NAME

pbmplus - enhanced portable bitmap toolkit

DESCRIPTION

The *pbmplus* toolkit allows conversions between image files of different format. By means of using common intermediate formats, only $2 \times N$ conversion filters are required to support N distinct formats, instead of the N^2 which would be required to convert directly between any one format and any other. The package also includes simple tools for manipulating portable bitmaps.

The package consists of four upwardly compatible sections:

pbm Supports monochrome bitmaps (1 bit per pixel).

pgm Supports grayscale images. Reads either *pbm* or *pgm* formats and writes *pgm* format.

ppm Supports full-color images. Reads either *pbm*, *pgm*, or *ppm* formats, writes *ppm* format.

pnm Supports content-independent manipulations on any of the three formats listed above, as well as external formats having multiple types. Reads either *pbm*, *pgm*, or *ppm* formats, and generally writes the same type as it read (whenever a *pnm* tool makes an exception and "promotes" a file to a higher format, it informs the user).

DESCRIPTION OF CONTENTS

atktopbm convert Andrew Toolkit raster object to portable bitmap

brushtopbm convert Xerox doodle brushes to portable bitmap

cmuwmtopbm convert CMU window manager format to portable bitmap

g3topbm convert Group 3 FAX to portable bitmap

icontopbm convert Sun icon to portable bitmap

gemtopbm convert GEM .img format to portable bitmap

macptopbm convert MacPaint to portable bitmap

mgrtopbm convert MGR format to portable bitmap

pbmmerge merge wrapper routine

pbmto10x convert portable bitmap to Gemini 10x printer graphics

pbmtoascii convert portable bitmap to ASCII graphic form

pbmtoatk convert portable bitmap to Andrew Toolkit raster object

pbmtobbnbg convert portable bitmap to BBN BitGraph graphics

pbmtocmuwm convert portable bitmap to CMU window manager format

pbmtoepson convert portable bitmap to Epson printer graphics

pbmtog3 convert portable bitmap to Group 3 FAX

pbmtogem convert portable bitmap into GEM .img file

pbmtogo convert portable bitmap to GraphOn graphics

pbmtoicon	convert portable bitmap to Sun icon
pbmtolj	convert portable bitmap to HP LaserJet graphics
pbmtomacp	convert portable bitmap to MacPaint
pbmtomgr	convert portable bitmap to MGR format
pbmtopi3	convert portable bitmap to Atari Degas .pi3
pbmtoplot	convert portable bitmap into Unix plot(5) file
pbmtoptx	convert portable bitmap to Printronix graphics
pbmtoxbm	convert portable bitmap to X11 bitmap
pbmtox10bm	convert portable bitmap to X10 bitmap
pbmtoybm	convert portable bitmap into Bennet Yee "face" file
pbmtozinc	convert portable bitmap to Zinc Interface Library icon
pbmlife	apply Conway's rules of Life to a portable bitmap
pbmmake	create a blank bitmap of a specified size and color
pbmmask	create a mask bitmap from a regular bitmap
pbmreduce	reduce a portable bitmap N times, using Floyd-Steinberg
pbmtext	render text into a bitmap
pbmupc	create a Universal Product Code bitmap
pi3topbm	convert Atari Degas .pi3 to portable bitmap
xbmtopbm	convert X10 or X11 bitmap to portable bitmap
ybmtopbm	convert Bennet Yee "face" file into portable bitmap
asciitopgm	convert ASCII graphics into a portable graymap
fstopgm	convert Usenix FaceSaver^{tm} format to portable graymap
hipstopgm	convert HIPS format to portable graymap
lispmtopgm	convert a Lisp Machine bitmap file into pgm format
pgmbentley	Bentleyize a portable graymap
pgmcrater	create cratered terrain by fractal forgery
pgmedge	edge-detect a portable graymap
pgmenhance	edge-enhance a portable graymap
pgmhist	print a histogram of the values in a portable graymap
pgmkernel	generate a convolution kernel
pgmmerge	merge wrapper routine
pgmnoise	create a graymap made up of white noise
pgmnorm	normalize contrast in a portable graymap

pgmoil	turn a portable graymap into an oil painting
pgmramp	generate a grayscale ramp
pgmtexture	calculate textural features on a portable graymap
pgmtofits	convert portable graymap to FITS format
pgmtofs	convert portable graymap to Usenix FaceSaver^{tm} format
pgmtolispm	convert a portable graymap into Lisp Machine format
pgmtopbm	convert portable graymap to portable bitmap
psidtopgm	convert PostScript "image" data to portable graymap
rawtopgm	convert raw grayscale bytes to portable graymap
giftoppm	convert GIF to portable pixmap
gouldtoppm	convert Gould scanner file to portable pixmap
ilbmtoppm	convert IFF ILBM to portable pixmap
imgtoppm	convert Img-whatnot to portable pixmap
mtvtoppm	convert MTV ray-tracer output to portable pixmap
pcxtoppm	convert PC Paintbrush format to portable pixmap
pgmtoppm	colorize a portable graymap into a portable pixmap
pi1toppm	convert Atari Degas .pi1 to portable pixmap
picttoppm	convert Macintosh PICT to portable pixmap
pjtoppm	convert HP PaintJet file to portable pixmap
ppmchange	change all pixels of one color to another in a portable pixmap
ppmdim	dim a portable pixmap down to total blackness
ppmdist	simplistic grayscale assignment for machine generated, color images
ppmdither	ordered dither for color images
ppmflash	brighten a picture up to complete white-out
ppmforge	fractal forgeries of clouds, planets, and starry skies
ppmhist	print a histogram of a portable pixmap
ppmmake	create a pixmap of a specified size and color
ppmmerge	merge wrapper routine
ppmmix	blend together two portable pixmaps
ppmpat	create a pretty pixmap
ppmquant	quantize colors down to a specified number
ppmquantall	script to run ppmquant on a set of pixmaps
ppmrelief	run a Laplacian Relief filter on a portable pixmap

ppmshift	shift lines of a portable pixmap left or right by a random amount
ppmspread	displace a portable pixmap's pixels by a random amount
ppmtoacad	convert portable pixmap to AutoCAD database or slide
ppmtoicr	convert portable pixmap to NCSA ICR graphics
ppmtoilbm	convert portable pixmap to IFF ILBM
ppmtomitsu	convert a portable pixmap to a Mitsubishi S340-10 file
ppmtopcx	convert portable pixmap to PC Paintbrush format
ppmtopgm	convert portable pixmap to portable graymap
ppmtopi1	convert portable pixmap to Atari Degas .pi1
ppmtopict	convert portable pixmap to Macintosh PICT
ppmtopj	convert portable pixmap to HP PaintJet file
ppmtopuzz	convert portable pixmap to X11 "puzzle" file
ppmtorgb3	separate a portable pixmap into three portable graymaps
ppmtosixel	convert portable pixmap to DEC sixel format
ppmtotga	convert portable pixmap to TrueVision Targa file
ppmtouil	convert portable pixmap to Motif UIL icon file
ppmtoxpm	convert portable pixmap to XPM format
ppmtoyuv	convert portable pixmap to Abekas YUV format
qrttoppm	convert QRT ray-tracer output to portable pixmap
rawtoppm	convert raw RGB bytes to portable pixmap
rgb3toppm	combine three portable graymaps into one portable pixmap
sldtoppm	convert an AutoCAD slide file into a portable pixmap
spctoppm	convert Atari compressed Spectrum to portable pixmap
sputoppm	convert Atari uncompressed Spectrum to portable pixmap
tgatoppm	convert TrueVision Targa file to portable pixmap
ximtoppm	convert Xim to portable pixmap
xvminitoppm	convert a XV "thumbnail" picture to PPM
xpmtoppm	convert XPM format to portable pixmap
yuvtoppm	convert Abekas YUV format to portable pixmap
anytopnm	script to attempt to convert any format to P?M
pnmalias	antialias a portable anyumap.
pnmarith	perform arithmetic on two portable anymaps
pnmcat	concatenate portable anymaps

pnmconvol	general MxN convolution on a portable anymap
pnmcrop	crop all like-colored borders off a portable anymap
pnmcut	select a rectangular region from a portable anymap
pnmdepth	change the maxval in a portable anymap
pnmenlarge	enlarge a portable anymap N times
pnmfile	describe a portable anymap
pnmflip	perform one or more flip operations on a portable anymap
pnmgamma	perform gamma correction on a portable anymap
pnmindex	script to build a visual index of a bunch of anymaps
pnminvert	invert a portable anymap
pnmmargin	script to add a margin to a portable anymap
pnmmerge	merge wrapper routine
pnmnoraw	force a portable anymap into ASCII format
pnmpaste	paste a rectangle into a portable anymap
pnmrotate	rotate a portable anymap
pnmscale	scale a portable anymap
pnmshear	shear a portable anymap
pnmsmooth	script that uses pnmconvol to smooth a anymap
pnmtile	replicate a portable anymap into a specified size
pnmtofits	convert a portable anymap into FITS format
pnmtogif	convert portable anymap to GIF
pnmtops	convert portable anymap to PostScript
pnmtorast	convert portable anymap to Sun raster file
pnmtotiff	convert portable anymap to TIFF file
pnmtoxwd	convert portable anymap to X11 window dump
fitstopnm	fitstopnm - convert a FITS file into a portable anymap
pstopnm	script to convert PS to portable anymap with GhostScript
rasttopnm	convert Sun raster file to portable anymap
tifftopnm	convert TIFF file to portable anymap
xwdtopnm	convert X10 or X11 window dump to portable anymap

SEE ALSO

There are a number of related image-manipulation tools:

IM Raster Toolkit A portable and efficient format toolkit. The format supports pixels of arbitrary channels, components, and bit precisions, while allowing compression and machine byte-order independence. Support for image manipulation, digital halftoning, and format conversion. Previously distributed on tape c/o the University of Waterloo (an *ftp* version is to appear later). Author: Alan Paeth (awpaeth@watcgl.uwaterloo.ca).

Utah RLE Toolkit Conversion and manipulation package, similar to *pbmplus*. Available via *ftp* as *cs.utah.edu: pub/toolkit-2.0.tar.Z* and *ucsd.edu: graphics-/utah-raster-toolkit.tar.Z*.

Fuzzy Pixmap Manipulation Conversion and manipulation package, similar to *pbmplus*. Version 1.0 available via *ftp* as *nl.cs.cmu.edu: /usr/mlm/ftp/fbm.tar.Z*, *uunet.uu.net: pub/fbm.tar.Z*, and *ucsd.edu: graphics/fbm.tar.Z*. Author: Michael Mauldin (mlm@nl.cs.cmu.edu).

Img Software Set Reads and writes its own image format, displaying results on an X11 screen, and does some image manipulations. Version 1.3 is available via *ftp* as *ftp.x.org: contrib/img_1.3.tar.Z*, and *venera.isi.edu: pub/img_1.3.tar.Z*, along with a large collection of color images. Author: Paul Raveling (raveling@venera.isi.edu).

Xim Reads and writes its own image format, displays on an X11 screen, and does some image manipulations. Available in your nearest X11R4 source tree as *contrib/clients/xim*. A more recent version is available via ftp from *video.mit.edu*. It uses X11R4 and the OSF/Motif toolkit to provide basic interactive image manipulation and reads/writes GIF, xwd, xbm, tiff, rle, xim, and other formats. Author: Philip R. Thompson.

xloadimage Reads in images in various formats and displays them on an X11 screen. Available via *ftp* as *ftp.x.org: contrib/xloadimage∗*, and in your nearest *comp.sources.x* archive. Author: Jim Frost (madd@std.-com).

TIFF Software Nice portable library for reading and writing TIFF files, plus a few tools for manipulating them and reading other formats. Available via *ftp* as *sgi.com: graphics/tiff/∗.tar.Z* or *uunet.uu.net: graphics/tiff.tar.Z*. Author: Sam Leffler (sam@sgi.com).

ALV A Sun-specific image toolkit. Version 2.0.6 posted to *comp.sources.-sun* on 11 December 1989. Also available via email to *alv-users-request @cs.bris.ac.uk*.

popi An image manipulation language. Version 2.1 posted to *comp.-sources.misc* on 12 December 1989.

ImageMagick, X11 package for display and interactive manipulation of images. Uses its own format (MIFF), and includes some converters. Available via *ftp* as *ftp.x.org: contrib/ImageMagick.∗.tar.Z*.

Khoros — A huge (~100 meg) graphical development environment based on X11R4. Components include a visual programming language, code generators for extending the visual language and adding new application packages to the system, an interactive user interface editor, an interactive image display package, an extensive library of image and signal processing routines, and 2D/3D plotting packages. Available via *ftp* as *pprg.unm.edu: pub/khoros/**.

JPEG package — JPEG is a a standardized compression method for full-color and gray-scale images of "real-world" scenes; this experimental package includes programs to compress gif and ppm format files to JPEG format (*cjpeg(1L)),* and to decompress them (*djpeg(1L)).* Available by *ftp* as *uunet.uu.net: graphics/jpeg/jpegsrc.v4.tar.Z.*

libpbm(3), libpgm(3), libpnm(3), libppm(3), pbm(5), pgm(5), pnm(5), ppm(5), rasterfile(1)

AUTHOR

Feedback and questions are welcome. Please send them to:

jef@netcom.com
jef@well.sf.ca.us
apple!well!jef

When sending bug reports, always include the output from running any pbmplus program with the -version flag, including descriptions of the type of system you are on, the compiler you use, and whether you are using Makefiles or Imakefiles.

When suggesting new formats or features, please include whatever documentation you have, and a uuencoded sample. The response time will depend upon my schedule and the complexity of the task; if you need it right away, or it is a complicated job, you might consider paying me.

The Usenet newsgroup *alt.graphics.pixutils* is a forum for discussion of image conversion and editing packages. Posting queries there may be better than mailing them to me, since it allows other people to help provide answers.

NAME

asciitopgm - convert ASCII graphics into a portable graymap

SYNOPSIS

asciitopgm *[*-d divisor*] height width* [*asciifile*]

DESCRIPTION

Reads ASCII data as input. Produces a portable graymap with pixel values which are an approximation of the "brightness" of the ASCII characters, assuming black-on-white printing. In other words, a capital M is very dark, a period is ver light, and a space is white. Input lines which are fewer than *width* characters are automatically padded with spaces.

The *divisor* argument is a floating-point number by which the output pixels are divided; the default value is 1.0. This can be used to adjust the brightness of the graymap: for example, if the image is too dim, reduce the divisor.

In keeping with (I believe) Fortran line-printer conventions, input lines beginning with a + (plus) character are assumed to "overstrike" the previous line, allowing a larger range of gray values.

This tool contradicts the message in the *pbmtoascii* manual: "Note that there is no asciitopbm tool - this transformation is one-way."

BUGS

The table of ASCII-to-grey values is subject to interpretation, and, of course, depends on the typeface intended for the input.

SEE ALSO

pbmtoascii(1), pgm(5)

AUTHOR

Wilson H. Bent. Jr. (whb@usc.edu)

NAME

atktopbm - convert Andrew Toolkit raster object to portable bitmap

SYNOPSIS

atktopbm [*atkfile*]

DESCRIPTION

Reads an Andrew Toolkit raster object as input. Produces a portable bitmap as output.

SEE ALSO

pbmtoatk(1), pbm(5)

AUTHOR

NAME

bioradtopgm - convert a Biorad confocal file into a portable graymap

SYNOPSIS

bioradtopgm [**-image#**] [*imagedata*]

DESCRIPTION

Reads a Biorad confocal file as input. Produces a portable graymap as output. If the resulting image is upside down, run it through **pnmflip -tb .**

OPTIONS

-image# A Biorad image file may contain more than one image. With this flag, you can specify which image to extract (only one at a time). The first image in the file has number zero. If no image number is supplied, only information about the image size and the number of images in the input is printed out. No output is produced.

BUGS

A Biorad image may be in word format. If PbmPlus is not compiled with the "BIGGRAYS" flag, word files can not be converted. See the Makefile.

SEE ALSO

pgm(5), pnmflip(1)

AUTHORS

©1993 by Oliver Trepte (oliver@fysik4.kth.se).

NAME

bmptoppm – convert a BMP file into a portable pixmap

SYNOPSIS

bmptoppm [*bmpfile*]

DESCRIPTION

Reads a Microsoft Windows or OS/2 BMP file as input. Produces a portable pixmap as output.

SEE ALSO

ppmtobmp(1), ppm(5)

AUTHOR

NAME

brushtopbm - convert a doodle brush file into a portable bitmap

SYNOPSIS

brushtopbm [*brushfile*]

DESCRIPTION

Reads a Xerox doodle brush file as input. Produces a portable bitmap as output.

Note that there is currently no pbmtobrush tool.

SEE ALSO

pbm(5)

AUTHOR

NAME

cmuwmtopbm - convert a CMU window manager bitmap into a portable bitmap

SYNOPSIS

cmuwmtopbm [*cmuwmfile*]

DESCRIPTION

Reads a CMU window manager bitmap as input. Produces a portable bitmap as output.

SEE ALSO

pbmtocmuwm(1), pbm(5)

AUTHOR

NAME

fitstopnm - convert a FITS file into a portable anymap

SYNOPSIS

fitstopnm [**-image** *N*] [**-noraw**] [**-scanmax**] [**-printmax**] [**-min** *f*] [**-max** *f*] [*FITSfile*]

DESCRIPTION

Reads a FITS file as input. Produces a portable pixmap if the FITS file consists of 3 image planes (NAXIS = 3 and NAXIS3 = 3), a portable graymap if the FITS file consists of 2 image planes (NAXIS = 2), or whenever the –**image** flag is specified. The results may need to be flipped top for bottom; if so, just pipe the output through **pnmflip -tb.**

OPTIONS

The **-image** option is for FITS files with three axes. The assumption is that the third axis is for multiple images, and this option lets you select which one you want.

Flags **-min** and **-max** can be used to override the min and max values as read from the FITS header or the image data if no DATAMIN and DATAMAX keywords are found. Flag **-scanmax** can be used to force the program to scan the data even when DATAMIN and DATAMAX are found in the header. If **-printmax** is specified, the program will just print the min and max values and quit. Flag **-noraw** can be used to force the program to produce an ASCII portable anymap.

The program will tell what kind of anymap is writing. All flags can be abbreviated to their shortest unique prefix.

REFERENCES

FITS stands for Flexible Image Transport System. A full description can be found in Astronomy & Astrophysics Supplement Series 44 (1981), page 363.

SEE ALSO

pnmtofits(1), pgm(5), pnmflip(1)

AUTHOR

NAME

fstopgm - convert a Usenix FaceSaver(tm) file into a portable graymap

SYNOPSIS

fstopgm [*fsfile*]

DESCRIPTION

Reads a Usenix FaceSaver(tm) file as input. Produces a portable graymap as output.

FaceSaver(tm) files sometimes have rectangular pixels. While *fstopgm* won't re-scale them into square pixels for you, it will give you the precise *pnmscale* command that will do the job. Because of this, reading a FaceSaver(tm) image is a two-step process. First you do:

```
fstopgm > /dev/null
```

This will tell you whether you need to use *pnmscale.* Then use one of the following pipelines:

```
fstopgm | pgmnorm
fstopgm | pnmscale -whatever | pgmnorm
```

To go to PBM, you want something more like one of these:

```
fstopgm | pnmenlarge 3 | pgmnorm | pgmtopbm
fstopgm | pnmenlarge 3 | pnmscale <whatever> | pgmnorm | pgmtopbm
```

You want to enlarge when going to a bitmap because otherwise you lose information; but enlarging by more than 3 does not look good.

FaceSaver is a registered trademark of Metron Computerware Ltd. of Oakland, CA.

SEE ALSO

pgmtofs(1), pgm(5), pgmnorm(1), pnmenlarge(1), pnmscale(1), pgmtopbm(1)

AUTHOR

NAME

g3topbm - convert a Group 3 fax file into a portable bitmap

SYNOPSIS

g3topbm [**-kludge**] [**-reversebits**] [**-stretch**] [*g3file*]

DESCRIPTION

Reads a Group 3 fax file as input. Produces a portable bitmap as output.

OPTIONS

-kludge Tells *g3topbm* to ignore the first few lines of the file; sometimes fax files have some junk at the beginning.

-reversebits Tells *g3topbm* to interpret bits least-significant first, instead of the default most-significant first. Apparently some fax modems do it one way and others do it the other way. If you get a whole bunch of "bad code word" messages, try using this flag.

-stretch Tells *g3topbm* to stretch the image vertically by duplicating each row. This is for the low-quality transmission mode.

All flags can be abbreviated to their shortest unique prefix.

REFERENCES

The standard for Group 3 fax is defined in CCITT Recommendation T.4.

BUGS

Probably.

SEE ALSO

pbmtog3(1), pbm(5)

AUTHOR

©1989 by Paul Haeberli (paul@manray.sgi.com).

NAME

gemtopbm - convert a GEM .img file into a portable bitmap

SYNOPSIS

gemtopbm [**-d**] *gemfile*

DESCRIPTION

Reads a GEM .img file as input. Produces a portable bitmap as output.

OPTIONS

-d Produce output describing the contents of the .img file.

BUGS

Does not support file containing more than one plane. Can't read from standard input.

SEE ALSO

pbmtogem(1), pbm(5)

AUTHOR

©1988 Diomidis D. Spinellis (dds@cc.ic.ac.uk).

NAME

giftopnm - convert a GIF file into a portable anymap

SYNOPSIS

giftopnm [**-verbose**] [**-comments**] [**-image** *N*] [*GIFfile*]

DESCRIPTION

Reads a GIF file for input, and outputs portable anymap.

OPTIONS

-verbose Produces verbose output about the GIF file input.

-comments Only outputs GIF89 comment fields.

-image Output the specified gif image from the input gif archive (where *N* is '1', '2', '20'...). Normally there is only one image per file, so this option is not needed.

All flags can be abbreviated to their shortest unique prefix.

BUGS

This does not correctly handle the Plain Text Extension of the GIF89 standard, since I did not have any example input files containing them.

SEE ALSO

ppmtogif(1), ppm(5)

AUTHOR

NAME

gouldtoppm - convert Gould scanner file into a portable pixmap

SYNOPSIS

gouldtoppm [*gouldfile*]

DESCRIPTION

Reads a file produced by the Gould scanner as input. Produces a portable pixmap as output.

SEE ALSO

ppm(5)

AUTHOR

NAME

hipstopgm - convert a HIPS file into a portable graymap

SYNOPSIS

hipstopgm [*hipsfile*]

DESCRIPTION

Reads a HIPS file as input. Produces a portable graymap as output.

If the HIPS file contains more than one frame in sequence, hipstopgm will concatenate all the frames vertically.

HIPS is a format developed at the Human Information Processing Laboratory, NYU.

SEE ALSO

pgm(5)

AUTHOR

NAME

icontopbm - convert a Sun icon into a portable bitmap

SYNOPSIS

icontopbm [*iconfile*]

DESCRIPTION

Reads a Sun icon as input. Produces a portable bitmap as output.

SEE ALSO

pbmtoicon(1), pbm(5)

AUTHOR

NAME

ilbmtoppm - convert an ILBM file into a portable pixmap

SYNOPSIS

ilbmtoppm [**-verbose**] [*ILBMfile*]

DESCRIPTION

Reads an IFF ILBM file as input. Produces a portable pixmap as output. Supported ILBM types are:

Normal ILBMs with 1-16 planes.

