The still image lossy compression standard - JPEG

Yair Wiseman

Computer Science Department

Bar-Ilan University

Ramat-Gan 52900

Israel

wiseman@cs.biu.ac.il

I. Introduction

Reducing image files will be an important procedure when we transmit files across networks (Wiseman, Schwan & Widener, 2004). or when we would like to archive libraries. Usually, JPEG can remove the less important data before the compression; hence JPEG will be able to compress images meaningfully, which produces a huge difference in the transmission time and the disk space. The processing time of the compression can be overlapped with the disk access time (Wiseman Y. & Feitelson, 2003).

Another advantage of JPEG is the capability of storing full color information: 24 bits/pixel or 16 million colors, while for example the GIF format, can store only 8 bits/pixel or 256 colors.

One more advantage of JPEG is automatic error recovery. The default compression algorithm of JPEG is based on Huffman codes (Huffman, 1952). Utilizing the tendency of Huffman codes to resynchronize

quickly, i.e. recovering after potential decoding errors in most cases, JPEG also recovers quickly after possible errors as is explained here in below.

II. BACKGROUND

"JPEG" stands for Joint Photographic Experts Group. JPEG is a well known standardized image compression technique for compressing pictures which do not have sharp changes e.g. landscape pictures. JPEG supports either color or grayscale images.

JPEG loses information, so the decompressed picture is not the same as the original one. By adjusting the compression parameters, the degree of loss can be adjusted. The wide use of JPEG is because of three fundamental reasons: reducing the size of image files, storing full color information, automatic error recovery.

The JPEG committee was established in 1986 by the CCITT and ISO Standards organizations with the aim of creating universal standards for image compression. The committee finalized the standard by early 1991 and latter the standard was approved as an International Standards Organization (ISO). Initially, JPEG targeted achieving a 15:1 average compression ratio; however, currently JPEG achieves even better compression ratios. Some real-time, full-motion and video applications also used JPEG. In the mid of the last decade of the 20^{th} century, a new standard called Moving or Motion-JPEG based on the JPEG algorithm was established. This standard aims at compression of video sequences and it stores JPEG format images in a sequential order.

III. OVERVIEW OF THE JPEG ALGORITHM:

JPEG compression algorithm consists of several steps. The steps are summarized in Figure 1. In this section the steps will be comprehensively explained.

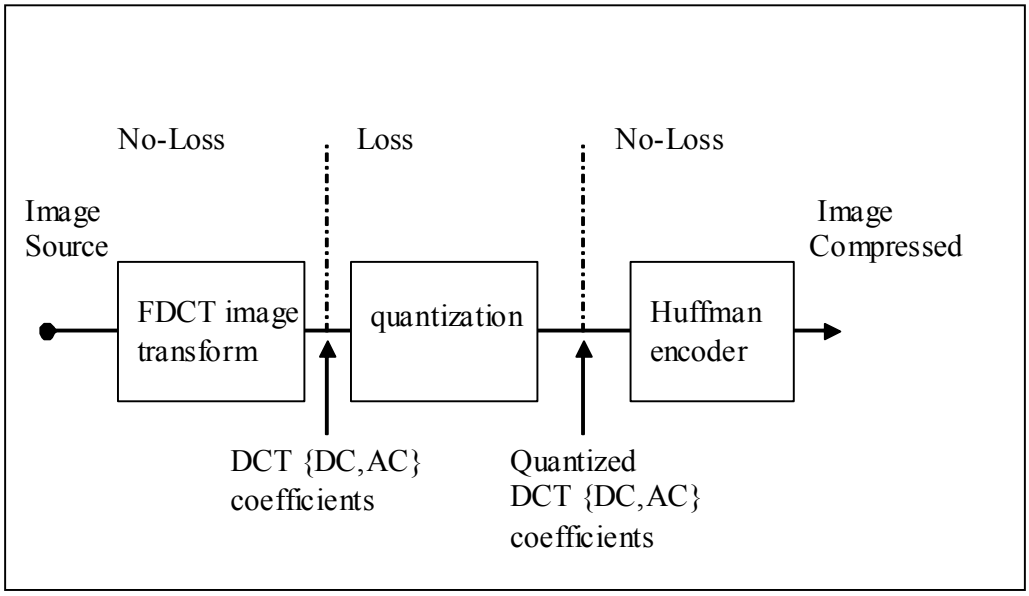


Figure 1: JPEG Model for a lossy image compression

The first step transforms the image color into a suitable color space. There are several methods to transform the image into a color space (Hearn & Baker, 1986), (Jain, 1986). The most common methods are the split into YUV components (Hunt, 1995) or the split into RGB components (Laplante & Stoyenko, 1996). These components are interleaved together within the compressed data. The ratio between these components is usually not one to one. When YUV components are used, usually the Y component will have a four times weight. The human eye is less sensitive to the frequency of chrominance information than to the frequency of luminance information which is represented by the Y component in the YUV format. Hence, the Y component gets a higher weight (Awcock, 1996).

