Introduction to JPEG

JPEG is an image compression standard which was developed by "Joint Photographic Experts Group". In 1992, it was accepted as an international standard. JPEG is a lossy image compression method. JPEG compression uses the DCT (Discrete Cosine Transform) method for coding transformation. It allows a tradeoff between storage size and the degree of compression can be adjusted.

JPEG compression is used in a number of image file formats. JPEG/Exif is the most common image format used by digital cameras and other photographic image capture devices; along with JPEG/JFIF, it is the most common format for storing and transmitting photographic images on the World Wide Web. These format variations are often not distinguished, and are simply called JPEG.

The MIME media type for JPEG is *image/jpeg*, except in older Internet Explorer versions, which provides a MIME type of *image/pjpeg* when uploading JPEG images. JPEG files usually have a filename extension of `.jpg` or `.jpeg`. JPEG/JFIF supports a maximum image size of 65,535×65,535 pixels, hence up to 4 gigapixels for an aspect ratio of 1:1. In 2000, the JPEG group introduced a format intended to be a successor, JPEG 2000, but it was unable to replace the original JPEG as the dominant image standard

JPEG standard

"JPEG" stands for Joint Photographic Experts Group, the name of the committee that created the JPEG standard and also other still picture coding standards. Founded in 1986, the group developed the JPEG standard during the late 1980s. Among several transform coding techniques they examined, they selected the discrete cosine transform (DCT), as it was by far the most efficient practical compression technique. The group published the JPEG standard in 1992.

The JPEG standard specifies the codec, which defines how an image is compressed into a stream of bytes and decompressed back into an image, but not the file format used to contain that stream. The Exif and JFIF standards define the commonly used file formats for interchange of JPEG-compressed images.

Typical usage

The JPEG compression algorithm operates at its best on photographs and paintings of realistic scenes with smooth variations of tone and color. For web usage, where reducing the amount of data used for an image is important for responsive presentation, JPEG's compression benefits make JPEG popular. JPEG/Exif is also the most common format saved by digital cameras.

However, JPEG is not well suited for line drawings and other textual or iconic graphics, where the sharp contrasts between adjacent pixels can cause noticeable artifacts. Such images are better saved in a lossless graphics format such as TIFF, GIF, or PNG. The JPEG standard includes a lossless coding mode, but that mode is not supported in most products.

As the typical use of JPEG is a lossy compression method, which reduces the image fidelity, it is inappropriate for exact reproduction of imaging data (such as some scientific and medical imaging applications and certain technical image processing work).

JPEG is also not well suited to files that will undergo multiple edits, as some image quality is lost each time the image is recompressed, particularly if the image is cropped or shifted, or if encoding parameters are changed – see digital generation loss for details. To prevent image information loss during sequential and repetitive editing, the first edit can be saved in a lossless format, subsequently edited in that format, then finally published as JPEG for distribution.

JPEG compression

JPEG uses a lossy form of compression based on the discrete cosine transform (DCT). This mathematical operation converts each frame/field of the video source from the spatial (2D) domain into the frequency domain (a.k.a. transform domain). A perceptual model based loosely on the human psycho visual system discards high-frequency information, i.e. sharp transitions in intensity, and color hue. In the transform domain, the process of reducing information is called quantization. In simpler terms, quantization is a method for optimally reducing a large number scale (with different occurrences of each number) into a smaller one, and the transform-domain is a convenient representation of the image because the high-frequency coefficients, which contribute less to the overall picture than other coefficients, are characteristically small-values with high compressibility. The quantized coefficients are then sequenced and losslessly packed into the output bit-stream. Nearly all software implementations of JPEG permit user control over the compression ratio (as well as other optional parameters), allowing the user to trade off picture-quality for smaller file size. In embedded applications (such as miniDV, which uses a similar DCT-compression scheme), the parameters are pre-selected and fixed for the application.

The compression method is usually lossy, meaning that some original image information is lost and cannot be restored, possibly affecting image quality. There is an optional lossless mode defined in the JPEG standard. However, this mode is not widely supported in products.

There is also an interlaced *progressive* JPEG format, in which data is compressed in multiple passes of progressively higher detail. This is ideal for large images that will be displayed while downloading over a slow connection, allowing a reasonable preview after receiving only a portion of the data. However, support for progressive JPEGs is not universal. When progressive JPEGs are received by programs that do not support them (such as versions of Internet Explorer before Windows 7) the software displays the image only after it has been completely downloaded.

