

## Introduction to conventional compression solutions

Storage constraints and bandwidth limitations in communication systems have necessitated the search for efficient image compression techniques. Uncompressed multimedia (graphics, audio and video) data requires considerable storage capacity and transmission bandwidth. Despite rapid progress in mass-storage density, processor speeds, and digital communication system performance, demand for data storage capacity and data-transmission bandwidth continues to outstrip the capabilities of available technologies. The recent growth of data intensive multimedia-based web applications have not only sustained the need for more efficient ways to encode signals and images but have made compression of such signals central to storage and communication technology.

### Principles behind Compression

A common characteristic of most images is that the neighboring pixels are correlated and therefore contain redundant information. The foremost task then is to find less correlated representation of the image. Two fundamental components of compression are redundancy and irrelevancy reduction. Redundancy reduction aims at removing duplication from the signal source (image/video). Irrelevancy reduction omits parts of the signal that will not be noticed by the signal receiver, namely the Human Visual System (HVS). In general, three types of redundancy can be identified:

- Spatial Redundancy or correlation between neighboring pixel values.
- Spectral Redundancy or correlation between different color planes or spectral bands.
- Temporal Redundancy or correlation between adjacent frames in a sequence of images (in video applications).

Image compression research aims at reducing the number of bits needed to represent an image by removing the spatial and spectral redundancies as much as possible.

### Classes of Compression Techniques

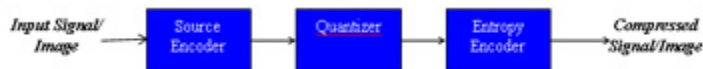
#### Lossless vs. Lossy Compression

There are basically two types of compression methods: lossy and lossless. Lossy compression creates smaller files by discarding some information about the original image. It removes details and color changes it deems too small for the human eye to differentiate. Lossless compression, on the other hand, never discards any information about the original file. In lossless compression schemes, the reconstructed image, after compression, is numerically identical to the original image. An image reconstructed following lossy compression contains degradation relative to the original. Often this is because the compression scheme completely discards redundant information. Under normal viewing conditions, no visible loss is perceived (visually lossless).

## Image Compression Process

### Typical Image Coder

A typical lossy image compression system consists of three closely connected components namely (a) Source Encoder (b) Quantizer, and (c) Entropy Encoder. Compression is accomplished by applying a linear transform to decorrelate the image data, quantizing the resulting transform coefficients, and entropy coding the quantized values.



### Source Encoder (or Linear Transformer)

Over the years, a variety of linear transforms have been developed which include Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT) and many more, each with its own advantages and disadvantages.

### Quantizer

A quantizer simply reduces the number of bits needed to store the transformed coefficients by reducing the precision of those values. Since this is a many-to-one mapping, it is a lossy process and is the main source of compression in an encoder. Quantization can be performed on each individual coefficient, which is known as Scalar Quantization (SQ). Quantization can also be performed on a group of coefficients together, and this is known as Vector Quantization (VQ). Both uniform and non-uniform quantizers can be used depending on the problem at hand.

### Entropy Encoder

An entropy encoder further compresses the quantized values losslessly to give better overall compression. It uses a model to accurately determine the probabilities for each quantized value and produces an appropriate code based on these probabilities so that the resultant output code stream will be smaller than the input stream. The most commonly used entropy encoders are the Huffman encoder and the arithmetic encoder, although for applications requiring fast execution, simple run-length encoding (RLE) has proven very effective.

It is important to note that a properly designed quantizer and entropy encoder are absolutely necessary along with optimum signal transformation to get the best possible compression.

