Applications of lossless and lossy compression

Lossless compression is mainly used to compress:

- images
- sound
- text

It is generally the technique of choice for detailed product images, photography showcases, text files and spreadsheet files, where the loss of picture quality, words or data (e.g., financial data) could pose a problem.

The Graphics Interchange File (GIF), an image format used on the internet, is generally compressed using lossless compression. RAW, BMP and PNG are also lossless image formats.

Lossy compression is mainly used to compress:

- images
- audio
- video

This technique is preferred for audio and video files where some amount of information loss is acceptable since it's unlikely to be detected by most users. The JPEG image file, commonly used for photographs and other complex still images on the web with no transparency, is generally compressed using lossy compression.

Using JPEG compression, the creator can decide how much loss to introduce, and how much of a trade-off is acceptable between file size and image quality.

Lossy compression is also suitable for websites featuring JPEG files and fast loading times since the compression ratio can be adjusted while maintaining the right balance.

Algorithms used in lossless and lossy compression

Different kinds of algorithms are used to reduce file sizes in lossless and lossy compression.

The algorithms used in lossless compression are:

- Run Length Encoding
- Lempel-Ziv-Welch (LZW)
- Huffman Coding
- Arithmetic Encoding

The algorithms used in lossy compression are:

- Transform Coding
- Discrete Cosine Transform
- Discrete Wavelet Transform
- Fractal Compression

Advantages and drawbacks of lossless compression

The key benefit of lossless compression is that the quality of the file (e.g., an image) can be retained while achieving a smaller file size.

In JPEG and PNG files, this is done by removing unnecessary metadata. For applications where it's important to retain the quality of the file, lossless compression is the better choice.

The drawback of this compression technique is that larger file sizes are required to maintain post-compression quality.

Advantages and drawbacks of lossy compression

Lossy compression results in a significantly reduced file size (smaller than lossless compression), which is its most noteworthy benefit. It is supported by many tools, plugins and software products that let the user choose their preferred degree of compression.

The disadvantage is that it also results in quality loss, which may not be acceptable for some applications or users. The higher the compression ratio, the more quality degradation. Additionally, the original file -- with original quality -- cannot be recovered after compressing.

Transform coding is a type of data compression for "natural" data like audio signals or photographic images. The transformation is typically lossless (perfectly reversible) on its own but is used to enable better (more targeted) quantization, which then results in a lower quality copy of the original input (lossy compression).

In transform coding, knowledge of the application is used to choose information to discard, thereby lowering its bandwidth. The remaining information can then be compressed via a variety of methods. When the output is decoded, the result may not be identical to the original input, but is expected to be close enough for the purpose of the application.

Colour television[edit]

Further information: YIQ

NTSC[edit]

One of the most successful transform encoding system is typically not referred to as such—the example being NTSC color television. After an extensive series of studies in the 1950s, Alda Bedford showed that the human eye has high resolution only for black and white, somewhat less for "mid-range" colors like yellows and greens, and much less for colors on the end of the spectrum, reds and blues.

Using this knowledge allowed RCA to develop a system in which they discarded most of the blue signal after it comes from the camera, keeping most of the green and only some of the red; this is chroma subsampling in the YIQ color space.

PAL and SECAM[edit]

The PAL and SECAM systems use nearly identical or very similar methods to transmit colour. In any case both systems are subsampled.

. Overview

JPEG is one of the most widely-used picture formats on the internet.

In this tutorial, we'll go through each step of the JPEG compression algorithm to learn how it works.

