or *gs_class.hqx*) along with the Ghostscript fonts from the archive *gs-fonts.hqx* into the same folder. The archive *gsmacsrc.hqx* is only needed for compiling an individual version of Ghostscript.

Launching and testing Ghostscript. Launch Ghostscript by double-clicking on the Mac GS Viewer icon. After initialization, the console window ("Talk to Ghostscript") will appear with the Ghostscript prompt

```
GS>
```

Although you can type in PostScript commands in the console window, it's much easier to use Mac-style menu commands. The sequence "File", "Open" lets you open EPSF or TEXT files, e.g., the test file *tiger.ps* in the Ghostscript folder. Activate balloon help to learn more about Ghostscript. After one page has been rendered on screen, the next page can be requested with "Ghostscript", "Next Page". The program can be closed with "Quit". The sequence "Edit", "Settings..." changes such internal settings as page size or the scaling factor of the window.

Device drivers and file output. By default, Ghostscript renders into a Macintosh window. It can also create instructions for other devices or graphics file formats. You can copy the bitmapped contents of a window to the clipboard, or save it as a PICT file with "File", "Save As...". In the "Devices" menu you can choose the output file. Although the Ghostscript source distribution contains many dozens of drivers, the Macintosh version only contains the screen driver, a few printer drivers, and drivers for the file formats PCX, PBM, and TIFF. This decision

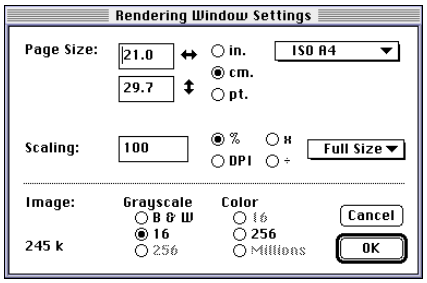


Fig. 4. The "Settings" panel changes page size, scaling factor, and color parameters.

has been made in order to reduce storage requirements. Since it is not possible to dynamically load drivers, you have to recompile and link Ghostscript if you want to use additional drivers.

Font configuration. Ghostscript uses PostScript fonts in ASCII format. Since PostScript fonts on the Mac are usually stored in a compact resource format, Ghostscript cannot use them directly (this is planned for a future version). However, you can convert standard Mac PostScript fonts for use with Ghostscript with the *unadobe* utility on the CD-ROM. Execute the following steps for each font you want to use:

- Start *unadobe* and open the font file. *unadobe* doesn't change the font data, but instead converts the resource representation to a textual representation suitable for Ghostscript. Save the generated font file in the Ghostscript folder (using the old file name or a new one).
- Edit the file *Fontmap* in the Ghostscript folder with a text editor (e.g., SimpleText or TeachText). Append a line similar to the following at the end of the file:

  ```
  /TheSerifBold-Italic     (TheSerBol)          ;
  ```

  This line contains the PostScript name of the font and the name of the font file that you used in the first step. The line must be terminated with a semicolon. If you're unsure about the exact PostScript name of the font, you can open the font file created by *unadobe* with a text editor and look for the entry /FontName.
- Launch Ghostscript to test the font. You can either use a suitable PostScript file or request the font manually in the console window:

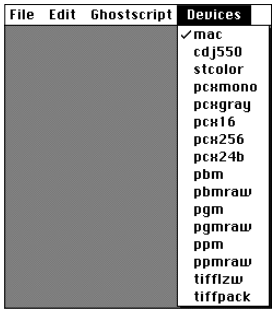


Fig. 5. In the "Devices" menu you can choose a driver for Mac GS Viewer.

```
/TheSerifBold-Italic findfont
```

Alternatively, you can install a fontmap file derived from another platform. For example, you can create the fontmap automatically on Windows (see section 2.4, »Font Configuration«) and install it in the Ghostscript folder together with the font files. It's important not to change the font file names so make sure that they correspond to the fontmap entries. Naturally, you have to observe the font manufacturer's license conditions if the installed fonts originate from another system.

2.3 Installation on Unix

Precompiled executables. The complete C sources for Ghostscript are found on the CD-ROM. An individually configured version of Ghostscript can be built with these source files. This is described in section 3, »Building Ghostscript from C Source«. To make your life easier, the CD-ROM contains precompiled Ghostscript executables for some common Unix flavors. The table below indicates operating system version and CPU platform on which the executables have been built. Check these to decide whether one of the precompiled binaries runs on your system. If so, the binary will save the hassle of building an individual version. Ghostview executables for these same systems are also found on the CD-ROM.

Operating system	CPU platform
Linux	*x86*
SunOS 4.1.1	*Sparc*
Solaris 2.3	*Sparc*
HP-UX 9.01	*HP-PA*
Unixware System V Release 4.2	*x86*
DEC OSF/1 V3.2	*DEC Alpha*
AIX 3.2	*RS/6000*

The installation of Ghostscript on Unix is controlled by the makefile. Therefore, you have to install the source package even if you can use one of the precompiled executables. The precompiled executables are configured with PostScript Level 2 and PDF support. They contain the most common device and file format drivers (see the table in section 3.3, »Configuration Options and Drivers«).

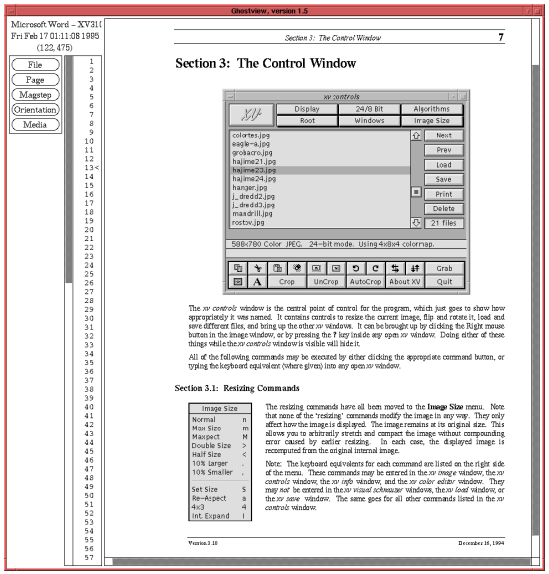


Fig. 6. With Ghostview for X11 you can easily control Ghostscript for screen and printer output. The page list allows quick access to random pages of DSC compliant documents.

To install one of the precompiled versions first decompress the Ghostscript package in a newly created directory, copy the suitable program file (e.g., for Linux), and create a makefile (the tar command creates the directory gs4xx, where 4xx stands for the version number):

```
gunzip -c /cdrom/software/gs/compress/gs4xxsrc.tgz |
    tar xvf -
cd gs4.xx
cp /cdrom/software/gs/linux/gs gs
ln -s unix-gcc.mak makefile
```

Since you don't want to compile the C source (and use the precompiled binary instead) you have to adapt the makefile. Change the following line in the makefile (near the end of the file):

```
install-exec: $(GS)
```

to

```
install-exec:
```

By default, the installation process copies files to the directories */usr/local/bin* and */usr/local/share/ghostscript/4.xx*. If necessary,

you can change these in the makefile too (variable *prefix*). Now install the files with the command

```
make install
```

(with *root* permission). You can delete the directory gs4xx afterwards. To finish the installation, install the Ghostscript fonts as described in the next section.

Ghostview installation. Like Ghostscript, Ghostview may be compiled from C source or installed from one of the precompiled versions on CD-ROM. To use one of these, copy the executable program and the X resource file to a suitable directory on disk. On Linux, for example, use the following commands to install the CD-ROM version:

```
install -s /cdrom/software/ghostv/linux/ghostv
    /usr/X11R6/bin/ghostview
install -m 0444 /cdrom/software/ghostv/linux/ghostv.ad
    /usr/X11R6/lib/X11/app-defaults/Ghostview
```

2.4 Font Configuration

This section applies to all operating systems except Macintosh. Font configuration for the Mac differs from other systems and has been covered with the Mac installation already. To apply the following description for Unix, you have to adapt the syntax for environment variables and directory paths appropriately.

For many years, the Ghostscript distribution contained only low quality public domain fonts. In 1996, the German company URW++ contributed commercial-quality PostScript Type 1 fonts for use with the Ghostscript package. These include look-alikes for the 35 PostScript core fonts found in most laser printers. The URW++ fonts are distributed under the Aladdin and the GNU license.