JPEG employs Chroma subsampling which is a technique of encoding images by using less resolution for chrominance information than for luminance information, taking advantage of the human eye's lower sensitiveness for color differences than for luminance differences. JPEG supports the obvious 4:1:1 chroma subsampling which denotes the color resolution is quartered, compared to the luminance

information i.e. for each sampled element as in figure 2, there is 4 numbers for luminance and just one number for chrominance; however the default chroma subsampling of JPEG is 4:2:0 which denotes the horizontal sampling is doubled compared to 4:1:1, but as the U and V components are only sampled on each alternate line.

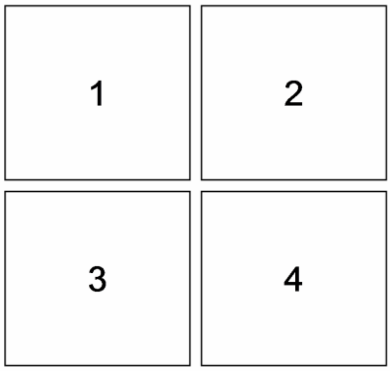


Figure 2: One JPEG sampled element

JPEG allows samples of 8 bits or 12 bits. In the original JPEG all values contained by the same source image must have the same precision. The values are shifted from unsigned integers with range $[0,2^{p}-1]$ to signed integers with range $[-2^{p-1},2^{p-1}-1]$, by reducing 2^{p-1} from the original values, where p can be either 8 or 12. These biased values are then sent to the next step.

The second step groups the pixels into blocks of 8X8 pixels. The order of the blocks is line by line and each line is read from left to right. After the group into blocks, JPEG transforms each block through a Forward Discrete Cosine Transform (FDCT) (Rao & Yip, 1990). The DCT gives a frequency map, with 8X8 or 64 elements. The transformation keeps the low frequency information which a human eye is sensitive to. In each block the DCT coefficients are composed of: A single Direct Current (DC) coefficient number, which represents the average intensity level value in each block and the remaining 63 are named Alternating Current (AC) coefficients. They reflect the frequency information of their row

and column. These coefficients indicate the amplitude of a specific frequency component of the input array. The frequency content of the sample set at each frequency is calculated by taking a weighted sum of the entire set.

Table 1 contains the values of weight for one row in a 8x8 matrix:

Index of value / Index of result	0	1	2	3	4	5	6	7
0	+0.707	+0.707	+0.707	+0.707	+0.707	+0.707	+0.707	+0.707
1	+0.981	+0.831	+0.556	+0.195	-0.195	-0.556	-0.831	-0.981
2	+0.924	+0.383	-0.383	-0.924	-0.924	-0.383	+0.383	+0.924
3	+0.831	-0.195	-0.981	-0.556	+0.556	+0.981	+0.195	-0.831
4	+0.707	-0.707	-0.707	+0.707	+0.707	-0.707	-0.707	+0.707
5	+0.556	-0.981	+0.195	+0.831	-0.831	-0.195	+0.981	-0.556
6	+0.383	-0.924	+0.924	-0.383	-0.383	+0.924	-0.924	+0.383
7	+0.195	-0.556	+0.831	-0.981	+0.981	-0.831	+0.556	-0.195

Table 1: Values of weight for one row in a 8x8 matrix:

In other words, these AC values of one row are the results of this formula:

$$F(u) = \frac{1}{2} \sum_{x=0}^{7} f(x) C(u) \cos[(2x+1)u\pi/16]$$

Where:

- x is the index of value.
- f(x) is the value itself.
- u is the index of result.
- $C(u) = 1/\sqrt{2}$ if u=0; $C(u)$=1 otherwise.

- F(u) is the result itself.

In reality, the computer, the digital camera or any other device that generate JPEG images do not compute the cosine values over and over again. It would be an unnecessary waste of CPU cycles, so they use this formula:

$$F(u) = \frac{1}{2}\sum_{x=0}^{7} f(x)T(u,x)$$

Where T is the matrix described in Table 1.

If f(x) (the intensity of each pixel) is equal in the entire row, each F(u) which holds u>0, will be zero. F(0) will be the sum of the row's values divided by $2\sqrt{2}$. This explains why JPEG is better pictures which do not have sharp changes. Pictures which have sharp changes like cartoons will have larger values and hence will be compressed not as good as the pictures without the sharp changes.

