

NAME

`cjpeg` – compress an image file to a JPEG file

SYNOPSIS

`cjpeg` [*options*] [*filename*]

DESCRIPTION

cjpeg compresses the named image file, or the standard input if no file is named, and produces a JPEG/JFIF file on the standard output. The currently supported input file formats are: PPM (PBMPLUS color format), PGM (PBMPLUS grayscale format), BMP, GIF [legacy feature], and Targa [legacy feature].

OPTIONS

All switch names may be abbreviated; for example, **-grayscale** may be written **-gray** or **-gr**. Most of the "basic" switches can be abbreviated to as little as one letter. Upper and lower case are equivalent (thus **-BMP** is the same as **-bmp**). British spellings are also accepted (e.g. **-greyscale**), though for brevity these are not mentioned below.

The basic switches are:

-quality *N*[,...]

Scale quantization tables to adjust image quality. Quality is 0 (worst) to 100 (best); default is 75. (See below for more info.)

-grayscale

Create monochrome JPEG file from color input. By specifying **-grayscale**, you'll get a smaller JPEG file that takes less time to process.

-rgb

Create RGB JPEG file. Using this switch suppresses the conversion from RGB colorspace input to the default YCbCr JPEG colorspace.

-optimize

Perform optimization of entropy encoding parameters. Without this, default encoding parameters are used. **-optimize** usually makes the JPEG file a little smaller, but **cjpeg** runs somewhat slower and needs much more memory. Image quality and speed of decompression are unaffected by **-optimize**.

-progressive

Create progressive JPEG file (see below). Implies **-optimize** unless **-arithmetic** is also specified.

-targa

Input file is Targa format [legacy feature]. Targa files that contain an "identification" field will not be automatically recognized by **cjpeg**. For such files, you must specify **-targa** to make **cjpeg** treat the input as Targa format. For most Targa files, you won't need this switch.

The **-quality** switch lets you trade off compressed file size against quality of the reconstructed image: the higher the quality setting, the larger the JPEG file, and the closer the output image will be to the original input. Normally you want to use the lowest quality setting (smallest file) that decompresses into something visually indistinguishable from the original image. For this purpose the quality setting should generally be between 50 and 95 (the default is 75) for photographic images. If you see defects at **-quality** 75, then go up 5 or 10 counts at a time until you are happy with the output image. (The optimal setting will vary from one image to another.)

-quality 100 will generate a quantization table of all 1's, minimizing loss in the quantization step (but there is still information loss in subsampling, as well as roundoff error.) For most images, specifying a quality value above about 95 will increase the size of the compressed file dramatically, and while the quality gain from these higher quality values is measurable (using metrics such as PSNR or SSIM), it is rarely perceivable by human vision.

In the other direction, quality values below 50 will produce very small files of low image quality. Settings around 5 to 10 might be useful in preparing an index of a large image library, for example. Try **-quality** 2 (or so) for some amusing Cubist effects. (Note: quality values below about 25 generate 2-byte quantization tables, which are considered optional in the JPEG standard. **cjpeg** emits a warning message when you give such a quality value, because some other JPEG programs may be unable to decode the resulting file. Use

-baseline if you need to ensure compatibility at low quality values.)

The **-quality** option has been extended in this version of **cjpeg** to support separate quality settings for luminance and chrominance (or, in general, separate settings for every quantization table slot.) The principle is the same as chrominance subsampling: since the human eye is more sensitive to spatial changes in brightness than spatial changes in color, the chrominance components can be quantized more than the luminance components without incurring any visible image quality loss. However, unlike subsampling, this feature reduces data in the frequency domain instead of the spatial domain, which allows for more fine-grained control. This option is useful in quality-sensitive applications, for which the artifacts generated by subsampling may be unacceptable.

The **-quality** option accepts a comma-separated list of parameters, which respectively refer to the quality levels that should be assigned to the quantization table slots. If there are more q-table slots than parameters, then the last parameter is replicated. Thus, if only one quality parameter is given, this is used for both luminance and chrominance (slots 0 and 1, respectively), preserving the legacy behavior of **cjpeg** v6b and prior. More (or customized) quantization tables can be set with the **-qtables** option and assigned to components with the **-qslots** option (see the "wizard" switches below.)

JPEG files generated with separate luminance and chrominance quality are fully compliant with standard JPEG decoders.