You can find the URW++ fonts in the *urw* directory on the CD-ROM. You can significantly enhance text representation on screen and paper by integrating high quality fonts.

Ghostscript uses all flavors of PostScript fonts and formats, i.e., Type 1, Type 3, ASCII, and binary. Ghostscript also is capable of processing TrueType fonts (see below). It is necessary for Ghostscript to have access to the font file (installing the font in the operating system, ATM, or Display PostScript is not enough).

If you installe Ghostscript with GSview, the URW font set is automatically installed. If you only want to use the PostScript core font set only, the font configuration is already finished.

Static font configuration. There are two choices for the font configuration. If the installed font base rarely changes, put one or more font directories in the environment variable GS_LIB. For example, if you want to launch Ghostscript directly from CD-ROM on MS-DOS or Windows, use the following:

```
set GS_LIB=g:\fonts\urw
```

On Unix, you have to export the variable using the command

```
GS_LIB=/usr/local/share/ghostscript/fonts
export GS_LIB
```

Another difference for Unix systems is the colon ":" used as path separator instead of the semicolon ";". A typical hard disk installation on MS-DOS or Windows uses the following:

```
set GS_LIB=C:\gs;C:\gs\fonts
```

or on Unix:

```
set GS_LIB=/usr/local/share/ghostscript/fonts
export GS_LIB
```

For now, the font configuration is already done.

Each directory in GS_LIB may contain the *Fontmap* file which defines the relation between font names and file names. To add new fonts, simply add a line similar to the following at the end of the *Fontmap* file:

```
/Fontname     (filename) ;
```

The file name in parentheses must be found in one of the font directories. In many cases you can save yourself the trouble of entering font and file names because there are prebuilt *Fontmap* files for several systems and font configurations. If you use one of the following, simply copy the appropriate file to the name *Fontmap*:

File name	Origin of fonts/name of operating system
Fontmap.atm	*Windows with standard ATM fonts*
Fontmap.atb	*Adobe Type Basics font package*
Fontmap.gs	*URW++ fonts (standard fontmap)*
Fontmap.os2	*OS/2 with integrated ATM*

File name	Origin of fonts/name of operating system
Fontmap.osf	*DEC OSF/1 with DPS*
Fontmap.sol	*Solaris 2.3 and higher with DPS*
Fontmap.ult	*Ultrix 4.3 and higher with DPS*
Fontmap.vms	*VAX/VMS with DECwindows/Motif and DPS*

In order to let Ghostscript access the fonts in Display PostScript systems, you have to include the appropriate path name in GS_LIB. A list of these (quite varying) path names can be found in section 7.2.3, »Installing Additional Fonts«.

Dynamic font configuration. For large or frequently changing font installations the second method is preferable. Ghostscript checks the GS_FONTPATH environment variable to determine the available font files and scans these files to find out the names of the fonts they contain. Similar to GS_LIB above, this variable contains one or more directory names. Since Ghostscript automatically recognizes fonts, this method is much more flexible. However, since Ghostscript checks all font directories every time a new font is requested, this method increases startup time. To avoid inconsistencies, make sure there is no *Fontmap* file in one of the GS_LIB directories if you use GS_FONTPATH or launch Ghostscript with the -dNOFONTMAP option.

TrueType fonts. Starting with Version 4.0, Ghostscript contains a rasterizer for TrueType fonts. Note that Ghostscript supports two flavors of TrueType fonts.

Type 42 fonts: These contain TrueType data wrapped in PostScript instructions and can be directly processed by the interpreter. Type 42 capability is integrated in all Ghostscript configurations which include Level 2 support (which is the case for all default makefiles). Level 2 capable Ghostscript versions accept PostScript files with embedded Type 42 fonts (created, for example, by the Windows PostScript driver).

Native TrueType files: Files in TTF format (as installed under Windows) contain raw TrueType data which a PostScript interpreter cannot directly process. Ghostscript contains some additional code for interpreting raw TrueType files. This code is activated by the "ttfont" makefile option. Since TrueType plays an important role for Windows users, the Windows Ghostscript versions are configured with the "ttfont" feature by default. This means that the Windows TrueType font directory can be configured for use with Ghostscript just as any PostScript font directory. However, Ghostscript only recognizes TTF font files when searching font files via the GS_FONTPATH variable; it's not possible to process TrueType files via the "run" operator. You can work around this with the following instruction:

```
(file.ttf) (r) file .loadttfont
```

Platform fonts on Windows and X11. Ghostscript always needs access to all PostScript fonts used in a document. It rasterizes all the characters itself instead of delegating this task to the operating system, ATM, or the X server (like other programs do). However, to speed up processing, Ghostscript sometimes uses system fonts. This feature is called *platform fonts* and is used only for certain font sizes and horizontal or vertical text. Even so, the PostScript font name must exactly match the system font name.

For small font sizes, platform fonts generally improve screen font representation. In some situations, however, they make it worse – e.g., if PostScript and system font metrics don't match (which results in ugly formatting). Another problem affects fonts with unusual character sets (encodings). In both cases you can turn off platform fonts by launching Ghostscript with the -dNOPLATFONTS option. There is also a special X Window resource to achieve this:

```
Ghostscript*useExternalFonts:false
```

3 Building Ghostscript from C Source

Since the C source code of Ghostscript is readily available, you can build an executable version of the interpreter if you have a C compiler for your system (and some experience compiling C programs). This way, you can link in additional drivers that are contained in the Ghostscript distribution but that (due to memory restrictions) are not compiled into the standard versions. More adventurous programmers also have the chance to implement their own extensions to Ghostscript. The following sections will give an overview of the compilation process and the configuration options.

3.1 A Little Tour to Ghostscript Files

The Ghostscript source directory contains several hundred files. The compilation may even double this number, so you may well get lost in this multitude of files. The following table lists the most important file types contained in the Ghostscript distribution.

Files	Contents
readme, news	*notes on the current version*
new-user.txt	*overview of Ghostscript*
use.txt	*information on using Ghostscript*
make.txt	*notes on compiling Ghostscript*
*other *.txt*	*additional documentation on special topics*
public	*"Aladdin Ghostscript Free Public License" (licensing conditions)*
**.1*	*Unix-style manual pages*
**.c, *.h, *.asm*	*source files*
**.mak, *.def, *.rc, *.icx*	*auxiliary files for building the program*
**.sh, *.bat, *.cmd*	*scripts and batch files used in the build process; several special applications of Ghostscript*
gs_*.ps	*initialization files for Ghostscript*
pdf_*.ps	*initialization files for the PDF interpreter*
*other *.ps*	*auxiliary PostScript files and sample programs*
*Fontmap.**	*Fontmap files for several systems*

3.2 Compiling the Standard Version

Requirements. To build Ghostscript, you need approximately 15 MB disk space. The source is written in ANSI-C suited for most current C compilers. However, you can also use an old Kernighan & Ritchie compiler. The auxiliary program *ansi2knr* converts the source files to K&R syntax before compilation. Another auxiliary program called *genarch* automatically creates an include file which describes hardware and compiler architecture. This includes bit and byte ordering, word size and other system-specific information. Ghostscript can be compiled on MS-DOS, Windows, OS/2, Amiga, many Unix systems, Macintosh, and VMS. The C source is highly portable, so compiling it on new systems shouldn't be much of a problem.

Makefiles. The compilation process is controlled by makefiles which you can adapt as necessary. To improve legibility, the makefiles are split in parts:

gs.mak	*Ghostscript "core" source*
lib.mak	*graphics library*
int.mak	*interpreter*
devs.mak	*device drivers*
jpeg.mak	*JPEG library*
libpng.mak	*PNG graphics file format library*
zlib.mak	*compression routines used for the PNG format*

All other **.mak* files are platform specific makefiles used to configure the development system. The compilation process for different platforms and the supported development systems are described in full detail in the file *make.txt*. The following description is not intended to replace this file, but to give you a jump start.