As was mention above, the blocks are of 8X8, so one-dimensional DCT is enough for only one row. The DCT for an entire block is done by applying one-dimensional DCT separately for each row of eight pixels. The result will be eight rows of frequency coefficients. Then, these 64 coefficients are taken as eight columns. The first column will contain all DC coefficients; the second column will contain the first AC coefficient from each row, and so on. At that time, JPEG apply one-dimensional DCT for each of these columns.

In reality the one-dimensional DCT is not used and the entire calculation is done by one formula that reflects this calculation:

$$F(u,v) = \frac{1}{4}\sum_{y=0}^{7}\sum_{x=0}^{7} f(x,y)C(u)\cos[\frac{(2x+1)u\pi}{16}]C(v)\cos[\frac{(2y+1)v\pi}{16}]$$

Where:

- x,y are the indices of value.
- f(x,y) is the value itself.
- u,v are the indices of result.
- $C(u) = 1/\sqrt{2}$ if u=0; $C(u)$=1 otherwise. $C(v) = 1/\sqrt{2}$ if v=0; $C(v)$=1 otherwise.
- F(u,v) is the result itself.

It should be noticed that index 0,0 contains the DC of the DCs. This value is called the DC of the 8x8 block.

The next step is the quantization. The 63 AC coefficients are ordered into a zig-zag sequence. Firstly JPEG puts the coefficient that its row index plus column index is 0 i.e. the DC. Then JPEG puts the coefficients that their row index plus column index is 1. Next, JPEG puts the coefficients that their row index plus column index is 2 and JPEG continues until putting the last coefficient that its row index plus column index is 14. This zig-zag algorithm arranges the coefficients into one dimensional array as described in Figure 3.

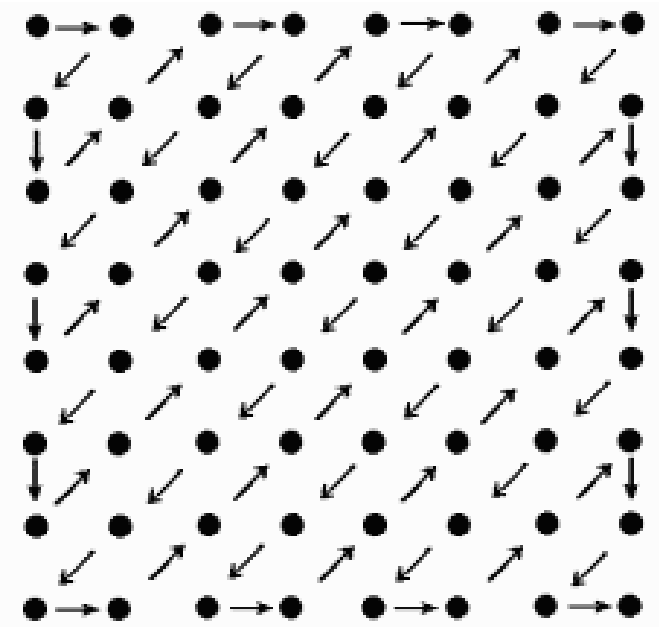


Figure 3. Zig-Zag order of an 8x8 block.

In each block, each of the 64 coefficients is divided by a separate “quantization coefficient”. The quantization coefficients are set according to the desired image quality. In point of fact, the image creator decides about a quality level (which is a number amongst 1 to 100) and according to this number the quantization coefficients are set. There is no standard method how to translate this quality level number into the quantization coefficients, so the quantization coefficients themselves are stored in the compressed image.

The results of the division are rounded to integers. This step loses some information because of the rounding. Furthermore, it can be noted that even if the quantization coefficient is 1, some information will be lost, because typically the DCT coefficients are real numbers.

The last step uses a traditional compression method with the aim of reducing the size of the data. The dictionary methods like the versions of Lempel-Ziv methods (Wiseman, 2007b) are more suitable for

text compression because it looks for an exact reappearance of strings which is less likely to occur in pictures, so JPEG encodes the reduced coefficients using either Huffman or Arithmetic coding. There are also other compression algorithms that fit JPEG, but are not officially supported (Wiseman, 2007a). Usually a strong correlation appears between DC coefficients of adjacent 8X8 blocks. Therefore, JPEG encodes the difference between each pair of adjacent DC coefficient. Baseline JPEG model uses two different Huffman trees to encode any given image, one Huffman tree for the DC coefficients' length and the other Huffman tree for the AC coefficients' length. The tree for the DC values encodes the lengths in bits (1 to 11) of the binary representations of the values in the DC fields.