2. What Is JPEG?

JPEG stands for Joint Photographic Experts Group and is a lossy compression algorithm that results in significantly smaller file sizes with little to no perceptible impact on picture quality and resolution. A JPEG-compressed image can be ten times smaller than the original one. **JPEG works by removing information that is not easily visible to the**

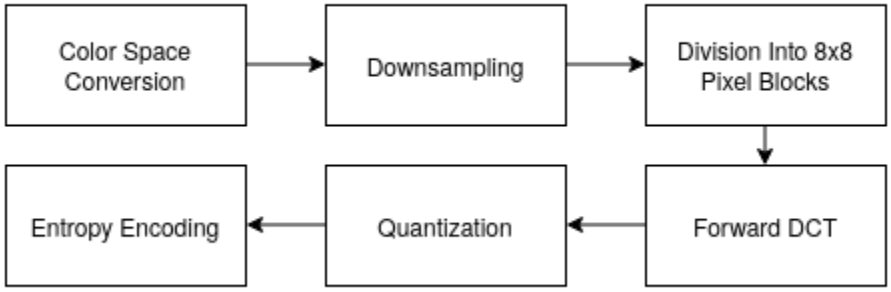
human eye while keeping the information that the human eye is good at perceiving.

2.1. What Is Lossy Compression?

A lossy compression algorithm is a compression algorithm that permanently removes some data from the original file, especially redundant data, when compressing it. On the other hand, a lossless compression algorithm is a compression algorithm that doesn't remove any information when compressing a file, and all information is restored after decompression.

3. JPEG Compression Steps

First, let's take a look at an overview of the steps that JPEG takes when compressing an image:



The above steps result in a *.jpg* image file.

3.1. Color Space Conversion

A color space is a specific organization of colors. RGB (Red, Green, and Blue) and CMYK (Cyan, Magenta, Yellow, and Key/Black) are examples of color spaces. Moreover, each pixel in a picture has its color space values.

The human eye is more sensitive to brightness than color. Therefore, the algorithm can exploit this by making significant changes to the picture's color information without affecting the picture's perceived quality. To do this, JPEG must first change the picture's color space from RGB to YCbCr.

YCbCr consists of 3 different channels: Y is the luminance channel containing brightness information, and Cb and Cr are chrominance blue and chrominance red channels containing color information. In this color space, JPEG can make changes to the chrominance channel to change the image's color information without affecting the brightness information that is located in the luminance channel.

After conversion, each pixel has three new values that represent the luminance, the chrominance blue, and the chrominance red information.

3.2. Downsampling

After we separate the color information from the brightness information, JPEG downsamples the chrominance channels to a quarter of their original size. Each block of 4 pixels is averaged into a single color value for all 4 pixels. As a result, some information is lost, and the size of the picture is halved, but since the human eye is not very sensitive to color, the changes are not easily distinguishable.

It's important to note that **downsampling is only applied to the chrominance channels, and the luminance channel keeps its original size**.