Compiling on MS-DOS, Windows, OS/2. First decompress the C source from the compressed archives *gs4xxsr1.zip* and *gs4xxsr2.zip*. The files in the other archive files *(jpeg_6a.zip, lpng089c.zip*, and *zlib102.zip)* contain additional libraries used by Ghostscript. They are stored in three subdirectories of the Ghostscript directory. The *-d* option of pkunzip creates the directory *gs4.xx* (*x:* represents the CD-ROM drive letter):

```
pkunzip -d x:\software\gs\compress\gs4xxsr1.zip
pkunzip -d x:\software\gs\compress\gs4xxsr2.zip
cd gs4.xx
mkdir jpeg-6a
cd jpeg-6a
pkunzip x:\software\gs\compress\jpeg_6a.zip
cd ..
mkdir libpng
cd libpng
pkunzip x:software\gs\compress\lpng089c.zip
cd ..
mkdir zlib
cd zlib
pkunzip x:software\gs\compress\zlib102.zip
```

The Ghostscript distribution contains makefiles for the Microsoft, Borland, Watcom and other C compilers (see *make.txt*). For example, to create the 32-bit Windows version of Ghost-

script using the Borland Compiler, you first have to create the makefile:

```
echo !include "bcwin32.mak" >makefile
```

Next, you can change some settings in *bcwin32.mak*, e.g., compiler and Ghostscript paths, optimizations for 386/486 CPUs or FPU, assembler accelerator modules or debugging options. Finally, launch *make*. In an intermediate step you have to manually start the Windows program *genarch* to create a system specific include file.

Compiling on Unix. Decompress the C source for Ghostscript and the JPEG, PNG, and ZLIB libraries from the compressed tar archives into a suitable directory. The tar commands creates the gs4.xx directory and three subdirectories:

```
gunzip -c /cdrom/software/gs/compress/gs4xxsrc.tgz |
    tar xvf -
cd gs4.xx
gunzip -c /cdrom/software/gs/compress/jpeg_6a.tgz |
    tar xvf -
gunzip -c /cdrom/software/gs/compress/lpng089c.tgz |
    tar xvf -
gunzip -c /cdrom/software/gs/compress/zlib102.tgz |
    tar xvf -
mv libpng-0.89c libpng
mv zlib-1.0.2 zlib
```

The mv commands are necessary because the tar archives include directory names containing the version numbers.

The Ghostscript distribution contains makefiles for several operating systems and compilers (ANSI-C, Kernighan&Ritchie, and GNU-C). Choose the appropriate makefile from **.mak* and create a symbolic link with the following command (assuming you use the GNU compiler):

```
ln -s unix-gcc.mak makefile
```

Use *unixansi.mak* or *unix-cc.mak*, respectively, for an ANSI or K&R compiler. On some systems you have to adapt the search path for X11 specific include files and additional libraries. These are controlled by the XINCLUDE und LDFLAGS variables in the makefile which you can change before launching make. By changing the *prefix* variable in the makefile you can adapt the install directory. Now compile and install Ghostscript with the command

```
make install
```

Consult the *make.txt* file if you have trouble with the build process or experience compilation errors. The install process needs *root* permission on most systems. After compiling Ghostscript, the install command copies the executable program to */usr/local/bin* and the auxiliary files to */usr/local/share/ghostscript/4.xx*. To complete the installation, you have to install fonts for Ghostscript as described in the next section.

Now you should install and compile the Ghostview source. This is accomplished with the commands

```
gunzip -c gsview15.tgz | tar xvf -
cd ghostview-1.5
xmkmf
make
make install
```

If your system doesn't have the *xmkmf* program, you have to adapt the Ghostview makefile manually (this shouldn't be too hard if you have ever worked with makefiles).

Compiling on the Macintosh. In addition to the source code in MS-DOS or Unix format, you need the archive *gsmacsrc.hqx* to compile Ghostscript for the Mac. This archive contains some additional Macintosh-specific files. The source files end up in several folders. You'll find an overview of the build process in the *Mac GS Viewer Manual* (part of the Ghostscript files) and some hints for compiling with the MPW or CodeWarrior compilers in the file *worksheet*.

Compiling on other systems. The file *readme* contains some remarks on Ghostscript ports to other systems, including VMS, Amiga, Atari ST, Acorn Archimedes and NEXTSTEP.

3.3 Configuration Options and Drivers

You can adjust the Ghostscript makefile to build a version that suits your needs with several extensions and an individual assortment of drivers. The most important options are PostScript Level 1, PostScript Level 2, and PDF. Standard configurations for most systems include PostScript Level 2, and on 32 bit systems additionally PDF. You can use the FEATURE_DEVS variable in the makefile to control the interpreter's configuration.

The second configuration option relates to the set of included drivers. Since it is not possible to load Ghostscript drivers dynamically at runtime, you have to choose the driver set when building the program. In doing so, you trade functionality for memory efficiency: If all available drivers were included in Ghostscript, the program would need far too much memory. For this reason the standard configuration for each platform contains only the most important screen, printer, and file format drivers for the respective platform. You can check the list of available Ghostscript drivers using the command line

```
gs -?
```

You can also obtain the driver list with the following commands at the Ghostscript prompt:

```
devicenames ==
```

or

```
devicedict {pop ==} forall
```

If you want Ghostscript to use a driver for which C source is included in the distribution but which is not compiled into the executable by default, you have to build your own version by using a modified makefile.

Many Ghostscript drivers have been contributed by users and later became part of the Ghostscript distribution. If you want to write a new driver, read the remarks on Ghostscript/driver interaction in *drivers.txt*.

In addition to the file format drivers listed in the table below, previous Ghostscript versions contained one for the GIF graphics format. In reaction to the licensing problems around the LZW compression technique used in GIF, Peter Deutsch dropped support for this format from the Ghostscript distribution. If you have to create GIF files you can integrate the GIF driver from an older release. However, it's easier to render to another graphics file format (e.g., TIFF or PBM) and convert it to GIF.

The table below lists all display, printer, and file format drivers available for Ghostscript 4.03. Each line lists the name of the device or format and the short name of the driver. The last column in the table tells you on which of the following platforms the particular driver is part of the standard configuration: MS-DOS 386 (D), Windows 32-bit (W), OS/2 (O), Unix (U), and Mac (M). The driver name is used in the makefile as well as for selecting a driver within Ghostscript. The makefile variables DEVICE_DEVS1 to DEVICE_DEVS15 contain the names of the drivers to be included in the program. If you don't supply a driver name at startup, Ghostscript uses the first driver in its list (which is a display driver on all platforms).

List of Ghostscript drivers available with version 4.03

Display drivers		
ATI Wonder SuperVGA, 256 colors	*atiw*	*D*
AT&T 3b1/Unixpc monochrome display	*att3b1*	*–*
Borland Graphics Interface	*bgi*	*–*
CRT sixels, e.g. VT240 compatible terminals	*sxlcrt*	*–*
EGA 640x350, 16 colors	*ega*	*D*
Hercules Graphics Display	*herc*	*–*
Linux PC with VGALIB	*vgalib*	*–*
Linux PC with VGALIB, 256 colors	*lvga256*	*–*
Macintosh window (QuickDraw)	*mac*	*M*
Microsoft Windows 3.1 DLL	*mswindll*	*W*
OS/2 DLL bitmap	*os2dll*	*O*
OS/2 Presentation Manager	*os2pm*	*O*
Private Eye display	*pe*	*–*
Sony Microsystems monochrome display	*sonyfb*	*–*
SunView window system	*sunview*	*–*
SuperVGA with S3 Chip 86C911	*s3vga*	*–*
SuperVGA 800x600, 16 colors	*svga16*	*D*
SuperVGA with Tseng Labs ET3000/4000 Chip, 256 colors	*tseng*	*D*
SuperVGA with VESA driver	*vesa*	*–*
Trident SuperVGA, 256 colors	*tvga*	*D*
VGA 640x480, 16 colors	*vga*	*D*
X Window System (X11), release 4 and higher	*x11*	*U*
X Window System as alpha device	*x11alpha*	*U*
X Window System as CMYK device, 1 bit per color	*x11cmyk*	*U*
X Window System as b/w device	*x11mono*	*U*
Printer drivers		
Apple Dot Matrix printer (also for Imagewriter)	*appledmp*	*–*
Apple Imagewriter, high resolution	*iwhi*	*–*
Apple Imagewriter, low resolution	*iwlo*	*–*
Apple Imagewriter LQ, 320 x 216 dpi	*iwlq*	*–*
CalComp raster format	*ccr*	*–*
Canon BubbleJet BJ10e	*bj10e*	*DWOU*
Canon BubbleJet BJ200	*bj200*	*DWOU*
Canon Color BubbleJet BJC-600 and BJC-4000	*bjc600*	*DWOU*