Amiga Extra-Halfbrite (EHB)

Amiga Hold-and-Modify (HAM) with 3-16 planes.

24 bit.

Color map (BMHD + CMAP chunk only, nPlanes = 0).

Unofficial direct color.	1-16 planes for each color component.
Chunks used:	BMHD, CMAP, CAMG (only HAM & EHB flags used), BODY unofficial DCOL chunk to identify direct color ILBM
Chunks ignored:	GRAB, DEST, SPRT, CRNG, CCRT, CLUT, DPPV, DRNG, EPSF

Other chunks (ignored but displayed in verbose mode): NAME, AUTH, (c), ANNO, DPI

Unknown chunks are skipped.

OPTIONS

-verbose Give some informaton about the ILBM file.

BUGS

Probably.

REFERENCES

Amiga ROM Kernel Reference Manual - Devices (3rd Ed.) Addison Wesley, ISBN 0–201–56775–X

SEE ALSO

ppm(5), ppmtoilbm(1)

AUTHORS

Modified June 1993 by Ingo Wilken (Ingo.Wilken@informatik.uni-oldenburg.de)

NAME

imgtoppm - convert an Img-whatnot file into a portable pixmap

SYNOPSIS

imgtoppm [*imgfile*]

DESCRIPTION

Reads an Img-whatnot file as input. Produces a portable pixmap as output. The Img-whatnot toolkit is available for FTP on venera.isi.edu, along with numerous images in this format.

SEE ALSO

ppm(5)

AUTHOR

Based on a simple conversion program posted to comp.graphics by Ed Falk.

NAME

libpbm - functions to support portable bitmap programs

SYNOPSIS

```
#include <pbm.h>
cc ... libpbm.a
```

DESCRIPTION - PACKAGE-WIDE ROUTINES

int pm_keymatch(char* str, char* keyword, int minchars)

Does a case-insensitive match of **str** against **keyword**. **str** can be a leading sunstring of **keyword**, but at least **minchars** must be present.

int pm_maxvaltobits(int maxval)
int pm_bitstomaxval(int bits)

Convert between a maxval and the minimum number of bits required to hold it.

void pm_message(char* fmt, ...)

printf() style routine to write an informational message.

void pm_error(char* fmt, ...)

printf() style routine to write an error message and abort.

void pm_usage(char* usage)

Write a usage message. The string should indicate what arguments are to be provided to the program.

FILE* pm_openr(char* name)

Open the given file for reading, with appropriate error checking. A filename of "-" is taken as equivalent to stdin.

FILE* pm_openw(char* name)

Open the given file for writing, with appropriate error checking.

void pm_close(FILE* fp)

Close the file descriptor, with appropriate error checking.

```
int pm_readbigshort( FILE* in, short* sP )
int pm_writebigshort( FILE* out, short s )
int pm_readbiglong( FILE* in, long* lP )
int pm_writebiglong( FILE* out, long l )
int pm_readlittleshort( FILE* in, short* sP )
int pm_writelittleshort( FILE* out, short s )
int pm_readlittlelong( FILE* in, long* lP )
int pm_writelittlelong( FILE* out, long l )
```

Routines to read and write short and long ints in either big- or little-endian byte order.

DESCRIPTION - PBM-SPECIFIC ROUTINES

```
typedef ... bit;
#define PBM_WHITE ...
#define PBM_BLACK ...
```

each **bit** should contain only the values of **PBM_WHITE** or **PBM_BLACK**.

```
#define PBM_FORMAT ...
#define RPBM_FORMAT ...
#define PBM_TYPE PBM_FORMAT
#define PBM_FORMAT_TYPE(f) ...
```

For distinguishing different file formats and types.

void pbm_init(int* argcP, char* argv[])

All PBM programs must call this routine.

bit** pbm_allocarray(int cols, int rows)

Allocate an array of bits.

bit* pbm_allocrow(int cols)

Allocate a row of the given number of bits.

void pbm_freearray(bit** bits, int rows)

Free the array allocated with **pbm_allocarray()** containing the given number of rows.

void pbm_freerow(bit* bitrow)

Free a row of bits.

void pbm_readpbminit(FILE* fp, int* colsP, int* rowsP, int* formatP)

Read the header from a PBM file, filling in the rows, cols and format variables.

void pbm_readpbmrow(FILE* fp, bit* bitrow, int cols, int format)

Read a row of bits into the bitrow array. Format and cols were filled in by **pbm_readpbminit()**.

bit** pbm_readpbm(FILE* fp, int* colsP, int* rowsP)

Read an entire bitmap file into memory, returning the allocated array and filling in the rows and cols variables. This function combines **pbm_readpbminit()**, **pbm_allocarray()** and **pbm_readpbmrow()**.

char* pm_read_unknown_size(FILE* fp, long* nread)

Read an entire file or input stream of unknown size to a buffer. Allocate memory more memory as needed. The calling routine has to free the allocated buffer with **free()**. **pm_read_unknown_size()** returns a pointer to the allocated buffer. The **nread** argument returns the number of bytes read.

void pbm_writepbminit(FILE* fp, int cols, int rows, int forceplain)

Write the header for a portable bitmap file. The forceplain flag forces a plain-format file to be written, as opposed to a raw-format one.

void pbm_writepbmrow(FILE* fp, bit* bitrow, int cols, int forceplain)

Write a row from a portable bitmap.

void pbm_writepbm(FILE* fp, bit** bits, int cols, int rows, int forceplain)

Write the header and all data for a portable bitmap. This function combines **pbm_writepbminit()** and **pbm_writepbmrow()**.

SEE ALSO

libpgm(3), libppm(3), libpnm(3)

AUTHOR

NAME

libpgm - functions to support portable graymap programs

SYNOPSIS

```
#include <pgm.h>
cc ... libpgm.a libpbm.a
```

DESCRIPTION

```
typedef ... gray;
#define PGM_MAXMAXVAL ...
extern gray pgm_pbmmaxval;
```

Each **gray** should contain only the values between **0** and **PGM_MAXMAXVAL**. **pgm_pbmmaxval** is the maxval used when a PGM program reads a PBM file. Normally it is 1; however, for some programs, a larger value gives better results.

```
#define PGM_FORMAT ...
#define RPGM_FORMAT ...
#define PGM_TYPE PGM_FORMAT
int PGM_FORMAT_TYPE( int format )
```

For distinguishing different file formats and types.

```
void pgm_init( int* argcP, char* argv[] )
```

All PGM programs must call this routine.

```
gray** pgm_allocarray( int cols, int rows )
```

Allocate an array of grays.

```
gray* pgm_allocrow( int cols )
```

Allocate a row of the given number of grays.

void pgm_freearray(gray** grays, int rows)

Free the array allocated with **pgm_allocarray()** containing the given number of rows.

void pgm_freerow(gray* grayrow)

Free a row of grays.

void pgm_readpgminit(FILE* fp, int* colsP, int* rowsP, gray* maxvalP, int* formatP)

Read the header from a PGM file, filling in the rows, cols, maxval and format variables.

void pgm_readpgmrow(FILE* fp, gray* grayrow, int cols, gray maxval, int format)

Read a row of grays into the grayrow array. Format, cols, and maxval were filled in by **pgm_readpgminit()**.

gray** pgm_readpgm(FILE* fp, int* colsP, int* rowsP, gray* maxvalP)

Read an entire graymap file into memory, returning the allocated array and filling in the rows, cols and maxval variables. This function combines **pgm_readpgminit()**, **pgm_allocarray()** and **pgm_readpgmrow()**.

void pgm_writepgminit(FILE* fp, int cols, int rows, gray maxval, int forceplain)

Write the header for a portable graymap file. The forceplain flag forces a plain-format file to be written, as opposed to a raw-format one.

void pgm_writepgmrow(FILE* fp, gray* grayrow, int cols, gray maxval, int forceplain)

Write a row from a portable graymap.

void pgm_writepgm(FILE* fp, gray** grays, int cols, int rows, gray maxval, int forceplain)

Write the header and all data for a portable graymap. This function combines **pgm_writepgminit()** and **pgm_writepgmrow()**.

SEE ALSO

libpbm(3), libppm(3), libpnm(3)

AUTHOR

NAME

libpnm - functions to support portable anymap programs

SYNOPSIS

```
#include <pnm.h>
cc ... libpnm.a libppm.a libpgm.a libpbm.a
```

DESCRIPTION

```
typedef ... xel;
typedef ... xelval;
#define PNM_MAXMAXVAL ...
extern xelval pnm_pbmmaxval;
```

Each **xel** contains three **xelval**s, each of which should contain only the values between **0** and **PNM_MAXMAXVAL**. **pnm_pbmmaxval** is the maxval used when a PNM program reads a PBM file. Normally it is 1; however, for some programs, a larger value gives better results.

xelval PNM_GET1(xel x)

This macro extracts a single value from an xel, when you know it's from a PBM or PGM file. When it's from a PPM file, use **PPM_GETR()**, **PPM_GETG()**, and **PPM_GETB()**.

void PNM_ASSIGN1(xel x, xelval v)

This macro assigns a single value to an xel, when you know it's from a PBM or PGM file. When it's from a PPM file, use **PPM_ASSIGN()**.

int PNM_EQUAL(xel x, xel y)

This macro checks two xels for equality.

int PNM_FORMAT_TYPE(int format)

For distinguishing different file types.

void pnm_init(int* argcP, char* argv[])

All PNM programs must call this routine.

xel** pnm_allocarray(int cols, int rows)

Allocate an array of xels.

xel* pnm_allocrow(int cols)

Allocate a row of the given number of xels.

void pnm_freearray(xel** xels, int rows)

Free the array allocated with **pnm_allocarray()** containing the given number of rows.

void pnm_freerow(xel* xelrow)

Free a row of xels.

void pnm_readpnminit(FILE* fp, int* colsP, int* rowsP, xelval* maxvalP, int* formatP)

Read the header from a PNM file, filling in the rows, cols, maxval and format variables.

void pnm_readpnmrow(FILE* fp, xel* xelrow, int cols, xelval maxval, int format)

Read a row of xels into the xelrow array. Format, cols, and maxval were filled in by **pnm_readpnminit()**.

xel** pnm_readpnm(FILE* fp, int* colsP, int* rowsP, xelval* maxvalP, int* formatP)

Read an entire anymap file into memory, returning the allocated array and filling in the rows, cols, maxval, and format variables. This function combines **pnm_readpnminit()**, **pnm_allocarray()** and **pnm_readpnmrow()**. Unlike the equivalent functions in PBM, PGM, and PPM, it returns the format so you can tell what type the file is.

void pnm_writepnminit(FILE* fp, int cols, int rows, xelval maxval, int format, int forceplain)

Write the header for a portable anymap file. Unlike the equivalent functions in PBM, PGM, and PPM, you have to specify the output type. The forceplain flag forces a plain-format file to be written, as opposed to a raw-format one.

void pnm_writepnmrow(FILE* fp, xel* xelrow, int cols, xelval maxval, int format, int forceplain)

Write a row from a portable anymap.

void pnm_writepnm(FILE* fp, xel** xels, int cols, int rows, xelval maxval, int format, int forceplain)

Write the header and all data for a portable anymap. This function combines **pnm_writepnminit()** and **pnm_writepnmrow()**.

void pnm_promoteformatrow(xel* xelrow, int cols, xelval maxval, int format, xelval newmaxval, int newformat)

Promote a row of xels from one maxval and format to a new set. Used when combining multiple anymaps of different types - just take the max of the maxvals and the max of the formats, and promote them all to that.

void pnm_promoteformat(xel** xels, int cols, int rows, xelval maxval, int format, xelval newmaxval, int newformat)

Promote an entire anymap.

xel pnm_whitexel(xelval maxval, int format)
xel pnm_blackxel(xelval maxval, int format)

Return a white or black xel for the given maxval and format.

void pnm_invertxel(xel* x, xelval maxval, int format)

Invert an xel.

xel pnm_backgroundxelrow(xel* xelrow, int cols, xelval maxval, int format)

Figure out an appropriate background xel based on this row.

xel pnm_backgroundxel(xel** xels, int cols, int rows, xelval maxval, int format)

Figure out a background xel based on an entire anymap. This can do a slightly better job than **pnm_backgroundxelrow()**.

SEE ALSO

pbm(3), pgm(3), ppm(3)

AUTHOR

NAME

libppm - functions to support portable pixmap programs

SYNOPSIS

```
#include <ppm.h>
cc ... libppm.a libpgm.a libpbm.a
```

DESCRIPTION

```
typedef ... pixel;
typedef ... pixval;
#define PPM_MAXMAXVAL ...
extern pixval ppm_pbmmaxval;
```

Each **pixel** contains three **pixval**s, each of which should contain only the values between **0** and **PPM_MAXMAXVAL**. **ppm_pbmmaxval** is the maxval used when a PPM program reads a PBM file. Normally it is 1; however, for some programs, a larger value gives better results.

```
#define PPM_FORMAT ...
#define RPPM_FORMAT ...
#define PPM_TYPE PPM_FORMAT
int PPM_FORMAT_TYPE( int format )
```

For distinguishing different file formats and types.

```
pixval PPM_GETR( pixel p )
pixval PPM_GETG( pixel p )
pixval PPM_GETB( pixel p )
```

These three macros retrieve the red, green or blue value from the given pixel.

```
void PPM_ASSIGN( pixel p, pixval red, pixval grn, pixval blu )
```

This macro assigns the given red, green and blue values to the pixel.

```
int PPM_EQUAL( pixel p, pixel q )
```

This macro checks two pixels for equality.

void PPM_DEPTH(pixel newp, pixel p, pixval oldmaxval, pixval newmaxval)

This macro scales the colors of pixel **p** according the old and new maximum values and assigns the new values to **newp**. It is intended to make writing ppmtowhatever easier.

float PPM_LUMIN(pixel p)

This macro determines the luminance of the pixel **p**.

pixel** ppm_allocarray(int cols, int rows)

Allocate an array of pixels.

pixel* ppm_allocrow(int cols)

Allocate a row of the given number of pixels.

void ppm_freearray(pixel** pixels, int rows)

Free the array allocated with **ppm_allocarray()** containing the given number of rows.

void pbm_freerow(pixel* pixelrow)

Free a row of pixels.

void ppm_readppminit(FILE* fp, int* colsP, int* rowsP, pixval* maxvalP, int* formatP)

Read the header from a PPM file, filling in the rows, cols, maxval and format variables.

void ppm_readppmrow(FILE* fp, pixel* pixelrow, int cols, pixval maxval, int format)

Read a row of pixels into the pixelrow array. Format, cols, and maxval were filled in by **ppm_readppminit()**.

pixel** ppm_readppm(FILE* fp, int* colsP, int* rowsP, pixval* maxvalP)

Read an entire pixmap file into memory, returning the allocated array and filling in the rows, cols and maxval variables. This function combines **ppm_readppminit()**, **ppm_allocarray()** and **ppm_readppmrow()**.

void ppm_writeppminit(FILE* fp, int cols, int rows, pixval maxval, int forceplain)

Write the header for a portable pixmap file. The forceplain flag forces a plain-format file to be written, as opposed to a raw-format one.

void ppm_writeppmrow(FILE* fp, pixel* pixelrow, int cols, pixval maxval, int forceplain)

Write a row from a portable pixmap.

void ppm_writeppm(FILE* fp, pixel** pixels, int cols, int rows, pixval maxval, int forceplain)

Write the header and all data for a portable pixmap. This function combines **ppm_writeppminit()** and **ppm_writeppmrow()**.

pixel ppm_parsecolor(char* colorname, pixval maxval)

Parses an ASCII color name into a pixel. The color can be specified in three ways. One, as a name, assuming that a pointer to an X11-style color names file was compiled in. Two, as an X11-style hexadecimal number: #rgb, #rrggbb, #rrrgggbbb, or #rrrrggggbbbb. Three, as a triplet of decimal floating point numbers separated by commas: r.r,g.g,b.b.

char* ppm_colorname(pixel* colorP, pixval maxval, int hexok)

Returns a pointer to a string describing the given color. If the X11 color names file is available and the color appears in it, that name is returned. Otherwise, if the hexok flag is true then a hexadecimal colorspec is returned; if hexok is false and the X11 color names file is available, then the closest matching color is returned; otherwise, it's an error.

SEE ALSO

pbm(3), pgm(3)

AUTHOR

NAME

lispmtopgm - convert a Lisp Machine bitmap file into pgm format

SYNOPSIS

lispmtopgm [*lispmfile*]

DESCRIPTION

Reads a Lisp Machine bitmap as input. Produces a portable graymap as output.

This is the file format written by the tv:write-bit-array-file function on TI Explorer and Symbolics lisp machines.

Multi-plane bitmaps on lisp machines are color; but the lispm image file format does not include a color map, so we must treat it as a graymap instead. This is unfortunate.

SEE ALSO

pgmtolispm(1), pgm(5)

BUGS

The Lispm bitmap file format is a bit quirky; Usually the image in the file has its width rounded up to the next higher multiple of 32, but not always. If the width is not a multiple of 32, we don't deal with it properly, but because of the Lispm microcode, such arrays are probably not image data anyway.

Also, the lispm code for saving bitmaps has a bug, in that if you are writing a bitmap which is not mod32 across, the file may be up to 7 bits too short! They round down instead of up, and we don't handle this bug gracefully.

No color.

AUTHOR

NAME

macptopbm - convert a MacPaint file into a portable bitmap

SYNOPSIS

macptopbm [**-extraskip** *N*] [*macpfile*]

DESCRIPTION

Reads a MacPaint file as input. Produces a portable bitmap as output.

OPTIONS

-extraskip This flag is to get around a problem with some methods of transferring files from the Mac world to the Unix world. Most of these methods leave the Mac files alone, but a few of them add the "finderinfo" data onto the front of the Unix file. This means an extra 128 bytes to skip over when reading the file. The symptom to watch for is that the resulting PBM file looks shifted to one side. If you get this, try **-extraskip** 128, and if that still doesn't look right try another value.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

picttoppm(1), pbmtomacp(1), pbm(5)

AUTHOR

NAME

mgrtopbm - convert a MGR bitmap into a portable bitmap

SYNOPSIS

mgrtopbm [*mgrfile*]

DESCRIPTION

Reads a MGR bitmap as input. Produces a portable bitmap as output.

SEE ALSO

pbmtomgr(1), pbm(5)

AUTHOR

NAME

mtvtoppm - convert output from the MTV or PRT ray tracers into a portable pixmap

SYNOPSIS

mtvtoppm [*mtvfile*]

DESCRIPTION

Reads an input file from Mark VanDeWettering's MTV ray tracer. Produces a portable pixmap as output.

The PRT raytracer also produces this format.

SEE ALSO

ppm(5)

AUTHOR

NAME

pbm - portable bitmap file format

DESCRIPTION

The portable bitmap format is a lowest common denominator monochrome file format. It was originally designed to make it reasonable to mail bitmaps between different types of machines using the typical stupid network mailers we have today. Now it serves as the common language of a large family of bitmap conversion filters. The definition is as follows:

- A "magic number" for identifying the file type. A pbm file's magic number is the two characters "P1".
- Whitespace (blanks, TABs, CRs, LFs).
- A width, formatted as ASCII characters in decimal.
- Whitespace.
- A height, again in ASCII decimal.
- Whitespace.
- Width * height bits, each either '1' or '0', starting at the top-left corner of the bitmap, proceeding in normal English reading order.
- The character '1' means black, '0' means white.
- Whitespace in the bits section is ignored.
- Characters from a "#" to the next end-of-line are ignored (comments).
- No line should be longer than 70 characters.

Here is an example of a small bitmap in this format:

```
P1
# feep.pbm
24 7
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0
0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0
0 1 1 1 0 0 0 1 1 1 0 0 0 1 1 1 0 0 0 1 1 1 1 0
0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0
0 1 0 0 0 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Programs that read this format should be as lenient as possible, accepting anything that looks remotely like a bitmap.

There is also a variant on the format, available by setting the RAWBITS option at compile time. This variant is different in the following ways:

- The "magic number" is "P4" instead of "P1".
- The bits are stored eight per byte, high bit first low bit last.
- No whitespace is allowed in the bits section, and only a single character of whitespace (typically a newline) is allowed after the height.
- The files are eight times smaller and many times faster to read and write.

SEE ALSO

atktopbm(1), brushtopbm(1), cmuwmtopbm(1), g3topbm(1), gemtopbm(1), icontopbm(1), macptopbm(1), mgrtopbm(1), pi3topbm(1), xbmtopbm(1), ybmtopbm(1), pbmto10x(1), pnmtoascii(1), pbmtoatk(1), pbmtobbnbg(1), pbmtocmuwm(1), pbmtoepson(1), pbmtog3(1), pbmtogem(1), pbmtogo(1), pbmtoicon(1), pbmtolj(1), pbmtomacp(1), pbmtomgr(1), pbmtopi3(1), pbmtoplot(1), pbmtoptx(1), pbmtox10bm(1), pbmtoxbm(1), pbmtoybm(1), pbmtozinc(1), pbmlife(1), pbmmake(1), pbmmask(1), pbmreduce(1), pbmtext(1), pbmupc(1), pnm(5), pgm(5), ppm(5)

AUTHOR

NAME

pbmclean - flip isolated pixels in portable bitmap

SYNOPSIS

pbmclean [-connect] [pbmfile]

DESCRIPTION

Reads a portable bitmap as input. Outputs a portable bitmap with every pixel which has less than *connect* identical neighbours inverted. Pbmclean can be used to clean up "snow" on bitmap images.

SEE ALSO

pbm(5)

AUTHOR

NAME

pbmlife - apply Conway's rules of Life to a portable bitmap

SYNOPSIS

pbmlife [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Applies the rules of Life to it for one generation, and produces a portable bitmap as output.

A white pixel in the image is interpreted as a live beastie, and a black pixel as an empty space.

SEE ALSO

pbm(5)

AUTHOR

NAME

pbmmake - create a blank bitmap of a specified size

SYNOPSIS

pbmmake [**-white**|**-black**|**-gray**] *width height*

DESCRIPTION

Produces a portable bitmap of the specified width and height. The color defaults to white.

OPTIONS

In addition to the usual **-white** and **-black**, this program implements **-gray**. This gives a simple 50% gray pattern with 1's and 0's alternating.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pbm(5), ppmmake(1)

AUTHOR

NAME

pbmmask - create a mask bitmap from a regular bitmap

SYNOPSIS

pbmmask [**-expand**] [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Creates a corresponding mask bitmap and writes it out.

The color to be interpreted as "background" is determined automatically. Regardless of which color is background, the mask will be white where the background is and black where the figure is.

This lets you do a masked paste like this, for objects with a black background:

```
pbmmask obj > objmask
pnmpaste < dest -and objmask <x> <y> | pnmpaste -or obj <x> <y>
```

For objects with a white background, you can either invert them or add a step:

```
pbmmask obj > objmask
pnminvert objmask | pnmpaste -and obj 0 0 > blackback
pnmpaste < dest -and objmask <x> <y> | pnmpaste -or blackback <x> <y>
```

Note that this three-step version works for objects with black backgrounds too, if you don't care about the wasted time.

You can also use masks with graymaps and pixmaps, using the *pnmarith* tool. For instance:

```
ppmtopgm obj.ppm | pgmtopbm -threshold | pbmmask > objmask.pbm
pnmarith -multiply dest.ppm objmask.pbm > t1.ppm
pnminvert objmask.pbm | pnmarith -multiply obj.ppm - > t2.ppm
pnmarith -add t1.ppm t2.ppm
```

An interesting variation on this is to pipe the mask through the *pnmsmooth* script before using it. This makes the boundary between the two images less sharp.

-expand Expands the mask by one pixel out from the image. This is useful if you want a little white border around your image. (A better solution might be to turn the *pbmlife* tool into a general cellular automaton tool...)

SEE ALSO

pnmpaste(1), pnminvert(1), pbm(5), pnmarith(1), pnmsmooth(1)

AUTHOR

NAME

pbmpscale - enlarge a portable bitmap with edge smoothing

SYNOPSIS

pbmpscale N [pbmfile]

DESCRIPTION

Reads a portable bitmap as input, and outputs a portable bitmap enlarged N times. Enlargement is done by pixel replication, with some additional smoothing of corners and edges.

SEE ALSO

pnmenlarge(1), ppmscale(1), pbm(5)

AUTHOR

NOTES

pbmpscale works best for enlargements of 2. Enlargements greater than 2 should be done by as many enlargements of 2 as possible, followed by an enlargement by the remaining factor.

NAME

pbmreduce - read a portable bitmap and reduce it N times

SYNOPSIS

pbmreduce [**-floyd**|**-fs**|**-threshold**] [**-value** *val*] *N* [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Reduces it by a factor of *N*, and produces a portable bitmap as output.

pbmreduce duplicates a lot of the functionality of *pgmtopbm;* you could do something like **pnmscale | pgmtopbm,** but *pbmreduce* is a lot faster.

pbmreduce can be used to "re-halftone" an image. Let's say you have a scanner that only produces black&white, not grayscale, and it does a terrible job of halftoning (most b&w scanners fit this description). One way to fix the halftoning is to scan at the highest possible resolution, say 300 dpi, and then reduce by a factor of three or so using *pbmreduce*. You can even correct the brightness of an image, by using the **-value** flag.

OPTIONS

By default, the halftoning after the reduction is done via boustrophedonic Floyd-Steinberg error diffusion; however, the **-threshold** flag can be used to specify simple thresholding. This gives better results when reducing line drawings.

The **-value** flag alters the thresholding value for all quantizations. It should be a real number between 0 and 1. Above 0.5 means darker images; below 0.5 means lighter.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pnmenlarge(1), pnmscale(1), pgmtopbm(1), pbm(5)

AUTHOR

NAME

pbmtext - render text into a bitmap

SYNOPSIS

pbmtext [**-font** *fontfile*] [*text*]

DESCRIPTION

Takes the specified text, either a single line from the command line or multiple lines from standard input, and renders it into a bitmap.

OPTIONS

By default, pbmtext uses a built-in font. You can also specify your own font with the **-font** flag. The *fontfile* is a pbm file, created in a very specific way. In your window system of choice, display the following text in the desired (fixed-width) font:

```
M ",/^_[`jpqy| M
/  !"#$%&'()*+ /
< ,-./01234567 <
> 89:;<=>?@ABC >
@ DEFGHIJKLMNO @
_ PQRSTUVWXYZ[ _
{ \]^_`abcdefg {
} hijklmnopqrs }
~ tuvwxyz{|}~  ~
M ",/^_[`jpqy| M
```

Do a screen grab or window dump of that text, using for instance *xwd*, *xgrabsc*, or *screendump*. Convert the result into a pbm file. If necessary, use *pnmcut* to remove everything except the text. Finally, run it through *pnmcrop* to make sure the edges are right up against the text. *pbmtext* can figure out the sizes and spacings from that.

SEE ALSO

pbm(5), pnmcut(1), pnmcrop(1)

AUTHOR

NAME

pbmto10x - convert a portable bitmap into Gemini 10X printer graphics

SYNOPSIS

pbmto10x [**-h**] [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces a file of Gemini 10X printer graphics as output. The 10x's printer codes are alleged to be similar to the Epson codes.

Note that there is no 10xtopbm tool - this transformation is one way.

OPTIONS

The resolution is normally 60H by 72V. If the **-h** flag is specified, resolution is 120H by 144V. You may find it useful to rotate landscape images before printing.

SEE ALSO

pbm(5)

AUTHOR

NAME

pbmtoascii - convert a portable bitmap into ASCII graphics

SYNOPSIS

pbmtoascii [**-1x2**|**-2x4**] [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces a somewhat crude ASCII graphic as output.

Note that there is no asciitopbm tool - this transformation is one-way.

OPTIONS

The **-1x2** and **-2x4** flags give you two alternate ways for the bits to get mapped to characters. With **1x2**, the default, each character represents a group of 1 bit across by 2 bits down. With **-2x4**, each character represents 2 bits across by 4 bits down. With the 1x2 mode you can see the individual bits, so it's useful for previewing small bitmaps on a non-graphics terminal. The 2x4 mode lets you display larger bitmaps on a standard 80-column display, but it obscures bit-level details. 2x4 mode is also good for displaying graymaps - "pnmscale -width 158 | pgmnorm | pgmtopbm -thresh" should give good results.

SEE ALSO

pbm(5)

AUTHOR

NAME

pbmtoatk - convert portable bitmap to Andrew Toolkit raster object

SYNOPSIS

pbmtoatk [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces a Andrew Toolkit raster object as output.

SEE ALSO

atktopbm(1), pbm(5)

AUTHOR

©1991 by Bill Janssen.

NAME

pbmtobg - convert a portable bitmap into BitGraph graphics

SYNOPSIS

pbmtobg [*rasterop*] [*x y*] < *pbmfile*

DESCRIPTION

Reads a portable bitmap as input. Produces BBN BitGraph terminal Display Pixel Data (DPD) sequence as output.

The rasterop can be specified on the command line. If this is omitted, 3 (replace) will be used. A position in (x,y) coordinates can also be specified. If both are given, the rasterop comes first. The portable bitmap is always taken from the standard input.

Note that there is no bgtopbm tool.

SEE ALSO

pbm(5)

AUTHOR

Copyright 1989 by Mike Parker.

NAME

pbmtocmuwm - convert a portable bitmap into a CMU window manager bitmap

SYNOPSIS

pbmtocmuwm [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces a CMU window manager bitmap as output.

SEE ALSO

cmuwmtopbm(1), pbm(5)

AUTHOR

NAME

pbmtoepsi - convert a portable bitmap into an encapsulated PostScript style preview bitmap

SYNOPSIS

pbmtoepsi [**-bbonly**] [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produce an encapsulated Postscript style bitmap as output. The output is not a stand alone postscript file, it is only a preview bitmap, which can be included in an encapsulated PostScript file. Note that there is no epsitopbm tool - this transformation is one way.

This utility is a part of the pstoepsi tool by Doug Crabill (dgc@cs.purdue.edu).

OPTIONS

-bbonly Only create a boundary box, don't fill it with the image.

SEE ALSO

pbm(5), pnmtops(1), psidtopgm(1)

AUTHOR

©1988 Jef Poskanzer
modified by Doug Crabill 1992.

NAME

pbmtoepson - convert a portable bitmap into Epson printer graphics

SYNOPSIS

pbmtoepson [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces a file of Epson printer graphics as output.

Note that there is no epsontopbm tool - this transformation is one way.

SEE ALSO

pbm(5)

AUTHOR

©1991 by John Tiller (tiller@galois.msfc.nasa.gov) and Jef Poskanzer.

NAME

pbmtog3 - convert a portable bitmap into a Group 3 fax file

SYNOPSIS

pbmtog3 [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as output. Produces a Group 3 fax file as input.

REFERENCES

The standard for Group 3 fax is defined in CCITT Recommendation T.4.

BUGS

Probably.

SEE ALSO

g3topbm(1), pbm(5)

AUTHOR

NAME

pbmtogem - convert a portable bitmap into a GEM .img file

SYNOPSIS

pbmtogem [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces a GEM .img file as output.

BUGS

It does not support compression of the data.

SEE ALSO

gemtopbm(1), pbm(5)

AUTHOR

NAME

pbmtogo - convert a portable bitmap into compressed GraphOn graphics

SYNOPSIS

pbmtogo [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces 2D compressed GraphOn graphics as output. Be sure to set up your GraphOn with the following modes: 8 bits / no parity; obeys no XON/XOFF; NULs are accepted. These are all on the Comm menu. Also, remember to turn off tty post processing. Note that there is no gotopbm tool.

SEE ALSO

pbm(5)

AUTHOR

NAME

pbmtoicon - convert a portable bitmap into a Sun icon

SYNOPSIS

pbmtoicon [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces a Sun icon as output.

SEE ALSO

icontopbm(1), pbm(5)

AUTHOR

NAME

pbmtolj - convert a portable bitmap into HP LaserJet format

SYNOPSIS

pbmtolj [**-resolution** *N*] [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces HP LaserJet data as output.

Note that there is no ljtopbm tool.

OPTIONS

-resolution Specifies the resolution of the output device, in dpi. Typical values are 75, 100, 150, 300. The default is 75.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pbm(5)

AUTHOR

NAME

pbmtoln03 - convert protable bitmap to DEC LN03+ Sixel output

SYNOPSIS

pbmtoln03 [**-rltbf**] *pbmfile*

DESCRIPTION

Reads a portable bitmap as input. Produces a DEC LN03+ Sixel output file.

OPTIONS

-l nn Use "nn" as value for left margin (default 0).

-r nn Use "nn" as value for right margin (default 2400).

-t nn Use "nn" as value for top margin (default 0).

-b nn Use "nn" as value for bottom margin (default 3400).

-f nn Use "nn" as value for form length (default 3400).

SEE ALSO

pbm(5)

AUTHOR

Tim Cook, 26 Feb 1992

NAME

pbmtolps - convert portable bitmap to PostScript

SYNOPSIS

pbmtolps [-dpi n] [pbmfile]

DESCRIPTION

Reads a portable bitmap as input, and outputs PostScript. The output Postscript uses lines instead of the image operator to generate a (device dependent) picture which will be imaged much faster.

The Postscript path length is constrained to be less that 1000 points so that no limits are overrun on the Apple Laserwriter and (presumably) no other printers.

SEE ALSO

pgmtops(1), ppmtops(1), pbm(5)

AUTHOR

©George Phillips (phillips@cs.ubc.ca).

NAME

pbmtomacp - convert a portable bitmap into a MacPaint file

SYNOPSIS

pbmtomacp [**-l** *left*] [**-r** *right*] [**-b** *bottom*] [**-t** *top*] [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. If no input-file is given, standard input is assumed. Produces a MacPaint file as output.

The generated file is only the data fork of a picture. You will need a program such as *mcvert* to generate a Macbinary or a BinHex file that contains the necessary information to identify the file as a PNTG file to MacOS.

OPTIONS

Left, right, bottom & top let you define a square into the pbm file, that must be converted. Default is the whole file. If the file is too large for a MacPaint-file, the bitmap is cut to fit from (left, top).

BUGS

The source code contains comments in a language other than English.

SEE ALSO

ppmtopict(1), macptopbm(1), pbm(5), mcvert(1)

AUTHOR

©1988 by Douwe van der Schaaf (...!mcvax!uvapsy!vdschaaf).

NAME

pbmtomgr - convert a portable bitmap into a MGR bitmap

SYNOPSIS

pbmtomgr [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces a MGR bitmap as output.

SEE ALSO

mgrtopbm(1), pbm(5)

AUTHOR

NAME

pbmtopgm - convert portable bitmap to portable graymap by averaging areas

SYNOPSIS

pbmtopgm <width> <height> [pbmfile]

DESCRIPTION

Reads a portable bitmap as input. Outputs a portable graymap created by averaging the number of pixels within a sample area of *width* by *height* around each point. Pbmtopgm is similar to a special case of ppmconvol. A ppmsmooth step may be needed after pbmtopgm.

Pbmtopgm has the effect of anti-aliasing bitmaps which contain distinct line features.

SEE ALSO

pbm(5)

AUTHOR

NOTES

Pbmtopgm works best with odd sample width and heights.

NAME

pbmtopi3 - convert a portable bitmap into an Atari Degas .pi3 file

SYNOPSIS

pbmtopi3 [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces an Atari Degas .pi3 file as output.

SEE ALSO

pi3topbm(1), pbm(5), ppmtopi1(1), pi1toppm(1)

AUTHOR

©1988 by David Beckemeyer (bdt!david) and Jef Poskanzer.

NAME

pbmtopk - convert a portable bitmap into a packed (PK) format font

SYNOPSIS

pbmtopk pkfile[.pk] tfmfile[.tfm] resolution [-s designsize] [-p num param...] [-C codingscheme] [-F family] [-f optfile] [-c num] [-W width] [-H height] [-D depth] [-I ital] [-h horiz] [-v vert] [-x xoff] [-y yoff] [pbmfile]...

DESCRIPTION

Reads portable bitmaps as input, and produces a packed (PK) font file and a TFM (TeX font metric) file as output. The resolution parameter indicates the resolution of the font, in dots per inch. If the filename "-" is used for any of the filenames, the standard input stream (or standard output where appropriate) will be used.

OPTIONS

-s designsize
: Sets the design size of the font, in TeX's points (72.27pt to the inch). The default design size is 1. The TFM parameters are given as multiples of the design size.

-p num param...
: Sets the first num font parameters for the font. The first seven parameters are the slant, interword spacing, interword space stretchability, interword space shrinkability, x-height, quad width, and post-sentence extra space of the font. Math and symbol fonts may have more parameters; see The TeXbook for a list of these. Reasonable default values are chosen for parameters which are not specified.

-C codingscheme
: Sets the coding scheme comment in the TFM file.

-F family
: Sets the font family comment in the TFM file.

-f optfile
: Reads the file optfile, which should contain a lines of the form:

    filename xoff yoff horiz vert width height depth ital

: The pbm files specified by the filename parameters are inserted consecutively in the font with the specified attributes. If any of the attributes are omitted, or replaced with "*", a default value will be calculated from the size of the bitmap. The settings of the -W, -H, -D, -I, -h, -v, -x, and -y options do not affected characters created in this way. The character number can be changed by including a line starting with "=", followed by the new number. Lines beginning with "%" or "#" are ignored.

-c num
: Sets the character number of the next bitmap encountered to num.

-W width
: Sets the TFM width of the next character to width (in design size multiples).

-H height
: Sets the TFM height of the next character to height (in design size multiples).

-D depth
: Sets the TFM depth of the next character to depth (in design size multiples).

-I ital
: Sets the italic correction of the next character to ital (in design size multiples).

-h horiz
: Sets the horizontal escapement of the next character to horiz (in pixels).

-v vert
: Sets the vertical escapement of the next character to vert (in pixels).

-x xoff
: Sets the horizontal offset of the next character to xoff (in pixels).

-y yoff
: Sets the vertical offset of the next character to yoff (in pixels, from the top row).

SEE ALSO

pktopbm(1), pbm(5)

AUTHOR

Adapted from Tom Rokicki's pxtopk by Angus Duggan (ajcd@dcs.ed.ac.uk).

NAME

pbmtoplot - convert a portable bitmap into a Unix plot(5) file

SYNOPSIS

pbmtoplot [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces a Unix *plot* file.

Note that there is no plottopbm tool - this transformation is one-way.

SEE ALSO

pbm(5), plot(5)

AUTHOR

NAME

pbmtoptx - convert a portable bitmap into Printronix printer graphics

SYNOPSIS

pbmtoptx [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces a file of Printronix printer graphics as output. Note that there is no ptxtopbm tool - this transformation is one way.

SEE ALSO

pbm(5)

AUTHOR

NAME

pbmtox10bm - convert a portable bitmap into an X10 bitmap

SYNOPSIS

pbmtox10bm [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces an X10 bitmap as output. This older format is maintained for compatibility.

Note that there is no x10bmtopbm tool, because *xbmtopbm* can read both X11 and X10 bitmaps.

SEE ALSO

pbmtoxbm(1), xbmtopbm(1), pbm(5)

AUTHOR

NAME

pbmtoxbm - convert a portable bitmap into an X11 bitmap

SYNOPSIS

pbmtoxbm [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces an X11 bitmap as output.

SEE ALSO

pbmtox10bm(1), xbmtopbm(1), pbm(5)

AUTHOR

NAME

pgmtoybm - convert a portable bitmap into a Bennet Yee "face" file

SYNOPSIS

pbmtoybm [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces as output a file acceptable to the *face* and *xbm* programs by Bennet Yee (bsy+@cs.cmu.edu).

SEE ALSO

ybmtopbm(1), pbm(5), face(1), face(5), xbm(1)

AUTHOR

NAME

pbmtozinc - convert a portable bitmap into a Zinc bitmap

SYNOPSIS

pbmtozinc [*pbmfile*]

DESCRIPTION

Reads a portable bitmap as input. Produces a bitmap in the format used by the Zinc Interface Library (ZIL) Version 1.0 as output.

SEE ALSO

pbm(5)

AUTHOR

©1988 by James Darrell McCauley (jdm5548@diamond.tamu.edu) and Jef Poskanzer.

NAME

pbmupc - create a Universal Product Code bitmap

SYNOPSIS

pbmupc [**-s1**|**-s2**] *type manufac product*

DESCRIPTION

Generates a Universal Product Code symbol. The three arguments are: a one digit product type, a five digit manufacturer code, and a five digit product code. For example, "0 72890 00011" is the code for Heineken.

As presently configured, *pbmupc* produces a bitmap 230 bits wide and 175 bits high. The size can be altered by changing the defines at the beginning of the program, or by running the output through *pnmenlarge* or *pnmscale*.

OPTIONS

The **-s1** and **-s2** flags select the style of UPC to generate. The default, **-s1**, looks more or less like this:

```
 |||||||||||||||
 |||||||||||||||
 |||||||||||||||
 |||||||||||||||
0||12345||67890||5
```

The other style, **-s2**, puts the product type digit higher up, and doesn't display the checksum digit:

```
 |||||||||||||||
 |||||||||||||||
0|||||||||||||||
 |||||||||||||||
 ||12345||67890||
```

SEE ALSO

pbm(5)

AUTHOR

©1989 by Jef Poskanzer.

NAME

pcxtoppm - convert a PCX file into a portable pixmap

SYNOPSIS

pcxtoppm [*pcxfile*]

DESCRIPTION

Reads a PCX file as input. Produces a portable pixmap as output.

SEE ALSO

ppmtopcx(1), ppm(5)

AUTHOR

©1990 by Michael Davidson.

NAME

pgm - portable graymap file format

DESCRIPTION

The portable graymap format is a lowest common denominator grayscale file format. The definition is as follows:

- A "magic number" for identifying the file type. A pgm file's magic number is the two characters "P2".
- Whitespace (blanks, TABs, CRs, LFs).
- A width, formatted as ASCII characters in decimal.
- Whitespace.
- A height, again in ASCII decimal.
- Whitespace.
- The maximum gray value, again in ASCII decimal.
- Whitespace.
- Width * height gray values, each in ASCII decimal, between 0 and the specified maximum value, separated by whitespace, starting at the top-left corner of the graymap, proceeding in normal English reading order. A value of 0 means black, and the maximum value means white.
- Characters from a "#" to the next end-of-line are ignored (comments).
- No line should be longer than 70 characters.

Here is an example of a small graymap in this format:

```
P2
# feep.pgm
24 7
15
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  3  3  3  3  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15 15 15 15  0
0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0 15  0
0  3  3  3  0  0  0  7  7  7  0  0  0 11 11 11  0  0  0 15 15 15 15  0
0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0  0  0
0  3  0  0  0  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

Programs that read this format should be as lenient as possible, accepting anything that looks remotely like a graymap.

There is also a variant on the format, available by setting the RAWBITS option at compile time. This variant is different in the following ways:

- The "magic number" is "P5" instead of "P2".
- The gray values are stored as plain bytes, instead of ASCII decimal.

- No whitespace is allowed in the grays section, and only a single character of whitespace (typically a newline) is allowed after the maxval.
- The files are smaller and many times faster to read and write.

Note that this raw format can only be used for maxvals less than or equal to 255. If you use the *pgm* library and try to write a file with a larger maxval, it will automatically fall back on the slower but more general plain format.

SEE ALSO

fitstopgm(1), fstopgm(1), hipstopgm(1), lispmtopgm(1), psidtopgm(1), rawtopgm(1), pgmbentley(1), pgmcrater(1), pgmedge(1), pgmenhance(1), pgmhist(1), pgmnorm(1), pgmoil(1), pgmramp(1), pgmtexture(1), pgmtofits(1), pgmtofs(1), pgmtolispm(1), pgmtopbm(1), pnm(5), pbm(5), ppm(5)

AUTHOR

NAME

pgmbentley - Bentleyize a portable graymap

SYNOPSIS

pgmbentley [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Performs The Bentley Effect, and writes a portable graymap as output.

The Bentley Effect is described in "Beyond Photography" by Holzmann, chapter 4, photo 4. It's a vertical smearing based on brightness.

SEE ALSO

pgmoil(1), ppmrelief(1), pgm(5)

AUTHOR

NAME

pgmcrater - create cratered terrain by fractal forgery

SYNOPSIS

pgmcrater 'ti 15 [**-number** *n*] [**-height**|**-ysize** *s*] [**-width**|**-xsize** *s*] [**-gamma** *g*]

DESCRIPTION

pgmcrater creates a portable graymap which mimics cratered terrain. The graymap is created by simulating the impact of a given number of craters with random position and size, then rendering the resulting terrain elevations based on a light source shining from one side of the screen. The size distribution of the craters is based on a power law which results in many more small craters than large ones. The number of craters of a given size varies as the reciprocal of the area as described on pages 31 and 32 of Peitgen and Saupe[1]; cratered bodies in the Solar System are observed to obey this relationship. The formula used to obtain crater radii governed by this law from a uniformly distributed pseudorandom sequence was developed by Rudy Rucker.

High resolution images with large numbers of craters often benefit from being piped through **pnmsmooth**. The averaging performed by this process eliminates some of the jagged pixels and lends a mellow "telescopic image" feel to the overall picture.

OPTIONS

-number *n*
Causes *n* craters to be generated. If no **-number** specification is given, 50000 craters will be generated. Don't expect to see them all! For every large crater there are many, many more tiny ones which tend simply to erode the landscape. In general, the more craters you specify the more realistic the result; ideally you want the entire terrain to have been extensively turned over again and again by cratering. High resolution images containing five to ten million craters are stunning but take quite a while to create.

-height *height*
Sets the height of the generated image to *height* pixels. The default height is 256 pixels.

-width *width*
Sets the width of the generated image to *width* pixels. The default width is 256 pixels.

-xsize *width*
Sets the width of the generated image to *width* pixels. The default width is 256 pixels.

-ysize *height*
Sets the height of the generated image to *height* pixels. The default height is 256 pixels.

-gamma *factor*
The specified *factor* is used to gamma correct the graymap in the same manner as performed by **pnmgamma**. The default value is 1.0, which results in a medium contrast image. Values larger than 1 lighten the image and reduce contrast, while values less than 1 darken the image, increasing contrast.

All flags can be abbreviated to their shortest unique prefix.

BUGS

The **-gamma** option isn't really necessary since you can achieve the same effect by piping the output from **pgmcrater** through **pnmgamma**. However, **pgmcrater** performs an internal gamma map anyway in the process of rendering the elevation array into a graymap, so there's no additional overhead in allowing a user-specified gamma.

