NAME

jpegtran – lossless transformation of JPEG files

SYNOPSIS

jpegtran [*options*] [*filename*]

DESCRIPTION

jpegtran performs various useful transformations of JPEG files. It can translate the coded representation from one variant of JPEG to another, for example from baseline JPEG to progressive JPEG or vice versa. It can also perform some rearrangements of the image data, for example turning an image from landscape to portrait format by rotation.

jpegtran works by rearranging the compressed data (DCT coefficients), without ever fully decoding the image. Therefore, its transformations are lossless: there is no image degradation at all, which would not be true if you used **djpeg** followed by **cjpeg** to accomplish the same conversion. But by the same token, **jpegtran** cannot perform lossy operations such as changing the image quality.

jpegtran reads the named JPEG/JFIF file, or the standard input if no file is named, and produces a JPEG/JFIF file on the standard output.

OPTIONS

All switch names may be abbreviated; for example, **–optimize** may be written **–opt** or **–o**. Upper and lower case are equivalent. British spellings are also accepted (e.g., **–optimise**), though for brevity these are not mentioned below.

To specify the coded JPEG representation used in the output file, **jpegtran** accepts a subset of the switches recognized by **cjpeg**:

–optimize
Perform optimization of entropy encoding parameters.

–progressive
Create progressive JPEG file.

–restart *N*
Emit a JPEG restart marker every N MCU rows, or every N MCU blocks if "B" is attached to the number.

–arithmetic
Use arithmetic coding.

–scans *file*
Use the scan script given in the specified text file.

See **cjpeg**(1) for more details about these switches. If you specify none of these switches, you get a plain baseline-JPEG output file. The quality setting and so forth are determined by the input file.

The image can be losslessly transformed by giving one of these switches:

–flip horizontal
Mirror image horizontally (left-right).

–flip vertical
Mirror image vertically (top-bottom).

–rotate 90
Rotate image 90 degrees clockwise.

–rotate 180
Rotate image 180 degrees.

–rotate 270
Rotate image 270 degrees clockwise (or 90 ccw).

–transpose

Transpose image (across UL-to-LR axis).

–transverse

Transverse transpose (across UR-to-LL axis).

The transpose transformation has no restrictions regarding image dimensions. The other transformations operate rather oddly if the image dimensions are not a multiple of the iMCU size (usually 8 or 16 pixels), because they can only transform complete blocks of DCT coefficient data in the desired way.

jpegtran's default behavior when transforming an odd-size image is designed to preserve exact reversibility and mathematical consistency of the transformation set. As stated, transpose is able to flip the entire image area. Horizontal mirroring leaves any partial iMCU column at the right edge untouched, but is able to flip all rows of the image. Similarly, vertical mirroring leaves any partial iMCU row at the bottom edge untouched, but is able to flip all columns. The other transforms can be built up as sequences of transpose and flip operations; for consistency, their actions on edge pixels are defined to be the same as the end result of the corresponding transpose-and-flip sequence.

For practical use, you may prefer to discard any untransformable edge pixels rather than having a strange-looking strip along the right and/or bottom edges of a transformed image. To do this, add the **–trim** switch:

–trim Drop non-transformable edge blocks.

Obviously, a transformation with **–trim** is not reversible, so strictly speaking **jpegtran** with this switch is not lossless. Also, the expected mathematical equivalences between the transformations no longer hold. For example, **–rot 270 -trim** trims only the bottom edge, but **–rot 90 -trim** followed by **–rot 180 -trim** trims both edges.

If you are only interested in perfect transformation, add the **–perfect** switch:

–perfect

Fails with an error if the transformation is not perfect.

For example you may want to do

(jpegtran –rot 90 -perfect *foo.jpg* **|| djpeg** *foo.jpg* **| pnmflip –r90 | cjpeg)**

to do a perfect rotation if available or an approximated one if not.

