NAME

pstogif - convert a single-page PostScript (PS or EPS) file
into a GIF, JPEG, PNG, TIFF or PNM image

SYNOPSIS

pstogif [options] ps_file [out_file]

DESCRIPTION

pstogif is a *bash* script to convert PostScript images into GIF, JPG, PNG, TIFF or PNM images, using Ghostscript ("**gs**") and several programs from the **Netbpm** package.

The main advantage of **pstogif** over conversion packages such as the widely used **convert**, is that **pstogif** performs any scaling of the image *before* conversion -- **convert** performs the scaling after the conversion, leading to strange artifacts in the image, notably with straight lines, and interpolation of colours.

The script is called "**pstogif**" as it started its life for making GIF images, back in 2004. The script has since been extended to cover more image formats and handle the most important options of the respective programs in the **Netbpm** package. The other image formats can be accessed using command line options, or by making a symbolic link to set the default output format:

```
ln -sf pstogif new_name
```

The script recognised the following names:

pstogif -- conversion to GIF == the standard name
pstopng -- conversion to PNG
pstojpg -- conversion to JPEG [or: **pstojpeg**]
pstotif -- conversion to TIFF [or: **pstotiff**]
pstopxm -- conversion to PNM [*]

[*] The name **pstopxm** is used here for conversion to PNM to avoide confusion: the **Netbpm** package contains a program called **pstopnm**. That program can do most of the things **pstogif** does - it also uses **gs** and other programs of the package. For more elaborate usage than **pstogif** can offer, one can start from **pstopnm**.

By the way, if either of the above links is made, one may wish to make similar links for this man page.

To download the full package, which includes some examples, visit *http://uv-vis.aeronomie.be/software/tools/pstogif.php*

COMMAND LINE OPTIONS

The options accepted by **pstogif** are listed below. Some of these options require an argument, others do not. If an option is missing, its default value is used. This default value can be seen in the brief overview shown when running **pstogif** without any option. If the same option occurs more than once, the last one is used. For mutually exclusive options, the last one given is used.

pstogif first uses **gs** to convert PostScript into *pnmraw*, setting the scale (resolution) and other image size details. Then a sequence of **Netbpm** programs is used to perform other steps -- cropping, adding a margin, rotating, etc. -- followed by the final *pnmtoXXX* conversion. If an option is used for one of the **Netbpm** programs, that program is given between square brackets; see the respective man page for more details.

Input and output file

The input file **ps_file** for the script should be a single file: the script cannot handle multiple files in one go; to process more than one input file, the script has to be run for each input file separtely.

The input file **ps_file** should furthermore be a single page PostScript (PS) or Encapsulated PostScript (EPS) file. If more than one page is present in the **ps_file**, only the first page is converted. If all pages of a multiple-page file need to be converted, first use the **pssplit** script that comes along with the package to split the input file into multiple files -- see the section **MULTIPLE-PAGE INPUT FILES** below.

If an **out_file** is given after the input file, that name is used for the output file of the script. If the **out_file** contains a directory path, this path is created in case it does not already exist.

If no **out_file** is given, its name is constructed from the name of the **ps_file** by adding the selected image format as suffix: one of .ps, .eps or .epsf (or their equivalent in capitals), that suffix is removed before adding the image format suffix. The **out_file** ends up in the same directory as where the **ps_file** is located.

NOTE: an existing **out_file** file is overwritten without warning.

Neither the **ps_file** nor the **out_file** name may start with a minus sign, as a minus sign designates an option.

General options

(none) If no argument is given, a short help message is shown, listing the main options (same as **-h**).

-h Shows a short help message, listing the main options.

-o Shows the other options.

-q, **-u** Normally some **Netbpm** programs write messages to *stderr*. Two options are provided to intercept these messages:
-q "quiet": the messages are redirected to the file *pstogif.err* in the current directory.
-u "ultra-quiet": the messages are redirected to */dev/null* -- use this option with care!
Option **-u** overrides **-q**.
Note that **gs** is always kept quiet, independent of any option.

Image format options

-g, **-j**, **-n**, **-f**, **-p**
These -- mutually exclusive -- options determine the image format of the output file: GIF, JPG, PNG, TIFF and PNM, respectively. Which of these acts as default is determined by the name with which the script is called. The standard name of the script is **pstogif** and its default image format is thus GIF. To get, for example, JPEG either use **pstogif -j** or the **pstojpg** symbolic link mentioned above.

Note that the PNM format covers the formats PBM ("portable bit map": black and white only), PGM ("portable gray map": greyscales) and PPM ("portable pixel map": colours) automatically. Use the **-k** option to set whether the image should be raw (binary) or plain (ascii).