Canon Color BubbleJet BJC-800	bjc800	DWOU
Canon LBP-8II laser printer	lbp8	OW
Canon LIPS III laser printer with CaPSL	lips3	–
Mitsubishi CP50 color printer	cp50	–
DEC LA50	la50	–
DEC LA70	la70	–
DEC LA70 with low resolution extensions	la70t	–
DEC LA75	la75	–
DEC LA75plus	la75plus	–
DEC LJ250 Companion color printer	lj250	OW
DEC LJ250, alternate driver	declj250	OW
DEC LN03	ln03	–
Epson AP3250	ap3250	–
Epson-compatible dot matrix printer (9 or 24 pin)	epson	DWO
Epson-compatible 9-pin, intermediate resolution	eps9mid	OW
Epson-compatible 9-pin, triple resolution	eps9high	DWO
Epson LQ-2550 and Fujitsu 3400/2400/1200 color printers	epsonc	OW
Epson Stylus Color	stcolor	WOM
Epson Stylus 800	st800	WO
HP DesignJet 650C	dnj650c	–
HP DeskJet and DeskJet Plus	deskjet	DWOU
HP DeskJet 500	djet500	DWOU
HP DeskJet 500C, 1 bit per pixel	cdeskjet	DWOU
HP DeskJet 500C, 24 bit per pixel, also for DeskJet 540C	cdjcolor	DWOU
HP DeskJet 500C (same as cdjcolor)	cdj500	–
HP DeskJet 500C (not for 550C/560C), alternate driver	djet500c	OW
HP DeskJet 500C b/w, also for DeskJet 510, 520, 540C	cdjmono	DWOU
HP DeskJet 550C/560C	cdj550	DWOUM
HP LaserJet	laserjet	DWOU
HP LaserJet Plus	ljetplus	DWOU
HP LaserJet IId/IIp/III* with TIFF compression	ljet2p	DWOU
HP LaserJet III* with delta row compression	ljet3	DWOU
HP LaserJet IIID with duplex function	ljet3d	–
HP LaserJet 4, 600 dpi	ljet4	DWOU
HP LaserJet 4 with Floyd-Steinberg dithering	lj4dith	–
HP PaintJet XL	pj	DWOU
HP PaintJet XL, alternate driver	pjetxl	–
HP PaintJet XL color printer	pjxl	DWOU
HP PaintJet XL color printer, alternate driver	paintjet	–
HP PaintJet XL 300 color printer, also for DeskJet 1200C	pjxl300	DWOU
HP 2563B line printer	lp2563	–
IBM Proprinter, 9 pin	ibmpro	DWO
IBM Jetprinter inkjet color printer (Modell #3852)	jetp3852	OW

Imagen ImPress	imagen	–
C. Itoh M8510	m8510	OW
Microsoft Windows system printer driver (DDB)	mswinprn	W
Microsoft Windows system printer driver (DIB)	mswinpr2	W
Mitsubishi CP50 color printer	cp50	–
NEC P6/P6+/P60, 360 x 360 DPI	necp6	OW
OCE 9050	oce9050	–
Okidata IBM-compatible dot matrix printer	okiibm	–
Okidata MicroLine 182	oki182	–
OS/2 system printer driver (only for OS/2 DLL)	os2prn	–
Ricoh 4081 laser printer	r4081	OW
Sony Microsystems NWP533 laser printer	nwp533	–
StarJet 48 inkjet printer	sj48	–
SPARCprinter	sparc	–
Tektronix 4693d color printer, 2 bits per RGB component	t4693d2	OW
Tektronix 4693d color printer, 4 bits per RGB component	t4693d4	OW
Tektronix 4693d color printer, 8 bits per RGB component	t4693d8	OW
Tektronix 4695/4696 inkjet plotter	tek4696	OW
Xerox XES 2700, 3700, 4045, and others	xes	–

Fax and file format drivers

BMP monochrome	bmpmono	WO
BMP 4 bits (EGA/VGA)	bmp16	WO
BMP 8 bits	bmp256	WO
BMP 24 bits	bmp16m	WO
CGM b/w, low level output only	cgmmono	–
CGM 8 bits, low level output only	cgm8	–
CGM 24 bits, low level output only	cgm24	–
CIF file format for VLSI	cif	–
DigiBoard DigiFAX, high resolution	dfaxhigh	O
DigiBoard DigiFAX, low resolution	dfaxlow	O
Fax group 3, with EOLs, no header or EOD	faxg3	U
Fax group 3 2-D, with EOLs, no header or EOD	faxg32d	U
Fax group 4, with EOLs, no header or EOD	faxg4	U
ImageMagick MIFF format, 24 bit color (RLE compressed)	miff24	–
MGR devices, 1 bit monochrome	mgrmono	–
MGR devices, 2 bits gray scale	mgrgray2	–
MGR devices, 4 bits gray scale	mgrgray4	–
MGR devices, 8 bits gray scale	mgrgray8	–
MGR devices, 4 bits color	mgr4	–
MGR devices, 8 bits color	mgr8	–
PCX monochrome	pcxmono	DWOUM
PCX, 8 bits gray scale	pcxgray	DWOUM

List of Ghostscript drivers available with version 4.03

PCX, 4 bits color	*pcx16*	*DWOUM*
PCX, 8 bits color	*pcx256*	*DWOUM*
PCX, 24 bits color	*pcx24b*	*DWOUM*
PDF (Portable Document Format)	*pdfwrite*	*WU*
PKM(Portable Inkmap),CMYK internally,RGB output,ASCII	*pkm*	*U*
PKM, (raw format)	*pkmraw*	*U*
Plain bits (raw format), monochrome	*bit*	*DWOU*
Plain bits (raw format), RGB	*bitrgb*	*DWOU*
Plain bits (raw format), CMYK	*bitcmyk*	*DWOU*
PBM (Portable Bitmap), ASCII format	*pbm*	*UM*
PBM, raw format	*pbmraw*	*UM*
PGM (Portable Graymap), ASCII format	*pgm*	*UM*
PGM, raw format	*pgmraw*	*UM*
PGM, optimizing to PBM ASCII if possible	*pgnm*	*U*
PGM, optimizing to PBM raw if possible	*pgnmraw*	*U*
PNG (Portable Network Graphics), monochrome	*pngmono*	*WOU*
PNG (Portable Network Graphics), 8 bits gray scale	*pnggray*	*WOU*
PNG (Portable Network Graphics), 4 bits color	*png16*	*WOU*
PNG (Portable Network Graphics), 8 bits color	*png256*	*WOU*
PNG (Portable Network Graphics), 24 bits color	*png16m*	*WOU*
PPM (Portable Pixmap), ASCII format (RGB)	*ppm*	*UM*
PPM, raw format (RGB)	*ppmraw*	*UM*
PPM, optimizing to PGM ASCII or PBM ASCII if possible	*pnm*	*U*
PPM, optimizing to PGM raw or PBM raw if possible	*pnmraw*	*U*
PostScript Level 1, monochrome bitmap	*psmono*	*DWOU*
SGI RGB pixmap format	*sgirgb*	–
TIFF b/w, CCITT RLE 1-dim (fax group 3 without EOLs)	*tiffcrle*	*DWOU*
TIFF b/w, fax group 3 (with EOLs)	*tiffg3*	*DWOU*
TIFF b/w, fax group 3 2-D	*tiffg32d*	*DWOU*
TIFF b/w, fax group 4	*tiffg4*	*DWOU*
TIFF b/w, LZW (compression tag 5)	*tifflzw*	*DWOUM*
TIFF b/w, PackBits (compression tag 32773)	*tiffpack*	*DWOUM*
TIFF 12 bit RGB color (no compression)	*tiff12nc*	*DWOU*
TIFF 24 bit RGB color (no compression)	*tiff24nc*	*DWOU*

4 Ghostscript Primer

This section is meant to give you a jump start into directly using Ghostscript. If you use GSview or Ghostview to drive Ghostscript, you only have to configure the appropriate Ghostscript call; the rest is handled by the frontend.