Real craters have two distinct morphologies. **pgmcrater** simulates only small craters, which are hemispherical in shape (regardless of the incidence angle of the impacting body, as long as the velocity is sufficiently high). Large craters, such as Copernicus and Tycho on the Moon, have a "walled plain" shape with a cross-section more like:

```
         /\                    /\

-----/  \ -----------/\ -----------/  \ -----
```

Larger craters should really use this profile, including the central peak, and totally obliterate the pre-existing terrain.

SEE ALSO

pgm(5), **pnmgamma**(1), **pnmsmooth**(1)

[1] Peitgen, H.-O., and Saupe, D. eds., The Science Of Fractal Images, New York: Springer Verlag, 1988.

AUTHOR

John Walker
Autodesk SA
Avenue des Champs-Montants 14b
CH-2074 MARIN
Suisse/Schweiz/Svizzera/Svizra/Switzerland

Usenet: kelvin@Autodesk.com

Fax: 038/33 88 15

Voice: 038/33 76 33

NAME

pgmedge - edge-detect a portable graymap

SYNOPSIS

pgmedge [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Outlines the edges, and writes a portable graymap as output. Piping the result through **pgmtopbm -threshold** and playing with the threshold value will give a bitmap of the edges.

The edge detection technique used is to take the Pythagorean sum of two Sobel gradient operators at 90 degrees to each other. For more details see "Digital Image Processing" by Gonzalez and Wintz, chapter 7.

SEE ALSO

pgmenhance(1), pgmtopbm(1), pgm(5), pbm(5)

AUTHOR

NAME

pgmenhance - edge-enhance a portable graymap

SYNOPSIS

pgmenhance [-*N*] [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Enhances the edges, and writes a portable graymap as output.

The edge enhancing technique is taken from Philip R. Thompson's "xim" program, which in turn took it from section 6 of "Digital Halftones by Dot Diffusion", D. E. Knuth, ACM Transaction on Graphics Vol. 6, No. 4, October 1987, which in turn got it from two 1976 papers by J. F. Jarvis *et. al.*

OPTIONS

The optional -*N* flag should be a digit from 1 to 9. 1 is the lowest level of enhancement, 9 is the highest, The default is 9.

SEE ALSO

pgmedge(1), pgm(5), pbm(5)

AUTHOR

NAME

pgmhist - print a histogram of the values in a portable graymap

SYNOPSIS

pgmhist [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Prints a histogram of the gray values.

SEE ALSO

pgmnorm(1), pgm(5), ppmhist(1)

AUTHOR

NAME

pgmkernel - generate a convolution kernel

SYNOPSIS

pgmkernel [–**weight** *w*] *width* [*height*]

DESCRIPTION

Generates a portable graymap array of size *width* x *height* (or *width* x *width* if *height* is not specified) to be used as a convolution file by **pnmconvol**. The data in the convolution array K are computed according to the formula:

K(i,j) = 1 / (1 + w * sqrt((i-width/2)^2 + (j-height/2)^2))

where *w* is a coefficient specified via the *–weight* flag, and *width* and *height* are the X and Y filter sizes.

The output PGM file is always written out in ASCII format.

OPTIONS

The optional -*weight* flag should be a real number greater than -1. The default value is 6.0.

BUGS

The computation time is proportional to *width* * *height*. This increases rapidly with the increase of the kernel size. A better approach could be using a FFT in these cases.

SEE ALSO

pnmconvol(1), pnmsmooth(1)

AUTHOR

Alberto Accomazzi (alberto@cfa.harvard.edu).

NAME

pgmnoise - create a graymap made up of white noise

SYNOPSIS

pgmnoise *width height*

DESCRIPTION

Creates a portable graymap that is made up of random pixels with gray values in the range of 0 to PGM_MAXMAXVAL (depends on the compilation, either 255 or 65535). The graymap has a size of width * height pixels.

SEE ALSO

pgm(5)

AUTHOR

NAME

pgmnorm - normalize the contrast in a portable graymap

SYNOPSIS

pgmnorm [**-bpercent** *N* | **-bvalue** *N*] [**-wpercent** *N* | **-wvalue** *N*] [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Normalizes the contrast by forcing the lightest pixels to white, the darkest pixels to black, and linearly rescaling the ones in between; and produces a portable graymap as output.

OPTIONS

By default, the darkest 2 percent of all pixels are mapped to black, and the lightest 1 percent are mapped to white. You can override these percentages by using the **-bpercent** and **-wpercent** flags, or you can specify the exact pixel values to be mapped by using the **-bvalue** and **-wvalue** flags. Appropriate numbers for the flags can be gotten from the *pgmhist* tool. If you just want to enhance the contrast, then choose values at elbows in the histogram; *e.g.*, if value 29 represents 3% of the image but value 30 represents 20%, choose 30 for *bvalue*. If you want to lighten the image, then set *bvalue* to 0 and just fiddle with *wvalue*; similarly, to darken the image, set *wvalue* to maxval and play with *bvalue*.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pgmhist(1), pgm(5)

AUTHOR

Partially based on the fbnorm filter in Michael Mauldin's "Fuzzy Pixmap" package.

NAME

pgmoil - turn a portable graymap into an oil painting

SYNOPSIS

pgmoil [**-n** *N*] [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Does an "oil transfer", and writes a portable graymap as output.

The oil transfer is described in "Beyond Photography" by Holzmann, chapter 4, photo 7. It's a sort of localized smearing.

OPTIONS

The optional **-n** flag controls the size of the area smeared. The default value is 3.

BUGS

Takes a long time to run.

SEE ALSO

pgmbentley(1), ppmrelief(1), pgm(5)

AUTHOR

NAME

pgmramp - generate a grayscale ramp

SYNOPSIS

pgmramp -lr|-tb | -rectangle|-ellipse *width height*

DESCRIPTION

Generates a graymap of the specified size containing a black-to-white ramp. These ramps are useful for multiplying with other images, using the *pnmarith* tool.

OPTIONS

-lr A left to right ramp.

-tb A top to bottom ramp.

-rectangle A rectangular ramp.

-ellipse An elliptical ramp.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pnmarith(1), pgm(5)

AUTHOR

NAME

pgmtexture - calculate textural features on a portable graymap

SYNOPSIS

pgmtexture [**-d** *d*] [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Calculates textural features based on spatial dependence matrices at 0, 45, 90, and 135 degrees for a distance *d* (default = 1). Textural features include:

(1) Angular Second Moment,
(2) Contrast,
(3) Correlation,
(4) Variance,
(5) Inverse Difference Moment,
(6) Sum Average,
(7) Sum Variance,
(8) Sum Entropy,
(9) Entropy,
(10) Difference Variance,
(11) Difference Entropy,
(12, 13) Information Measures of Correlation, and
(14) Maximal Correlation Coefficient.

Algorithm taken from:
Haralick, R.M., K. Shanmugam, and I. Dinstein. 1973. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybertinetics,* SMC-3(6):610-621.

BUGS

The program can run incredibly slow for large images (larger than 64 x 64) and command line options are limited. The method for finding (14) the maximal correlation coefficient, which requires finding the second largest eigenvalue of a matrix Q, does not always converge.

REFERENCES

IEEE Transactions on Systems, Man, and Cybertinetics, SMC-3(6):610-621.

SEE ALSO

pgm(5), pnmcut(1)

AUTHOR

NAME

pgmtofits - convert a portable graymap into FITS format

SYNOPSIS

pgmtofits [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Produces a FITS file as output.

FITS stands for Flexible Image Transport System. A full description can be found in Astronomy & Astrophysics Supplement Series 44 (1981), page 363.

SEE ALSO

fitstopgm(1), pgm(5)

AUTHOR

NAME

pgmtofs - convert portable graymap to Usenix FaceSaver(tm) format

SYNOPSIS

pgmtofs [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Produces Usenix FaceSaver(tm) format as output.

FaceSaver is a registered trademark of Metron Computerware Ltd. of Oakland, CA.

SEE ALSO

fstopgm(1), pgm(5)

AUTHOR

NAME

pgmtolispm - convert a portable graymap into Lisp Machine format

SYNOPSIS

pgmtolispm [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Produces a Lisp Machine bitmap as output.

This is the file format read by the tv:read-bit-array-file function on TI Explorer and Symbolics lisp machines.

Given a pgm (instead of a pbm) a multi-plane image will be output. This is probably not useful unless you have a color lisp machine.

Multi-plane bitmaps on lisp machines are color; but the lispm image file format does not include a color map, so we must treat it as a graymap instead. This is unfortunate.

SEE ALSO

lispmtopgm(1), pgm(5)

BUGS

Output width is always rounded up to the nearest multiple of 32; this might not always be what you want, but it probably is (arrays which are not modulo 32 cannot be passed to the Lispm BITBLT function, and thus cannot easily be displayed on the screen).

No color.

AUTHOR

NAME

pgmtopbm - convert a portable graymap into a portable bitmap

SYNOPSIS

pgmtopbm [**-floyd**|**-fs**|**-threshold** |**-dither8**|**-d8**|**-cluster3** |**-c3**|**-cluster4**|**-c4** |**-cluster8**|**-c8**] [**-value** *val*] [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Produces a portable bitmap as output.

Note that there is no pbmtopgm converter, because any pgm program can read pbm files automagically.

OPTIONS

The default quantization method is boustrophedonic Floyd-Steinberg error diffusion (**-floyd** or **-fs**). Also available are simple thresholding (**-threshold**); Bayer's ordered dither (**-dither8**) with a 16x16 matrix; and three different sizes of 45-degree clustered-dot dither (**-cluster3**, **-cluster4**, **-cluster8**).

Floyd-Steinberg will almost always give the best looking results; however, looking good is not always what you want. For instance, thresholding can be used in a pipeline with the *pnmconvol* tool, for tasks like edge and peak detection. And clustered-dot dithering gives a newspaper-ish look, a useful special effect.

The **-value** flag alters the thresholding value for Floyd-Steinberg and simple thresholding. It should be a real number between 0 and 1. Above 0.5 means darker images; below 0.5 means lighter.

All flags can be abbreviated to their shortest unique prefix.

REFERENCES

The only reference you need for this stuff is "Digital Halftoning" by Robert Ulichney, MIT Press, ISBN 0–262–21009–6.

SEE ALSO

pbmreduce(1), pgm(5), pbm(5), pnmconvol(1)

AUTHOR

NAME

pgmtoppm - colorize a portable graymap into a portable pixmap

SYNOPSIS

pgmtoppm *colorspec* [*pgmfile*]
pgmtoppm *colorspec1***-***colorspec2* [*pgmfile*]
pgmtoppm -map *mapfile* [*pgmfile*]

DESCRIPTION

Reads a portable graymap as input. Colorizes it by multiplying the the gray values by specified color or colors, and produces a portable pixmap as output.

If only one color is specified, black in the pgm file stays black and white in the pgm file turns into the specified color in the ppm file. If two colors (separated by a dash) are specified, then black gets mapped to the first color and white gets mapped to the second.

The color can be specified in five ways:

- A name, assuming that a pointer to an X11-style color names file was compiled in.
- An X11-style hexadecimal specifier: rgb:r/g/b, where r g and b are each 1- to 4-digit hexadecimal numbers.
- An X11-style decimal specifier: rgbi:r/g/b, where r g and b are floating point numbers between 0 and 1.
- For backwards compatibility, an old-X11-style hexadecimal number: #rgb, #rrggbb, #rrrgggbbb, or #rrrrggggbbbb.
- For backwards compatibility, a triplet of numbers separated by commas: r,g,b, where r g and b are floating point numbers between 0 and 1. (This style was added before MIT came up with the similar rgbi style.)

Also, the **-map** flag lets you specify an entire colormap to be used. The mapfile is just a *ppm* file; it can be any shape, all that matters is the colors in it and their order. In this case, black gets mapped into the first color in the map file, and white gets mapped to the last.

SEE ALSO

rgb3toppm(1), ppmtopgm(1), ppmtorgb3(1), ppm(5), pgm(5)

AUTHOR

NAME

pi1toppm - convert an Atari Degas .pi1 into a portable pixmap

SYNOPSIS

pi1toppm [*pi1file*]

DESCRIPTION

Reads an Atari Degas .pi1 file as input. Produces a portable pixmap as output.

SEE ALSO

ppmtopi1(1), ppm(5), pi3topbm(1), pbmtopi3(1)

AUTHOR

©1991 by Steve Belczyk (seb3@gte.com) and Jef Poskanzer.

NAME

pi3topbm - convert an Atari Degas .pi3 file into a portable bitmap

SYNOPSIS

pi3topbm [*pi3file*]

DESCRIPTION

Reads an Atari Degas .pi3 file as input. Produces a portable bitmap as output.

SEE ALSO

pbmtopi3(1), pbm(5), pi1toppm(1), ppmtopi1(1)

AUTHOR

NAME

picttoppm - convert a Macintosh PICT file into a portable pixmap

SYNOPSIS

picttoppm [**-verbose**] [**-fullres**] [**-noheader**] [*pictfile*]

DESCRIPTION

Reads a PICT file (version 1 or 2) and outputs a portable pixmap. Useful as the first step in converting a scanned image to something that can be displayed on Unix.

OPTIONS

–**fullres** Force any images in the PICT file to be output with at least their full resolution. A PICT file may indicate that a contained image is to be scaled down before output. This option forces images to retain their sizes and prevent information loss.

–**noheader** Do not skip the 512 byte header that is present on all PICT files. This is useful when you have PICT data that was not stored in the data fork of a PICT file.

–**verbose** Turns on verbose mode which prints a a whole bunch of information that only *picttoppm* hackers really care about.

BUGS

The PICT file format is a general drawing format. *picttoppm* only supports a small subset of its operations but is still very useful for files produced by scanning software. In particular, text added to a scanned image will be silently ignored.

SEE ALSO

Inside Macintosh volume 5, ppmtopict(1), ppm(5)

AUTHOR

NAME

pjtoppm - convert an HP PaintJet file to a portable pixmap

SYNOPSIS

pjtoppm [*paintjet*]

DESCRIPTION

Reads an HP PaintJet file as input and converts it into a portable pixmap. This was a quick hack to save some trees, and it only handles a small subset of the paintjet commands. In particular, it will only handle enough commands to convert most raster image files.

REFERENCES

HP PaintJet XL Color Graphics Printer User's Guide

SEE ALSO

ppmtopj(1)

AUTHOR

NAME

pktopbm - convert packed (PK) format font into portable bitmap(s)

SYNOPSIS

pktopbm pkfile[.pk] [-c num] pbmfile ...

DESCRIPTION

Reads a packed (PK) font file as input, and produces portable bitmaps as output. If the filename "-" is used for any of the filenames, the standard input stream (or standard output where appropriate) will be used.

OPTIONS

-c num Sets the character number of the next bitmap written to num.

SEE ALSO

pbmtopk(1), pbm(5)

AUTHOR

Adapted from Tom Rokicki's pxtopk by Angus Duggan (ajcd@dcs.ed.ac.uk.

NAME

pnm - portable anymap file format

DESCRIPTION

The *pnm* programs operate on portable bitmaps, graymaps, and pixmaps, produced by the *pbm, pgm,* and *ppm* segments. There is no file format associated with *pnm* itself.

SEE ALSO

anytopnm(1), rasttopnm(1), tifftopnm(1), xwdtopnm(1), pnmtops(1), pnmtorast(1), pnmtotiff(1), pnmtoxwd(1), pnmarith(1), pnmcat(1), pnmconvol(1), pnmcrop(1), pnmcut(1), pnmdepth(1), pnmenlarge(1), pnmfile(1), pnmflip(1), pnmgamma(1), pnmindex(1), pnminvert(1), pnmmargin(1), pnmnoraw(1), pnmpaste(1), pnmrotate(1), pnmscale(1), pnmshear(1), pnmsmooth(1), pnmtile(1), ppm(5), pgm(5), pbm(5)

AUTHOR

NAME

pnmalias - antialias a portable anyumap.

SYNOPSIS

pnmalias [**-bgcolor** *color*] [**-fgcolor** *color*] [**-bonly**] [**-fonly**] [**-balias**] [**-falias**] [**-weight** *w*] [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input, and applies anti-aliasing to background and foreground pixels. If the input file is a portable bitmap, the output anti-aliased image is promoted to a graymap, and a message is printed informing the user of the change in format.

OPTIONS

–**bgcolor** *colorb,* –**fgcolor** *colorf*

set the background color to *colorb,* and the foreground to color to *colorf.* Pixels with these values will be anti-aliased. by default, the background color is taken to be black, and foreground color is assumed to be white. The colors can be specified in five ways:

- A name, assuming that a pointer to an X11-style color names file was compiled in.
- An X11-style hexadecimal specifier: rgb:r/g/b, where r g and b are each 1- to 4-digit hexadecimal numbers.
- An X11-style decimal specifier: rgbi:r/g/b, where r g and b are floating point numbers between 0 and 1.
- For backwards compatibility, an old-X11-style hexadecimal number: #rgb, #rrggbb, #rrrgggbbb, or #rrrrggggbbbb.
- For backwards compatibility, a triplet of numbers separated by commas: r,g,b, where r g and b are floating point numbers between 0 and 1. (This style was added before MIT came up with the similar rgbi style.)

Note that even when dealing with graymaps, background and foreground colors need to be specified in the fashion described above. In this case, background and foreground pixel values are taken to be the value of the red component for the given color.

–**bonly**, –**fonly**

Apply anti-aliasing only to background (–**bonly**), or foreground (–**fonly**) pixels.

–**balias**, –**falias**

Apply anti-aliasing to all pixels surrounding background (–**balias**), or foreground (–**falias**) pixels. By default, anti-aliasing takes place only among neighboring background and foreground pixels.

–**weight** *w*

Use *w* as the central weight for the aliasing filter. *W* must be a real number in the range $0 < w < 1$. The lower the value of *w* is, the "blurrier" the output image is. The default is w = 1/3.

SEE ALSO

pbmtext(1), pnmsmooth(1), pnm(5)

AUTHOR

NAME

pnmarith - perform arithmetic on two portable anymaps

SYNOPSIS

pnmarith -add|-subtract|-multiply|-difference *pnmfile1 pnmfile2*

DESCRIPTION

Reads two portable anymaps as input. Performs the specified arithmetic operation, and produces a portable anymap as output. The two input anymaps must be the same width and height.

The arithmetic is performed between corresponding pixels in the two anymaps, as if maxval was 1.0, black was 0.0, and a linear scale in between. Results that fall outside of [0..1) are truncated.

The operator *-difference* calculates the absolute value of *pnmarith -subtract pnmfile1 pnmfile2, i.e.*, no truncation is done.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pbmmask(1), pnmpaste(1), pnminvert(1), pnm(5)

AUTHOR

NAME

pnmcat - concatenate portable anymaps

SYNOPSIS

pnmcat [**-white**|**-black**] **-leftright**|**-lr** [**-jtop**|**-jbottom**] *pnmfile pnmfile* ...
pnmcat [**-white**|**-black**] **-topbottom**|**-tb** [**-jleft**|**-jright**] *pnmfile pnmfile* ...

DESCRIPTION

Reads portable anymaps as input. Concatenates them either left to right or top to bottom, and produces a portable anymap as output.

OPTIONS

If the anymaps are not all the same height (left-right) or width (top-bottom), the smaller ones have to be justified with the largest. By default, they get centered, but you can specify one side or the other with one of the -j* flags. So, **-topbottom -jleft** would stack the anymaps on top of each other, flush with the left edge.

The **-white** and **-black** flags specify what color to use to fill in the extra space when doing this justification. If neither is specified, the program makes a guess.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pnm(5)

AUTHOR

NAME

pnmcomp - composite two portable anymap files together

SYNOPSIS

pnmcomp [-*invert*] [-*xoff*N] [-*yoff*N] [-*alpha*pgmfile] overlay [*pnm-input*] [*pnm-output*]

DESCRIPTION

Reads in a portable any map image and put a overlay upon it, with optional alpha mask. The -*alpha pgmfile* allows you to also add an alpha mask file to the compositing process, the range of max and min can be swapped by using the -*invert* option. The -*xoff* and -*yoff* arguments can be negative, allowing you to shift the overlay off the top corner of the screen.

SEE ALSO

pnm(5)

AUTHOR

©1992 by David Koblas (koblas@mips.com).

NAME

pnmconvol - general MxN convolution on a portable anymap

SYNOPSIS

pnmconvol *convolutionfile* [*pnmfile*]

DESCRIPTION

Reads two portable anymaps as input. Convolves the second using the first, and writes a portable anymap as output.

Convolution means replacing each pixel with a weighted average of the nearby pixels. The weights and the area to average are determined by the convolution matrix. The unsigned numbers in the convolution file are offset by -maxval/2 to make signed numbers, and then normalized, so the actual values in the convolution file are only relative.

Here is a sample convolution file; it does a simple average of the nine immediate neighbors, resulting in a smoothed image:

```
P2
3 3
18
10 10 10
10 10 10
10 10 10
```

To see how this works, do the above-mentioned offset: 10 - 18/2 gives 1. The possible range of values is from 0 to 18, and after the offset that's -9 to 9. The normalization step makes the range -1 to 1, and the values get scaled correspondingly so they become 1/9 - exactly what you want. The equivalent matrix for 5x5 smoothing would have maxval 50 and be filled with 26.

The convolution file will usually be a graymap, so that the same convolution gets applied to each color component. However, if you want to use a pixmap and do a different convolution to different colors, you can certainly do that.

SEE ALSO

pnmsmooth(1), pnm(5)

AUTHOR

NAME

pnmcrop - crop a portable anymap

SYNOPSIS

pnmcrop [**-white**|**-black**] [**-left**] [**-right**] [**-top**] [**-bottom**] [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Removes edges that are the background color, and produces a portable anymap as output.

OPTIONS

By default, it makes a guess as to what the background color is. You can override the default with the **-white** and **-black** flags.

The options **-left, -right, -top** and **-bottom** restrict cropping to the sides specified. The default is to crop all sides of the image.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pnmcut(1), pnm(5)

AUTHOR

NAME

pnmcut - cut a rectangle out of a portable anymap

SYNOPSIS

pnmcut *x y width height* [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Extracts the specified rectangle, and produces a portable anymap as output. The *x* and *y* can be negative, in which case they are interpreted relative to the right and bottom of the anymap, respectively.

SEE ALSO

pnm(5)

AUTHOR

NAME

pnmdepth - change the maxval in a portable anymap

SYNOPSIS

pnmdepth *newmaxval* [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Scales all the pixel values, and writes out the image with the new maxval. Scaling the colors down to a smaller maxval will result in some loss of information.

Be careful of off-by-one errors when choosing the new maxval. For instance, if you want the color values to be five bits wide, use a maxval of 31, not 32.

SEE ALSO

pnm(5), ppmquant(1), ppmdither(1)

AUTHOR

NAME

pnmenlarge - read a portable anymap and enlarge it N times

SYNOPSIS

pnmenlarge *N* [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Replicates its pixels *N* times, and produces a portable anymap as output.

pnmenlarge can only enlarge by integer factors. The slower but more general *pnmscale* can enlarge or reduce by arbitrary factors, and *pbmreduce* can reduce by integer factors, but only for bitmaps.

If you enlarge by a factor of 3 or more, you should probably add a *pnmsmooth* step; otherwise, you can see the original pixels in the resulting image.

