Advanced usage instructions for the Independent JPEG Group's JPEG software

==========================================================================

This file describes cjpeg's "switches for wizards".

The "wizard" switches are intended for experimentation with JPEG by persons

who are reasonably knowledgeable about the JPEG standard. If you don't know

what you are doing, DON'T USE THESE SWITCHES. You'll likely produce files

with worse image quality and/or poorer compression than you'd get from the

default settings. Furthermore, these switches must be used with caution

when making files intended for general use, because not all JPEG decoders

will support unusual JPEG parameter settings.

Quantization Table Adjustment

-----------------------------

Ordinarily, cjpeg starts with a default set of tables (the same ones given

as examples in the JPEG standard) and scales them up or down according to

the -quality setting. The details of the scaling algorithm can be found in

jcparam.c. At very low quality settings, some quantization table entries

can get scaled up to values exceeding 255. Although 2-byte quantization

values are supported by the IJG software, this feature is not in baseline

JPEG and is not supported by all implementations. If you need to ensure

wide compatibility of low-quality files, you can constrain the scaled

quantization values to no more than 255 by giving the -baseline switch.

Note that use of -baseline will result in poorer quality for the same file

size, since more bits than necessary are expended on higher AC coefficients.

You can substitute a different set of quantization values by using the

-qtables switch:

-qtables file Use the quantization tables given in the named file.

The specified file should be a text file containing decimal quantization

values. The file should contain one to four tables, each of 64 elements.

The tables are implicitly numbered 0,1,etc. in order of appearance. Table

entries appear in normal array order (NOT in the zigzag order in which they

will be stored in the JPEG file).

Quantization table files are free format, in that arbitrary whitespace can

appear between numbers. Also, comments can be included: a comment starts

with '#' and extends to the end of the line. Here is an example file that

duplicates the default quantization tables:

# Quantization tables given in JPEG spec, section K.1

# This is table 0 (the luminance table):

16 11 10 16 24 40 51 61

12 12 14 19 26 58 60 55

14 13 16 24 40 57 69 56

14 17 22 29 51 87 80 62

18 22 37 56 68 109 103 77

24 35 55 64 81 104 113 92

49 64 78 87 103 121 120 101

72 92 95 98 112 100 103 99

# This is table 1 (the chrominance table):

17 18 24 47 99 99 99 99

18 21 26 66 99 99 99 99

24 26 56 99 99 99 99 99

47 66 99 99 99 99 99 99

99 99 99 99 99 99 99 99

99 99 99 99 99 99 99 99

99 99 99 99 99 99 99 99

99 99 99 99 99 99 99 99

If the -qtables switch is used without -quality, then the specified tables

are used exactly as-is. If both -qtables and -quality are used, then the

tables taken from the file are scaled in the same fashion that the default

tables would be scaled for that quality setting. If -baseline appears, then

the quantization values are constrained to the range 1-255.

By default, cjpeg will use quantization table 0 for luminance components and

table 1 for chrominance components. To override this choice, use the -qslots

switch:

-qslots N[,...] Select which quantization table to use for

each color component.

The -qslots switch specifies a quantization table number for each color

component, in the order in which the components appear in the JPEG SOF marker.

For example, to create a separate table for each of Y,Cb,Cr, you could

provide a -qtables file that defines three quantization tables and say

"-qslots 0,1,2". If -qslots gives fewer table numbers than there are color

components, then the last table number is repeated as necessary.

Sampling Factor Adjustment

--------------------------

By default, cjpeg uses 2:1 horizontal and vertical downsampling when

compressing YCbCr data, and no downsampling for all other color spaces.

You can override this default with the -sample switch:

-sample HxV[,...] Set JPEG sampling factors for each color

component.

The -sample switch specifies the JPEG sampling factors for each color

component, in the order in which they appear in the JPEG SOF marker.

If you specify fewer HxV pairs than there are components, the remaining

components are set to 1x1 sampling. For example, the default YCbCr setting

is equivalent to "-sample 2x2,1x1,1x1", which can be abbreviated to

"-sample 2x2".

There are still some JPEG decoders in existence that support only 2x1

sampling (also called 4:2:2 sampling). Compatibility with such decoders can

be achieved by specifying "-sample 2x1". This is not recommended unless

really necessary, since it increases file size and encoding/decoding time

with very little quality gain.