4.1 Launching Ghostscript

In the following examples, *gs* always represents the name of the Ghostscript executable file. Depending on your platform, the actual name may vary.

File search path. First you have to make sure that Ghostscript finds its initialization and font files. When searching for files without an absolute file name, Ghostscript uses the following search order:

- The current directory.
- The directories listed at the *-I* command line option, e.g., on MS-DOS, Windows or OS/2:

  ```
  gs -Id:/gstools/gs4.xx;d:/gstools/gs4.xx/fonts
  ```

  or on Unix:

  ```
  gs -I/usr/local/lib/gs:/usr/local/psfonts
  ```

- The directories listed in the GS_LIB environment variable.
- Predefined directories selected at build time using the GS_LIB_DEFAULT makefile variable (*C:\gs* on MS-DOS, Windows or OS/2; */usr/local/share/ghostscript/4.xx* on Unix).

Watch the tiger. Ghostscript accepts the names of the PostScript files to display or print:

```
gs file1.ps file2.ps ...
```

The interpreter processes the files one after the other. Then, Ghostscript prompts for PostScript commands:

```
GS>
```

At this prompt you can issue PostScript commands. If it is clumsy or uncommon to pass file names on the command line (e.g., on Windows), you can open files with the *run* command at the prompt. Try the *tiger.ps* sample file included in the Ghostscript distribution:

```
GS>(file.ps) run
```

On the Mac, it's even easier to use the "File", "Open" menu command.

Note concerning MS-DOS path names: Since the backslash "\" escapes the next character in PostScript strings, you have to use double backslashes in path names. However, a single Unix-style slash "/" also works as a separator in path names, for example

```
GS>(c:/gs/tiger.ps)run
```

The following command exits the interpreters:

```
GS>quit
```

Alternatively, you can append *-c quit* to the command line.

Selecting a driver. Usually, Ghostscript uses the first driver in its internal list (configured at build time). This driver outputs to the screen in the standard configurations on all operating systems. You can select another driver on the command line:

```
gs -sDEVICE=laserjet file.ps -c quit
```

This instructs Ghostscript to produces output for the particular device or file format (*laserjet* in the example above). Using the following commands at the Ghostscript prompt, you can change the driver at any time:

```
GS>(epson) selectdevice
GS>(file.ps) run
```

For printers with multiple resolutions you can also set the desired print resolution using the *-r* option:

```
gs -sDEVICE=epson -r60x72 -c quit
```

Redirecting printer data. On MS-DOS, Ghostscript sends printer data directly to the parallel port. On Unix, the printer data is sent to a temporary file. You can also redirect it to your own print file:

```
gs -sDEVICE=laserjet -sOutputFile=laserjet.prn file.ps
    -c quit
```

If the output file name contains the variable *%d* (e.g. *laser%d.prn*), Ghostscript produces one output file per page and replaces the *%d* with the page number. On Unix, you can also redirect the data to a pipe using the "\|" syntax:

```
gs -sDEVICE=laserjet -sOutputFile=\|lp file.ps -c quit
```

Finally, Ghostscript sends the printer data to its standard output with the following command line (the *-q* option suppresses messages):

```
gs -q -sOutputFile=- file.ps -c quit
```

On Windows and OS/2 the easiest way to redirect printer data is to use GSview. This frontend presents a menu for choosing the printer interface.

Page size. Ghostscript uses U.S. letter size by default. To change this, use a text editor to locate the following line in the initialization file *gs_init.ps*:

```
% (a4) /PAPERSIZE where { pop pop } { /PAPERSIZE exch
    def } ifelse
```

In this line, remove the "%" comment sign at the beginning to use A4 format. You can also replace the "a4" in parentheses by any other known format. A list of all formats known to Ghostscript (and their dimensions) can be found in the *gs_statd.ps* file. Alternatively, you can change the page size on the Ghostscript command line:

```
gs -sPAPERSIZE=legal file.ps
```

4.2 Printing with Ghostscript

Printing on MS-DOS. To reduce memory problems, you should use *gs386.exe* if possible. The following longish call processes a PostScript file for output on a Laserjet 4 printer connected to the parallel interface:

```
gs386 -q -dNOPAUSE -sDEVICE=ljet4 file.ps -c quit
```

The *-c quit* option is used to quit Ghostscript after the PostScript file is completely rendered. Due to the *-q* option, Ghostscript itself works quietly. However, the 386 MS-DOS extender still presents its copyright banner. You can suppress it with an environment variable:

```
set DOS4G=quiet
```

If your printer is not connected to the parallel interface or you want to bring the printer data to another machine, you can redirect it to a file:

```
gs386 -q -dNOPAUSE -sDEVICE=ljet4 -sOutputFile=ljet.prn
    file.ps -c quit
```

To print this file, send it to the printer interface with the *copy* command:

```
copy /b ljet.prn lpt1:
```

The */b* (binary) option is important because otherwise the copy command may not completely transfer the printer data.

Printing on Windows 3.x. In Windows, it is easiest to use GSview for printing with Ghostscript. After installing this Ghostscript frontend correctly, you can select a PostScript file using "File", "Print...". In the subsequent menus you can select printer driver, resolution (if the printer supports multiple resolutions), and – in the case of DSC compatible files – the page range you want to print. When Ghostscript has finished processing the file, you can select the printer interface for forwarding the data in a dialog box.

Printing on Windows 95 and NT. For the newer systems of the Windows family you can use the methods described above for MS-DOS and Windows 3.x. Additionally, you can select a printer queue using its UNC name:

```
gswin32 -q -dNOPAUSE -sDEVICE=ljet4 file.ps
    -sOutputFile="\\spool\<printer name>" -c quit
```

This spools the printer data to the given printer queue. Using the *-sOutputFile="\\spool"* option instructs Ghostscript to present a dialog box in which you can select the desired printer queue or interface.

Printing on Unix. In Unix it's possible to integrate Ghostscript in the printing process seamlessly. However, some experience with Unix systems administration is required. The variety of available Unix derivatives doesn't really simplify the task of describing the integration of a PostScript emulation for printers. The following notes are not supposed to be a complete description, but should help you get started.

Assuming other system components (especially spooler and backend) are already set up correctly and are able to transfer binary data to the printer unmodified, you can manually use Ghostscript for printing:

```
gs -q -dNOPAUSE -sDEVICE=ljet4 -sOutputFile=\|lp
    file.ps -c quit
```

You can find hints on integrating Ghostscript in systems with a printcap database in the file *unix-lpr.txt*. The accompanying shell script *lprsetup.sh* automatically creates some necessary directories and links as well as printcap entries. Use a text editor to adapt the list of device drivers in this script that Ghostscript is supposed to use. Obviously, these drivers must be compiled into the Ghostscript executable. You can also set up additional printcap filters with *lprsetup.sh*. By default, it creates an input filter consisting of a shell script with the actual Ghostscript call.

After executing lprsetup.sh, follow the instructions in *unix-lpr.txt*, i.e., create some links as indicated in the file, integrate the generated *printcap.insert* file into the system printcap, and adjust the new entries to your local setup (serial interface parameters, etc.). The */usr/local/lib/ghostscript/filt* directory contains several links to the unix-lpr.sh file. In this file you have to add the *-I* option if you didn't install Ghostscript in the standard directories.