SEE ALSO

pbmreduce(1), pnmscale(1), pnmsmooth(1), pnm(5)

AUTHOR

NAME

pnmfile - describe a portable anymap

SYNOPSIS

pnmfile [*pnmfile*] ...

DESCRIPTION

Reads one or more portable anymaps as input. Writes out short descriptions of the image type, size, etc. This is mostly for use in shell scripts, so the format is not particularly pretty.

SEE ALSO

pnm(5), file(1)

AUTHOR

NAME

pnmflip - perform one or more flip operations on a portable anymap

SYNOPSIS

pnmflip [**-leftright**|**-lr**] [**-topbottom**|**-tb**] [**-transpose**|**-xy**] [**-rotate90**|**-r90**|**-ccw**] [**-rotate270**|**-r270**|**-cw**] [**-rotate180**|**-r180**] [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Performs one or more flip operations, in the order specified, and writes out a portable anymap.

OPTIONS

The flip operations available are: left for right (**-leftright** or **-lr**); top for bottom (**-topbottom** or **-tb**); and transposition (**-transpose** or **-xy**). In addition, some canned concatenations are available: **-rotate90** or **-ccw** is equivalent to **-transpose -topbottom**; **-rotate270** or **-cw** is equivalent to **-transpose -leftright**; and **-rotate180** is equivalent to **-leftright -topbottom**.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pnmrotate(1), pnm(5)

AUTHOR

NAME

pnmgamma - perform gamma correction on a portable anymap

SYNOPSIS

pnmgamma *value* [*pnmfile*]
pnmgamma *redvalue greenvalue bluevalue* [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Performs gamma correction, and produces a portable anymap as output.

The arguments specify what gamma value(s) to use. A value of 1.0 leaves the image alone, less than one darkens it, and greater than one lightens it.

SEE ALSO

pnm(5)

AUTHOR

NAME

pnminvert - invert a portable anymap

SYNOPSIS

pnminvert [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Inverts it black for white and produces a portable anymap as output.

SEE ALSO

pnm(5)

AUTHOR

NAME

pnmnlfilt - non-linear filters: smooth, alpha trim mean, optimal estimation smoothing, edge enhancement.

SYNOPSIS

pnmnlfilt alpha radius [*pnmfile*]

DESCRIPTION

This is something of a swiss army knife filter. It has 3 distinct operating modes. In all of the modes each pixel in the image is examined and processed according to it and its surrounding pixels values. Rather than using the 9 pixels in a 3x3 block, 7 hexagonal area samples are taken, the size of the hexagons being controlled by the radius parameter. A radius value of 0.3333 means that the 7 hexagons exactly fit into the center pixel (*i.e.*, there will be no filtering effect). A radius value of 1.0 means that the 7 hexagons exactly fit a 3x3 pixel array.

Alpha trimmed mean filter. (0.0 <= alpha <= 0.5)

The value of the center pixel will be replaced by the mean of the 7 hexagon values, but the 7 values are sorted by size and the top and bottom alpha portion of the 7 are excluded from the mean. This implies that an alpha value of 0.0 gives the same sort of output as a normal convolution (*i.e.*, averaging or smoothing filter), where radius will determine the "strength" of the filter. A good value to start from for subtle filtering is alpha = 0.0, radius = 0.55 For a more blatant effect, try alpha 0.0 and radius 1.0

An alpha value of 0.5 will cause the median value of the 7 hexagons to be used to replace the center pixel value. This sort of filter is good for eliminating "pop" or single pixel noise from an image without spreading the noise out or smudging features on the image. Judicious use of the radius parameter will fine tune the filtering. Intermediate values of alpha give effects somewhere between smoothing and "pop" noise reduction. For subtle filtering try starting with values of alpha = 0.4, radius = 0.6 For a more blatant effect try alpha = 0.5, radius = 1.0

Optimal estimation smoothing. (1.0 <= alpha <= 2.0)

This type of filter applies a smoothing filter adaptively over the image. For each pixel the variance of the surrounding hexagon values is calculated, and the amount of smoothing is made inversely proportional to it. The idea is that if the variance is small then it is due to noise in the image, while if the variance is large, it is because of "wanted" image features. As usual the radius parameter controls the effective radius, but it probably advisable to leave the radius between 0.8 and 1.0 for the variance calculation to be meaningful. The alpha parameter sets the noise threshold, over which less smoothing will be done. This means that small values of alpha will give the most subtle filtering effect, while large values will tend to smooth all parts of the image. You could start with values like alpha = 1.2, radius = 1.0 and try increasing or decreasing the alpha parameter to get the desired effect. This type of filter is best for filtering out dithering noise in both bitmap and color images.

Edge enhancement. (-0.1 >= alpha >= -0.9)

This is the opposite type of filter to the smoothing filter. It enhances edges. The alpha parameter controls the amount of edge enhancement, from subtle (-0.1) to blatant (-0.9). The radius parameter controls the effective radius as usual, but useful values are between 0.5 and 0.9. Try starting with values of alpha = 0.3, radius = 0.8

Combination use.

The various modes of **pnmnlfilt** can be used one after the other to get the desired result. For instance to turn a monochrome dithered image into a grayscale image you could try one or two passes of the smoothing filter, followed by a pass of the optimal estimation filter, then some subtle edge enhancement. Note that using edge enhancement is only likely to be useful after one of the non-linear filters (alpha trimmed mean or optimal estimation filter), as edge enhancement is the direct opposite of smoothing.

For reducing color quantization noise in images (*i.e.*, turning .gif files back into 24 bit files) you could try a pass of the optimal estimation filter (alpha 1.2, radius 1.0), a pass of the median filter (alpha 0.5, radius 0.55), and possibly a pass of the edge enhancement filter. Several passes of the optimal estimation filter with declining alpha values are more effective than a single pass with a large alpha value. As usual, there is a tradeoff between filtering effectiveness and loosing detail. Experimentation is encouraged.

References:

The alpha-trimmed mean filter is based on the description in IEEE CG&A May 1990 Page 23 by Mark E. Lee and Richard A. Redner, and has been enhanced to allow continuous alpha adjustment.

The optimal estimation filter is taken from an article "Converting Dithered Images Back to Gray Scale" by Allen Stenger, Dr Dobb's Journal, November 1992, and this article references "Digital Image Enhancement and Noise Filtering by Use of Local Statistics", Jong-Sen Lee, IEEE Transactions on Pattern Analysis and Machine Intelligence, March 1980.

The edge enhancement details are from pgmenhance(1), which is taken from Philip R. Thompson's "xim" program, which in turn took it from section 6 of "Digital Halftones by Dot Diffusion", D. E. Knuth, ACM Transaction on Graphics Vol. 6, No. 4, October 1987, which in turn got it from two 1976 papers by J. F. Jarvis *et. al.*

SEE ALSO

pgmenhance(1), pnmconvol(1), pnm(5)

BUGS

Integers and tables may overflow if PPM_MAXMAXVAL is greater than 255.

AUTHOR

Graeme W. Gill graeme@labtam.oz.au

NAME

pnmnoraw - force a portable anymap into plain format

SYNOPSIS

pnmnoraw [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Writes it out in plain (non-raw) format. This is fairly useless if you haven't defined the PBMPLUS_RAWBITS compile-time option.

SEE ALSO

pnm(5)

AUTHOR

NAME

pnmpad - add borders to portable anymap

SYNOPSIS

pnmpad [-white|-black] [-l#] [-r#] [-t#] [-b#] [pnmfile]

DESCRIPTION

Reads a portable anymap as input. Outputs a portable anymap with extra borders of the sizes specified. The colour of the borders can be set to black or white (default black).

SEE ALSO

pbmmake(1), pnmpaste(1), pbm(5)

AUTHOR

NAME

pnmpaste - paste a rectangle into a portable anymap

SYNOPSIS

pnmpaste [**-replace**|**-or**|**-and** |**-xor**] *frompnmfile x y* [*intopnmfile*]

DESCRIPTION

Reads two portable anymaps as input. Inserts the first anymap into the second at the specified location, and produces a portable anymap the same size as the second as output. If the second anymap is not specified, it is read from stdin. The *x* and *y* can be negative, in which case they are interpreted relative to the right and bottom of the anymap, respectively.

This tool is most useful in combination with *pnmcut.* For instance, if you want to edit a small segment of a large image, and your image editor cannot edit the large image, you can cut out the segment you are interested in, edit it, and then paste it back in.

Another useful companion tool is *pbmmask.*

The optional flag specifies the operation to use when doing the paste. The default is **-replace**. The other, logical operations are only allowed if both input images are bitmaps. These operations act as if white is TRUE and black is FALSE.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pnmcut(1), pnminvert(1), pnmarith(1), pnm(5), pbmmask(1)

AUTHOR

NAME

pnmrotate - rotate a portable anymap by some angle

SYNOPSIS

pnmrotate [**-noantialias**] *angle* [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Rotates it by the specified angle and produces a portable anymap as output. If the input file is in color, the output will be too, otherwise it will be grayscale. The angle is in degrees (floating point), measured counter-clockwise. It can be negative, but it should be between -90 and 90. Also, for rotations greater than 45 degrees you may get better results if you first use *pnmflip* to do a 90 degree rotation and then *pnmrotate* less than 45 degrees back the other direction

The rotation algorithm is Alan Paeth's three-shear method. Each shear is implemented by looping over the source pixels and distributing fractions to each of the destination pixels. This has an "anti-aliasing" effect - it avoids jagged edges and similar artifacts. However, it also means that the original colors or gray levels in the image are modified. If you need to keep precisely the same set of colors, you can use the **-noantialias** flag. This does the shearing by moving pixels without changing their values. If you want anti-aliasing and don't care about the precise colors, but still need a limited *number* of colors, you can run the result through *ppmquant*.

All flags can be abbreviated to their shortest unique prefix.

REFERENCES

"A Fast Algorithm for General Raster Rotation" by Alan Paeth, Graphics Interface '86, pp. 77-81.

SEE ALSO

pnmshear(1), pnmflip(1), pnm(5), ppmquant(1)

AUTHOR

NAME

pnmscale - scale a portable anymap

SYNOPSIS

pnmscale *s* [*pnmfile*]
pnmscale -xsize|**-width**|**-ysize**| **-height** *s* [*pnmfile*]
pnmscale -xscale|**-yscale** *s* [*pnmfile*]
pnmscale -xscale|**-xsize**|**-width** *s* **-yscale**|**-ysize**|**-height** *s* [*pnmfile*]
pnmscale -xysize *x y* [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Scales it by the specified factor or factors and produces a portable anymap as output. If the input file is in color, the output will be too, otherwise it will be grayscale. You can both enlarge (scale factor > 1) and reduce (scale factor < 1).

You can specify one dimension as a pixel size, and the other dimension will be scaled correspondingly.

You can specify one dimension as a scale, and the other dimension will not be scaled.

You can specify different sizes or scales for each axis.

Or, you can use the special **-xysize** flag, which fits the image into the specified size without changing the aspect ratio.

All flags can be abbreviated to their shortest unique prefix.

If you enlarge by a factor of 3 or more, you should probably add a *pnmsmooth* step; otherwise, you can see the original pixels in the resulting image.

SEE ALSO

pbmreduce(1), pnmenlarge(1), pnmsmooth(1), pnm(5)

AUTHOR

NAME

pnmshear - shear a portable anymap by some angle

SYNOPSIS

pnmshear [**-noantialias**] *angle* [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Shears it by the specified angle and produces a portable anymap as output. If the input file is in color, the output will be too, otherwise it will be grayscale. The angle is in degrees (floating point), and measures this:

```
+------+ +------+
|      | |\      \
| OLD  | | \ NEW  \
|      | |an\      \
+------+ |gle+------+
```

If the angle is negative, it shears the other way:

```
+------+ |-an+------+
|      | |gl/      /
| OLD  | |e/ NEW  /
|      | |/      /
+------+ +------+
```

The angle should not get too close to 90 or -90, or the resulting anymap will be unreasonably wide.

The shearing is implemented by looping over the source pixels and distributing fractions to each of the destination pixels. This has an "anti-aliasing" effect - it avoids jagged edges and similar artifacts. However, it also means that the original colors or gray levels in the image are modified. If you need to keep precisely the same set of colors, you can use the **-noantialias** flag. This does the shearing by moving pixels without changing their values. If you want anti-aliasing and don't care about the precise colors, but still need a limited *number* of colors, you can run the result through *ppmquant*.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pnmrotate(1), pnmflip(1), pnm(5), ppmquant(1)

AUTHOR

NAME

pnmtile - replicate a portable anymap into a specified size

SYNOPSIS

pnmtile *width height* [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Replicates it until it is the specified size, and produces a portable anymap as output.

SEE ALSO

pnm(5)

AUTHOR

NAME

pnmtofits - convert a portable anymap into FITS format

SYNOPSIS

pnmtofits [–**max** *f*] [–**min** *f*] [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Produces a FITS (Flexible Image Transport System) file as output. The resolution of the output file is either 8 bits/pixel, or 16 bits/pixel, depending on the value of maxval in the input file. If the input file is a portable bitmap or a portable graymap, the output file consists of a single plane image (NAXIS = 2). If instead the input file is a portable pixmap, the output file will consist of a three-plane image (NAXIS = 3, NAXIS3 = 3). A full description of the FITS format can be found in Astronomy & Astrophysics Supplement Series 44 (1981), page 363.

OPTIONS

Flags –**min** and –**max** can be used to set DATAMAX, DATAMIN, BSCALE and BZERO in the FITS header, but do not cause the data to be rescaled.

SEE ALSO

fitstopnm(1), pgm(5)

AUTHOR

NAME

pnmtops - convert portable anymap to PostScript

SYNOPSIS

pnmtops [**-scale** *s*] [**-turn**|**-noturn**] [**-rle**|**-runlength**] [**-dpi** *n*] [**-width** *n*] [**-height** *n*] [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Produces Encapsulated PostScript as output.

If the input file is in color (PPM), a color PostScript file gets written. Some PostScript interpreters can't handle color PostScript. If you have one of these you will need to run your image through *ppmtopgm* first.

Note that there is no pstopnm tool - this transformation is one-way, because a pstopnm tool would be a full-fledged PostScript interpreter, which is beyond the scope of this package. However, see the *psidtopgm* tool, which can read grayscale non-runlength PostScript image data. Also, if you're willing to install the fairly large GhostScript package, it comes with a pstoppm script.

OPTIONS

The **-scale** flag controls the scale of the result. The default scale is 1, which on a 300 dpi printer such as the Apple LaserWriter makes the output look about the same size as the input would if it was displayed on a typical 72 dpi screen. To get one PNM pixel per 300 dpi printer pixel, use "-scale 0.25".

The **-turn** and **-noturn** flags control whether the image gets turned 90 degrees. Normally, if an image is wider than it is tall, it gets turned automatically to better fit the page. If the **-turn** flag is specified, it will be turned no matter what its shape; and if the **-noturn** flag is specified, it will *not* be turned no matter what its shape.

The **-rle** or **-runlength** flag specifies run-length compression. This may save time if the host-to-printer link is slow; but normally the printer's processing time dominates, so **-rle** makes things slower.

The **-dpi** flag lets you specify the dots per inch of your output device. The default is 300 dpi. In theory PostScript is device-independent and you don't have to worry about this, but in practice its raster rendering can have unsightly bands if the device pixels and the image pixels aren't in sync.

The **-width** and **-height** flags let you specify the size of the page. The default is 8.5 inches by 11 inches.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pnm(5), psidtopgm(1)

AUTHOR

NAME

pnmtorast - convert a portable pixmap into a Sun rasterfile

SYNOPSIS

pnmtorast [**-standard**|**-rle**] [*pnmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces a Sun rasterfile as output.

Color values in Sun rasterfiles are eight bits wide, so *pnmtorast* will automatically scale colors to have a maxval of 255. An extra *pnmdepth* step is not necessary.

OPTIONS

The **-standard** flag forces the result to be in RT_STANDARD form; the **-rle** flag, RT_BYTE_ENCODED, which is smaller but, well, less standard. The default is **-rle**.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

rasttopnm(1), pnm(5)

AUTHOR

NAME

pnmtosir - convert a portable anymap into a Solitaire format

SYNOPSIS

pnmtosir [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Produces a Solitaire Image Recorder format.

pnmtosir produces an MGI TYPE 17 file for *pbm* and *pgm* files. For *ppm*, it writes a MGI TYPE 11 file.

SEE ALSO

sirtopnm(1), pnm(5)

BUGS

AUTHOR

NAME

pnmtotiff - convert a a portable anymap into a TIFF file

SYNOPSIS

pnmtotiff [**-none**|**-packbits**| **-lzw**|**-g3**|**-g4**] [**-2d**] [**-fill**] [**-predictor** *n*] [**-msb2lsb**|**-lsb2msb**] [**-rowsperstrip** *n*] [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Produces a TIFF file as output.

OPTIONS

By default, *pnmtotiff* creates a TIFF file with LZW compression. This is your best bet most of the time. However, some TIFF readers can't deal with it. If you want to try another compression scheme or tweak some of the other even more obscure output options, there are a number of flags to play with.

The **-none**, **-packbits**, **-lzw**, **-g3**, and **-g4** options are used to override the default and set the compression scheme used in creating the output file. The CCITT Group 3 and Group 4 compression algorithms can only be used with bilevel data. The **-2d** and **-fill** options are meaningful only with Group 3 compression: **-2d** requests 2-dimensional encoding, while **-fill** requests that each encoded scanline be zero-filled to a byte boundry. The **-predictor** option is only meaningful with LZW compression: a predictor value of 2 causes each scanline of the output image to undergo horizontal differencing before it is encoded; a value of 1 forces each scanline to be encoded without differencing. By default, *pnmtotiff* creates a TIFF file with msb-to-lsb fill order. The **-msb2lsb** and **-lsb2msb** options are used to override the default and set the fill order used in creating the file. The **-rowsperstrip** option can be used to set the number of rows (scanlines) in each strip of data in the output file. By default, the output file has the number of rows per strip set to a value that will ensure each strip is no more than 8 kilobytes long.

BUGS

This program is not self-contained. To use it you must fetch the TIFF Software package listed in the OTHER.SYSTEMS file and configure PBMPLUS to use libtiff. See PBMPLUS's Makefile for details on this configuration.

SEE ALSO

tifftopnm(1), pnm(5)

AUTHOR

Derived by Jef Poskanzer from ras2tiff.c, which is ©1990 by Sun Microsystems, Inc.
Author: Patrick J. Naughton (naughton@wind.sun.com).

NAME

pnmtoxwd - convert a portable anymap into an X11 window dump

SYNOPSIS

pnmtoxwd [**-pseudodepth** *n*] [**-directcolor**] [*pnmfile*]

DESCRIPTION

Reads a portable anymap as input. Produces an X11 window dump as output. This window dump can be displayed using the xwud tool.

Normally, pnmtoxwd produces a StaticGray dump file for *pbm* and *pgm* files. For *ppm*, it writes a PseudoColor dump file if there are up to 256 colors in the input, and a DirectColor dump file otherwise. The **-directcolor** flag can be used to force a DirectColor dump. And the **-pseudodepth** flag can be used to change the depth of PseudoColor dumps from the default of 8 bits / 256 colors.

SEE ALSO

xwdtopnm(1), pnm(5), xwud(1)

AUTHOR

NAME

ppm - portable pixmap file format

DESCRIPTION

The portable pixmap format is a lowest common denominator color image file format. The definition is as follows:

- A "magic number" for identifying the file type. A ppm file's magic number is the two characters "P3".
- Whitespace (blanks, TABs, CRs, LFs).
- A width, formatted as ASCII characters in decimal.
- Whitespace.
- A height, again in ASCII decimal.
- Whitespace.
- The maximum color-component value, again in ASCII decimal.
- Whitespace.
- Width * height pixels, each three ASCII decimal values between 0 and the specified maximum value, starting at the top-left corner of the pixmap, proceeding in normal English reading order. The three values for each pixel represent red, green, and blue, respectively; a value of 0 means that color is off, and the maximum value means that color is maxxed out.
- Characters from a "#" to the next end-of-line are ignored (comments).
- No line should be longer than 70 characters.

Here is an example of a small pixmap in this format:

```
P3
# feep.ppm
4 4
15
 0  0  0    0  0  0    0  0  0   15  0 15
 0  0  0    0 15  7    0  0  0    0  0  0
 0  0  0    0  0  0    0 15  7    0  0  0
15  0 15    0  0  0    0  0  0    0  0  0
```

Programs that read this format should be as lenient as possible, accepting anything that looks remotely like a pixmap.

There is also a variant on the format, available by setting the RAWBITS option at compile time. This variant is different in the following ways:

- The "magic number" is "P6" instead of "P3".
- The pixel values are stored as plain bytes, instead of ASCII decimal.
- Whitespace is not allowed in the pixels area, and only a single character of whitespace (typically a newline) is allowed after the maxval.

- The files are smaller and many times faster to read and write.

Note that this raw format can only be used for maxvals less than or equal to 255. If you use the *ppm* library and try to write a file with a larger maxval, it will automatically fall back on the slower but more general plain format.

SEE ALSO

giftoppm(1), gouldtoppm(1), ilbmtoppm(1), imgtoppm(1), mtvtoppm(1), pcxtoppm(1), pgmtoppm(1), pi1toppm(1), picttoppm(1), pjtoppm(1), qrttoppm(1), rawtoppm(1), rgb3toppm(1), sldtoppm(1), spctoppm(1), sputoppm(1), tgatoppm(1), ximtoppm(1), xpmtoppm(1), yuvtoppm(1), ppmtoacad(1), ppmtogif(1), ppmtoicr(1), ppmtoilbm(1), ppmtopcx(1), ppmtopgm(1), ppmtopi1(1), ppmtopict(1), ppmtopj(1), ppmtopuzz(1), ppmtorgb3(1), ppmtosixel(1), ppmtotga(1), ppmtouil(1), ppmtoxpm(1), ppmtoyuv(1), ppmdither(1), ppmforge(1), ppmhist(1), ppmmake(1), ppmpat(1), ppmquant(1), ppmquantall(1), ppmrelief(1), pnm(5), pgm(5), pbm(5)

AUTHOR

NAME

ppmbrighten - change an images Saturation and Value from an HSV map

SYNOPSIS

ppmbrighten [-n] [-s <+- saturation>] [-v <+- value>] <ppmfile>

DESCRIPTION

Reads a portable pixmap as input. Converts the image from RGB space to HSV space and changes the Value by <+- value> as a percentage. Likewise with the Saturation. Doubling the Value would involve

ppmbrighten -v 100

to add 100 percent to the Value.

The 'n' option normalizes the Value to exist between 0 and 1 (normalized).

SEE ALSO

pgmnorm(1), ppm(5)

AUTHOR

NOTES

This program does not change the number of colors.

NAME

ppmchange - change all pixels of one color to another in a portable pixmap

SYNOPSIS

ppmchange *oldcolor newcolor [...]* [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Changes all pixels of *oldcolor* to *newcolor*, leaving all others unchanged. Up to 256 colors may be replaced by specifying couples of colors on the command line.