JPEG files

The file format known as "JPEG Interchange Format" (JIF) is specified in Annex B of the standard. However, this "pure" file format is rarely used, primarily because of the difficulty of programming encoders and decoders that fully implement all aspects of the standard and because of certain shortcomings of the standard:

- Color space definition
- Component sub-sampling registration
- Pixel aspect ratio definition.

Several additional standards have evolved to address these issues. The first of these, released in 1992, was the JPEG File Interchange Format (or JFIF), followed in recent years by Exchangeable image file format (Exif) and ICC color profiles. Both of these formats use the actual JIF byte layout, consisting of different *markers*, but in addition, employ one of the JIF standard's extension points, namely the *application markers*: JFIF uses APP0, while Exif uses APP1. Within these segments of the file that were left for future use in the JIF standard and are not read by it, these standards add specific metadata.

Thus, in some ways, JFIF is a cut-down version of the JIF standard in that it specifies certain constraints (such as not allowing all the different encoding modes), while in other ways, it is an extension of JIF due to the added metadata. The documentation for the original JFIF standard states:

JPEG File Interchange Format is a minimal file format which enables JPEG bitstreams to be exchanged between a wide variety of platforms and applications. This minimal format does not include any of the advanced features found in the TIFF JPEG specification or any application specific file format. Nor should it, for the only purpose of this simplified format is to allow the exchange of JPEG compressed images.

Image files that employ JPEG compression are commonly called "JPEG files", and are stored in variants of the JIF image format. Most image capture devices (such as digital cameras) that output JPEG are actually creating files in the Exif format, the format that the camera industry has standardized on for metadata interchange. On the other hand, since the Exif standard does not allow color profiles, most image editing software stores JPEG in JFIF format, and also includes the APP1 segment from the Exif file to include the metadata in an almost-compliant way; the JFIF standard is interpreted somewhat flexibly.

JPEG filename extensions

The most common filename extensions for files employing JPEG compression are **.jpg** and **.jpeg**, though .jpe, .jfif and .jif are also used. It is also possible for JPEG data to be embedded in other file types – TIFF encoded files often embed a JPEG image as a thumbnail of the main image; and MP3 files can contain a JPEG of cover art in the ID3v2 tag.

Color profile

Many JPEG files embed an ICC color profile (color space). Commonly used color profiles include sRGB and Adobe RGB. Because these color spaces use a non-linear transformation, the dynamic range of an 8-bit JPEG file is about 11 stops; see gamma curve.

The general approach to image compression implies the following stages

RGB color space to YCbCr color space conversion

The human visual system can perceive minor changes of brightness, though it’s far less responsive to changes of color (chroma components of the image) for the regions with the same brightness. That’s why we can apply stronger compression to chroma to get less image size of the compressed image. We take an RGB image and convert it to luma/chroma representation in order to separate luma from chroma and to process them separately.

An digital image in RGB format that is a combination of Red, Green, Blue color channel is converted to YCbCr color channels. Y is the brightness of the image and Cb is the blue difference relative to the green color and Cr is the red difference relative to the red color.

Sampling

The transformation into the $Y'C_BC_R$ color model enables the next usual step, which is to reduce the spatial resolution of the Cb and Cr components (called "downsampling" or "chroma subsampling").

The ratios at which the downsampling is ordinarily done for JPEG images are 4:4:4 (no downsampling), 4:2:2 (reduction by a factor of 2 in the horizontal direction), or (most commonly) 4:2:0 (reduction by a factor of 2 in both the horizontal and vertical directions). For the rest of the compression process, Y', Cb and Cr are processed separately and in a very similar manner

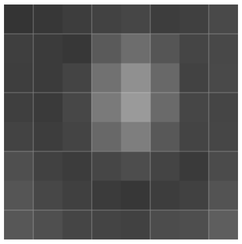
As soon as we can consider chroma components to be less important than luma, we can decrease the total number of chroma pixels. For example, we can average chroma in horizontal or vertical direction. At the most extreme case, we can average 4 neighbor chroma values in the rectangle 2x2 to get just one new value. That mode is called 4:2:0 and this is the most popular choice for subsampling.