On System V, Release 4, and related systems you can define print filters for specific file types. The spooler launches these filters for printing on devices which are not supported directly. To define Ghostscript as a filter, change to the directory */etc/lp/fd* and create a file for the printer, say *ljet_ps.fd*:

```
Input types: postscript,ps
Output types: simple
Command: /usr/local/bin/gs -sDEVICE=ljet4 -q
-sOutputFile=- -
```

Integrate this filter in the spool system:

```
lpfilter -f ljet_ps -F /etc/lp/fd/ljet_ps.fd
```

To print a PostScript file, simply declare the file type on the command line:

```
lp -T postscript tiger.ps
```

For serial connections, make sure the backend doesn't change the printer data by using the stty options

```
-opost -cs8 -parenb
```

4.3 Ghostscript as Viewer for a WWW Browser

World Wide Web browsers and many E-mail programs classify files according to MIME types (*Multipurpose Internet Mail Extension*). PostScript files use a MIME type of

```
application/postscript
```

MIME types and corresponding viewers are generally configured in a configuration file or menu. The details vary according to the particular program. Let's take a look at the Windows version of the well-known Netscape WWW browser as an example:

- Launch Netscape Navigator.
- Choose "Options", "General Preferences..." and the submenu "Helpers".
- For the MIME type *application/postscript* enter *ai, ps, eps* in the extensions field (if the entry doesn't exist already). Check "Launch the Application" for "Action" and enter the path of GSview, Ghostview oder Mac GS Viewer as appropriate, e.g., *c:\gs\gsview32.exe*.
- Click "OK" and "Options", "Save Options".

Unix systems generally use the *.mailcap* file for configuring MIME types. Use the following entry for Ghostview (see below for the *-safer* option):

```
application/postscript; ghostview -safer  %s
```

The relation between MIME types and file extensions is controlled by the *.mime.types* file. If the following line doesn't already exist, add it to the *.mime.types* file:

```
application/postscript   ai, ps, eps
```

Like PostScript, you can also configure Ghostscript and GSview as helper application for PDF files. Proceed as above, using the pdf suffix and a MIME type of

```
application/pdf
```

Note that Ghostview doesn't yet directly support PDF.

PostScript files and security. PostScript – being a full-blown programming language – contains operators for modifying and deleting files. This opens a security gap when downloading unknown files. In the worst case, a file pretending to be a harmless PostScript image may delete files from your local hard disk – possibly even with *root* permission! Although there are no known cases of such "trojan horses", you should protect yourself against this kind of attack. Ghostscript's *-dSAFER* option disables critical file operators; the interpreter refuses to open files other than read-only. GSview launches Ghostscript with this option by default, Ghostview for Unix uses the option if launched with the *-safer* option itself.

5 Ghostscript Reference Section

5.1 Command Line Options

On all platforms, Ghostscript evaluates several command line options used to control the interpreter:

```
-h
-?
--help
```
These options cause Ghostscript to print a brief help message and a list of available (i.e., built-in) device drivers on screen.

```
@<filename>
```
Ghostscript reads the specified file and treats its contents the same as the command line. This makes it easier to use command line options on Windows or to use command lines longer than 128 characters on MS-DOS.

```
-- <file.ps> arg1 ...
-+ <file.ps> arg1 ...
```
Ghostscript treats the file name as usual but stores the remaining arguments in an array named ARGUMENTS in userdict. This way PostScript programs can access options and command line arguments.

```
-@ <filename> arg1 ...
```
Same as -- and -+, but expands arguments from *argfile*.

```
-c tokens ...
```
Interprets arguments up to the next "-" as PostScript code and executes them. Each argument must be exactly one token.

`-Dname=token`
`-dname=token`
Defines a name in systemdict with the given definition (equivalent to `/name token def`). This option is mainly used for special names (see below).

`-Dname`
`-dname`
Defines *name* in systemdict with a value of *true*.

`-Sname=string`
`-sname=string`
Defines a name in systemdict with the given string definition (equivalent to `/name (string) def`).

`-q`
(*quiet*) Suppress normal startup messages.

`-f<filename>`
Execute the given file, even if its name begins with a "-" or "@". *-f* provides a way to terminate the token list for *-c*.

`-g<number1>x<number2>`
Equivalent to -dDEVICEWIDTH=*number1* and -dDEVICEHEIGHT= *number2* (see below).

`-r<number>`
`-r<number1>x<number2>`
Equivalent to -dDEVICEXRESOLUTION=*number1* and -dDEVICEYRESOLUTION=*number2* (see below). This is intended for devices that support different horizontal and vertical resolutions, especially dot matrix printers.

`-u<name>`
Undefines a name, cancelling *-d* or *-s*.

`-I<directories>`
Add a list of directories to the search path for initalization and font files. Multiple directories are separated with a semicolon ";" (MS-DOS, Windows, OS/2) or colon ":" (Unix).

`-P`
Ghostscript first searches the current directory for library files. This is the default.

`-P-`
Ghostscript doesn't search the current directory for library files, but uses the search path only.

`-`
Instructs Ghostscript to read standard input from file or pipe (instead of from the keyboard). Ghostscript reads and processes data from standard input and exits. Note that it's not possible to read PDF files from standard input.

Special PostScript names used as switches. The *use.txt* file contains some more options for debugging Ghostscript. A couple of names with special meanings is being interpreted by the PostScript code in Ghostscript's initialization files. They work in a similar way to command line options:

`-dDEVICEWIDTH=<number>`
`-dDEVICEHEIGHT=<number>`
Sets width and height of the device, respectively (in pixels).

`-dDEVICEXRESOLUTION=<number>`
`-dDEVICEYRESOLUTION=<number>`
Sets the device horizontal resp. vertical device resolution in dpi.

`-dCOLORSCREEN`
On devices with at least 150dpi resolution forces the use of separate halftone screens with different angles for the process colors (this produces the best-quality output).

`-dCOLORSCREEN=0`
Uses separate screens with the same frequency and angle for the process colors.

`-dCOLORSCREEN=false`
Forces the use of a single binary screen. If COLORSCREEN is not specified, the default is to use separate screens with different angles if the device has fewer than 5 bits per color.

`-dDISKFONTS`
Causes character outlines in fonts to be loaded from disk on demand only. This slows down text rendering but increases the number of fonts which may be loaded into RAM. This technique is mainly intended for low-memory systems such as MS-DOS.

`-dDITHERPPI=<lpi>`
forces all devices to be considered high-resolution, and forces use of a halftone screen or screens with lpi lines per inch, disregarding the actual device resolution. Reasonable values for lpi are N/5 to N/20, where N is the resolution in dots per inch.

`-dFirstPage=<n>`
Starts interpreting on the given page of a PDF document.

`-dFIXEDMEDIA`
Causes the media size to be fixed after initialization. Pages are scaled or rotated if necessary.

`-dFIXEDRESOLUTION`
Causes the output resolution to be fixed.

`-dLastPage=<n>`
Stops interpreting after the given page of a PDF document.

`-dLOCALFONTS`
This is a compatibility option for certain obsolete fonts. This option makes Ghostscript load type 1 fonts always to local VM.

`-dNOBIND`
Disables the *bind* operator (useful for debugging).

`-dNOCACHE`
Disables the font cache (useful for debugging).

`-dNOCIE`
substitutes DeviceGray and DeviceRGB for CIEBasedA and CIEBasedABC color spaces respectively (useful on very slow systems where color accuracy is less important).

`-dNODISPLAY`
Suppresses normal initialization of the output device. This is useful for debugging and also for PostScript converters that don't produce any screen or printer output (e.g., *ps2ai*).

`-dNOFONTMAP`
Suppresses loading of the Fontmap file.

`-dNOGC`
Disables the level 2 *garbage collector* (useful for debugging).

`-dNOPAUSE`
Disables the prompt and pause at the end of each page. This is useful for driving Ghostscript from another program.

`-dNOPLATFONTS`
Disables platform fonts for X Windows or Microsoft Windows (see section 2.4, »Font Configuration«).

`-dNOPROMPT`
Disables the prompt (but not the pause) at the end of each page. This prevents text and graphics output from being mixed on PC displays.

`-dORIENT1`
Exchanges the meaning of the values 0 and 1 for indicating page orientation with *setpageparams*. This is needed for the PostScript code of certain applications.

`-dSAFER`
Disables the PostScript operators for writing or deleting disk files. This is intended for using Ghostscript as viewer for a Web browser in a secure mode.

`-dSHORTERRORS`
Brackets several error messages with %%[and]%% (as Adobe Interpreters do).

`-dWRITESYSTEMDICT`
Systemdict remains writable. This is necessary for some utility programs that must bypass normal PostScript access protection, such as *font2c* and *pcharstr*.

`-sDEVICE=<device>`
Select the initial output device driver.

`-sFONTMAP=<filename1>:<filename2>...`
Defines one or more file names for the font file mapping table. Several file names are separated by a semicolon ";" under Windows and OS/2 and a colon ":" under Unix.