## Further Elaboration on Source Encoder

### Discrete Cosine Transformation (DCT)

The discrete cosine transform (DCT) helps separate the image into parts (or spectral sub-bands) of differing importance (with respect to the image's visual quality). The DCT is similar to the discrete Fourier transform: it transforms a signal or image from the spatial domain to the frequency domain. With an input image, A, the coefficients for the output "image," B, are:

$$B(k_1, k_2) = \sum_{i=0}^{N_1-1} \sum_{j=0}^{N_2-1} A(i, j) \cdot \cos\left[\frac{\pi \cdot k_1}{2 \cdot N_1} (2i + 1)\right] \cdot \cos\left[\frac{\pi \cdot k_2}{2 \cdot N_2} (2j + 1)\right]$$

The input image is  $N_2$  pixels wide by  $N_1$  pixels high;  $A(i, j)$  is the intensity of the pixel in row  $i$  and column  $j$ ;  $B(k_1, k_2)$  is the DCT coefficient in row  $k_1$  and column  $k_2$  of the DCT matrix. All DCT multiplications are real. This lowers the number of required multiplications, as compared to the discrete Fourier transform. The DCT input is an 8 by 8 array of integers. This array contains each pixel's gray scale level; 8 bit pixels have levels from 0 to 255. The output array of DCT coefficients contains integers; these can range from -1024 to 1023. For most images, much of the signal energy lies at low frequencies; these appear in the upper left corner of the DCT. The lower right values represent higher frequencies, and are often small - small enough to be neglected with little visible distortion.

### Wavelets and Image Compression

Wavelets are functions defined over a finite interval and having an average value of zero. The basic idea of the wavelet transform is to represent any arbitrary function  $f(t)$  as a superposition of a set of such wavelets or basis functions. These basis functions or baby wavelets are obtained from a single prototype wavelet called the mother wavelet, by dilations or contractions (scaling) and translations (shifts). The Discrete Wavelet Transform of a finite length signal  $x(n)$  having  $N$  components, for example, is expressed by an  $N \times N$  matrix.

#### *Wavelet-based Compression*

Despite all the advantages of JPEG compression schemes based on DCT namely simplicity, satisfactory performance, and availability of special purpose hardware for implementation; these are not without their shortcomings. Since the input image needs to be "blocked," correlation across the block boundaries is not eliminated. This results in noticeable and annoying "blocking artifacts" particularly at low bit rates. Lapped Orthogonal Transforms (LOT) attempt to solve this problem by using smoothly overlapping blocks. Although blocking effect are reduced in LOT compressed images, increased computational complexity of such algorithms do not justify wide replacement of DCT by LOT.

Over the past several years, the wavelet transform has gained widespread acceptance in signal processing in general and in image compression research in particular. In many applications wavelet-based schemes (also referred as subband coding) outperform other coding schemes like the one based on DCT. Since there is no need to block the input image and its basis functions have variable length, wavelet coding schemes at higher compression avoid blocking artifacts. Wavelet-based coding is more robust under

transmission and decoding errors, and also facilitates progressive transmission of images. Because of their inherent multiresolution nature, wavelet coding schemes are especially suitable for applications where scalability and tolerable degradation are important.

### **JPEG (DCT-Based)**

JPEG is the image compression standard developed by the Joint Photographic Experts Group. It works best on natural images (scenes). The discovery of DCT in 1974 was an important achievement for the research community working on image compression. The DCT can be regarded as a discrete-time version of the Fourier-Cosine series. It is a close relative of DFT, a technique for converting a signal into elementary frequency components. Thus DCT can be computed with a Fast Fourier Transform (FFT) like algorithm in  $O(n \log n)$  operations. Unlike DFT, DCT is real-valued and provides a better approximation of a signal with fewer coefficients. The DCT of a discrete signal  $x(n)$ ,  $n=0,1, \dots, N-1$  is defined as:

In 1992, JPEG established the first international standard for still image compression where the encoders and decoders are DCT-based. The JPEG standard specifies three modes namely sequential, progressive and hierarchical for lossy encoding, and one mode of lossless encoding.

Entropy Coding (EC) achieves additional compression losslessly by encoding the quantized DCT coefficients more compactly based on their statistical characteristics. The JPEG proposal specifies both Huffman coding and arithmetic coding. The baseline sequential codec uses Huffman coding, but codecs with both methods are specified for all modes of operation. Arithmetic coding, though more complex, normally achieves 5-10% better compression than Huffman coding.