The second tree encodes information about the sequence of AC coefficients. As many of them are zero, and most of the non-zero values are often concentrated in the upper left part of the 8 × 8 block, the AC coefficients are scanned in the zig-zag order specified by the quantization step i.e. processing elements on a diagonal close to the upper left corner before those on such diagonals further away from that corner, that is the order is given by (0, 1), (1, 0), (2, 0), (1, 1), (0, 2), (0, 3), (1, 2), etc. The second Huffman tree encodes pairs of the form (n, l), where n (limited to 0 to 15) is the number of elements that are zero, preceding a non-zero element in the given order and l is the length in bits (1 to 10) of the binary representation of the non-zero quantized AC value. The second tree also includes codewords for EOB, which is used when no non-zero elements are left in the scanning order and for a sequence of 16 consecutive zeros in the AC sequence (ZRL), necessary to encode zero-runs that are longer than 15. The Huffman trees used in baseline JPEG are well-known and fixed. The trees can be found in (ISO/IEC, 1993). There is also an option to use a specific tree; however this option is only on the odd occasion used.

Each 8X8 block is then encoded by an alternating sequence of Huffman codewords and binary

integers (except that the codeword for ZRL is not followed by any integer), the first codeword belonging to the first tree and relating to the DC value, the other codewords encoding the (n, l) pairs for the AC values, with the last codeword in each block representing EOB.

20	**1**	0	0	0	0	0	0
0	**3**	0	0	0	0	0	0
0	0	0	0	0	0	0	0
-2	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Table 2: Example of block compression

Table 2 brings an example of a JPEG 8X8 block after DCT and after quantization, with the DC value is in the upper left corner. The DC value is more often than not very similar to the previous DC value, so it is computed as a difference between the current value and the previous DC value. Suppose the previous DC value in this example was 15, the result of the compression will be:

```
100 101 00  1 11111001 11 1111111000 01 1010
 ↓   ↓  ↓  ↓    ↓      ↓      ↓       ↓   ↓
 3   5 0,1 1   2,2     3      4,2     -2 EOB
```

The first value (110) is the Huffman code for the DC length in bits (3). Then, the difference between

the current DC value and the previous DC value is explicitly written (5). Next, the AC values are written as sequences of several zeros followed by one non-zero value i.e. 0,1 is no zero and then a number with one bit. After this pair explicitly appears the value of the one bit number (1). 2,2 is 2 zeros and then a number with two bits. After this pair explicitly appears the value of the two bits number (3). 4,2 is 4 zeros and then a number with two bits. After this pair explicitly appears the value of the two bits number (-2). At the end a special code EOB (1010) is written. It should be noted that negative numbers are written in JPEG in one's complement representation.

Finally, the compression parameters are written in the file header, so that the decoder module will be able to decode the compressed image.

The decompression process performs an inverse procedure:

- It decompresses the Huffman or the Arithmetic codes.

- Then, it makes the inversion of the Quantization step. In this stage, the decoder raises the small numbers by a multiplication of them by the quantization coefficients. The results are not accurate, but they are close to the original numbers of the DCT coefficients.

- Finally, an Inverse Discrete Cosine Transform (IDCT) is performed on the data received from the previous step.

JPEG has some disadvantages. Unfortunately, even with a JPEG viewer, it takes a longer time to decode and view a JPEG image than to view an image of a simpler format such as GIF, BMP, etc.;

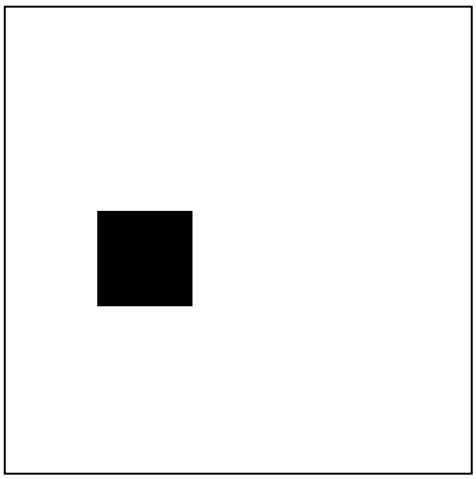
however it is still a very short time as can be indicated in every digital camera. Another disadvantage is the compression method that does not take into account the temporal correlation of the coefficients.

IV. Features of Jpeg

As mentioned above the JPEG standard is based on the DCT paradigm. The DCT changes the picture into frequency space. The frequency coefficients, which are very low magnitude, are rounded to zero. When most of the coefficients in a block are zero or very low magnitude: The compression algorithm will give a very short bits sequence for such a block. Zero sequences are treated very efficiently by JPEG compression and the results will be only few bytes.