Multiple Scan / Progression Control

-----------------------------------

By default, cjpeg emits a single-scan sequential JPEG file. The

-progressive switch generates a progressive JPEG file using a default series

of progression parameters. You can create multiple-scan sequential JPEG

files or progressive JPEG files with custom progression parameters by using

the -scans switch:

-scans file Use the scan sequence given in the named file.

The specified file should be a text file containing a "scan script".

The script specifies the contents and ordering of the scans to be emitted.

Each entry in the script defines one scan. A scan definition specifies

the components to be included in the scan, and for progressive JPEG it also

specifies the progression parameters Ss,Se,Ah,Al for the scan. Scan

definitions are separated by semicolons (';'). A semicolon after the last

scan definition is optional.

Each scan definition contains one to four component indexes, optionally

followed by a colon (':') and the four progressive-JPEG parameters. The

component indexes denote which color component(s) are to be transmitted in

the scan. Components are numbered in the order in which they appear in the

JPEG SOF marker, with the first component being numbered 0. (Note that these

indexes are not the "component ID" codes assigned to the components, just

positional indexes.)

The progression parameters for each scan are:

Ss Zigzag index of first coefficient included in scan

Se Zigzag index of last coefficient included in scan

Ah Zero for first scan of a coefficient, else Al of prior scan

Al Successive approximation low bit position for scan

If the progression parameters are omitted, the values 0,63,0,0 are used,

producing a sequential JPEG file. cjpeg automatically determines whether

the script represents a progressive or sequential file, by observing whether

Ss and Se values other than 0 and 63 appear. (The -progressive switch is

not needed to specify this; in fact, it is ignored when -scans appears.)

The scan script must meet the JPEG restrictions on progression sequences.

(cjpeg checks that the spec's requirements are obeyed.)

Scan script files are free format, in that arbitrary whitespace can appear

between numbers and around punctuation. Also, comments can be included: a

comment starts with '#' and extends to the end of the line. For additional

legibility, commas or dashes can be placed between values. (Actually, any

single punctuation character other than ':' or ';' can be inserted.) For

example, the following two scan definitions are equivalent:

0 1 2: 0 63 0 0;

0,1,2 : 0-63, 0,0 ;

Here is an example of a scan script that generates a partially interleaved

sequential JPEG file:

0; # Y only in first scan

1 2; # Cb and Cr in second scan

Here is an example of a progressive scan script using only spectral selection

(no successive approximation):

# Interleaved DC scan for Y,Cb,Cr:

0,1,2: 0-0, 0, 0 ;

# AC scans:

0: 1-2, 0, 0 ; # First two Y AC coefficients

0: 3-5, 0, 0 ; # Three more

1: 1-63, 0, 0 ; # All AC coefficients for Cb

2: 1-63, 0, 0 ; # All AC coefficients for Cr

0: 6-9, 0, 0 ; # More Y coefficients

0: 10-63, 0, 0 ; # Remaining Y coefficients

Here is an example of a successive-approximation script. This is equivalent

to the default script used by "cjpeg -progressive" for YCbCr images:

# Initial DC scan for Y,Cb,Cr (lowest bit not sent)

0,1,2: 0-0, 0, 1 ;

# First AC scan: send first 5 Y AC coefficients, minus 2 lowest bits:

0: 1-5, 0, 2 ;

# Send all Cr,Cb AC coefficients, minus lowest bit:

# (chroma data is usually too small to be worth subdividing further;

# but note we send Cr first since eye is least sensitive to Cb)

2: 1-63, 0, 1 ;

1: 1-63, 0, 1 ;

# Send remaining Y AC coefficients, minus 2 lowest bits:

0: 6-63, 0, 2 ;

# Send next-to-lowest bit of all Y AC coefficients:

0: 1-63, 2, 1 ;

# At this point we've sent all but the lowest bit of all coefficients.

# Send lowest bit of DC coefficients

0,1,2: 0-0, 1, 0 ;

# Send lowest bit of AC coefficients

2: 1-63, 1, 0 ;

1: 1-63, 1, 0 ;

# Y AC lowest bit scan is last; it's usually the largest scan

0: 1-63, 1, 0 ;

It may be worth pointing out that this script is tuned for quality settings

of around 50 to 75. For lower quality settings, you'd probably want to use

a script with fewer stages of successive approximation (otherwise the

initial scans will be really bad). For higher quality settings, you might

want to use more stages of successive approximation (so that the initial

scans are not too large).