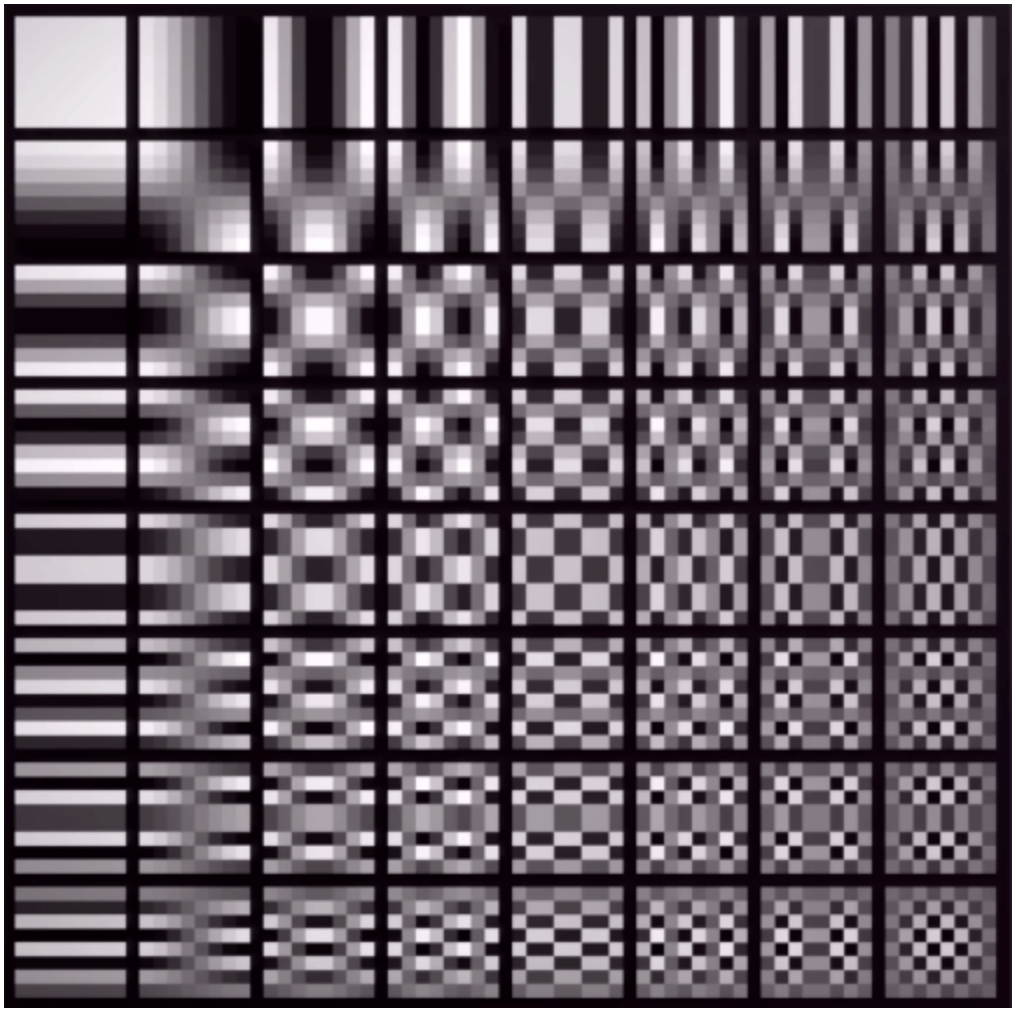
3.3. Division Into 8×8 Pixel Blocks

After downsampling, the pixel data of each channel is divided into 8×8 blocks of 64 pixels. From now on, the algorithm processes each block of pixels independently.

3.4. Forward DCT (Discrete Cosine Transform)

First, each pixel value for each channel is subtracted by 128 to make the value range from -128 to +127.

Using DCT, for each channel, each block of 64 pixels can be reconstructed by multiplying a constant set of base images by their corresponding weight values and then summing them up together. Here is what the base images look like:



Applying forward DCT on each block of 64 pixels gives us a matrix containing the corresponding weight values for each base image. These values show how much of each base image is used to reconstruct the original 8×8 pixel block, and it looks something like this:

For every block of 64 pixels, we'll have three weight matrices, one for luminance and two for chrominance. In addition, no information is lost in this phase.

3.5. Quantization

The human eye is very good at perceiving high-frequency elements in an image. In this step, JPEG removes some of this high-frequency information without affecting the perceived quality of the image.

To do this, the weight matrix is divided by a precalculated quantization table, and the results are rounded to the closest integer. Here is what a quantization table for chrominance channels looks like:

$$\begin{bmatrix} 10 & 8 & 9 & 9 & 9 & 8 & 10 & 9 \\ 9 & 9 & 10 & 10 & 10 & 11 & 12 & 17 \\ 13 & 12 & 12 & 12 & 12 & 20 & 16 & 16 \\ 14 & 17 & 18 & 20 & 23 & 23 & 22 & 20 \\ 25 & 25 & 25 & 25 & 25 & 25 & 25 & 25 \\ 25 & 25 & 25 & 25 & 25 & 25 & 25 & 25 \\ 25 & 25 & 25 & 25 & 25 & 25 & 25 & 25 \\ 25 & 25 & 25 & 25 & 25 & 25 & 25 & 25 \end{bmatrix}$$