When there is a drastic change in a block of 8X8, the value of many frequency coefficients will be high. Such a sequence will be compressed into many more bits.

Original image



JPEG format of uniform area

0	0	1	0	1	0

JPEG format of the upper left corner of the black square

-12	18	-8	-2	3	-1	-1	1
22	-25	8	2	-3	0	1	-1
-18	20	-7	-2	2	0	-1	1
16	14	4	1	-1	0	1	1
-11	10	-2	-1	0	-1	-1	0
6	-5	1	-3	0	0	0	0
-2	2	-1	0	0	0	0	0
1	-1	0	0	0	0	0	0

Figure 4. A sample image and how JPEG can be used for contouring

Figure 4 demonstrates how JPEG is used for contour extraction. The original image was compressed in grayscale baseline JPEG format with 75% quality. The left image shows the original image, which is a high-resolution picture of 1000X1000 pixels. The upper-right image shows the JPEG format in the white or black area. The lower-right image shows the JPEG format in the upper left corner of the black square. The size of the black square is 200X200 pixels and the square is not aligned relative to JPEG's 8X8 blocks.

The JPEG file reports the difference of magnitude between the DC's coefficients of a previous block

relative to the current block. In the case of a white or black area there are no changes in the coefficients' magnitudes. This type of block is encoded as six bits by the JPEG standard.

The output of the JPEG file is shown in Figure 4. The "00" reflects that there are no differences between the values of the previous and the current block's DC coefficients, and "1010" symbolizes the end of the block. If there is a difference between the intensity of the DC coefficients of the previous and the current blocks, the size of the encoding block will be slightly bigger. For example, a block which encodes a sharp change from white to black is represented by a wide range of frequency coefficients.

Figure 4 shows a sample of the block, which contains the upper left corner of the black square. In order to compress these values in JPEG standard, 243 bits are needed. The difference between 6 to 243 is obviously significant. By using three parameters the length of the block, its magnitude and the number of consecutive blocks the threshold can extract the contour with a range of scalar values (Wiseman & Fredj, 2001), (Fredj & Wiseman, 2001). The extra parameters allow more control over the resulting mechanism. Such an extraction can be useful for many applications e.g. vehicular objective (Wiseman, 2010), (Wiseman, 2013a), (Wiseman, 2013b), (Wiseman, 2013c), (Wiseman, 2014).

V. Error Recovery

JPEG's default compression algorithm is based on Huffman coding scheme which has a tendency to resynchronize immediately. JPEG also recovers after decoding errors in the usual course of events Klein & Wiseman, 2000), (Klein & Wiseman, 2003); however there can be some incorrectness in the decoded picture; although the picture will be generally almost the same.

Any Huffman code is complete, in the sense that any binary string can be 'decoded' as if it were some

Huffman encoding, so that errors in the binary file will stay undetected unless the end of the file is reached within a codeword. In the particular case of JPEG, errors may be detected in certain circumstances: keeping track of the number of AC elements by summing the n fields of the (n, l) pairs and adding the number of non-zero coefficients, if this number exceeds 63, there must obviously have been an error. In addition, the particular Huffman trees used are not complete and, in fact, certain codewords are missing (for example 111111111 in the first Huffman tree, used for the DC coefficients). The appearance of such an invalid codeword is therefore a sign that some error has occurred. The error however will be almost certainly fixed when the decoder will continue to decode more blocks.

.

Another problem can be with the block positioning within the picture. As the compressed data has variable length, the location of each block in the image cannot be known accurately, if an error has been occurred. The wrong positioning of a decoded block may cause the cutting of a line of blocks into two parts: the left part may appear at the right end of a line of blocks in the image, whereas the right part will be at the left border of the following line. For many images this will result in a discontinuity which is often easily detectable by a human eye, as for example in the lower part of Figure 5.

Figure 5. The right picture is the original one and the left picture has a decoding error.

As mentioned, the DC values are not encoded themselves, but rather as the difference between the current value and that of the previous block. When the decoding algorithm does not know the exact previous DC, the exact DC for the current blocks is not known as well. A wrong DC may result in a biased image, which can be too bright or too dark for grayscale pictures, if the change was in the luminance component; a change in the chrominance component of color pictures may turn the image too reddish or bluish. This is still better than not seeing this part of the image at all. The left image in Figure 3 is too bright due to incorrect DC values.

Note that the new standard JPEG–2000 has built-in synchronization codewords which make the synchronization faster.