### **JPEG2000 (DWT Based)**

JPEG 2000, as noted previously, is the next ISO/ITU-T standard for still image coding. JPEG 2000 is based on the discrete wavelet transform (DWT), scalar quantization, context modeling, arithmetic coding and post compression rate allocation. The DWT is dyadic and can be performed with either a reversible filter, which provides for lossless coding, or a non-reversible one, which provides for higher compression but does not do lossless. The quantizer follows an embedded dead-zone scalar approach and is independent for each sub-band. Each sub-band is divided into blocks, typically 64x64, and entropy coded using context modeling and bit-plane arithmetic coding. The coded data is organized in so called layers, which are quality levels, using the post-compression rate allocation and output to the code stream in packets. The generated code-stream is parseable and can be resolution, layer (i.e. SNR), position or component progressive, or any combination thereof. JPEG 2000 also supports error-resilience, arbitrarily shaped region of interest, random access, multicomponent images, palletized color, compressed domain lossless flipping and simple rotation, to mention a few times of the different algorithms on a Linux workstation. This only gives an appreciation of the involved complexity.

## Features

### Smaller File Sizes

JPEG2000 uses wavelet compression technology to outperform the existing JPEG standard. Wavelets enable lossy or pure lossless compression, while JPEG always imposes some sort of loss. (Pure lossless compression means that there is no loss of data the compressed image is pixel-identical to the original image. Lossy compression discards data that is considered visually less important). For lossy compression at a given quality, wavelet-compressed files are generally three to five times smaller than a JPEG compressed file. For a given file size, a wavelet file offers better quality than a JPEG. For lossless compression, wavelets can cut the file size by at least half its original size, offering a smaller file than LZW or Zip.

### Better Image Quality

Traditional JPEG compression uses the Discrete Cosine Transformation (DCT), which compresses an image in 8x8 blocks and results in visible artifacts at high compression rates. JPEG artifacts include visible seams at the tile edges, dubbed as "blocking artifacts". The wavelet transform encodes an image in a continuous stream allowing it to avoid the artifacts that result from DCT's division of an image into discrete compression blocks. Wavelet artifacts take the form of blurring high contrast lines, merely making the image look softer. The wavelet transform performs what's called, multi-resolution compression—it stores image information in a series of bands, with the most important image information at the beginning of the file. Each band contains a representation of the entire image, with the various bands containing details of the image at every level, from coarse resolution and textures to fine details.

### Progressive Image Downloading

An inherent benefit of the wavelet's multi-resolution architecture, is the ability to progressively access the encoded image in a smooth continuous fashion without having to download, decode and/or print the entire file. Wavelet compressed images appear first as an image with coarse resolution and then finer resolution details are progressively filled in. Since the most important details are stored at the front of the image file, users will first see a blurry version of the image and the remaining details appearing as the bitstream arrives. Usually with about 10% of the image data, the user can tell what the image will be and can decide whether or not to wait for higher resolution. The current JPEG is single-resolution, so with 10% of the data, the user will have barely gotten a peek at the top of the image and has wait for the entire download.

### Entropy Encoding

Lossless compression techniques frequently involve some form of entropy encoding and are based in information theoretic techniques. Entropy encoding is an example of lossless encoding as the decompression process regenerates the data completely. The raw data and the decompressed data are identical, no information is lost. Entropy encoding just manipulates bit streams without regarding what the bits mean. It usually transforms the bit pattern into a different form for transmitting. These methods make use exclusively of the redundancy of the data. There are several of these kinds. Entropy encoding is used regardless of the media's specific characteristics. The data stream to be compressed is considered to be a simple digital sequence, and the semantic of the data is ignored.

## Run Length Encoding (RLE)

Run length coding is an example of entropy encoding. If a byte occurs at least four consecutive times the number of occurrences is counted. The compressed data contains this byte followed by a special flag, called M-byte, and the number of its occurrences. The exclamation mark "!" can be defined as this M-byte. A single occurrence of this exclamation mark is interpreted as M-byte during the decompression; two consecutive exclamation marks are interpreted as an exclamation mark occurring within the data.

This algorithm is very easy to implement and does not require much CPU horsepower. RLE compression is only efficient with files that contain lots of repetitive data. These can be text files if they contain lots of spaces for indenting but line-art images that contain large white or black areas are far more suitable. Computer generated colour images (e.g. architectural drawings) can also give fair compression ratios.

