## NAME

ppmshadow - add simulated shadows to a portable pixmap image

## SYNOPSIS

**ppmshadow** [**−b** *blur_size*] [**−k**] [**−t**] [**−x** *xoffset*] [**−y** *yoffset*] [**−u**] [*pnmfile*]

## DESCRIPTION

**ppmshadow** adds a simulated shadow to an image, giving the appearance that the contents of the image float above the page, casting a diffuse shadow on the background. Shadows can either be black, as cast by opaque objects, or translucent, where the shadow takes on the colour of the object which casts it. You can specify the extent of the shadow and its displacement from the image with command line options.

## OPTIONS

**−b** *blur_size*

Sets the distance of the light source from the image. Larger values move the light source closer, casting a more diffuse shadow, while smaller settings move the light further away, yielding a sharper shadow. *blur_size* defaults to 11 pixels.

**−k** Keep the intermediate temporary image files. When debugging, these intermediate files provide many clues as to the source of an error. See**FILES** belo w for a list of the contents of each file.

**−t** Consider the non-background material in the image translucent -- it casts shadows of its own colour rather than a black shadow, which is default. This often results in fuzzy, difficult-to-read images but in some circumstances may look better.

**−u** Print command syntax and a summary of options.

**−x** *xoffset*

Specifies the displacement of the light source to the left of the image. Larger settings of **xoffset** displace the shadow to the right, as would be cast by a light further to the left. If not specified, the horizontal offset is half of *blur_size* (above), to the left.

**−y** *yoffset*

Specifies the displacement of the light source above the top of the image. Larger settings displace the shadow downward, corresponding to moving the light further above the top of the image. If you don't specify **−y**, the vertical offset defaults to the same as the horizontal offset (above), upward.

## FILES

Input is an anymap named by the *pnmfile* command line argument; if you don't specify *pnmfile*, the input is the Standard Input file.

Output is a always a PPM file, written to Standard Output.

**pnmfile** creates a number of temporary files as it executes. It creates them in the /tmp directory, with names of the form:

**_PPMshadow***pid*-*N***.ppm**

where *pid* is the process number of the **ppmshadow** process and *N* is a number identifying the file as described below. In normal operation,**ppmshado w** deletes temporary files as soon as it is done with them and leaves no debris around after it completes. To preserve the intermediate files for debugging, use the **−k**

command line option.

*N* in the filename means:

**1** Positive binary mask

**2** Convolution kernel for blurring shadow

**3** Blurred shadow image

**4** Clipped shadow image, offset as requested

**5** Blank image with background of source image

**6** Offset shadow

**7** Inverse mask file

**8** Original image times inverse mask

**9** Generated shadow times positive mask

**10** Shadow times background colour

## LIMITATIONS

The source image must contain sufficient space on the edges in the direction in which the shadow is cast to contain the shadow -- if it doesn't some of the internal steps may fail. You can usually expand the border of a too-tightly-cropped image with **pnmmargin** before processing it with **ppmshadow**.

Black pixels and pixels with the same color as the image background don't cast a shadow. If this causes unintentional "holes" in the shadow, fill the offending areas with a color which differs from black or the background by RGB values of 1, which will be imperceptible to the viewer. Since the comparison is exact, the modified areas will now cast shadows.

The background color of the source image (which is preserved in the output) is deemed to be the color of the pixel at the top left of the input image. If that pixel isn't part of the background, simply add a one-pixel border at the top of the image, generate the shadow image, then delete the border from it.

If something goes wrong along the way, the error messages from the various Netpbm programs **ppmshadow** calls will, in general, provide little or no clue as to where **ppmshadow** went astray. In this case, Specify the **−k** option and examine the intermediate results in the temporary files (which this option causes to be preserved). If you manually run the commands that**ppmshado w** runs on these files, you can figure out where the problem is. In problem cases where you want to manually tweak the image generation process along the way, you can keep the intermediate files with the **−k** option, modify them appropriately with an image editor, then recombine them with the steps used by the code in **ppmshadow**. See the **ppmshadow.doc** document for additional details and examples of the intermediate files.

Shadows are by default black, as cast by opaque material in the image occluding white light. Use the **−t** option to simulate translucent material, where the shadow takes on the colour of the object that casts it. If the contrast between the image and background is insufficient, the **−t** option may yield unattractive results which resemble simple blurring of the original image.

Because Netpbm used to have a maximum maxval of 255, which meant that the largest convolution kernel **pnmconvol** could use was 11 by 11, **ppmshadow** includes a horrid, CPU-time-burning kludge which, if a blur of greater than 11 is requested, performs an initial convolution with an 11×11 kernel, then calls **pnmsmooth** (which is actually a script that calls pnmconvol with a 3×3 kernel) as many times as the requested blur exceeds 11. It's ugly, but it gets the job done on those rare occasions where you need a blur greater than 11.

If you wish to generate an image at high resolution, then scale it to publication size with **pnmscale** in order to eliminate jagged edges by resampling, it's best to generate the shadow in the original high resolution image, prior to scaling it down in size. If you scale first and then add the shadow, you'll get an unsightly jagged stripe between the edge of material and its shadow, due to resampled pixels intermediate between the image and background obscuring the shadow.

## EXIT STATUS

**ppmshadow** returns status 0 if processing was completed without errors, and a nonzero Unix error code if an error prevented generation of output. Some errors may result in the script aborting, usually displaying error messages from various Netpbm components it uses, without returning a nonzero error code. When this happens, the output file will be empty, so be sure to test this if you need to know if the program succeeded.

## SEE ALSO

**pnm**(5), **pnmmargin**(1), **pnmconvol**(1), **pnmscale**(1), **pnmsmooth**(1), **ppm**(5)

## AUTHOR

John Walker <http://www.fourmilab.ch> August 8, 1997

## COPYRIGHT