The colors can be specified in five ways:

- o A name, assuming that a pointer to an X11-style color names file was compiled in.
- o An X11-style hexadecimal specifier: rgb:r/g/b, where r g and b are each 1- to 4-digit hexadecimal numbers.
- o An X11-style decimal specifier: rgbi:r/g/b, where r g and b are floating point numbers between 0 and 1.
- o For backwards compatibility, an old-X11-style hexadecimal number: #rgb, #rrggbb, #rrrgggbbb, or #rrrrggggbbbb.
- o For backwards compatibility, a triplet of numbers separated by commas: r,g,b, where r g and b are floating point numbers between 0 and 1. (This style was added before MIT came up with the similar rgbi style.)

SEE ALSO

pgmtoppm(1), ppm(5)

AUTHOR

Wilson H. Bent. Jr. (whb@usc.edu) with modifications by Alberto Accomazzi (alberto@cfa.harvard.edu)

NAME

ppmdim - dim a portable pixmap down to total blackness

SYNOPSIS

ppmdim *dimfactor* [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Diminishes its brightness by the specified dimfactor down to total blackness. The dimfactor may be in the range from 0.0 (total blackness, deep night, nada, null, nothing) to 1.0 (original picture's brightness).

As *pnmgamma* does not do the brightness correction in the way I wanted it, this small program was written.

ppmdim is similar to *ppmbrighten* , but not exactly the same.

SEE ALSO

ppm(5), ppmflash(1), pnmgamma(1), ppmbrighten(1)

AUTHOR

Copyright (C) 1993 by Frank Neumann

NAME

ppmdist - simplistic grayscale assignment for machine generated, color images

SYNOPSIS

ppmdist [**-intensity**|**-frequency**] [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input, performs a simplistic grayscale assignment intended for use with grayscale or bitmap printers.

Often conversion from ppm to pgm will yield an image with contrast too low for good printer output. The program maximizes contrast between the gray levels output.

A ppm input of n colors is read, and a pgm of n gray levels is written. The gray levels take on the values 0..n-1, while maxval takes on n-1.

The mapping from color to stepped grayscale can be performed in order of input pixel intensity, or input pixel frequency (number of repetitions).

OPTIONS

-frequency Sort input colors by the number of times a color appears in the input, before mapping to evenly distributed graylevels of output. **-intensity** Sort input colors by their grayscale intensity, before mapping to evenly distributed graylevels of output. This is the default.

BUGS

Helpful only for images with a very small number of colors. Perhaps should have been an option to ppmtopgm(1).

SEE ALSO

ppmtopgm(1), ppmhist(1), ppm(5)

AUTHOR

NAME

ppmdither - ordered dither for color images

SYNOPSIS

ppmdither [**-dim** *dimension*] [**-red** *shades*] [**-green** *shades*] [**-blue** *shades*] [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input, and applies dithering to it to reduce the number of colors used down to the specified number of shades for each primary. The default number of shades is red=5, green=9, blue=5, for a total of 225 colors. To convert the image to a binary rgb format suitable for color printers, use -red 2 -green 2 -blue 2. The maximum number of colors that can be used is 256 and can be computed as the product of the number of red, green and blue shades.

OPTIONS

-dim *dimension* The size of the dithering matrix. Must be a power of 2.

-red *shades* The number of red shades to be used; minimum of 2.

-green *shades* The number of green shades to be used; minimum of 2.

-blue *shades* The number of blue shades to be used; minimum of 2.

SEE ALSO

pnmdepth(1), ppmquant(1), ppm(5)

AUTHOR

NAME

ppmflash - brighten a picture up to complete white-out

SYNOPSIS

ppmflash *flashfactor* [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Increases its brightness by the specified flashfactor up to a total white-out image. The flashfactor may be in the range from 0.0 (original picture's brightness) to 1.0 (full white-out, The Second After).

As *pnmgamma* does not do the brightness correction in the way I wanted it, this small program was written.

This program is similar to *ppmbrighten* , but not exactly the same.

SEE ALSO

ppm(5), ppmdim(1), pnmgamma(1), ppmbrighten(1)

AUTHOR

Copyright (C) 1993 by Frank Neumann

NAME

ppmforge - fractal forgeries of clouds, planets, and starry skies

SYNOPSIS

ppmforge [**-clouds**] 'in +9n [**-night**] [**-dimension** *dimen*] [**-hour** *hour*] [**-inclination**|**-tilt** *angle*] [**-mesh** *size*] [**-power** *factor*] [**-glaciers** *level*] [**-ice** *level*] [**-saturation** *sat*] [**-seed** *seed*] [**-stars** *fraction*] [**-xsize**|**-width** *width*] [**-ysize**|**-height** *height*]

DESCRIPTION

ppmforge generates three kinds of "random fractal forgeries," the term coined by Richard F. Voss of the IBM Thomas J. Watson Research Center for seemingly realistic pictures of natural objects generated by simple algorithms embodying randomness and fractal self-similarity. The techniques used by **ppmforge** are essentially those given by Voss[1], particularly the technique of spectral synthesis explained in more detail by Dietmar Saupe[2].

The program generates two varieties of pictures: planets and clouds, which are just different renderings of data generated in an identical manner, illustrating the unity of the fractal structure of these very different objects. A third type of picture, a starry sky, is synthesised directly from pseudorandom numbers.

The generation of planets or clouds begins with the preparation of an array of random data in the frequency domain. The size of this array, the "mesh size," can be set with the **-mesh** option; the larger the mesh the more realistic the pictures but the calculation time and memory requirement increases as the square of the mesh size. The fractal dimension, which you can specify with the **-dimension** option, determines the roughness of the terrain on the planet or the scale of detail in the clouds. As the fractal dimension is increased, more high frequency components are added into the random mesh.

Once the mesh is generated, an inverse two dimensional Fourier transform is performed upon it. This converts the original random frequency domain data into spatial amplitudes. We scale the real components that result from the Fourier transform into numbers from 0 to 1 associated with each point on the mesh. You can further modify this number by applying a "power law scale" to it with the **-power** option. Unity scale leaves the numbers unmodified; a power scale of 0.5 takes the square root of the numbers in the mesh, while a power scale of 3 replaces the numbers in the mesh with their cubes. Power law scaling is best envisioned by thinking of the data as representing the elevation of terrain; powers less than 1 yield landscapes with vertical scarps that look like glacially-carved valleys; powers greater than one make fairy-castle spires (which require large mesh sizes and high resolution for best results).

After these calculations, we have a array of the specified size containing numbers that range from 0 to 1. The pixmaps are generated as follows:

Clouds A colour map is created that ranges from pure blue to white by increasing admixture (desaturation) of blue with white. Numbers less than 0.5 are coloured blue, numbers between 0.5 and 1.0 are coloured with corresponding levels of white, with 1.0 being pure white.

Planet The mesh is projected onto a sphere. Values less than 0.5 are treated as water and values between 0.5 and 1.0 as land. The water areas are coloured based upon the water depth, and land based on its elevation. The random depth data are used to create clouds over the oceans. An atmosphere approximately like the Earth's is simulated; its light absorption is calculated to create a blue cast around the limb of the planet. A function that rises from 0 to 1 based on latitude is modulated by the local elevation to generate polar ice caps–high altitude terrain carries glaciers farther from the pole. Based on the position of the star with respect to the observer, the apparent colour of each pixel of the planet is calculated by ray-tracing from the star to the planet to the observer and applying a lighting model that sums ambient light and diffuse reflection (for most planets ambient light is zero, as their primary star is the only source of illumination). Additional random data are used to generate stars around the planet.

Night A sequence of pseudorandom numbers is used to generate stars with a user specified density.

Cloud pictures always contain 256 or fewer colours and may be displayed on most colour mapped devices without further processing. Planet pictures often contain tens of thousands of colours which must be compressed with **ppmquant** or **ppmdither** before encoding in a colour mapped format. If the display resolution is high enough, **ppmdither** generally produces better looking planets. **ppmquant** tends to create discrete colour bands, particularly in the oceans, which are unrealistic and distracting. The number of colours in starry sky pictures generated with the **-night** option depends on the value specified for **-saturation**. Small values limit the colour temperature distribution of the stars and reduce the number of colours in the image. If the **-saturation** is set to 0, none of the stars will be coloured and the resulting image will never contain more than 256 colours. Night sky pictures with many different star colours often look best when colour compressed by **pnmdepth** rather than **ppmquant** or **ppmdither**. Try *newmaxval* settings of 63, 31, or 15 with **pnmdepth** to reduce the number of colours in the picture to 256 or fewer.

OPTIONS

-clouds Generate clouds. A pixmap of fractal clouds is generated. Selecting clouds sets the default for fractal dimension to 2.15 and power scale factor to 0.75.

-dimension *dimen* Sets the fractal dimension to the specified *dimen*, which may be any floating point value between 0 and 3. Higher fractal dimensions create more "chaotic" images, which require higher resolution output and a larger FFT mesh size to look good. If no dimension is specified, 2.4 is used when generating planets and 2.15 for clouds.

-glaciers *level* The floating point *level* setting controls the extent to which terrain elevation causes ice to appear at lower latitudes. The default value of 0.75 makes the polar caps extend toward the equator across high terrain and forms glaciers in the highest mountains, as on Earth. Higher values make ice sheets that cover more and more of the land surface, simulating planets in the midst of an ice age. Lower values tend to be boring, resulting in unrealistic geometrically-precise ice cap boundaries.

-hour *hour* When generating a planet, *hour* is used as the "hour angle at the central meridian." If you specify **-hour 12**, for example, the planet will be fully illuminated, corresponding to high noon at the longitude at the centre of the screen. You can specify any floating point value between 0 and 24 for *hour*, but values which place most of the planet in darkness (0 to 4 and 20 to 24) result in crescents which, while pretty, don't give you many illuminated pixels for the amount of computing that's required. If no **-hour** option is specified, a random hour angle is chosen, biased so that only 25% of the images generated will be crescents.

-ice *level* Sets the extent of the polar ice caps to the given floating point *level*. The default level of 0.4 produces ice caps similar to those of the Earth. Smaller values reduce the amount of ice, while larger **-ice** settings create more prominent ice caps. Sufficiently large values, such as 100 or more, in conjunction with small settings for **-glaciers** (try 0.1) create "ice balls" like Europa.

-inclination|-tilt *angle* The inclination angle of the planet with regard to its primary star is set to *angle*, which can be any floating point value from -90 to 90. The inclination angle can be thought of as specifying, in degrees, the "season" the planet is presently experiencing or, more precisely, the latitude at which the star transits the zenith at local noon. If 0, the planet is at equinox; the star is directly overhead at the equator. Positive values represent summer in the northern hemisphere, negative values summer in the southern hemisphere. The Earth's inclination angle, for example, is about 23.5 at the June solstice, 0 at the equinoxes in March and September, and -23.5 at the December solstice. If no inclination angle is specified, a random value between -21.6 and 21.6 degrees is chosen.

-mesh *size* A mesh of *size* by *size* will be used for the fast Fourier transform (FFT). Note that memory requirements and computation speed increase as the square of *size*; if you double the mesh size, the program will use four times the memory and run four times as long. The default mesh is 256x256, which produces reasonably good looking pictures while using half a megabyte for the 256x256 array of single precision complex numbers required by the FFT. On machines with limited memory capacity, you may have to reduce the mesh size to avoid running out of RAM. Increasing the mesh size produces better looking pictures; the difference becomes particularly noticeable when generating high resolution images with relatively high fractal dimensions (between 2.2 and 3).

-night A starry sky is generated. The stars are created by the same algorithm used for the stars that surround planet pictures, but the output consists exclusively of stars.

-power *factor* Sets the "power factor" used to scale elevations synthesised from the FFT to *factor*, which can be any floating point number greater than zero. If no factor is specified a default of 1.2 is used if a planet is being generated, or 0.75 if clouds are selected by the **-clouds** option. The result of the FFT image synthesis is an array of elevation values between 0 and 1. A non-unity power factor exponentiates each of these elevations to the specified power. For example, a power factor of 2 squares each value, while a power factor of 0.5 replaces each with its square root. (Note that exponentiating values between 0 and 1 yields values that remain within that range.) Power factors less than 1 emphasise large-scale elevation changes at the expense of small variations. Power factors greater than 1 increase the roughness of the terrain and, like high fractal dimensions, may require a larger FFT mesh size and/or higher screen resolution to look good.

-saturation *sat* Controls the degree of colour saturation of the stars that surround planet pictures and fill starry skies created with the **-night** option. The default value of 125 creates stars which resemble the sky as seen by the human eye from Earth's surface. Stars are dim; only the brightest activate the cones in the human retina, causing colour to be perceived. Higher values of *sat* approximate the appearance of stars from Earth orbit, where better dark adaptation, absence of skyglow, and the concentration of light from a given star onto a smaller area of the retina thanks to the lack of atmospheric turbulence enhances the perception of colour. Values greater than 250 create "science fiction" skies that, while pretty, don't occur in this universe.

Thanks to the inverse square law combined with Nature's love of mediocrity, there are many, many dim stars for every bright one. This population relationship is accurately reflected in the skies created by **ppmforge**. Dim, low mass stars live much longer than bright massive stars, consequently there are many reddish stars for every blue giant. This relationship is preserved by **ppmforge**. You can reverse the proportion, simulating the sky as seen in a starburst galaxy, by specifying a negative *sat* value.

-seed *num* Sets the seed for the random number generator to the integer *num*. The seed used to create each picture is displayed on standard output (unless suppressed with the **-quiet**

option). Pictures generated with the same seed will be identical. If no **-seed** is specified, a random seed derived from the date and time will be chosen. Specifying an explicit seed allows you to re-render a picture you particularly like at a higher resolution or with different viewing parameters.

-stars *fraction* Specifies the percentage of pixels, in tenths of a percent, which will appear as stars, either surrounding a planet or filling the entire frame if **-night** is specified. The default *fraction* is 100.

-xsize|-width *width* Sets the width of the generated image to *width* pixels. The default width is 256 pixels. Images must be at least as wide as they are high; if a width less than the height is specified, it will be increased to equal the height. If you must have a long skinny pixmap, make a square one with **ppmforge**, then use **pnmcut** to extract a portion of the shape and size you require.

-ysize|-height *height* Sets the height of the generated image to *height* pixels. The default height is 256 pixels. If the height specified exceeds the width, the width will be increased to equal the height.

All flags can be abbreviated to their shortest unique prefix.

BUGS

The algorithms require the output pixmap to be at least as wide as it is high, and the width to be an even number of pixels. These constraints are enforced by increasing the size of the requested pixmap if necessary.

You may have to reduce the FFT mesh size on machines with 16 bit integers and segmented pointer architectures.

SEE ALSO

pnmcut(1), **pnmdepth**(1), **ppmdither**(1), **ppmquant**(1), **ppm**(5)

[1] Voss, Richard F., "Random Fractal Forgeries," in Earnshaw *et. al.*, Fundamental Algorithms for Computer Graphics, Berlin: Springer-Verlag, 1985.

[2] Peitgen, H.-O., and Saupe, D. eds., The Science Of Fractal Images, New York: Springer Verlag, 1988.

AUTHOR

John Walker
Autodesk SA
Avenue des Champs-Montants 14b
CH-2074 MARIN
Suisse/Schweiz/Svizzera/Svizra/Switzerland

Usenet: kelvin@Autodesk.com

Fax: 038/33 88 15

Voice: 038/33 76 33

PLUGWARE! If you like this kind of stuff, you may also enjoy "James Gleick's Chaos–The Software" for MS-DOS, available for $59.95 from your local software store or directly from Autodesk, Inc., Attn: Science Series, 2320 Marinship Way, Sausalito, CA 94965, USA. Telephone: (800) 688-2344 toll-free or, outside the U.S. (415) 332-2344 Ext 4886. Fax: (415) 289-4718. "Chaos–The Software" includes a more comprehensive fractal forgery generator which creates three-dimensional landscapes as well as clouds and planets, plus five more modules which explore other aspects of Chaos. The user guide of more than 200 pages includes an introduction by James Gleick and detailed explanations by Rudy Rucker of the mathematics and algorithms used by each program.

NAME

ppmhist - print a histogram of a portable pixmap

SYNOPSIS

ppmhist [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Generates a histogram of the colors in the pixmap.

SEE ALSO

ppm(5), pgmhist(1)

AUTHOR

NAME

ppmmake - create a pixmap of a specified size and color

SYNOPSIS

ppmmake *color width height*

DESCRIPTION

Produces a portable pixmap of the specified color, width, and height.

The color can be specified in five ways:

- A name, assuming that a pointer to an X11-style color names file was compiled in.
- An X11-style hexadecimal specifier: rgb:r/g/b, where r g and b are each 1- to 4-digit hexadecimal numbers.
- An X11-style decimal specifier: rgbi:r/g/b, where r g and b are floating point numbers between 0 and 1.
- For backwards compatibility, an old-X11-style hexadecimal number: #rgb, #rrggbb, #rrrgggbbb, or #rrrrggggbbbb.
- For backwards compatibility, a triplet of numbers separated by commas: r,g,b, where r g and b are floating point numbers between 0 and 1. (This style was added before MIT came up with the similar rgbi style.)

SEE ALSO

ppm(5), pbmmake(1)

AUTHOR

NAME

ppmmix - blend together two portable pixmaps

SYNOPSIS

ppmmix *fadefactor ppmfile1 ppmfile2*

DESCRIPTION

Reads two portable pixmaps as input. Mixes them together using the specified fade factor. The fade factor may be in the range from 0.0 (only ppmfile1's image data) to 1.0 (only ppmfile2's image data). Anything in between gains a smooth blend between the two images.

The two pixmaps must have the same size.

SEE ALSO

ppm(5)

AUTHOR

©1993 by Frank Neumann.

NAME

ppmpat - make a pretty pixmap

SYNOPSIS

ppmpat -gingham2|-g2|-gingham3| -g3|-madras|-tartan| -poles|-squig|-camo| -anticamo *width height*

DESCRIPTION

Produces a portable pixmap of the specified width and height, with a pattern in it.

This program is mainly to demonstrate use of the ppmdraw routines, a simple but powerful drawing library. See the ppmdraw.h include file for more info on using these routines. Still, some of the patterns can be rather pretty. If you have a color workstation, something like **ppmpat -squig 300 300 | ppmquant 128** should generate a nice background.

OPTIONS

The different flags specify various different pattern types:

-gingham2 A gingham check pattern. Can be tiled.

-gingham3 A slightly more complicated gingham. Can be tiled.

-madras A madras plaid. Can be tiled.

-tartan A tartan plaid. Can be tiled.

-poles Color gradients centered on randomly-placed poles. May need to be run through *ppmquant*.

-squig Squiggley tubular pattern. Can be tiled. May need to be run through *ppmquant*.

-camo Camouflage pattern. May need to be run through *ppmquant*.

-anticamo Anti-camouflage pattern - like -camo, but ultra-bright colors. May need to be run through *ppmquant*.

All flags can be abbreviated to their shortest unique prefix.

REFERENCES

Some of the patterns are from "Designer's Guide to Color 3" by Jeanne Allen.

SEE ALSO

pnmtile(1), ppmquant(1), ppm(5)

AUTHOR

NAME

ppmquant - quantize the colors in a portable pixmap down to a specified number

SYNOPSIS

ppmquant [**-floyd**|**-fs**] *ncolors* [*ppmfile*]
ppmquant [**-floyd**|**-fs**] **-map** *mapfile* [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Chooses *ncolors* colors to best represent the image, maps the existing colors to the new ones, and writes a portable pixmap as output.

The quantization method is Heckbert's "median cut".

Alternately, you can skip the color-choosing step by specifying your own set of colors with the **-map** flag. The *mapfile* is just a *ppm* file; it can be any shape, all that matters is the colors in it. For instance, to quantize down to the 8-color IBM TTL color set, you might use:

```
P3
8 1
255
  0   0   0
255   0   0
  0 255   0
  0   0 255
255 255   0
255   0 255
  0 255 255
255 255 255
```

If you want to quantize one pixmap to use the colors in another one, just use the second one as the mapfile. You don't have to reduce it down to only one pixel of each color, just use it as is.

The **-floyd/-fs** flag enables a Floyd-Steinberg error diffusion step. Floyd-Steinberg gives vastly better results on images where the unmodified quantization has banding or other artifacts, especially when going to a small number of colors such as the above IBM set. However, it does take substantially more CPU time, so the default is off.

All flags can be abbreviated to their shortest unique prefix.

REFERENCES

"Color Image Quantization for Frame Buffer Display" by Paul Heckbert, SIGGRAPH '82 Proceedings, page 297.

SEE ALSO

ppmquantall(1), pnmdepth(1), ppmdither(1), ppm(5)

AUTHOR

NAME

ppmquantall - run ppmquant on a bunch of files all at once, so they share a common colormap

SYNOPSIS

ppmquantall *ncolors ppmfile* ...

DESCRIPTION

Takes a bunch of portable pixmap as input. Chooses *ncolors* colors to best represent all of the images, maps the existing colors to the new ones, and **overwrites the input files** with the new quantized versions.

Verbose explanation: Let's say you've got a dozen pixmaps that you want to display on the screen all at the same time. Your screen can only display 256 different colors, but the pixmaps have a total of a thousand or so different colors. For a single pixmap you solve this problem with *ppmquant*; this script solves it for multiple pixmaps. All it does is concatenate them together into one big pixmap, run *ppmquant* on that, and then split it up into little pixmaps again.

(Note that another way to solve this problem is to pre-select a set of colors and then use *ppmquant*'s **-map** option to separately quantize each pixmap to that set.)

SEE ALSO

ppmquant(1), ppm(5)

BUGS

It's a csh script. Csh scripts are not portable to System V. Scripts in general are not portable to non-Unix environments.

AUTHOR

'

NAME

ppmqvga - 8 plane quantization '

SYNOPSIS

ppmqvga [options] [input file] '

DESCRIPTION

ppmqvga quantizes PPM files to 8 planes, with optional Floyd-Steinberg dithering. Input is a PPM file from the file named, or standard input of no file is provided. **-d** dither. Apply Floyd-Steinberg dithering to the data

-q quiet. Produces no progress reporting, and no terminal output unless and error occurs.

-v verbose. Produces additional output describing the number of colors found, and some information on the resulting mapping. May be repeated to generate loads of internal table output, but generally only useful once.

EXAMPLES

ppmqvga -d my_image.ppm | ppmtogif >my_image.gif

tgatoppm zombie.tga | ppmqvga | ppmtotif > zombie.tif

SEE ALSO

ppmquant

DIAGNOSTICS

Error messages if problems, various levels of optional progress reporting.

LIMITATIONS

none known.

AUTHOR

Original by Lyle Rains (lrains@netcom.com) as ppmq256 and ppmq256fs combined, documented, and enhanced by Bill Davidsen (davidsen@crd.ge.com)

Copyright

NAME

ppmrelief - run a Laplacian relief filter on a portable pixmap

SYNOPSIS

ppmrelief [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Does a Laplacian relief filter, and writes a portable pixmap as output.

The Laplacian relief filter is described in "Beyond Photography" by Holzmann, equation 3.19. It's a sort of edge-detection.

SEE ALSO

pgmbentley(1), pgmoil(1), ppm(5)

AUTHOR

NAME

ppmshift - shift lines of a portable pixmap left or right by a random amount

SYNOPSIS

ppmshift *shift* [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Shifts every row of image data to the left or right by a certain amount. The 'shift' parameter determines by how many pixels a row is to be shifted at most.

Another one of those effects I intended to use for MPEG tests. Unfortunately, this program will not help me here - it creates too random patterns to be used for animations. Still, it might give interesting results on still images.