VI. JPEG-2000

JPEG-2000 is a new standard for image and video compression (Taubman & Marcellin, 2002), (Christopoulos, Skodras & Ebrahimi, 2000), (Skodras, Christopoulos & Ebrahimi, 2001a), (Skodras, Christopoulos & Ebrahimi, 2001b), (Usevitch, 2001). Several latest digital cameras support JPEG-2000 and Motion JPEG-2000 e. g. (Yamauchi, Okada, Taketa & Ohyama, 2004), (Fang, Huang, Chang, Wang, Tseng, Lian & Chen, 2004), (Liu, Chen, Meng, Zhang, Wang & Chen, 2004), (Seo & Kim, 2007). Motion JPEG is employed in numerous applications from professional video editing to capture cards in many hardware settings.

By means of JPEG-2000, an image stored on a disk has lower quality levels embedded in the compressed file. If a user requests to preview a JPEG-2000 image, only a lower quality image will be restored. No modification of compressed data is needed. If the user decides to see a better quality image, further information will be restored from the JPEG-2000 image file. There will be no need to redownload the entire image. The overall quality of a JPEG-2000 image is typically higher than a

traditional JPEG image. In addition to the advantages of improved quality and scalability, JPEG-2000 also generates about 20% better compression ratio than JPEG and at several cases JPEG-2000 even produces a considerably better compression ratio.

Unlike MPEG, by means of Motion JPEG-2000 each frame is encoded independently. MPEG employs three kinds of frames for encoding:

- I−frames do not need other video frames to be decoded, but the compression ratio is not as good as the other frame types.
- P−frames employ data of previous frames to be decoded and the compression ratio is typically better than I−frames.
- B−frames employ data of both previous and forward frames for the decoding and typically obtain the best compression ratio.

The three types of frames allow improved compression efficiency but the coding technique is more complex and therefore requires more computation time. Motion JPEG-2000 employs only Intra-frame blocks which let the user the ability to randomly access a file at any fame. In addition, the using of only Intra-frame blocks reduces the complexity of the compression and decompression processes and therefore the decoding time is faster.

There is another major modification in JPEG-2000. The designers of JPEG-2000 have came to a decision to depart form the block based DCT coding used by the traditional JPEG and MPEG standards in favor of a wavelet based compression - the discrete wavelet transform (DWT). The DWT provides improved quality than JPEG and also provides scalability without having to store surplus data.

The forward DWT at the encoder is successive functions of a pair of low-pass and high-pass filters,

followed by division by a factor of two after each filtering function so that odd indexed samples will be discarded. The low-pass and high-pass filter pair is known as analysis filter-bank. The low-pass filter keeps the low frequencies whereas the high frequencies are significantly diminished and therefore the result is only a vague form of the original signal. Conversely, the high-pass filter keeps only the high frequencies of a signal such as contours, whereas the low frequencies are significantly diminished.

Like Original JPEG, JPEG-2000 also employs Quantization. The output of the filters is divided by predefined numbers with the aim of reducing them at the expense of quality. The predefined numbers can be larger for a better compression or smaller for better quality. If all the predefined numbers are equal 1 no quantization is actually performed and it is in point of fact a lossless compression.

Lossless compress also stipulates some other characterizations for the compression process: In any image, the code blocks of the image are put in a single sub-band and except the code blocks at the edges of the image. The bits of all quantized coefficients of a code block are encoded from the most significant bits to the less significant bits by a process called EBCOT stands for "Embedded Block Coding with Optimal Truncation" In EBCOT, there are three coding passes:

1. Significance Propagation - Encoding bits (and signs) of insignificant coefficients with significant neighbors.
2. Magnitude Refinement - Refinement bits of significant coefficients.
3. Cleanup pass - Handling coefficients without significant neighbors.

When loosy mode is selected, insignificant bits can be dropped, depends of the chosen quality; however, if lossless mode is selected, EBCOT will encode the entire data and no data can be left out.

Actually, the generated data from the execution of JPEG-2000 is effortlessly scalable i.e. by

truncating the output at any point; a representation of the same image at a lower quality will be created. A larger truncation will unsurprisingly produce a poorer image. Such a truncation occurs when there is an attempt to store the image in a location too short to hold its entire extent. Image truncation may occur automatically such as when a large image is written to a smaller buffer, or deliberately when only a portion of the image is wanted.

Unlike original JPEG which uses an 8×8 block-size discrete cosine transform as was mention above, JPEG 2000 employs two different wavelet transforms – an irreversible transform that depends on the precision of the decoder and set a quantization noise according to this precision and a reversible transform that makes use of only integer coefficients. Clearly integer numbers do not need rounding; therefore this transform does not use any quantization noise and thus this transform is used for lossless coding.