`-sFONTPATH=<dirname1>:<dirname2>...`
Defines one or more directory names to be searched for font definitions. Several file names are separated by a semicolon ";" under Windows and OS/2 and a colon ":" under Unix.

`-sOutputFile=<filename>`
Selects an output file name or pipe. If the file name contains the characters "%d", Ghostscript replaces the "%d" with the actual page number and creates one file for each page, e.g., *page%d.prn* yields *page1.prn*, *page2.prn* and so on.

On OS/2, Windows 95 and Windows NT you can use printer queue names: `-sOutputFile="\\spool\printername"` sends the output to the named printer queue. If the printer name is missing, Ghostscript prompts for the name of the (connected) printer (Windows) or uses the default queue (OS/2).

On Unix, you can also redirect the output to another program via pipe: `-sOutputFile=\|lp`. The special name "-" for the output file instructs Ghostscript to send the data to its standard output.

`-sPAPERSIZE=<papersize>`
Selects a page size, e.g., *a4*. The file *gs_statd.ps* contains a list of supported page size names.

`-sPSFile=<file.ps>`
Defines the output file name for PDF to PostScript conversion.

`-sSUBSTFONT=<fontname>`
Selects the named font as substitute for all missing fonts. This disables Ghostscript's normal font substitution mechanism.

5.2 Environment Variables

`GS_DEVICE=<device>`
Defines the initial output device driver.

`GS_FONTPATH=<path>`
Specifies a list of directories that should be scanned for fonts at startup (see section 2.4, »Font Configuration«).

`GS_LIB=<path>`
Search path for initialization and font files.

`GS_OPTIONS=<options>`
Defines a list of command line arguments to be processed before the ones specified on the command line. All command line options are also allowed in this environment variable.

`TEMP=<directory>`
Directory name for temporary files. On Windows and OS/2 this variable must point to an existing directory in order to have the printing feature work properly.

`DOS4G=quiet`
Suppresses the usual startup message of the DOS extender for the 386 MS-DOS version.

5.3 X Window System Resources

Ghostscript evaluates several X resources under the program name *ghostscript* and the class name *Ghostscript*. You can use X resources to define user preferences or to activate bug workarounds for several X servers. In the *use.txt* file you can find more information on resources. The table below lists all resources together with their default values.

Name	Class	Default value
background	*Background*	*white*
foreground	*Foreground*	*black*
borderColor	*BorderColor*	*black*
borderWidth	*BorderWidth*	*1*
geometry	*Geometry*	*NULL*
xResolution	*Resolution*	*(calc. from screen size)*
yResolution	*Resolution*	*(calc. from screen size)*
useExternalFonts	*UseExternalFonts*	*true*
useScalableFonts	*UseScalableFonts*	*true*
logExternalFonts	*LogExternalFonts*	*false*
externalFontTolerance	*ExternalFontTolerance*	*10.0*
palette	*Palette*	*Color*
maxGrayRamp	*MaxGrayRamp*	*128*
maxRGBRamp	*MaxRGBRamp*	*5*
maxDynamicColors	*MaxDynamicColors*	*256*
useBackingPixmap	*UseBackingPixmap*	*true*
useXPutImage	*UseXPutImage*	*true*
useXSetTile	*UseXSetTile*	*true*
regularFonts	*RegularFonts*	*(see use.txt)*
symbolFonts	*SymbolFonts*	*(see use.txt)*
dingbatFonts	*DingbatFonts*	*(see use.txt)*

As an example, the resources below select a resolution of 72 dpi (independent of actual screen size and resolution) and disable platform fonts:

```
Ghostscript*useExternalFonts:   false
Ghostscript*xResolution: 72
Ghostscript*yResolution: 72
```

If you want Ghostscript to use the same resource settings every time, it's best to put the resources into a file and load it with the *xrdb* program.

5.4 Configuration Error Messages

Ghostscript issues the usual PostScript error messages (see Chapter 2). Additionally, there are some messages relating to Ghostscript installation or configuration errors instead of PostScript errors.

`/undefinedfilename in (Fontmap)`
Ghostscript can't find the *Fontmap* file, and the GS_FONTPATH environment variable isn't set. Install a Fontmap file or set GS_FONTPATH to point to an appropriate font directory.

`Can't find (or open) initialization file gs_init.ps.`
Ghostscript can't find its main initialization file. Use the *-I* option or the GS_LIB enviroment variable to point Ghostscript to the directory containing the gs_*.ps files.

`Can't find (or can't open) font file xxx`
The Fontmap file contains a font file entry for a nonexistent file or a file that Ghostscript can't open. Under Unix, check the file permissions.

`Substituting font Courier for xxx`
Ghostscript can't find a requested font and substitutes for it with another font. Processing continues.

`Unable to load default font xxx!  Giving up.`
Ghostscript can't find the default font file and hence isn't able to do text output at all. Therefore processing stops. Check the font configuration.

`Can't find library 'libXt.so.6'`
Unix versions of Ghostscript generally are linked dynamically. For this reason, several libraries must be accessible at runtime. Use the *ldd* command to find out which libraries are needed, locate these on your hard disk, and set the LD_LIBRARY_PATH environment variable appropriately. Another solution is to link Ghostscript statically.

`Unknown device: xxx`
Ghostscript has been launched with an unknown device driver name. If you want to use drivers which are not available in the standard configuration, you have to recompile and link Ghostscript with the necessary drivers.

`gs: Interpreter revision (403) does not match gs_init.ps revision (353).`
Ghostscript found an initialization file that doesn't match the program version. Make sure the GS_LIB environment variable or the *-I* command line option don't point to an obsolete Ghostscript version on your hard disk.

6 More Ghostscript Applications

Many file format conversions and other special applications are possible with the help of Ghostscript drivers and auxiliary programs. Some of these applications are not PostScript interpreter tasks at first sight. The descriptions in the following sections provide a summary of the most important of these applications. Usually you can find more detailed information in the appropriate documentation or source files (*.txt, *.ps, *.c).

6.1 Graphics File Formats

Displaying and printing graphics file formats. The utility programs viewgif.ps, viewjpeg.ps, viewpbm.ps, and viewpcx.ps – written in PostScript – display or print raster graphic files in the GIF, JPEG, PBM, or PCX file formats without converting them to PostScript. Launch Ghostscript with the appropriate utility and load a graphics file using one of the procedures viewGIF, viewJPEG, viewpbm, or viewpcx, e.g.,

```
gs viewjpeg.ps
GS>(file.jpg) viewJPEG
```

Converting PostScript to raster graphics formats. Several file format drivers enable Ghostscript to convert PostScript files to TIFF, PBM, PCX, BMP, etc., given the appropriate driver has been compiled into the Ghostscript executable. The Ghostscript call contains the name of the driver and (optionally) the resolution. For example, to create a 600 dpi bitmapped TIFF version of a file, use the following command:

```
gs -sDEVICE=tiffpack -r600 -sOutputFile=page%d.tif
    file.ps -c quit
```

Ghostscripts replaces the "%d" variable in the filename with the actual page number (*page1.tif*, *page2.tif* etc.). By default, the TIFF drivers use a resolution of 204 x 196 dpi (standard fax resolution).

Enhanced rendering with anti-aliasing. A technique called anti-aliasing tries to improve text or graphics rendering by making use of gray levels for smoothing. Anti-aliasing is implemented in a couple of Ghostscript drivers: Windows and OS/2 display drivers and the Portable Graymap Format (PGM) und Portable Pixmap Format (PPM) file format driver. To make use of anti-aliasing, the bit depth must be at least 8 bits. The following command line can be used to convert a PostScript file to PGM with anti-aliasing:

```
gs -sDEVICE=pgm -dTextAlphaBits=4 -sOutputFile=
    file.ppm file.ps -c quit
```

Possible alpha values are 1 (=no anti-aliasing), 2, and 4. Anti-aliasing for graphics can (independent of text anti-aliasing) similarly be achieved using the `-dGraphicsAlphaBits=4` option.