We also offer a lossless-crop option, which discards data outside a given image region but losslessly preserves what is inside. Like the rotate and flip transforms, lossless crop is restricted by the current JPEG format: the upper left corner of the selected region must fall on an iMCU boundary. If this does not hold for the given crop parameters, we silently move the upper left corner up and/or left to make it so, simultaneously increasing the region dimensions to keep the lower right crop corner unchanged. (Thus, the output image covers at least the requested region, but may cover more.)

The image can be losslessly cropped by giving the switch:

–crop WxH+X+Y

Crop to a rectangular subarea of width W, height H starting at point X,Y.

Other not-strictly-lossless transformation switches are:

–grayscale

Force grayscale output.

This option discards the chrominance channels if the input image is YCbCr (ie, a standard color JPEG), resulting in a grayscale JPEG file. The luminance channel is preserved exactly, so this is a better method of reducing to grayscale than decompression, conversion, and recompression. This switch is particularly handy for fixing a monochrome picture that was mistakenly encoded as a color JPEG. (In such a case, the space savings from getting rid of the near-empty chroma channels won't be large; but the decoding time for a grayscale JPEG is substantially less than that for a color JPEG.)

−scale *M/N*
: Scale the output image by a factor M/N.

Currently supported scale factors are M/N with all M from 1 to 16, where N is the source DCT size, which is 8 for baseline JPEG. If the /N part is omitted, then M specifies the DCT scaled size to be applied on the given input. For baseline JPEG this is equivalent to M/8 scaling, since the source DCT size for baseline JPEG is 8. **Caution:** An implementation of the JPEG SmartScale extension is required for this feature. SmartScale enabled JPEG is not yet widely implemented, so many decoders will be unable to view a SmartScale extended JPEG file at all.

jpegtran also recognizes these switches that control what to do with "extra" markers, such as comment blocks:

−copy none
Copy no extra markers from source file. This setting suppresses all comments and other excess baggage present in the source file.

−copy comments
Copy only comment markers. This setting copies comments from the source file, but discards any other inessential (for image display) data.

−copy all
Copy all extra markers. This setting preserves miscellaneous markers found in the source file, such as JFIF thumbnails, Exif data, and Photoshop settings. In some files these extra markers can be sizable.

The default behavior is **−copy comments**. (Note: in IJG releases v6 and v6a, **jpegtran** always did the equivalent of **−copy none**.)

Additional switches recognized by jpegtran are:

−maxmemory *N*
Set limit for amount of memory to use in processing large images. Value is in thousands of bytes, or millions of bytes if "M" is attached to the number. For example, **−max 4m** selects 4000000 bytes. If more space is needed, temporary files will be used.

−outfile *name*
Send output image to the named file, not to standard output.

−verbose
Enable debug printout. More **−v**'s give more output. Also, version information is printed at startup.

−debug
Same as **−verbose**.

EXAMPLES

This example converts a baseline JPEG file to progressive form:

jpegtran −progressive *foo.jpg* **>** *fooprog.jpg*

This example rotates an image 90 degrees clockwise, discarding any unrotatable edge pixels:

jpegtran −rot 90 -trim *foo.jpg* **>** *foo90.jpg*

ENVIRONMENT

JPEGMEM
If this environment variable is set, its value is the default memory limit. The value is specified as described for the **−maxmemory** switch. **JPEGMEM** overrides the default value specified when the program was compiled, and itself is overridden by an explicit **−maxmemory**.

SEE ALSO

cjpeg(1), **djpeg**(1), **rdjpgcom**(1), **wrjpgcom**(1)
Wallace, Gregory K. "The JPEG Still Picture Compression Standard", Communications of the ACM, April 1991 (vol. 34, no. 4), pp. 30-44.

AUTHOR

Independent JPEG Group

BUGS

The transform options can't transform odd-size images perfectly. Use **−trim** or **−perfect** if you don't like the results.

The entire image is read into memory and then written out again, even in cases where this isn't really necessary. Expect swapping on large images, especially when using the more complex transform options.