CAUTION: For this setting to be useful, be sure to pass an argument of **-sample 1x1** to **cjpeg** to disable chrominance subsampling. Otherwise, the default subsampling level (2x2, AKA "4:2:0") will be used.

The **-progressive** switch creates a "progressive JPEG" file. In this type of JPEG file, the data is stored in multiple scans of increasing quality. If the file is being transmitted over a slow communications link, the decoder can use the first scan to display a low-quality image very quickly, and can then improve the display with each subsequent scan. The final image is exactly equivalent to a standard JPEG file of the same quality setting, and the total file size is about the same --- often a little smaller.

Switches for advanced users:

-precision *N*

Create JPEG file with *N*-bit data precision. *N* is 2 to 16; default is 8. If *N* is not 8 or 12, then **-lossless** must also be specified. Note that only the PBMPLUS input file format supports data precisions other than 8. Note also that PBMPLUS input files are silently scaled to the target data precision, even if it is lower than the precision of the input file. Passing an argument of **-verbose** to **cjpeg** will cause it to print information about the precision of the input file. **Caution:** only 8-bit data precision is widely implemented, so many decoders will be unable to handle JPEG files with other data precisions.

-precision 12 implies **-optimize** unless **-arithmetic** is also specified.

-lossless *psv[,Pt]*

Create a lossless JPEG file using the specified predictor selection value (1 through 7) and optional point transform (0 through *precision* - 1, where *precision* is the JPEG data precision in bits). A point transform value of 0 (the default) is necessary in order to create a fully lossless JPEG file. (A non-zero point transform value right-shifts the input samples by the specified number of bits, which is effectively a form of lossy color quantization.) **Caution:** lossless JPEG is not yet widely implemented, so many decoders will be unable to handle a lossless JPEG file at all. In most cases, compressing and decompressing a lossless JPEG file is considerably slower than compressing and decompressing a lossy JPEG file, and lossless JPEG files are much larger than lossy JPEG files. Also note that the following features will be unavailable when compressing or decompressing a lossless JPEG file:

- Quality/quantization table selection
- Color space conversion (the JPEG image will use the same color space as the input image)
- Color quantization

- DCT/IDCT algorithm selection
- Smoothing
- Downsampling/upsampling
- IDCT scaling
- Partial image decompression
- Transformations using **jpegtran**

Any switches used to enable or configure those features will be ignored.

-arithmetic

Use arithmetic coding. **Caution:** arithmetic-coded JPEG is not yet widely implemented, so many decoders will be unable to handle an arithmetic-coded JPEG file at all.

-dct int

Use accurate integer DCT method (default).

-dct fast

Use less accurate integer DCT method [legacy feature]. When the Independent JPEG Group's software was first released in 1991, the compression time for a 1-megapixel JPEG image on a mainstream PC was measured in minutes. Thus, the **fast** integer DCT algorithm provided noticeable performance benefits. On modern CPUs running libjpeg-turbo, however, the compression time for a 1-megapixel JPEG image is measured in milliseconds, and thus the performance benefits of the **fast** algorithm are much less noticeable. On modern x86/x86-64 CPUs that support AVX2 instructions, the **fast** and **int** methods have similar performance. On other types of CPUs, the **fast** method is generally about 5-15% faster than the **int** method.

For quality levels of 90 and below, there should be little or no perceptible quality difference between the two algorithms. For quality levels above 90, however, the difference between the **fast** and **int** methods becomes more pronounced. With quality=97, for instance, the **fast** method incurs generally about a 1-3 dB loss in PSNR relative to the **int** method, but this can be larger for some images. Do not use the **fast** method with quality levels above 97. The algorithm often degenerates at quality=98 and above and can actually produce a more lossy image than if lower quality levels had been used. Also, in libjpeg-turbo, the **fast** method is not fully accelerated for quality levels above 97, so it will be slower than the **int** method.

-dct float

Use floating-point DCT method [legacy feature]. The **float** method does not produce significantly more accurate results than the **int** method, and it is much slower. The **float** method may also give different results on different machines due to varying roundoff behavior, whereas the integer methods should give the same results on all machines.

-icc file

Embed ICC color management profile contained in the specified file.

