Paper from Proceeding of the National Conference on Innovative Developments in Science, Technology & Management (NCIDSTM-2015) Organized by Ganga Technical Campus, Soldha, Bahadurgarh, Haryana (India) March 1st 2015 Published by International Journal of Engineering Sciences Paradigms and Researches (IJESPR) with ISSN (Online): 2319-6564, Impact Factor: 2.20 and Website: www.ijesonline.com

Basic Image Compression Algorithm and Introduction to JPEG Standard

Nidhi Sharma[1], Dr. Anurag Mishra[2]
[1]Assistant Professor, (ECE Deptt.)
Delhi Technical Campus
Delhi, India

[2]Associate Professor
Delhi University
Delhi, India
[1]nidhi22sharma@gmail.com

Abstract

Because of the explosively increasing information of image and video in various storage devices and Internet, the image and video compression technique becomes more and more important. This paper introduces the basic concept of data compression which is applied to modern image and video compression techniques such as JPEG, MPEG, MPEG-4 and so on. The basic idea of data compression is to reduce the data correlation. By applying Discrete Cosine Transform (DCT), the data in time (spatial) domain can be transformed into frequency domain. Because of the less sensitivity of human vision in higher frequency, we can compress the image or video data by suppressing its high frequency components but do no change to our eye. Moving pictures such as video are data in three-dimensional space consists of spatial plane and time axis. Therefore, in addition to reducing spatial correlation, we need to reduce the time correlation. We introduce a method called Motion Estimation (ME). In this method, we find similar part of image in previous or future frames. Then replace the image by a Motion Vector (MV) in order to reduce time correlation. In this paper, we also introduce JPEG standard and MPEG standard which are the well-known image and video compression standard, respectively.

Keywords: *MPEG, MPEG, DCT, correlation, compression, ME, MV*

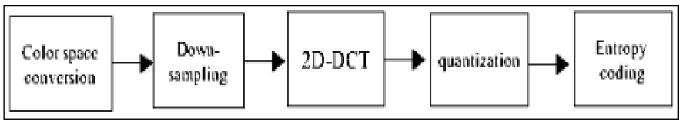
1. Introduction

1.1 Basic introduction

Nowadays, the size of storage media increases day by day. Although the largest capacity of hard disk is about two Terabytes, it is not enough large if we storage a video file without compressing it. For example, if we have a color video file stream, that is, with three 720x480 sized layer, 30 frames per second and 8 bits for each pixel. Then we need ! This equals to about 31.1MB per second. For a 650MB CD-ROM, we can only storage a video about 20 seconds long. That is why we want to do image and video compression though the capacity of storage media is quite large now. In this, we will introduce the basic concept of data compression. The main idea of data compressing is reducing the data correlation and replacing them with simpler data form. Then we will discuss the method that is common used in image/video compression. We will also introduce quantization and entropy coding. After reducing data correlation, the amounts of data are not really reduced. We use quantization and entropy coding to compress the data. Further, we give an example of image compression − JPEG standard. The JPEG standard has been widely used in image and photo compression recently. We discuss how to reduce time correlation with a method called Motion Estimation (ME). And then we give an example of video compression − MPEG standard.

1.2 JPEG Image Compression

Data compression method is different depending on the type of data. For information in the form of images, one of the most popular compression method is JPEG. JPEG stands for Joint Photographic Expert Group. Accordingly widely used in JPEG image included on the internet web pages. Use JPEG create a web page with a picture can be accessed faster than a



web page with an image without compression. Color image JPEG compression consists of five steps . This is shown in figure . The steps are: color space conversion, down sampling, 2-D DCT, quantization and entropy coding. Grayscale image compression uses only last three steps.

Figure . JPEG compression steps of color images

However, this paper's main interest is only on hardware implementation of 2-D DCT combined with quantization and zig-zag process. To achieve high throughput, this paper uses pipelined architecture, rather than single clock architecture designed.

2. Basic Concept of Data Compression

The motivation of data compression is using less quantity of data to represent the original data without distortion of them. Consider the system in Fig. 1, when the encoder receives the target image, it converts the image into bit stream *b*. On the other hand, the decoder receives the bit stream and then converts it back to the image *I*′ . If the quantity of bit stream *b* less than the original image then we call this process *Image Compression Coding*.