Block splitting

For further processing, we divide the whole image into blocks 8x8 dimensions for luma and chroma. That partitioning scheme lets us process each block independently, though we will have to remember coordinates of each block which are essential at image decoding. Divide an image into blocks with each having dimensions of 8 x8.

First an image needs to be separated for 8*8 pixel blocks. That means each block has 8*8 pixels and it is 64 pixels in one block. Let's assume that the dimension of this image is 240*320. That means this image has 76800 pixels. if we divide this in 64 we can get the number of blocks. 76800 = 64 * 1200. That means we have 1200 blocks in this image. Following image is the partitioned image.

Let's take one block zoomed and the each size of these pixels. Let's take the zoomed block in 3rd column on 10th row.



One block of the image/ 8×8 sub-image shown in 8-bit gray-scale

Following is the pixel matrix of this chosen block

```
005 176 193 168 168 170 167 165
006 176 158 172 162 177 168 151
005 167 172 232 158 061 145 214
033 179 169 174 005 005 135 178
008 104 180 178 172 197 188 169
063 005 102 101 160 142 133 139
051 047 063 005 180 191 165 005
049 053 043 005 184 170 168 074
```

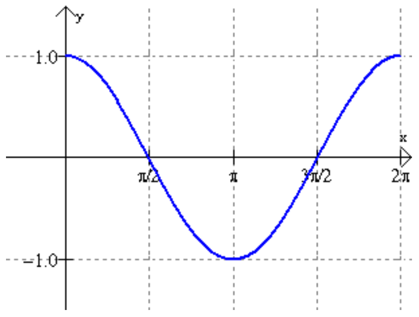
Before computing the DCT of the 8×8 block, its values are shifted from a positive range to one center on zero. For an 8-bit image, each entry in the original block falls in the range [0,255]. The midpoint of the range (in this case, the value 128) is subtracted from each entry to produce a data range that is centered on zero, so that the modified range is [-128,127] . This step reduces the dynamic range requirements in the DCT processing stage that follows.

This step results in the following values:

122 0049 0066 0041 0041 0043 0040 0038
–121 0049 0031 0045 0035 0050 0041 0024
–122 0040 0045 0105 0031 –066 0018 0087
–094 0052 0042 0047 –122 –122 0008 0051
–119 –023 0053 0051 0045 0070 0061 0042
–064 –122 –025 –026 0033 0015 0006 0012
–076 –080 –064 –122 0053 0064 0038 –122
–078 –074 –084 –122 0057 0043 0041 –053

DCT transformation

This transformation stands for discrete cosine transformation and this increases the effectiveness in the 5th step encoding. Before going to this transform I'll bit talk about the cosine signal. Cosine signal lies in between the 1 and -1. You can see in the following diagram.



What we do with this transform is that we represent the image pixels with different of cosine waves. By doing this we eliminate high frequencies in the signal. Because human eyes are not sensitive to the very high-frequency changes of the image. It is the graphical explanation of this transformation. If we talk about the numerical explanation, this transformation creates a new matrix that has the values in left upper corner of the matrix and other places are almost nearly zero. Following is the new matrix after the transformation.

-27.500	-213.468	-149.608	-95.281	-103.750	-46.946	-58.717	27.226
168.229	51.611	-21.544	-239.520	-8.238	-24.495	-52.657	-96.621
-27.198	-31.236	-32.278	173.389	-51.141	-56.942	4.002	49.143
30.184	-43.070	-50.473	67.134	-14.115	11.139	71.010	18.039
19.500	8.460	33.589	-53.113	-36.750	2.918	-5.795	-18.387
-70.593	66.878	47.441	-32.614	-8.195	18.132	-22.994	6.631
12.078	-19.127	6.252	-55.157	85.586	-0.603	8.028	11.212
71.152	-38.373	-75.924	29.294	-16.451	-23.436	-4.213	15.624

DCT values

Here what this step does is the values near to 0 are converted to the 0 and other elements also will be shrinking towards zero. Then each value of the resultant matrix is divided by another matrix called standard jpeg quantization table. There are two tables for luminance and chrominance components as shown in the following diagram.

```
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
```