RLE compression can be used in the following file formats:

- TIFF files
- PDF files
- BMP files
- PCX files

## Huffman Encoding

Huffman encoding is an example of entropy encoding. It is based on statistical methods. Given the character that must be encoded, together with the probability of their occurrences, the Huffman encoding algorithm determines the optimal code using the minimum number of bits. Hence the length (number of bits) of the coded characters will differ. In text, the shortest code is assigned to those characters that occur most frequently. To determine a Huffman code, it is useful to construct a binary tree. The nodes of this tree represent the characters that are to be encoded. Every node contains the occurrence probability of one of the characters belonging to this subtree. 0 and 1 are assigned to the edges of the tree. The two characters with the lowest probabilities are combined in the first binary tree. Their root node is labeled with these characters and the combined probability. The edges are labeled with 1 and 0 respectively. (This assignment is arbitrary; therefore, with the same data one can get different Huffman codes). The nodes below this root node won't be considered anymore. Again, the two nodes with the lowest probabilities are combined into a binary subtree, the root node and the edges are labeled as before. Continue in this way until the node with the whole used alphabet and the probability 1 is reached. The encoded data for the characters are the paths through the tree to their nodes. The result is stored in a table. If the information of an image can be transformed into a bit stream, such a table can be used to compress the data without loss.

This compression algorithm is mainly efficient in compressing text or program files. Images like they are often used in prepress are better handled by other compression algorithms.

Huffman compression is mainly used in compression programs like pkZIP, lha, gz, zoo and arj. It is also used within JPEG and MPEG compression.

## Arithmetic Coding

In arithmetic coding, a message is encoded as a real number in an interval from one to zero. Arithmetic coding typically has a better compression ratio than Huffman coding, as it produces a single symbol rather than several separate code words. Arithmetic coding is a lossless coding technique. There are a few disadvantages of arithmetic coding. One is that the whole codeword must be received to start decoding the symbols, and if there is a corrupt bit in the codeword, the entire message could become corrupt. Another is that there is a limit to the precision of the number which can be encoded, thus limiting the number of symbols to encode within a codeword. There also exist many patents upon arithmetic coding, so the use of some of the algorithms also call upon royalty fees.

Arithmetic coding is used within JPEG2000 compression.

## Lempel-Ziv-Welch Algorithm (LZW)

Lempel-Ziv-Welch is a sophisticated lossless compression method which analyzes the data and looks for repeating patterns. If LZW sees "010101" (for example), it is a clever-enough algorithm to spot the trend of alternating characters and replace each instance with a single character, thereby compressing the information. LZW compression replaces strings of characters with single codes. It does not do any analysis of the incoming text. Instead, it just adds every new string of characters it sees to a table of strings. Compression occurs when a single code is output instead of a string of characters. The code that the LZW algorithm outputs can be of any arbitrary length, but it must have more bits in it than a single character. The first 256 codes (when using eight bit characters) are by default assigned to the standard character set. The remaining codes are assigned to strings as the algorithm proceeds.

LZW compression works best for files containing lots of repetitive data. This is often the case with text and monochrome images. Files that are compressed but that do not contain any repetitive information at all can even grow bigger. LZW compression is fast. Royalties have to be paid to use LZW compression algorithms within applications (see below).

LZW compression can be used in a variety of file formats:

- TIFF files
- GIF files

## Burrows-Wheeler Transformation (BWT)

The BWT is an algorithm that takes a block of data and rearranges it using a sorting algorithm. The resulting output block contains exactly the same data elements that it started with, differing only in their ordering. The transformation is reversible, meaning the original ordering of the data elements can be restored with no loss of fidelity.

The BWT is performed on an entire block of data at once. Most of today's familiar lossless compression algorithms operate in streaming mode, reading a single byte or a few bytes at a time. But with this new transform, there is a need to operate on the largest chunks of data possible. Since the BWT operates on data in memory, there would be files too big to process in one fell swoop. In these cases, the file must be split up and processed a block at a time.

BWT can be used in a variety of file formats

- bzip2
- zip2