EXAMPLE

Check this out: Save your favourite model's picture from something like alt.binaries.pictures.supermodels (ok, or from any other picture source), convert it to ppm, and process it e.g. like this, assuming the picture is 800x600 pixels:

```
# take the upper half, and leave it like it is
pnmcut 0 0 800 300 cs.ppm >upper.ppm
# take the lower half, flip it upside down, dim it and distort it a little
pnmcut 0 300 800 300 cs.ppm | pnmflip -tb | ppmdim 0.7 |
   ppmshift 10 >lower.ppm
# and concatenate the two pieces
pnmcat -tb upper.ppm lower.ppm >newpic.ppm
```

The resulting picture looks like the image being reflected on a water surface with slight ripples.

SEE ALSO

ppm(5), pnmcut(1), pnmflip(1), ppmdim(1), pnmcat(1)

AUTHOR

©1993 by Frank Neumann

NAME

ppmspread - displace a portable pixmap's pixels by a random amount

SYNOPSIS

ppmspread *amount* [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Moves every pixel around a bit relative to its original position. amount determines by how many pixels a pixel is to be moved around at most.

Pictures processed with this filter will seem to be somewhat dissolved or unfocussed (although they appear more coarse than images processed by something like *pnmconvol*).

SEE ALSO

ppm(5), pnmconvol(1)

AUTHOR

NAME

ppmtoacad - convert portable pixmap to AutoCAD database or slide

SYNOPSIS

ppmtoacad 'in 15n [**-dxb**] [**-poly**] [**-background** *colour*] [**-white**] [**-aspect** *ratio*] [**-8**] [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces an AutoCADlide file or binary database import (.dxb) file as output. If no *ppmfile* is specified, input is read from standard input.

OPTIONS

-dxb An AutoCAD binary database import (.dxb) file is written. This file is read with the DXBIN command and, once loaded, becomes part of the AutoCAD geometrical database and can be viewed and edited like any other object. Each sequence of identical pixels becomes a separate object in the database; this can result in very large AutoCAD drawing files. However, if you want to trace over a bitmap, it lets you zoom and pan around the bitmap as you wish.

-poly If the **-dxb** option is not specified, the output of **ppmtoacad** is an AutoCAD slide file. Normally each row of pixels is represented by an AutoCAD line entity. If **-poly** is selected, the pixels are rendered as filled polygons. If the slide is viewed on a display with higher resolution than the source pixmap, this will cause the pixels to expand instead of appearing as discrete lines against the screen background colour. Regrettably, this representation yields slide files which occupy more disc space and take longer to display.

-background *colour* Most AutoCAD display drivers can be configured to use any available colour as the screen background. Some users perfer a black screen background, others white, while splinter groups advocate burnt ocher, tawny puce, and shocking grey. Discarding pixels whose closest AutoCAD colour representation is equal to the background colour can substantially reduce the size of the AutoCAD database or slide file needed to represent a bitmap. If no **-background** colour is specified, the screen background colour is assumed to be black. Any AutoCAD colour number may be specified as the screen background; colour numbers are assumed to specify the hues defined in the standard AutoCAD 256 colour palette.

-white Since many AutoCAD users choose a white screen background, this option is provided as a short-cut. Specifying **-white** is identical in effect to **-background 7**.

-aspect *ratio* If the source pixmap had non-square pixels, the ratio of the pixel width to pixel height should be specified as *ratio*. The resulting slide or .dxb file will be corrected so that pixels on the AutoCAD screen will be square. For example, to correct an image made for a 320x200 VGA/MCGA screen, specify **-aspect 0.8333**.

-8 Restricts the colours in the output file to the 8 RGB shades.

All flags can be abbreviated to their shortest unique prefix.

BUGS

AutoCAD has a fixed palette of 256 colours, distributed along the hue, lightness, and saturation axes. Pixmaps which contain many nearly-identical colours, or colours not closely approximated by AutoCAD's palette, may be poorly rendered.

ppmtoacad works best if the system displaying its output supports the full 256 colour AutoCAD palette. Monochrome, 8 colour, and 16 colour configurations will produce less than optimal results.

When creating a .dxb file or a slide file with the **-poly** option, **ppmtoacad** finds both vertical and horizontal runs of identical pixels and consolidates them into rectangular regions to reduce the size of the output file. This is effective for images with large areas of constant colour but it's no substitute for true raster to vector conversion. In particular, thin diagonal lines are not optimised at all by this process.

Output files can be huge.

SEE ALSO

AutoCAD Reference Manual: *Slide File Format* and *Binary Drawing Interchange (DXB) Files*, **ppm**(5)

AUTHOR

John Walker
Autodesk SA
Avenue des Champs-Montants 14b
CH-2074 MARIN
Suisse/Schweiz/Svizzera/Svizra/Switzerland

Usenet:	kelvin@Autodesk.com
Fax:	038/33 88 15
Voice:	038/33 76 33

NAME

ppmtobmp – convert a portable pixmap into a BMP file

SYNOPSIS

ppmtobmp [*–windows*] [*–os2*] [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces a Microsoft Windows or OS/2 BMP file as output.

OPTIONS

–**windows** Tells the program to produce a Microsoft Windows BMP file.

–**os2** Tells the program to produce an OS/2 BMP file. (This is the default.)

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

bmptoppm(1), ppm(5)

AUTHOR

©1992 by David W. Sanderson.

NAME

ppmtogif - convert a portable pixmap into a GIF file

SYNOPSIS

ppmtogif [**-interlace**] [**-sort**] [**-map** *mapfile]* [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces a GIF file as output.

OPTIONS

-interlace Tells the program to produce an interlaced GIF file.

-sort Produces a GIF file with a sorted color map.

-map *mapfile*

Uses the colors found in the *mapfile* to create the colormap in the GIF file, instead of the colors from *ppmfile.* The *mapfile* can be any *ppm* file; all that matters is the colors in it. If the colors in *ppmfile* do not match those in *mapfile* , they are matched to a "best match". A (much) better result can be obtained by using the following filter in advance:

ppmquant -floyd -map *mapfile*

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

giftoppm(1), ppmquant(1), ppm(5)

AUTHOR

Based on GIFENCOD by David Rowley (mgardi@watdcsu.waterloo.edu). Lempel-Ziv compression based on "compress".

NAME

ppmtoicr - convert a portable pixmap into NCSA ICR format

SYNOPSIS

ppmtoicr [**-windowname** *name*] [**-expand** *expand*] [**-display** *display*] [**-rle**] [*ppmfile*]

DESCRIPTION

Reads a portable pixmap file as input. Produces an NCSA Telnet Interactive Color Raster graphic file as output. If *ppmfile* is not supplied, *ppmtoicr* will read from standard input.

Interactive Color Raster (ICR) is a protocol for displaying raster graphics on workstation screens. The protocol is implemented in NCSA Telnet for the Macintosh version 2.3. The ICR protocol shares characteristics of the Tektronix graphics terminal emulation protocol. For example, escape sequences are used to control the display.

ppmtoicr will output the appropriate sequences to create a window of the dimensions of the input pixmap, create a colormap of up to 256 colors on the display, then load the picture data into the window.

Note that there is no icrtoppm tool - this transformation is one way.

OPTIONS

-windowname*name* Output will be displayed in *name* (Default is to use *ppmfile* or "untitled" if standard input is read.)

-expand*expand* Output will be expanded on display by factor *expand* (For example, a value of 2 will cause four pixels to be displayed for every input pixel.)

-display*display* Output will be displayed on screen numbered *display*

-rle Use run-length encoded format for display. (This will nearly always result in a quicker display, but may skew the colormap.)

EXAMPLES

To display a *ppm* file using the protocol:

```
ppmtoicr ppmfile
```

This will create a window named *ppmfile* on the display with the correct dimensions for *ppmfile,* create and download a colormap of up to 256 colors, and download the picture into the window. The same effect may be achieved by the following sequence:

```
ppmtoicr ppmfile > filename
cat filename
```

To display a GIF file using the protocol in a window titled after the input file, zoom the displayed image by a factor of 2, and run-length encode the data:

```
giftoppm giffile | ppmtoicr -w giffile -r -e 2
```

BUGS

The protocol uses frequent *fflush* calls to speed up display. If the output is saved to a file for later display via *cat,* drawing will be much slower. In either case, increasing the Blocksize limit on the display will speed up transmission substantially.

SEE ALSO

ppm(5)
NCSA Telnet for the Macintosh, University of Illinois at Urbana-Champaign (1989)

AUTHOR

©1990 by Kanthan Pillay (svpillay@Princeton.EDU), Princeton University Computing and Information Technology.

NAME

ppmtoilbm - convert a portable pixmap into an ILBM file

SYNOPSIS

ppmtoilbm [-maxplanes|-mp *N*] **[-fixplanes|-fp** *N*] **[-ham6|-ham8] [-dcbits|-dcplanes**rgb] **[-normal|-hamif|-hamforce|-24if|-24force|** -dcif|-dcforce|-cmaponly] **[-ecs|-aga] [-map**ppmfile] **[ppmfile]**

DESCRIPTION

Reads a portable pixmap as input. Produces an ILBM file as output. Supported ILBM types are:

Normal ILBMs with 1-16 planes.

Amiga Hold-and-Modify (HAM) with 3-16 planes.

24 bit.

Color map (BMHD + CMAP chunk only, nPlanes = 0).

Unofficial direct color. 1-16 planes for each color component.

Chunks written: BMHD, CMAP, CAMG (only for HAM), BODY (not for colormap files) unofficial DCOL chunk for direct color ILBM

Chunks ignored: GRAB, DEST, SPRT, CRNG, CCRT, CLUT, DPPV, DRNG, EPSF

Other chunks (ignored but displayed in verbose mode): NAME, AUTH, (c), ANNO, DPI

OPTIONS

Options marked with (*) can be prefixed with a "no", *e.g.*, "-nohamif". All options can be abbreviated to their shortest unique prefix.

-maxplanes | -mp n (default 5, minimum 1, maximum 16) Maximum planes to write in a normal ILBM. If the pixmap does not fit into <n> planes, ppmtoilbm writes a HAM file (if -hamif is used), a 24bit file (if -24if is used) or a direct color file (if -dcif is used) or aborts with an error.

-fixplanes | -fp n (min 1, max 16) If a normal ILBM is written, it will have exactly <n> planes.

-hambits | -hamplanes n (default 6, min 3, max 16) Select number of planes for HAM picture. The current Amiga hardware supports 6 and 8 planes, so for now you should only use this values.

-normal (default) Turns off -hamif/-24if/-dcif, -hamforce/-24force/-dcforce and -cmaponly.

-hamif (*)

-24if (*)

-dcif (*) Write a HAM/24bit/direct color file if the pixmap does not fit into <maxplanes> planes.

-hamforce (*)

-24force (*)

-dcforce (*) Write a HAM/24bit/direct color file.

-dcbits | -dcplanes r g b (default 5, min 1, max 16). Select number of bits for red, green & blue in a direct color ILBM.

-ecs (default) Shortcut for: -hamplanes 6 -maxplanes 5

-aga

Shortcut for: -hamplanes 8 -maxplanes 8

-ham6

Shortcut for: -hamplanes 6 -hamforce

-ham8 Shortcut for: -hamplanes 8 -hamforce

-map ppmfile Write a normal ILBM using the colors in <ppmfile> as the colormap. The colormap file also determines the number of planes, a -maxplanes or -fixplanes option is ignored.

-cmaponly Write a colormap file: only BMHD and CMAP chunks, no BODY chunk, nPlanes = 0.

BUGS

Needs a real colormap selection algorithm for HAM pictures, instead of using a grayscale colormap.

REFERENCES

Amiga ROM Kernel Reference Manual - Devices (3rd Ed.) Addison Wesley, ISBN 0–201–56775–X

SEE ALSO

ppm(5), ilbmtoppm(1)

AUTHORS

©1989 by Jef Poskanzer.
Modified August 1993 by Ingo Wilken (Ingo.Wilken@informatik.uni-oldenburg.de).

NAME

ppmtomitsu - convert a portable pixmap to a Mitsubishi S340-10 file

SYNOPSIS

ppmtomitsu [-sharpness *val*] [**-enlarge** *val*] [**-media** *string*] [**-copy** *val*] [**-dpi300**] [**-tiny**] [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input and converts it into a format suitable to be printed by a Mitsubishi S340-10 printer, or any other Mitsubishi color sublimation printer.

The Mitsubishi S340-10 Color Sublimation printer supports 24bit color. Images of the available sizes take so long to transfer that there is a fast method, employing a lookuptable, that ppmtomitsu will use if there is a maximum of 256 colors in the pixmap. ppmtomitsu will try to position your image to the center of the paper, and will rotate your image for you if xsize is larger than ysize. If your image is larger than the media allows, ppmtomitsu will quit with an error message. (We decided that the media were too expensive to have careless users produce misprints.) Once data transmission has started, the job can't be stopped in a sane way without resetting the printer. The printer understands putting together images in the printers memory; ppmtomitsu doesn't utilize this as pnmcat etc provide the same functionality and let you view the result on-screen, too. The S340-10 is the lowest common denominator printer; for higher resolution printers there's the dpi300 option. The other printers also support higher values for enlarge eg, but I don't think that's essential enough to warrant a change in the program.

-sharpness *1-4*
'sharpness' designation. Default is to use the current sharpness.

-enlarge *1-3*
Enlarge by a factor; Default is 1 (no enlarge)

-media *A, A4, AS, A4S*
Designate the media you're using. Default is 1184 x 1350, which will fit on any media. A is 1216 x 1350, A4 is 1184 x 1452, AS is 1216 x 1650 and A4S is 1184 x 1754. A warning: If you specify a different media than the printer currently has, the printer will wait until you put in the correct media or switch it off.

-copy *1-9*
The number of copies to produce. Default is 1.

-dpi300
Double the number of allowed pixels for a S3600-30 Printer in S340-10 compatibility mode. (The S3600-30 has 300 dpi).

-tiny
Memory-safing, but always slow. The printer will get the data line-by-line in 24bit. It's probably a good idea to use this if your machine starts paging a lot without this option.

REFERENCES

Mitsubishi Sublimation Full Color Printer S340-10 Specifications of Parallel Interface LSP-F0232F

SEE ALSO

ppmquant(1), pnmscale(1), ppm(5)

BUGS

We didn't find any - yet. (Besides, they're called features anyway :-) If you should find one, my email-adress is below.

AUTHOR

©1992, 93 by S.Petra Zeidler, MPIfR Bonn, Germany. (spz@specklec.mpifr-bonn.mpg.de)

NAME

ppmtomap - extract all colors from a portable pixmap

SYNOPSIS

ppmtomap [**-sort**] [**-square**] [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces a portable pixmap as output, representing a color map of the input file. All N different colors found are put in an Nx1 portable pixmap. This color map file can be used as a mapfile for *ppmquant* or *ppmtogif.*

OPTIONS

-sort Produces a portable pixmap with the colors in some sorted order.

-square Produces a (more or less) square output file, instead of putting all colors on the top row.

All flags can be abbreviated to their shortest unique prefix.

WARNING

If you want to use the output file as a mapfile for *ppmtogif,* you first have to do a *ppmquant 256,* since *ppmtomap* is not limited to 256 colors (but to 65536).

SEE ALSO

ppmtogif(1), ppmquant(1), ppm(5)

AUTHOR

Marcel Wijkstra (wijkstra@fwi.uva.nl).

NAME

ppmtopcx - convert a portable pixmap into a PCX file

SYNOPSIS

ppmtopcx [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces a PCX file as output.

SEE ALSO

pcxtoppm(1), ppm(5)

AUTHOR

NAME

ppmtopgm - convert a portable pixmap into a portable graymap

SYNOPSIS

ppmtopgm [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces a portable graymap as output. The quantization formula used is .299 r + .587 g + .114 b.

Note that although there is a *pgmtoppm* program, it is not necessary for simple conversions from *pgm* to *ppm*, because any ppm program can read *pgm* (and *pbm*) files automagically. *pgmtoppm* is for colorizing a *pgm* file. Also, see *ppmtorgb3* for a different way of converting color to gray.

QUOTE

Cold-hearted orb that rules the night
Removes the colors from our sight
Red is gray, and yellow white
But we decide which is right
And which is a quantization error.

SEE ALSO

pgmtoppm(1), ppmtorgb3(1), rgb3toppm(1), ppm(5), pgm(5)

AUTHOR

NAME

ppmtopi1 - convert a portable pixmap into an Atari Degas .pi1 file

SYNOPSIS

ppmtopi1 [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces an Atari Degas .pi1 file as output.

SEE ALSO

pi1toppm(1), ppm(5), pbmtopi3(1), pi3topbm(1)

AUTHOR

©1991 by Steve Belczyk (seb3@gte.com) and Jef Poskanzer.

NAME

ppmtopict - convert a portable pixmap into a Macintosh PICT file

SYNOPSIS

ppmtopict [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces a Macintosh PICT file as output.

The generated file is only the data fork of a picture. You will need a program such as *mcvert* to generate a Macbinary or a BinHex file that contains the necessary information to identify the file as a PICT file to MacOS.

Even though PICT supports 2 and 4 bits per pixel, *ppmtopict* always generates an 8 bits per pixel file.

BUGS

The picture size field is only correct if the output is to a file since writing into this field requires seeking backwards on a file. However the PICT documentation seems to suggest that this field is not critical anyway since it is only the lower 16 bits of the picture size.

SEE ALSO

picttoppm(1), ppm(5), mcvert(1)

AUTHOR

©1990 by Ken Yap (ken@cs.rocester.edu).

NAME

ppmtopj - convert a portable pixmap to an HP PaintJet file

SYNOPSIS

ppmtopj [-gamma *val*] [**-xpos** *val*] [**-ypos** *val*] [**-back dark**|**lite**] [**-rle**] [**-center**] [**-render none**|**snap**|**bw**|**dither**|**diffuse**|**monodither**|**monodiffuse**|**clusterdither**|**monoclusterdither**] [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input and converts it into a format suitable to be printed by an HP PaintJet printer.

For best results, the input file should be in 8-color RGB form; *i.e.*, it should have only the 8 binary combinations of full-on and full-off primaries. You could get this by sending the input file through *ppmquant -map* with a map file such as:

```
P3
8 1
255
0 0 0       255 0 0     0 255 0     0 0 255
255 255 0   255 0 255   0 255 255   255 255 255
```

Or else you could use use *ppmdither -red 2 -green 2 -blue 2.*

OPTIONS

-rle Run length encode the image. (This can result in larger images)

-back Enhance the foreground by indicating if the background is light or dark compated to the foreground.

-render *alg* Use an internal rendering algorithm (default dither).

-gamma *int* Gamma correct the image using the integet parameter as a gamma (default 0).

-center Center the image to an 8.5 by 11 page

-xpos *pos* Move by pos pixels in the x direction.

-ypos *pos* Move by pos pixels in the y direction.

REFERENCES

HP PaintJet XL Color Graphics Printer User's Guide

SEE ALSO

pnmdepth(1), ppmquant(1), ppmdither(1), ppm(5)

BUGS

Most of the options have not been tested because of the price of the paper.

AUTHOR

NAME

ppmtopjxl - convert a portable pixmap into an HP PaintJet XL PCL file

SYNOPSIS

ppmtopjxl [-nopack] [-gamma <*n*>] [-presentation] [-dark] [-diffuse] [-cluster] [-dither] [-xshift <*s*>] [-yshift <*s*>] [-xshift <*s*>] [-yshift <*s*>] [-xsize|-width|-xscale <*s*>] [-ysize|-height|-yscale <*s*>] [ppmfile]

DESCRIPTION

Reads a portable pixmap as input. Produces a PCL file suitable for printing on an HP PaintJet XL printer as output.

The generated file is not suitable for printing on a normal PrintJet printer. The –**nopack** option generates a file which does not use the normal TIFF 4.0 compression method. This file might be printable on a normal PaintJet printer (not an XL).

The –**gamma** option sets the gamma correction for the image. The useful range for the PaintJet XL is approximately 0.6 to 1.5.

The rendering algorithm used for images can be altered with the **-dither, -cluster,** and **-diffuse** options. These options select ordered dithering, clustered ordered dithering, or error diffusion respectively. The –**dark** option can be used to enhance images with a dark background when they are reduced in size. The –**presentation** option turns on presentation mode, in which two passes are made over the paper to increase ink density. This should be used only for images where quality is critical.

The image can be resized by setting the –**xsize** and –**ysize** options. The parameter to either of these options is interpreted as the number of dots to set the width or height to, but an optional dimension of '**pt**' (points), '**dp**' (decipoints), '**in**' (inches), or '**cm**' (centimetres) may be appended. If only one dimension is specified, the other will be scaled appropriately.

The options –**width** and –**height** are synonyms of –**xsize** and –**ysize.**

The –**xscale** and –**yscale** options can alternatively be used to scale the image by a simple factor.

The image can be shifted on the page by using the –**xshift** and –**yshift** options. These move the image the specified dimensions right and down.

SEE ALSO

ppm(5)

AUTHOR

Angus Duggan

NAME

ppmtopuzz - convert a portable pixmap into an X11 "puzzle" file

SYNOPSIS

ppmtopuzz [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces an X11 "puzzle" file as output. A "puzzle" file is for use with the *puzzle* program included with the X11 distribution - *puzzle*'s **-picture** flag lets you specify an image file.

SEE ALSO

ppm(5), puzzle(1)

AUTHOR

NAME

ppmtorgb3 - separate a portable pixmap into three portable graymaps

SYNOPSIS

ppmtorgb3 [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Writes three portable graymaps as output, one each for red, green, and blue.

The output filenames are constructed by taking the input filename, stripping off any extension, and appending ".red", ".grn", and ".blu". For example, separating lenna.ppm would result in lenna.red, lenna.grn, and lenna.blu. If the input comes from stdin, the names are noname.red, noname.grn, and noname.blu.

SEE ALSO

rgb3toppm(1), ppmtopgm(1), pgmtoppm(1), ppm(5), pgm(5)

AUTHOR

NAME

ppmtosixel - convert a portable pixmap into DEC sixel format

SYNOPSIS

ppmtosixel [**-raw**] [**-margin**] [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces sixel commands (SIX) as output. The output is formatted for color printing, *e.g.*, for a DEC LJ250 color inkjet printer.

If RGB values from the PPM file do not have maxval=100, the RGB values are rescaled. A printer control header and a color assignment table begin the SIX file. Image data is written in a compressed format by default. A printer control footer ends the image file.

OPTIONS

-raw If specified, each pixel will be explicitly described in the image file. If **-raw** is not specified, output will default to compressed format in which identical adjacent pixels are replaced by "repeat pixel" commands. A raw file is often an order of magnitude larger than a compressed file and prints much slower.

-margin If **-margin** is not specified, the image will be start at the left margin (of the window, paper, or whatever). If **-margin** is specified, a 1.5 inch left margin will offset the image.

PRINTING

Generally, sixel files must reach the printer unfiltered. Use the lpr -x option or cat filename > /dev/tty0?.

BUGS

Upon rescaling, truncation of the least significant bits of RGB values may result in poor color conversion. If the original PPM maxval was greater than 100, rescaling also reduces the image depth. While the actual RGB values from the ppm file are more or less retained, the color palette of the LJ250 may not match the colors on your screen. This seems to be a printer limitation.