VII. Further Research Directions

Large data centers such as Google, Yahoo, Facebook, Microsoft, Amazon and Apple that put in storage several billions of photos would be happy to reduce their storage costs. Lately, JPEGmini (Gennuth, 2011) has been disclosed. JPEGmini may perhaps be the next generation of image compression tool. The algorithms of JPEGmini are based on the human eye. JPEGmini analyzes the image and decides what the highest level of compression that can be applied is. Obviously, this compression should be done without creating any visual imperfections. As a result, JPEGmini compresses each image to the maximum extent possible without damaging the perceived quality of the image.

Other potential users of JPEGmini are Content Delivery Networks (CDNs) that can save bandwidth and costs for themselves and their customers using JPEGmini. In addition, websites with many pictures

can improve the user experience, because of the reduced image size resulting in reduced web-page load time. This advantage is also significant to developers of flash applications and games, who can reduce the size of images in their applications, and therefore ease the traffic on the network lines and the time it takes to download the application to the browser.

The new JPEGmini is compatible with traditional JPEG although it is capable to reduce the image size by up to 5 times without any visible loss of image quality.

VIII. CONCLUDING REMARKS

JPEG is most excellent for images of realistic pictures having smooth changes of color and tune. JPEG can significantly reduce the size of such images while still keep the quality of the images. JPEG is also very tolerant for errors in the image file – an image with errors still can be shown with minor inaccuracy. In the web where the size of data used for an image is of great consequence, JPEG is widely used. In addition, JPEG is the most widespread format supported by digital cameras.

IX. REFERENCES

Awcock G. J. (1996). Applied Image Processing McGraw-Hill Book Company, pp. 260-261.

Christopoulos C. A., Skodras A. N., & Ebrahimi, T. (2000). The JPEG 2000 still image coding system: An overview, IEEE Trans. Consumer Electron., vol. 46, pp. 1103-1127.

Fang, H. C., Huang, C. T., Chang, Y. W., Wang, T. C., Tseng, P. C., Lian, C. J. & Chen, L. G. (2004). 81 MS/s JPEG 2000 single-chip encoder with rate-distortion optimization, in Dig. Tech. Papers IEEE Int. Solid-State Circuits Conf., p. 328.

Fredj E. & Wiseman Y. (2001). An O(n) Algorithm for Edge Detection in Photos Compressed by JPEG Format, Proc. IASTED International Conference on Signal and Image Processing SIP-2001, Honolulu, Hawaii, pp. 304-308.

Gennuth I., (2011), JPEGmini – the Future of Image Compression?, megapixel, article no. 21907, http://www.megapixel.co.il/english/archive/21907.

Hearn D. & Baker M. P. (1986), Computer Graphic Prentice Hall, Englewood Cliffs, NJ, pp. 295-307.

Huffman, D. (1952). A method for the construction of minimum redundancy codes. In Proc. IRE, Kansas City, 9 September 1952, vol. 40, pp. 1098–1101. The Institute of Electronics and Radio Engineers, London.

Hunt R. W. G. (1995).The Reproduction of Colour Fountain Press England, pp. 507-511.

ISO/IEC 10918-1 (1993). Information technology—digital compression and coding of continuous–tone still images requirements and guidelines. International Standard ISO/IEC, Geneva, Switzerland.

Jain A. K. (1986). Fundamental of Digital Image Processing Prentice Hall Information and Sytem Sciences Series, pp. 553-557.

Klein S. T. & Wiseman Y. (2000). Parallel Huffman Decoding, Proc. Data Compression Conference DCC-2000, Snowbird, Utah, USA, pp. 383-392.

Klein S. T. & Wiseman Y. (2003). Parallel Huffman Decoding with Applications to JPEG Files, The Computer Journal, Oxford University Press, Swindon, UK, Vol. 46(5), pp. 487-497.

Laplante P. A. & Stoyenko A. D. (1996). Real Time Imaging, Theory, Techniques and Applications IEEE Press Inc. NY, pp. 122-124.

Liu, L., Chen N., Meng, H., Zhang, L., Wang, Z. & Chen, H. (2004). A VLSI architecture of JPEG2000 encoder, IEEE J. Solid-State Circuits, vol. 39, pp. 2032–2040.

Rao K. R. & Yip P. (1990). Discrete Cosine Transform Algorithms, Advantages, Applications Academic Press Inc., London.

Seo, Y. H. & Kim, D. W. (2007). VLSI Architecture of Line-Based Lifting Wavelet Transform for Motion JPEG2000. IEEE Journal of Solid-State Circuits, vol. 42, pp.431-440.

Skodras, A. N., Christopoulos, C. & Ebrahimi, T. (2001). JPEG 2000: The upcoming still image compression standard, Pattern Recognition Lett., vol. 22, pp. 1337-1345.