-t *colour* Mark the specified *colour* as transparent in the GIF or PNG image; if the specified colour is not available in the image, its nearest neighbour is used. The default is no transparency.

The *colour* is either given as a name from the list in the *rgb.txt* colour definition file of the X11 system, or in the form of a hexadecimal or decimal coding. For example:

```
-t green
-t #00ff00
-t rgb:00/ff/00
```

-t rgbi:0.0/1.0/0.0

Note that in the latter three cases the colour string may have to be given between quotes.

[Option for *ppmtogif* and *pnmtopng*]

-k *number* Compression option, the functionality of which depends on the selected image format:

GIF No functionality.

JPEG Quality factor for the JPEG image. The default is 75; sensible values are between 50 and 95; the script allows for values of 5 to 100.

PNG Compression factor for the PNG image. The default is 6; the range is 0 (no compression) to 9.

TIFF Determines the compression of the TIFF image:

0 : use no compression.

1 : use the "flate" compression (the same format as used for PNG); this is the default.

PNM Determines the format of the PNM image:

0 : raw (binary) format; this is the default.

1 : plain (ascii) format.

[Option for *pnmtojpeg*, *pnmtopng*, *pnmtotiff*, *pnmtopnm*, resp.]

-i Produce an interlaced GIF or PNG image. The default is not interlaced.

[Option for *ppmtogif* and *pnmtopng*]

-y Produce a true-colour TIFF image, i.e. the 24-bit RGB form. Without this option, a colour-mapped (paletted) TIFF image is made, unless there are more than 256 colours. The default is no option.

[Option for *ppmtotiff*]

-z *ncols* Quantise the number of colours; this may be necessary for GIF images. Selecting 0 (zero) for *ncols* turns off quantisation. The default is no quantisation.

[Option for *pnmquant*]

Scaling and rotation options

-s *scale* Scale the image with factor *scale* before any conversion (the scaling is done by Ghostscript). The *scale* should be a decimal number larger than 0.1.

- Using *scale* equal 1 (one), the resulting image is of the same size as the input PostScript image. This is the default.
- Using values less than one reduces the size of the image; to avoid crashing the script, *scale* should be larger than 0.1.
- Using values larger than one increases the size of the image. Be careful with this! The larger the image, the more memory space the script needs to do the conversion, the slower the conversion is, and the larger the output file is.

[Option for *ghostscript*]

-r *angle* Rotate the image over *angle* degrees; negative values designate clockwise rotation. The default is no rotation.

In case of angles which are not multiples of 90 degrees, the background colour of the extra space (i.e. the corners of the rectangle around the image) is determined automatically, by taking an average of the colours of the two top corners of the image. The rotation is done after adding the margin(s) -- see below -- and so the background colour is determined

by the colour of the (outer) margin.

[Option for *pnmrotate* and *pnmflip*]

-a Perform anti-aliasing when rotating. The default is no anti-aliasing, which is OK in most cases. When converting a PS file with only black lines on a white background and using **-a** one can get the following message:

pnmrotate: promoting from PBM to PGM - use
-noantialias to avoid this

while without **-a** one can see this message:

ppmtogif: maxval is not 255 - automatically
rescaling colors

Neither of the two messages seems to be harmfull. And neither appear in case there are more colours than black and white.

[Option for *pnmrotate*]

Margin options

-m *width*
-c *colour* Add a *colour* margin of *width* pixels wide around the image, which is done after(!) scaling and cropping the image. The default is no margin, and for a non-zero margin the default colour is white. See the **-t** option above on how to provide the colour.

[Options for *pnmmargin*]

-M *width*
-C *colour* Same as **-m** and **-c**.
Puts an additional margin around the plot. This can be used, for example, to limit the plot with a black line around a white margin:

-m 5 -c white -M 1 -C black

The margin determined by **-M** is put around the margin determined by **-m**. If the latter is missing, only one margin is made.

[Options for *pnmmargin*]

Special options

Use these options with care: they may not work under all circumstances.

-x *file* Add the text in *file* as comment to the image file. This option works with GIF, JPEG and PNG files; it does not work with TIFF or PNM files. The default is no comments added.

The format of the *file* is a little tricky and is slightly different for PNG than for the other formats. The reason for this is that the comment added to the GIF and JPEG files are strickly speaking only single strings, whereas the comment in PNG files are made of "keyword value" combinations.

Regarding comment text in a PNG image, the "keyword" is the first word, i.e. the string starting in the very first column of the *file* up to the first space or tab. If the first position on a line is a space, then "keyword" is left empty. To have a "keyword" with a space in it, enclose it between double quotes.