And for the luminance channel:

$$\begin{bmatrix} 6 & 4 & 4 & 6 & 10 & 16 & 20 & 24 \\ 5 & 5 & 6 & 8 & 10 & 23 & 24 & 22 \\ 6 & 5 & 6 & 10 & 16 & 23 & 28 & 22 \\ 6 & 7 & 9 & 12 & 20 & 35 & 32 & 25 \\ 7 & 9 & 15 & 22 & 27 & 44 & 41 & 31 \\ 10 & 14 & 22 & 26 & 32 & 42 & 45 & 37 \\ 20 & 26 & 31 & 35 & 41 & 48 & 48 & 40 \\ 29 & 37 & 38 & 39 & 45 & 40 & 41 & 40 \end{bmatrix}$$

We can see that the quantization tables have higher numbers at the bottom right, where high-frequency data is located, and have lower numbers at the top left, where low-frequency data is located. As a result, after dividing each weight value, high-frequency data is rounded to 0:

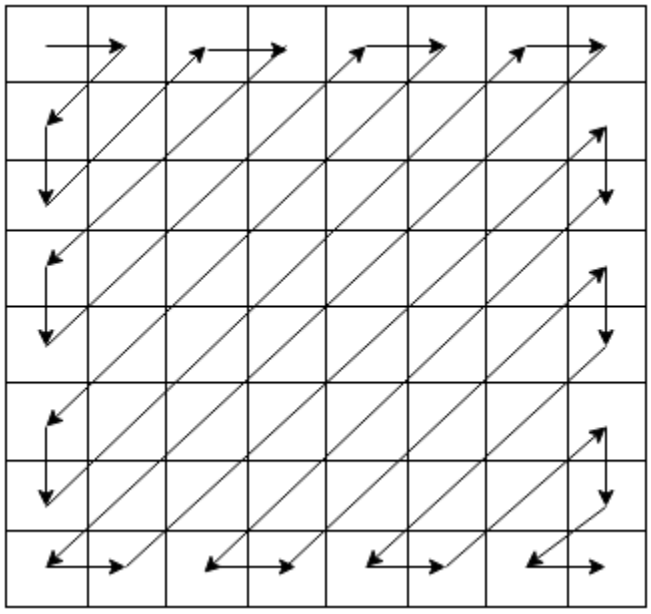
Now we have three matrices containing integer values for every block of 64 pixels, one for luminance and two for chrominance.

3.6. Entropy Encoding

Now all that's left to do is to list the values that are inside each matrix, run RLE (Run Length Encoding) since we have a lot of zeroes, and then run the Huffman Coding algorithm before storing the data.

RLE and Huffman Coding are lossless compression algorithms that reduce the space needed to store information without removing any data.

As we can see, the result of quantization has many 0s toward the bottom left corner. To keep the 0 redundancy, the JPEG algorithm does a zig-zag scanning pattern that keeps all zeroes together:



After zig-zag scanning, we get a list of integer values, and after running RLE on the list, we get each number and how many times they're repeated.

Finally, JPEG runs Huffman coding on the result, and the values that have more frequency (like the zeroes) are represented with fewer bits than less frequent values.

No data is lost during this phase.

4. JPEG Decoding

Whenever we open a .jpg file using an image viewer, the file needs to be decoded before it can be displayed. **To decode a *.jpg* image, the image viewer does all the above steps in reverse order**:

1. Run Huffman Decoding on the file data
2. Deconstruct RLE
3. Put the list of numbers into an 8×8 matrix
4. Multiply the integer values by their corresponding quantization table
5. Multiply the weight values by their corresponding base images, then sum them up to get the pixel values
6. Upscale the chrominance channels
7. Change the color space from YCbCr to RGB
8. Display the image