Skodras, A., Christopoulos, C. & Ebrahimi, T. (2001). The JPEG2000 still image compression standard, IEEE Signal Processing Mag., vol. 18, pp. 36-58.

Taubman D. S. & Marcellin M. W. (2002). JPEG 2000: Image Compression Fundamentals, Standards and Practice.. Kluwer International Series in Engineering and Computer Science.

Usevitch, B. E. (2001). A tutorial on modern lossy wavelet image compression: Foundations of JPEG 2000, IEEE Signal Processing Mag., vol. 18, pp. 22-35.

Wiseman Y. & Feitelson D. G. (2003). Paired Gang Scheduling, IEEE Transactions on Parallel and Distributed Systems, Vol. 14(6), pp. 581-592.

Wiseman Y. & Fredj E. (2001). Contour Extraction of Compressed JPEG Images, ACM - Journal of Graphic Tools, Vol. 6 No. 3, pp. 37-43.

Wiseman Y. (2007). Burrows-Wheeler Based JPEG, Data Science Journal, Vol. 6, pp. 19-27.

Wiseman Y. (2007). The Relative Efficiency of LZW and LZSS, Data Science Journal, Vol. 6, pp. 1-6.

Wiseman Y. (2010). Take a Picture of Your Tire!, Proc. IEEE Conference on Vehicular Electronics and Safety (IEEE ICVES-2010) Qingdao, ShanDong, China, pp. 151-156.

Wiseman Y. (2013). Camera That Takes Pictures of Aircraft and Ground Vehicle Tires Can Save Lives, Journal of Electronic Imaging, Vol. 22(4), paper no. 041104.

Wiseman Y. (2013). Fuselage Damage Locator System, Advanced Science and Technology Letters, Vol. 37, pp. 1-4, Jeju Island, Korea.

Wiseman Y. (2013). The Effectiveness of JPEG Images Produced By a Standard Digital Camera to Detect Damaged Tyres, World Review of Intermodal Transportation Research, Vol. 4(1), pp. 23-36.

Wiseman Y. (2014). Device for Detection of Fuselage Defective Parts, Information Journal, Tokyo, Japan.

Wiseman Y., Schwan K. & Widener P. (2004). Efficient End to End Data Exchange Using Configurable Compression, Proc. The 24th IEEE Conference on Distributed Computing Systems (ICDCS 2004), Tokyo, Japan, pp. 228-235.

Yamauchi, H., Okada, S., Taketa, K. & Ohyama, T. (2004). A single chip JPEG2000 encode procesor capable of compressing D1-images at 30 frame/sec without tile division, IEICE Trans. Elec., vol. E87-C, no. 4, pp. 448-456.

Additional Readings

Chao, J., Chen, H., & Steinbach, E. (2013, September). On the design of a novel JPEG quantization table for improved feature detection performance. In Proc. IEEE International Conference on Image Processing, Melbourne, Australia.

Huang, F., Huang, J., & Shi, Y. Q. (2010). Detecting double JPEG compression with the same quantization matrix. IEEE Transactions on Information Forensics and Security, Vol. 5(4), pp. 848-856.

Liang, Z. G., Du, X. Y., Liu, J. B., & Li, K. C. (2012). Effects of JPEG 2000 Compression Techniques on Diagnostic Accuracies of Breast Microcalcification. Chinese Medical Equipment Journal, Vol. 1, article 026.

Luo, W., Huang, J., & Qiu, G. (2010). JPEG error analysis and its applications to digital image forensics. IEEE Transactions on Information Forensics and Security, Vol. 5(3), pp. 480-491.

Stamm, M. C., Tjoa, S. K., Lin, W. S., & Liu, K. R. (2010, March). Anti-forensics of JPEG compression. In (ICASSP), 2010 IEEE International Conference on Acoustics Speech and Signal Processing, pp. 1694-1697.

Valenzise, G., Nobile, V., Tagliasacchi, M., & Tubaro, S. (2011, September). Countering JPEG anti-forensics. In 2011 18th IEEE International Conference on Image Processing (ICIP), pp. 1949-1952.

Yeung, C., & Lin, S. (2012). Method and system for frame rotation within a JPEG compressed pipeline, U.S. Patent No. 8,098,959. Washington, DC: U.S. Patent and Trademark Office.

Key Terms & Definitions:

Compression: Encoding data by fewer bits than the original representation.

Lossy Encoding: Some information is lost in encoding process; expectantly insignificant information.

Chroma subsampling: Encoding images with less information for chrominance components than for luminance components.

YUV: A color representation of an image where Y stands for the brightness components whereas U and V stand for the color components.

Error recovery: Ability to salvage the remaining data after a read error in the decoding process.