The way the comments added to an image is shown by a viewer depends on the viewer, so it is best to keep the text file as simple as possible, i.e. limit the text to standard ASCII coding when possible. Some viewers may be able to handle accented characters or control characters (such

as "tab"), others will not.

[Option for *pnmtogif*, *pnmtojpeg*, *pnmtopng*, *pnmtopnm*]

-b Use the BoundingBox defined in the PostScript image to select the page size use by Ghostscript. In the absence of a BoundingBox definition, the paper size is set to that of A4 paper. Omitting this option lets **gs** use its internal default paper size; this is the default.

This is an "expert option" and will usually not be necessary. The option may be necessary if the image in the PostScript file is larger than the default paper size of **gs**. The option uses a temporary file (called */tmp/p2gNNNN.tmp*, with NNNN the process number), which is deleted after completion of the conversion.

[Option for *ghostscript*]

MULTIPLE-PAGE INPUT FILES

The script **pstogif** can only handle *single page* PostScript (PS) or Encapsulated PostScript (EPS) files. The script does *not* check for this: it simply converts the first page only. If a (Encapsulated) PostScript file with mulitple pages needs to be converted, it first has to be split into single-page files. After which each file has to be converted separately, as **pstogif** cannot handle multiple input files in one go.

Splitting a multiple-page PS or EPS file can be done using the **pssplit** script that comes along with the **pstogif** package. This script performs sequential calls to the **psselect** program, part of the *PSUtils* package. The **psselect** program is designed to export one or more selected pages of one given input file to one given output file. It does not create separate files for separate pages, hence the need for the **pssplit** script.

The usage of **pssplit** is as follows:

pssplit [page] infile outfilemask

where *infile* is a PS or EPS file and *outfilemask* is the basis for the name of the output files for the separate pages:

outfilemask_<pagenumber>.[ps/eps]

with <*pagenumber*> is a 4-digit string (0001, 0002, etc.), and an extention depending on the type of the infile. If the *outfilemask* contains a directory path, the script creates the directory path in case it does not already exist. When processing, **pssplit** writes the total number of pages and the page being extracted to the screen.

If the number of a specific *page* is given as first command line argument, then only that page is extracted to an output file; the *page* number should be a positive integer.

Some EPS files do not contain page number instructions and **psselect** cannot split such files. To overcome this problem, **pssplit** first passes such an input file through the **eps2eps** program of the *Ghostscript* package, which optimises the EPS file and appears to add an appropriate pagination. If all goes well, the user does not notice this extra step.

SYSTEM and PROGRAM REQUIREMENTS

For **pstogif** to work, Ghostscript (**gs**) and the **Netbpm** package must be installed. The first one is usually integral part of UNIX/LINUX distributions, the **Netbpm** package often too. If not available, they can be found at *http://www.ghostscript.com/* and *http://netpbm.sourceforge.net/* respectively. Ghostscript should support the *pnmraw* format, which most versions will -- to check, type "**gs -help**" and have a look at the list

of available devices written to the screen.

In addition, the script uses standard UNIX/LINUX commands (such as **test**, **awk**, **grep**, **cat**, etc.), as well as the *rgb.txt* colour table definition file of the X11 system. The programs should be available in the path accessed by the script. To check for this, have a look at the first 25 or so lines of the script.

The conversion from PostScript to the final format is done via a series of pipes, i.e. memory: no temporary files are used (except for the **-b** option, which needs a temporary file, but is hardly ever necessary; see the description above).

Note that as part of the LaTeX2HTML package comes the script **pstoimg**, which converts a PostScript file to a bitmap image, also using **gs** and **Netbpm**, and it has some useful options, but it can -- depending on the installation -- only provide PNG and GIF images.

The **pssplit** script uses the **psselect** program, part of the package *PSUtils*, which usually is part of UNIX/LINUX distributions. If not available, see e.g. *http://www.tardis.ed.ac.uk/~ajcd/psutils/*. The script furthermore uses the **eps2eps** program of the *Ghostscript* package and a temporary file in the */tmp/* directory.

Note that there is a **psplit** program in the *NCAR GRAPHICS* package, but this package is not really standard for Linux distributions. Beside, could not locate the web page where to download 'psplit'. Using **psselect** seems to be more general though.

LICENSES and TRADEMARKS

This software is released under the GNU General Public License (GPL).

PostScript is a trademark of Adobe Systems Incorporated.

AUTHOR

Copyright (C) 2004-2011 -- Jos van Geffen; cf. **pstogif -v**

Website: *http://uv-vis.aeronomie.be/software/tools/pstogif.php*