There is an example in Fig. using JPEG image compression standard. The compression ratio is 15138 / 83261, about 0.1818 , around one fifth of the original size. Besides, we can see that the decoded image and the original image are only slightly different. In fact, the two images are not completely

Fig. Example of image compression using JPEG standard

same, that is, parts of information are lost during the image compression process. For this reason, the decoder cannot rebuild the image perfectly. This kind of image compression is called *non-reversible coding* or *lossy coding*. On the contrary, there is another form called *reversible coding* that can perfectly rebuild the original image without any distortion. But the compression ratio of reversible coding is much lower. For lossy coding, there is a distortion between the original image and the decoded image. In order to evaluate the coding efficiency, we need a method to evaluate the degree of distortion. There are two common evaluation tools, which are *Mean Square Error* (MSE) and *Peak Signal to Noise Ratio* (PSNR). They are defined as following:

$$\mathrm{MSE} = \sqrt{\frac{\sum_{x=0}^{W-1}\sum_{y=0}^{H-1}\left(f(x,y)| - f'(x,y)\right)^2}{WH}}$$

$$\mathrm{PSNR} = 20\log_{10}\frac{255}{\mathrm{MSE}}$$

f (*x*, *y*) and *f* ′(*x*, *y*) denote the original image and decoded image, respectively. The image size is *W* ×*H* . In Eq. (2), the PSNR formula is common used for 8-bits image. Note that the larger the PSNR, that smaller the degree of distortion.

Now, we want to know how and why we can make an image compressed. Generally speaking, the neighboring pixels in an image have highly correlation to each other. That is why images can be compressed in a high compression ratio. The image coding algorithm today consists of reducing correlation between pixels, quantization and entropy coding. We will discuss these parts one by one in the following chapters. The coding algorithm system model is shown in Fig.

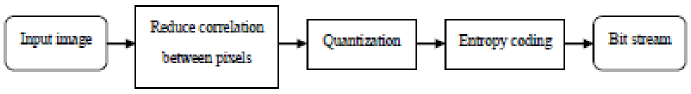


Fig. general constitution of image coding algorithm

3. DCT block transforms and quantization

The source image is first broken into 8×8 blocks. The pixel values syx in each of these blocks are then transformed by the FDCT into an 8 × 8 block of 64 DCT coefficients. Figure illustrates this process with the DCT coefficients denoted by the variables Svu where v and u are integer frequency variables between 0 and 7. The coefficient S00 is known as the DC coefficient because it represents the average value of the block, whereas the remaining values are known as the AC coefficients. The FDCT and IDCT are defined as follows:

$$\begin{aligned} &FDCT: \\ &S_{vu} = \frac{1}{4}C_uC_v\sum_{x=0}^{7}\sum_{y=0}^{7} s_{yx}cos\frac{(2x+1)u\pi}{16}cos\frac{(2y+1)v\pi}{16} \\ &IDCT: \\ &s_{yx} = \frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7} C_uC_vS_{vu}cos\frac{(2x+1)u\pi}{16}cos\frac{(2y+1)v\pi}{16} \end{aligned}$$

where C_u and C_v are defined by

$$C_u = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } u = 0 \\ 1 & \text{for } u \neq 0 \end{cases}$$

4. Differential Encoding and the Zig-Zag Scan Pattern

Among the quantized 64 DCT coefficients, the DC coefficient is treated separately from the other 63 AC coefficients. (In the following sequel, when DCT coefficients are mentioned, they implicitly mean the quantized values.) This is because it corresponds to the average gray level of each 8 × 8 block. Generally, the DC coefficient has the highest energy of the DCT coefficients; so a fine quantization step size will lead to a large number of bits to code.

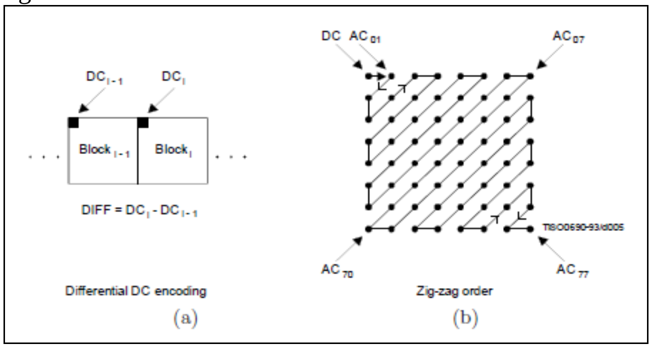


Figure: Encoding of DC and AC coefficients in JPEG.
(a) The DC coefficient is differentially encoded from block to block using a raster ordering of the blocks.
(b) The AC coefficients are ordered within each block using a Zig-zag scan pattern. The pattern is used so that small coefficients that are likely to be zero are grouped together.

Alternatively, a large quantization step size will result in a substantial DC shift, which can cause blocking artifacts in the restored image. Another important property of the DC coefficient is that it tends to be highly correlated among adjacent image blocks. This is because the average gray level of adjacent image blocks is likely to be similar. To improve image quality and reduce bit rate, the DC coefficient is differentially encoded. This means that, using a raster ordering of the blocks, only the difference between the current and previous DC coefficients is coded. The differential DC coefficient for the k_{th} block is computed by

$$\mathrm{DIFF}(k) = Q_{(k)00} - Q_{(k-1)00} \quad (2)$$

Where,
$Q_{(k)00}$ is the current DC coefficient and $Q_{(k-1)00}$ is the DC coefficient from the previous block in raster order. For the first block of the image,
$Q_{(k-1)\,00}$ is set to zero.
The remaining 63 AC coefficients have a high probability of being zero after quantization because the higher frequency coefficients usually have lower energy and the higher frequency quantization table entries are usually larger. Therefore, it is very important to order the coefficients so that coefficients of zeros are likely to be grouped together, forming long “run lengths” of zeros.