Converting PostScript to PostScript raster graphics. This conversion makes it possible to print PostScript Level 2 files on devices with PostScript Level 1 interpreters. Ghostscript's *psmono* driver produces (not DSC-compliant) PostScript files containing a raster version of the file as a PostScript Level 1 bitmap. Similar to the TIFF conversion you can select the resolution:

```
gs -sDEVICE=psmono -r600 -sOutputFile=file1.ps file2.ps
```

By default, the *psmono* driver uses a resolution of 300 dpi.

Additional features for EPS files. The shell script/batch file *ps2epsi* or *ps2epsi.bat* creates ASCII previews for EPSI files. Usage notes can be found in *ps2epsi.txt*. On Windows and OS/2 many additional EPS functions are possible with GSview, including determining a correct bounding box and creating or deleting preview bitmaps. More details can be found in section 3.5.2, »Adding a Preview Image«.

Converting to the Adobe Illustrator format. As of version 4.03, a special Ghostscript driver for creating Adobe Illustrator format is not yet available. Instead, Ghostscript uses the sophisticated PostScript program *ps2ai.ps*. Although this conversion is bound to some restrictions, in many cases it yields graphics files which may be opened and edited with any Illustrator-compatible program. On Unix, redirect Ghostscript's output to a file. On Windows, OS/2 and the Macintosh it's easier to have Ghostscript create the file directly. To achieve this, change the variable */fout* at the beginning of *ps2ai.ps* to a value of true. Assign the name of the AI file you want to create to the */foutput* variable. Now launch Ghostscript with the converter and the PostScript file to be converted to AI format:

```
gs -dNODISPLAY ps2ai.ps file.ps
```

For EPS files that are missing the showpage operator at the end, you have to type this operator at the Ghostscript prompt before quitting in order to completely render the page. For the conversion to be successful, it is necessary for Ghostscript to have access to all fonts used in the PostScript file. Note that the conversion isn't perfect: there may be problems concerning color ramps and grouped objects in the converted AI files.

6.2 PDF Files

Displaying and printing PDF files. You can list PDF files on Ghostscript's command line just like you can PostScript files because Ghostscript recognizes PDF files automatically. In order to be able to process PDF files, Ghostscript must be configured with the PDF interpreter. By default, this is true on all 32-bit systems except the Macintosh. Ghostscript interprets the printable contents of PDF files only, and ignores hypertext elements (such as links, annotations, and bookmarks) and thumbnails. Although there is no font substitution with Multiple Mas-

ter fonts, Ghostscript replaces a missing font with a similar one and adjusts the metrics of the substituted font to those of the missing font.

Contrary to PostScript files, it's possible to access random pages of PDF files. Therefore you can select a page range for PDF files when launching Ghostscript:

```
gs -dFirstPage=<n> -dLastPage=<m> file.pdf
```

This works for displaying and printing PDF files and for converting them to PostScript (see below).

Converting PDF files to PostScript. Ghostscript is able to convert PDF files back to PostScript Level 2. In fact, if the PDF file doesn't contain compressed raster graphics or color spaces, the generated PostScript files use Level 1 only. The following command line creates the corresponding PostScript file for a PDF file:

```
gs -dNODISPLAY -sPSFile=file.ps file.pdf
```

A little shell script/batch file contained in the Ghostscript distribution makes this even easier:

```
pdf2ps file.pdf file.ps
```

The following additional options are available for PDF to PostScript conversion:

`-dPSBinaryOK`
Allows the generated PostScript files to contain binary data.

`-dPSLevel1`
Generates PostScript Level 1 output.

`-dPSNoProcSet`
Does not include the PostScript prolog (procset) in the output. Note that the prolog is necessary when sending the generated file to a PostScript printer.

Converting PostScript to PDF (distilling). As of version 4.0, Ghostscript is also able to "distill" PostScript files to PDF. Ghostscript must be compiled and linked with the *pdfwrite* device which is the case for Unix, OS/2, and Windows by default. The following command line can be used to distill PostScript files to PDF:

```
gs -q -dNOPAUSE -sDEVICE=pdfwrite -sOutputFile=file.pdf
    file.ps -c quit
```

Again, a shell script/batch file simplifies the process:

```
ps2pdf file.ps file.pdf
```

There are no further options available for PS to PDF conversion. Ghostscript recognizes the *pdfmark* and *setdistillerparams* operators. However, the distiller parameters are ignored with the exception of ASCII85 encoding. Text in any other font than the 13 Acrobat base fonts is converted to bitmaps. Also, there are problems with fonts that use non-standard encodings. Both limitations will be lifted in the future.

PDF files and GSview/Ghostview. Note that because of the idiosyncrasies of the PDF format Ghostscript isn't able to read PDF files from its standard input. Due to this fact, displaying PDF files with Ghostview has its limitations. You can use the following command line for Ghostview on Unix to browse PDF files one page after the other. It's not possible to jump to arbitrary page numbers:

```
ghostview -arguments file.pdf quit.ps
```

GSview for Windows and OS/2 has already been adapted to processing PDF files. GSview handles PDF files just as PostScript files. Jumping to an arbitrary page in the file is also possible.

Extracting text from PDF files. Using the *ps2ascii.ps* or *pstotext* utilities you can extract text from PDF files (see below for details).

6.3 Printing and Extracting Text

Printing text files. The *gslp.ps* utility program implements a lineprinter emulation, i.e., printing simple text files with PostScript commands. This includes some formating such as headers and footers, page numbers and tabs. Ghostscript optionally creates PostScript code or renders the text on screen or printer. Details on using the emulator and possible options may be found in the *gslp.ps* file. Example for printing a text file on a laserjet printer:

```
gs -q -sDEVICE=laserjet -r300 -dNOPAUSE -- gslp.ps
    file.txt
```

You may use several options following the file name *gslp.ps*. The *-p* option creates a PostScript file instead of rendering the text on screen or printer:

```
gs -q -dNOPAUSE -- gslp.ps -p file.ps file.txt
```

Extracting text from PostScript and PDF files. The text extraction feature is a counterpart of the lineprinter emulation as it extracts the textual contents from a PostScript file. There are at least two options for extracting text with Ghostscript: the *ps2ascii.ps* PostScript program and the more sophisticated *pstotext* package.

ps2ascii.ps is included in the Ghostscript distribution. Depending on the -dSIMPLE option, the utility creates simple or complex output. Simple output consists of text only, whereas complex output additionally contains information about font type and string positions. The positions are given in tenth of a point. More details may be found at the beginning of *ps2ascii.ps*, some usage samples in the batch file *ps2ascii.bat*. A typical command line:

```
gs -dNODISPLAY -dNOBIND -dWRITESYSTEMDICT -dSIMPLE
    ps2ascii.ps file.ps -c quit >file.txt
```

Note that there is good reason to call the utility ps*2ascii*: If the PostScript file contains any special characters (German umlauts, for example), they don't make their way into the output file but get substituted with two-character sequences of ASCII symbols.

The *pstotext* package is available as a Ghostscript add-on from the following URL:

```
http://www.research.digital.com/SRC/virtualpaper/
     pstotext.html
```

The pstotext code is also included in the GSview distribution as a separate DLL. Simply use GSview's "Edit", "Text Extract..." feature to make use of it. If you check "Options", "Quick Text", a different method will be used for text extraction which is less accurate but faster. Note that unlike ps2ascii.ps, pstotext preserves umlauts and other special characters.

6.4 Font Tools

Creating AFM files. The *printafm.ps* utility creates an AFM file for any given PostScript font. The metrics files created only contain character width information; kerning data are missing because they cannot be reconstructed from the font. To use the utility, enter the PostScript font name at the end of *printafm.ps*. The program print the AFM file to its standard output which you have to redirect to a file. If the font isn't configured for Ghostscript (via the Fontmap file or the GS_FONTPATH environment variable), enter the font file name on the Ghostscript command line (unfortunately, the redirection doesn't work on Windows or on the Mac):

```
gs -dNODISPLAY -q fontfile -- printafm.ps fontname
    >fontname.afm
```

Printing font tables. The *prfont.ps* utility prints character tables for any given PostScript font. It first prints all characters contained in the encoding vector of the font, then uncoded characters. If the font isn't configured in Ghostscript, enter the font file name on the Ghostscript command line. Then type the font name and the procedure name at the Ghostscript prompt:

```
gs fontfile prfont.ps
GS>/FontName DoFont
```