-restart N

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

In typical JPEG images, an MCU (Minimum Coded Unit) is the minimum set of interleaved "data units" (8x8 DCT blocks if the image is lossy or samples if the image is lossless) necessary to represent at least one data unit per component. (For example, an MCU in an interleaved lossy JPEG image that uses 4:2:2 subsampling consists of two luminance blocks followed by one block for each chrominance component.) In single-component or non-interleaved JPEG images, an MCU is the same as a data unit. An MCU row is a row of MCUs spanning the entire width of the image.

-restart 0 (the default) means no restart markers.

- smooth** *N*
Smooth the input image to eliminate dithering noise. *N*, ranging from 1 to 100, indicates the strength of smoothing. 0 (the default) means no smoothing.
- 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, an error will occur.
- outfile** *name*
Send output image to the named file, not to standard output.
- memdst**
Compress to memory instead of a file. This feature was implemented mainly as a way of testing the in-memory destination manager (`jpeg_mem_dest()`), but it is also useful for benchmarking, since it reduces the I/O overhead.
- report**
Report compression progress.
- strict** Treat all warnings as fatal. Enabling this option will cause the compressor to abort if an LZW-compressed GIF input image contains incomplete or corrupt image data.
- verbose**
Enable debug printout. More **-v**'s give more output. Also, version information is printed at start-up.
- debug**
Same as **-verbose**.
- version**
Print version information and exit.

The **-restart** option inserts extra markers that allow a JPEG decoder to resynchronize after a transmission error. Without restart markers, any damage to a compressed file will usually ruin the image from the point of the error to the end of the image; with restart markers, the damage is usually confined to the portion of the image up to the next restart marker. Of course, the restart markers occupy extra space. We recommend **-restart 1** for images that will be transmitted across unreliable networks such as Usenet.

The **-smooth** option filters the input to eliminate fine-scale noise. This is often useful when converting dithered images to JPEG: a moderate smoothing factor of 10 to 50 gets rid of dithering patterns in the input file, resulting in a smaller JPEG file and a better-looking image. Too large a smoothing factor will visibly blur the image, however.

Switches for wizards:

- baseline**
Force baseline-compatible quantization tables to be generated. This clamps quantization values to 8 bits even at low quality settings. (This switch is poorly named, since it does not ensure that the output is actually baseline JPEG. For example, you can use **-baseline** and **-progressive** together.)
- qtables** *file*
Use the quantization tables given in the specified text file.
- qslots** *N[,...]*
Select which quantization table to use for each color component.
- sample** *HxV[,...]*
Set JPEG sampling factors for each color component.
- scans** *file*
Use the scan script given in the specified text file.

The "wizard" switches are intended for experimentation with JPEG. If you don't know what you are doing, **don't use them**. These switches are documented further in the file `wizard.txt`.

EXAMPLES

This example compresses the PPM file `foo.ppm` with a quality factor of 60 and saves the output as `foo.jpg`:

```
cjpeg -quality 60 foo.ppm > foo.jpg
```

HINTS

Color GIF files are not the ideal input for JPEG; JPEG is really intended for compressing full-color (24-bit through 48-bit) images. In particular, don't try to convert cartoons, line drawings, and other images that have only a few distinct colors. GIF works great on these; JPEG does not. If you want to convert a GIF to JPEG, you should experiment with **cjpeg**'s **-quality** and **-smooth** options to get a satisfactory conversion. **-smooth 10** or so is often helpful.

Avoid running an image through a series of JPEG compression/decompression cycles. Image quality loss will accumulate; after ten or so cycles the image may be noticeably worse than it was after one cycle. It's best to use a lossless format while manipulating an image, then convert to JPEG format when you are ready to file the image away.

The **-optimize** option to **cjpeg** is worth using when you are making a "final" version for posting or archiving. It's also a win when you are using low quality settings to make very small JPEG files; the percentage improvement is often a lot more than it is on larger files. (At present, **-optimize** mode is always selected when generating progressive JPEG files.)

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

djpeg(1), **jpegtran**(1), **rdjpgcom**(1), **wrjpgcom**(1)
ppm(5), **pgm**(5)

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

This file was modified by The libjpeg-turbo Project to include only information relevant to libjpeg-turbo, to wordsmith certain sections, and to describe features not present in libjpeg.

ISSUES

Not all variants of BMP and Targa file formats are supported.