SEE ALSO

ppm(5)

AUTHOR

NAME

ppmtotga - convert portable pixmap into a TrueVision Targa file

SYNOPSIS

ppmtotga [**-mono**|**-cmap**|**-rgb**] [**-norle**] [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces a TrueVision Targa file as output.

OPTIONS

-mono Forces Targa file to be of type 8 bit monochrome. Input must be a portable bitmap or a portable graymap.

-cmap Forces Targa file to be of type 24 bit colormapped. Input must be a portable bitmap, a portable graymap or a portable pixmap containing no more than 256 distinct colors.

-rgb Forces Targa file to be of type 24 bit unmapped color.

-norle Disables run-length encoding, in case you have a Targa reader which can't read run-length encoded files.

All flags can be abbreviated to their shortest unique prefix. If no file type is specified the most highly constained compatible type is used, where monochrome is more constained than colormapped which is in turn more constained than unmapped.

BUGS

Does not support all possible Targa file types. Should really be in PNM, not PPM.

SEE ALSO

tgatoppm(1), ppm(5)

AUTHOR

©1989, 1991 by Mark Shand and Jef Poskanzer.

NAME

ppmtouil - convert a portable pixmap into a Motif UIL icon file

SYNOPSIS

ppmtouil [**-name** *uilname*] [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces a Motif UIL icon file as output.

If the program was compiled with an rgb database specified, and a RGB value from the ppm input matches a RGB value from the database, then the corresponding color name mnemonic is printed in the UIL's colormap. If no rgb database was compiled in, or if the RGB values don't match, then the color will be printed with the #RGB, #RRGGBB, #RRRGGGBBB, or #RRRRGGGGBBBB hexadecimal format.

OPTIONS

-name Allows you to specify the prefix string which is printed in the resulting UIL output. If not specified, will default to the filename (without extension) of the ppmfile argument. If **-name** is not specified and no ppmfile is specified (*i.e.*, piped input), the prefix string will default to the string "noname".

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

ppm(5)

AUTHOR

Converted by Jef Poskanzer from ppmtoxpm.c, which is ©1990 by Mark W. Snitily.

NAME

ppmtoxpm - convert a portable pixmap into an X11 pixmap

SYNOPSIS

ppmtoxpm [-name <xpmname>] [-rgb <rgb-textfile>] [<ppmfile>]

DESCRIPTION

Reads a portable pixmap as input. Produces X11 pixmap (version 3) as output which can be loaded directly by the XPM library.

The **-nameoption** allows one to specify the prefix string which is printed in the resulting XPM output. If not specified, will default to the filename (without extension) of the <ppmfile> argument. If **-nameis** not specified and <ppmfile> is not specified (*i.e.*, piped input), the prefix string will default to the string "noname".

The **-rgboption** allows one to specify an X11 rgb text file for the lookup of color name mnemonics. This rgb text file is typically the /usr/lib/X11/rgb.txt of the MIT X11 distribution, but any file using the same format may be used. When specified and a RGB value from the ppm input matches a RGB value from the <rgb-textfile>, then the corresponding color name mnemonic is printed in the XPM's colormap. If **-rgbis** not specified, or if the RGB values don't match, then the color will be printed with the #RGB, #RRGGBB, #RRRGGGBBB, or #RRRRGGGGBBBB hexadecimal format.

All flags can be abbreviated to their shortest unique prefix.

For example, to convert the file "dot" (found in /usr/include/X11/bitmaps), from xbm to xpm one could specify

```
xbmtopbm dot | ppmtoxpm -name dot
```

or, with a rgb text file (in the local directory)

```
xbmtopbm dot | ppmtoxpm -name dot -rgb rgb.txt
```

BUGS

An option to match the closest (rather than exact) color name mnemonic from the rgb text would be a desirable enhancement.

Truncation of the least significant bits of a RGB value may result in nonexact matches when performing color name mnemonic lookups.

SEE ALSO

ppm(5)
XPM Manual by Arnaud Le Hors (lehors@mirsa.inria.fr).

AUTHOR

This tool was developed for Schlumberger Technologies, ATE Division, and with their permission is being made available to the public with the above copyright notice and permission notice.

Upgraded to XPM2 by Paul Breslaw, Mecasoft SA, Zurich, Switzerland (paul@mecazh.uu.ch) Thu Nov 8 16:01:17 1990

Upgraded to XPM version 3 by Arnaud Le Hors (lehors@mirsa.inria.fr) Tue Apr 9 1991

NAME

ppmtoyuv - convert a portable pixmap into an Abekas YUV file

SYNOPSIS

ppmtoyuv [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces an Abekas YUV file as output.

SEE ALSO

yuvtoppm(1), ppm(5)

AUTHOR

Marc Boucher (marc@PostImage.COM), based on Example Conversion Program, A60/A64 Digital Video Interface Manual, page 69.

NAME

ppmtoyuvsplit - convert a portable pixmap into 3 subsampled raw YUV files

SYNOPSIS

ppmtoyuvsplit *basename* [*ppmfile*]

DESCRIPTION

Reads a portable pixmap as input. Produces 3 raw files basename.Y, basename.U and basename.V as output. These files are the subsampled raw YUV representation of the input pixmap, as required by the Stanford MPEG codec. The subsampling is done by arithmetic mean of 4 pixels colors into one. The YUV values are scaled according to CCIR.601, as assumed by MPEG.

SEE ALSO

mpeg(1), ppm(5)

AUTHOR

Based on ppmtoyuv.c.

NAME

psidtopgm - convert PostScript "image" data into a portable graymap

SYNOPSIS

psidtopgm *width height bits/sample* [*imagedata*]

DESCRIPTION

Reads the "image" data from a PostScript file as input. Produces a portable graymap as output.

This is a very simple and limited program, and is here only because so many people have asked for it. To use it you have to **manually** extract the readhexstring data portion from your PostScript file, and then give the width, height, and bits/sample on the command line. Before you attempt this, you should **at least** read the description of the "image" operator in the PostScript Language Reference Manual.

It would probably not be too hard to write a script that uses this filter to read a specific variety of PostScript image, but the variation is too great to make a general-purpose reader. Unless, of course, you want to write a full-fledged PostScript interpreter...

SEE ALSO

pnmtops(1), pgm(5)

AUTHOR

NAME

qrttoppm - convert output from the QRT ray tracer into a portable pixmap

SYNOPSIS

qrttoppm [*qrtfile*]

DESCRIPTION

Reads a QRT file as input. Produces a portable pixmap as output.

SEE ALSO

ppm(5)

AUTHOR

NAME

rasttopnm - convert a Sun rasterfile into a portable anymap

SYNOPSIS

rasttopnm [*rastfile*]

DESCRIPTION

Reads a Sun rasterfile as input. Produces a portable anymap as output. The type of the output file depends on the input file - if it's black & white, a *pbm* file is written, else if it's grayscale a *pgm* file, else a *ppm* file. The program tells you which type it is writing.

SEE ALSO

pnmtorast(1), pnm(5)

AUTHOR

NAME

rawtopgm - convert raw grayscale bytes into a portable graymap

SYNOPSIS

rawtopgm [**-headerskip** *N*] [**-rowskip** *N*] [**-tb**|**-topbottom**] [*width height*] [*imagedata*]

DESCRIPTION

Reads raw grayscale bytes as input. Produces a portable graymap as output. The input file is just grayscale bytes. If you don't specify the width and height on the command line, the program will check the size of the image and try to make a quadratic image of it. It is an error to supply a non quadratic image without specifying width and height. The maxval is assumed to be 255.

OPTIONS

-headerskip If the file has a header, you can use this flag to skip over it.

-rowskip If there is padding at the ends of the rows, you can skip it with this flag. Note that rowskip can be a real number. Amazingly, I once had an image with 0.376 bytes of padding per row. This turned out to be due to a file-transfer problem, but I was still able to read the image.

-tb -topbottom Flips the image upside down. The first pixel in a pgm file is in the lower left corner of the image. For conversion from images with the first pixel in the upper left corner (*e.g.*, the Molecular Dynamics and Leica confocal formats) this flips the image right. This is equivalent to **rawtopgm [file] | pnmflip -tb .**

BUGS

If you don't specify the image width and height, the program will try to read the entire image to a memory buffer. If you get a message that states that you are out of memory, try to specify the width and height on the command line. Also, the -tb option consumes much memory.

SEE ALSO

pgm(5), rawtoppm(1), pnmflip(1)

AUTHORS

©1989 by Jef Poskanzer.
Modified June 1993 by Oliver Trepte (oliver@fysik4.kth.se).

NAME

rawtoppm - convert raw RGB bytes into a portable pixmap

SYNOPSIS

rawtoppm [**-headerskip** *N*] [**-rowskip** *N*] [**-rgb**|**-rbg**|**-grb** |**-gbr**|**-brg**|**-bgr**]
[**-interpixel**|**-interrow**] *width height* [*imagedata*]

DESCRIPTION

Reads raw RGB bytes as input. Produces a portable pixmap as output. The input file is just RGB bytes. You have to specify the width and height on the command line, since the program obviously can't get them from the file. The maxval is assumed to be 255. If the resulting image is upside down, run it through **pnmflip -tb .**

OPTIONS

-headerskip If the file has a header, you can use this flag to skip over it.

-rowskip If there is padding at the ends of the rows, you can skip it with this flag.

-rgb -rbg -grb -gbr -brg -bgr These flags let you specify alternate color orders. The default is **-rgb**.

-interpixel -interrow These flags let you specify how the colors are interleaved. The default is **-interpixel**, meaning interleaved by pixel. A byte of red, a byte of green, and a byte of blue, or whatever color order you specified. **-interrow** means interleaved by row - a row of red, a row of green, a row of blue, assuming standard rgb color order. An **-interplane** flag - all the red pixels, then all the green, then all the blue - would be an obvious extension, but is not implemented. You could get the same effect by splitting the file into three parts (perhaps using *dd*), turning each part into a PGM file with rawtopgm, and then combining them with rgb3toppm.

SEE ALSO

ppm(5), rawtopgm(1), rgb3toppm(1), pnmflip(1)

AUTHOR

NAME

rgb3toppm - combine three portable graymaps into one portable pixmap

SYNOPSIS

rgb3toppm *redpgmfile greenpgmfile bluepgmfile*

DESCRIPTION

Reads three portable graymaps as input. Combines them and produces one portable pixmap as output.

SEE ALSO

ppmtorgb3(1), pgmtoppm(1), ppmtopgm(1), ppm(5), pgm(5)

AUTHOR

NAME

sirtopnm - convert a Solitaire file into a portable anymap

SYNOPSIS

sirtopnm [*sirfile*]

DESCRIPTION

Reads a Solitaire Image Recorder file as input. Produces a portable anymap as output. The type of the output file depends on the input file - if it's an MGI TYPE 17 file, a *pgm* file is written. If it's an MGI TYPE 11 file, a *ppm* file is written. The program tells you which type it is writing.

BUGS

SEE ALSO

pnmtosir(1), pnm(5)

AUTHOR

NAME

sldtoppm - convert an AutoCAD slide file into a portable pixmap

SYNOPSIS

sldtoppm 'in 14n [**-adjust**] [**-dir**] [**-height**|**-ysize** *s*] [**-info**] [**-lib**|**-Lib** *name*] [**-scale** *s*] [**-verbose**] [**-width**|**-xsize** *s*] [*slidefile*]

DESCRIPTION

Reads an AutoCADlide file and outputs a portable pixmap. If no *slidefile* is specified, input is read from standard input. The ppmdraw library is used to convert the vector and polygon information in the slide file to a pixmap; see the file ppmdraw.h for details on this package.

OPTIONS

-adjust If the display on which the slide file was created had non-square pixels, when the slide is processed with **sldtoppm** and the **-adjust** option is not present, the following warning will appear:

Warning - pixels on source screen were non-square.
Specifying **-adjust** will correct image width to compensate.

Specifying the **-adjust** option causes **sldtoppm** to scale the width of the image so that pixels in the resulting portable pixmap are square (and hence circles appear as true circles, not ellipses). The scaling is performed in the vector domain, before scan converting the objects. The results are, therefore, superior in appearance to what you'd obtain were you to perform the equivalent scaling with **pnmscale** after the bitmap had been created.

-dir The input is assumed to be an AutoCAD slide library file. A directory listing each slide in the library is printed on standard error.

-height *size* Scales the image in the vector domain so it is *size* pixels in height. If no **-width** or **-xsize** option is specified, the width will be adjusted to preserve the pixel aspect ratio.

-info Dump the slide file header on standard error, displaying the original screen size and aspect ratio among other information.

-lib *name* Extracts the slide with the given *name* from the slide library given as input. The specified *name* is converted to upper case.

-Lib *name* Extracts the slide with the given *name* from the slide library given as input. The *name* is used exactly as specified; it is not converted to upper case.

-scale *s* Scales the image by factor *s*, which may be any floating point value greater than zero. Scaling is done after aspect ratio adjustment, if any. Since scaling is performed in the vector domain, before rasterisation, the results look much better than running the output of **sldtoppm** through **pnmscale**.

-verbose Dumps the slide file header and lists every vector and polygon in the file on standard error.

-width *size* Scales the image in the vector domain so it is *size* pixels wide. If no **-height** or **-ysize** option is specified, the height will be adjusted to preserve the pixel aspect ratio.

-xsize *size* Scales the image in the vector domain so it is *size* pixels wide. If no **-height** or **-ysize** option is specified, the height will be adjusted to preserve the pixel aspect ratio.

-ysize *size* Scales the image in the vector domain so it is *size* pixels in height. If no **-width** or **-xsize** option is specified, the width will be adjusted to preserve the pixel aspect ratio.

All flags can be abbreviated to their shortest unique prefix.

BUGS

Only Level 2 slides are converted. Level 1 format has been obsolete since the advent of AutoCAD Release 9 in 1987, and was not portable across machine architectures.

Slide library items with names containing 8 bit (such as ISO) or 16 bit (Kanji, for example) characters may not be found when chosen with the **-lib** option unless **sldtoppm** has been built with character set conversion functions appropriate to the locale. You can always retrieve slides from libraries regardless of the character set by using the **-Lib** option and specifying the precise name of library member. Use the **-dir** option to list the slides in a library if you're unsure of the exact name.

SEE ALSO

AutoCAD Reference Manual: *Slide File Format*, **pnmscale**(1), **ppm**(5)

AUTHOR

John Walker
Autodesk SA
Avenue des Champs-Montants 14b
CH-2074 MARIN
Suisse/Schweiz/Svizzera/Svizra/Switzerland

Usenet: kelvin@Autodesk.com

Fax: 038/33 88 15

Voice: 038/33 76 33

NAME

spctoppm - convert an Atari compressed Spectrum file into a portable pixmap

SYNOPSIS

spctoppm [*spcfile*]

DESCRIPTION

Reads an Atari compressed Spectrum file as input. Produces a portable pixmap as output.

SEE ALSO

sputoppm(1), ppm(5)

AUTHOR

©1991 by Steve Belczyk (seb3@gte.com) and Jef Poskanzer.

NAME

spottopgm – convert SPOT satellite images to Portable Greymap format

SYNTAX

spottopgm [–1|2|3] [Firstcol Firstline Lastcol Lastline] inputfile

OPTIONS

–1|2|3 Extract the given colour from the SPOT image. The colours are infra-red, visible light and ultra-violet, although I don't know which corresponds to which number. If the image is in colour, this will be announced on standard error. The default colour is 1.

Firstcol Firstline Lastcol Lastline Extract the specified rectangle from the SPOT image. Most SPOT images are 3000 lines long and 3000 or more columns wide. Unfortunately the SPOT format only gives the width and not the length. The width is printed on standard error. The default rectangle is the width of the input image by 3000 lines.

DESCRIPTION

Spottopgm converts the named **inputfile** into Portable Greymap format, defaulting to the first color and the whole SPOT image unless specified by the options.

INSTALLATION

You **must** edit the source program and either define BIGENDIAN or LITTLEENDIAN, and fix the typedefs for uint32t, uint16t and uint8t appropriately.

BUGS

Currently *spottopgm* doesn't determine the length of the input file; this would involve two passes over the input file. It defaults to 3000 lines instead.

Spottopgm could extract a three-color image (ppm), but I didn't feel like making the program more complicated than it is now. Besides, there is no one-to-one correspondence between red, green, blue and infra-red, visible and ultra-violet.

I've only had a limited number of SPOT images to play with, and therefore wouldn't guarantee that this will work on any other images.

AUTHOR

Warren Toomey wkt@csadfa.cs.adfa.oz.au

SEE ALSO

The rest of the Pbmplus suite.

NAME

sputoppm - convert an Atari uncompressed Spectrum file into a portable pixmap

SYNOPSIS

sputoppm [*spufile*]

DESCRIPTION

Reads an Atari uncompressed Spectrum file as input. Produces a portable pixmap as output.

SEE ALSO

spctoppm(1), ppm(5)

AUTHOR

NAME

tgatoppm - convert TrueVision Targa file into a portable pixmap

SYNOPSIS

tgatoppm [**-debug**] [*tgafile*]

DESCRIPTION

Reads a TrueVision Targa file as input. Produces a portable pixmap as output.

OPTIONS

-debug Causes the header information to be dumped to stderr.

All flags can be abbreviated to their shortest unique prefix. Should really be in PNM, not PPM.

SEE ALSO

ppmtotga(1), ppm(5)

AUTHOR

Partially based on tga2rast, version 1.0, by Ian J. MacPhedran.

NAME

tifftopnm - convert a TIFF file into a portable anymap

SYNOPSIS

tifftopnm [**-headerdump**] tifffile

DESCRIPTION

Reads a TIFF file as input. Produces a portable anymap as output. The type of the output file depends on the input file - if it's black & white, a *pbm* file is written, else if it's grayscale a *pgm* file, else a *ppm* file. The program tells you which type it is writing.

OPTIONS

-headerdump Dump TIFF file information to stderr. This information may be useful in debugging TIFF file conversion problems.

All flags can be abbreviated to their shortest unique prefix.

SEE ALSO

pnmtotiff(1), pnm(5)

BUGS

This program is not self-contained. To use it you must fetch the TIFF Software package listed in the OTHER.SYSTEMS file and configure PBMPLUS to use libtiff. See PBMPLUS's Makefile for details on this configuration.

AUTHOR

Derived by Jef Poskanzer from tif2ras.c, which is ©1990 by Sun Microsystems, Inc.
Author: Patrick J. Naughton (naughton@wind.sun.com).

NAME

xbmtopbm - convert an X11 or X10 bitmap into a portable bitmap

SYNOPSIS

xbmtopbm [*bitmapfile*]

DESCRIPTION

Reads an X11 or X10 bitmap as input. Produces a portable bitmap as output.

SEE ALSO

pbmtoxbm(1), pbmtox10bm(1), pbm(5)

AUTHOR

NAME

ximtoppm - convert an Xim file into a portable pixmap

SYNOPSIS

ximtoppm [*ximfile*]

DESCRIPTION

Reads an Xim file as input. Produces a portable pixmap as output. The Xim toolkit is included in the contrib tree of the X.V11R4 release.

SEE ALSO

ppm(5)

AUTHOR

NAME

xpmtoppm - convert an X11 pixmap into a portable pixmap

SYNOPSIS

xpmtoppm [*xpmfile*]

DESCRIPTION

Reads an X11 pixmap (XPM version 1 or 3) as input. Produces a portable pixmap as output.

KNOWN BUGS

The support to XPM version 3 is limited. Comments can only be single lines and there must be for every pixel a default colorname for a color type visual.

SEE ALSO

ppmtoxpm(1), ppm(5)
XPM Manual by Arnaud Le Hors (lehors@mirsa.inria.fr).

AUTHOR

©1991 by Jef Poskanzer. Upgraded to support XPM version 3 by Arnaud Le Hors (lehors@mirsa.inria.fr) Tue Apr 9 1991.

NAME

xvminitoppm - convert a XV "thumbnail" picture to PPM

SYNOPSIS

xvminitoppm [*xvminipic*]

DESCRIPTION

Reads a XV "thumbnail" picture (a miniature picture generated by the "VisualSchnauzer" browser) as input. Produces a portable pixmap as output.

SEE ALSO

ppm(5), xv(1)

AUTHOR

Copyright (C) 1993 by Ingo Wilken

NAME

xwdtopnm - convert a X11 or X10 window dump file into a portable anymap

SYNOPSIS

xwdtopnm [*xwdfile*]

DESCRIPTION

Reads a X11 or X10 window dump file as input. Produces a portable anymap as output. The type of the output file depends on the input file - if it's black & white, a *pbm* file is written, else if it's grayscale a *pgm* file, else a *ppm* file. The program tells you which type it is writing.

Using this program, you can convert anything on an X workstation's screen into an anymap. Just display whatever you're interested in, do an xwd, run it through xwdtopnm, and then use pnmcut to select the part you want.

BUGS

I haven't tested this tool with very many configurations, so there are probably bugs. Please let me know if you find any.

SEE ALSO

pnmtoxwd(1), pnm(5), xwd(1)

AUTHOR

NAME

ybmtopbm - convert a Bennet Yee "face" file into a portable bitmap

SYNOPSIS

ybmtopbm [*facefile*]

DESCRIPTION

Reads a file acceptable to the *face* and *xbm* programs by Bennet Yee (bsy+@cs.cmu.edu). Writes a portable bitmap as output.

SEE ALSO

pbmtoybm(1), pbm(5), face(1), face(5), xbm(1)

AUTHOR

NAME

yuvplittoppm - convert a Y- an U- and a V-file into a portable pixmap.

SYNOPSIS

yuvsplittoppm *basename width height* [-ccir601]

DESCRIPTION

Reads three files, containing the YUV components, as input. These files are *basename* Y, *basename* U and *basename* .V . Produces a portable pixmap on stdout.

Since the YUV files are raw files, the dimensions *width* and *height* must be specified on the command line.

OPTIONS

-ccir601 Assumes that the YUV triplets are scaled into the smaller range of the CCIR 601 (MPEG) standard. Else, the JFIF (JPEG) standard is assumed.

SEE ALSO

ppmtoyuvsplit(1), yuvtoppm(1), ppm(5)

AUTHOR

Marcel Wijkstra (wijkstra@fwi.uva.nl), based on *ppmtoyuvsplit.*

NAME

yuvtoppm - convert Abekas YUV bytes into a portable pixmap

SYNOPSIS

yuvtoppm *width height* [*imagedata*]

DESCRIPTION

Reads raw Abekas YUV bytes as input. Produces a portable pixmap as output. The input file is just YUV bytes. You have to specify the width and height on the command line, since the program obviously can't get them from the file. The maxval is assumed to be 255.

SEE ALSO

ppmtoyuv(1), ppm(5)

AUTHOR

Marc Boucher (marc@PostImage.COM), based on Example Conversion Program, A60/A64 Digital Video Interface Manual, page 69.

NAME

zeisstopnm - convert a Zeiss confocal file into a portable anymap

SYNOPSIS

zeisstopnm [-*pgm* | -*ppm*] [*zeissfile*]

DESCRIPTION

Reads a Zeiss confocal file as input. Produces a portable anymap as output. The type of the output file depends on the input file - if it's grayscale a *pgm* file, else a *ppm* file will be produced. The program tells you which type it is writing.

OPTIONS

-pgm Force the output to be a *pgm* file.

-ppm Force the output to be a *ppm* file.

SEE ALSO

pnm(5)

AUTHOR

©1993 by Oliver Trepte (oliver@fysik4.kth.se).