Chrominance quantization table

Luminance Quantization 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
```

Following matrix is the resultant matrix after the quantization.

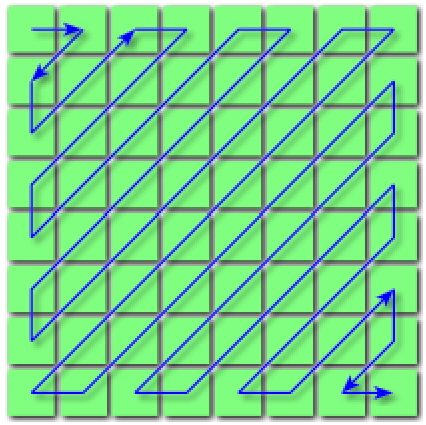
-2	-19	-15	-6	-4	-1	-1	0
14	4	-2	-13	0	0	-1	-2
-2	-2	-2	7	-1	-1	0	1
2	-3	-2	2	0	0	1	0
1	0	1	-1	-1	0	0	0
-3	2	1	-1	0	0	0	0
0	0	0	-1	1	0	0	0
1	0	-1	0	0	0	0	0

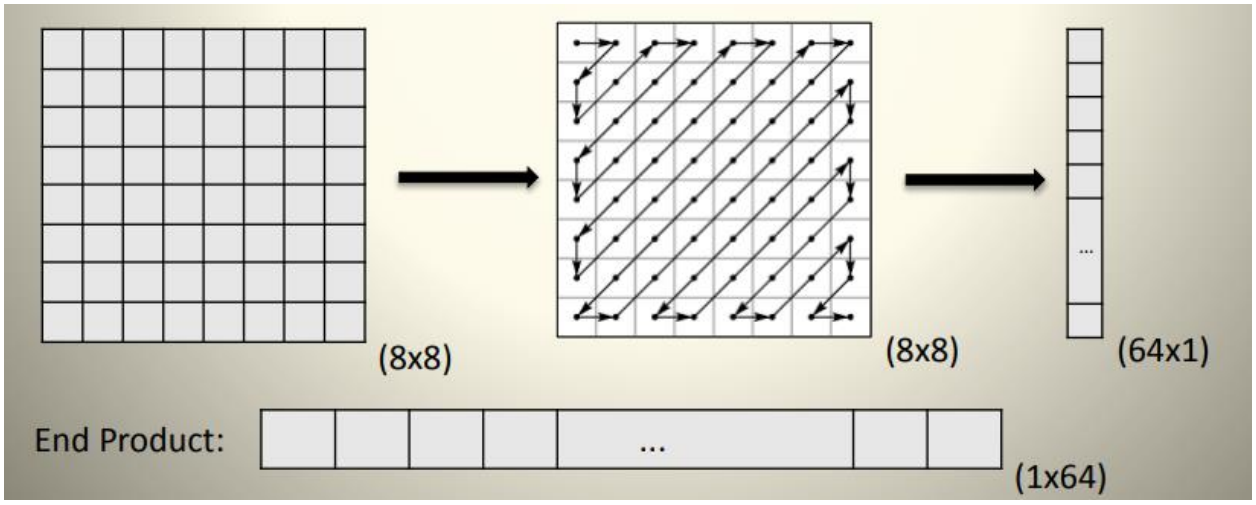
Entropy coding

Entropy coding is a special form of lossless data compression. It involves arranging the image components in a "zigzag" order employing run-length encoding (RLE) algorithm that groups similar frequencies together, inserting length coding zeros, and then using Huffman coding on what is left.

Maps 8 x 8 matrix to a 1 x 64 vector. • Why zigzag scanning? – To group low frequency coefficients at the top of the vector and high frequency coefficients at the bottom. • In order to exploit the presence of the large number of zeros in the quantized matrix, a zigzag of the matrix is used.

The JPEG standard also allows, but does not require, decoders to support the use of arithmetic coding, which is mathematically superior to Huffman coding. However, this feature has rarely been used, as it was historically covered by patents requiring royalty-bearing licenses, and because it is slower to encode and decode compared to Huffman coding.





Lossless Encoding

This is the final step and this method uses Huffman coding and it reduces a large amount of memory without losing any detail of the image. Most of the times it saves 70% memory. What this encoding method does is it gets the frequency of different pixels and stores that pixels and the frequency instead of storing each pixel.