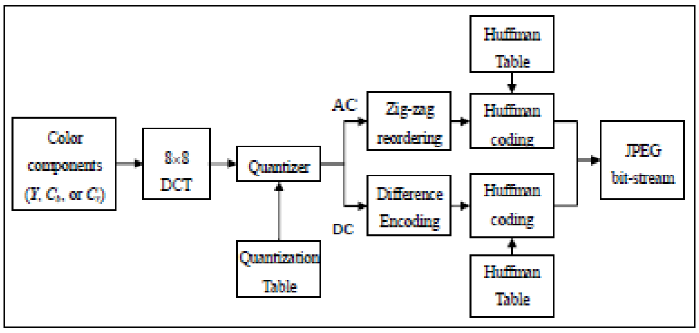
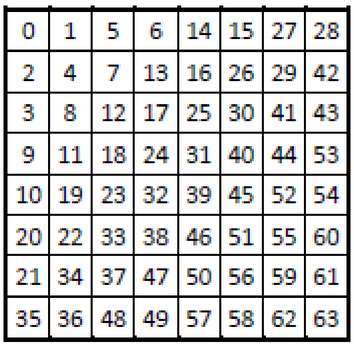


Fig. Baseline JPEG encoder



0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	63

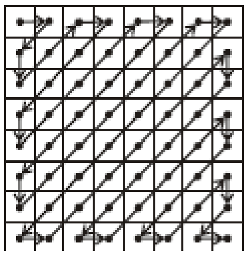


Fig. Zig-Zag reordering matrix

5. Zero Run Length Coding Of Ac Coefficient

Now we have the one dimensional quantized vector with a lot of consecutive zeroes. We can process this by run length coding of the consecutive zeroes. Let's consider the 63 AC coefficients in the original 64 quantized vectors first. For example, we have:
57, 45, 0, 0, 0, 0, 23, 0, -30, -16, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, ..., 0

We encode for each value which is not 0, than add the number of consecutive zeroes preceding that value in front of it. The RLC (run length coding) is:
(0,57) ; (0,45) ; (4,23) ; (1,-30) ; (0,-16) ; (2,1) ; EOB
The EOB (End of Block) is a special coded value. If we have reached in a position in the vector from which we have till the end of the vector only zeroes, we'll mark that position with EOB and finish the RLC of the quantized vector. Note that if the quantized vector does not finishes with zeroes (the last element is not 0), we do not add the EOB marker. Actually, EOB is equivalent to (0,0), so we have :
(0,57) ; (0,45) ; (4,23) ; (1,-30) ; (0,-16) ; (2,1) ; (0,0)

The JPEG Huffman coding makes the restriction that the number of previous 0's to be coded as a 4-bit value, so it can't

overpass the value 15 (0xF). So, this example would be coded as :
(0,57) ; (15,0) ; (2,3) ; (4,2) ; (15,0) ; (15,0) ; (1,895) ; (0,0)
(15,0)
is a special coded value which indicates that there are 16 consecutive zeroes.

6. Difference Coding of DC Coefficient

Because the DC coefficients in the blocks are highly correlated with each other. Moreover, DC coefficients contain a lot of energy so they usually have much larger value than AC coefficients. That is why we have to reduce the correlation before doing encoding. The JPEG standard encodes the difference between the DC coefficients. We compute the difference value between adjacent DC values by the following eq uation:

$$Diff_i = DC_i - DC_{i-1}$$

Note that the initial DC value is set to zero. Then the difference is Huffman encoded together with the encoding of AC coefficients. The difference coding process is shown in Fig.

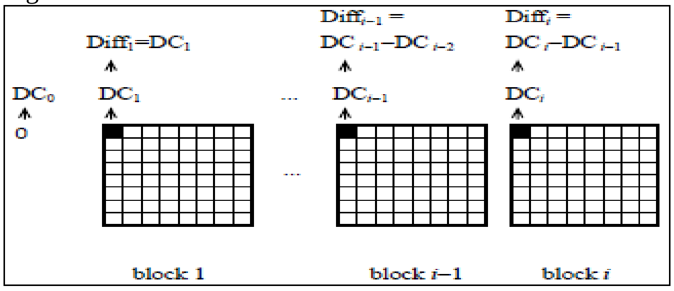


Fig. Difference coding of DC coefficients

7. Conclusion and Future Scope

2D-DCT combined with quantization and zigzag buffer is designed using VHDL. System is tested with real grayscale image. In this paper, a new fully parallel architecture based on row-column decomposition has been proposed for the computation of the 2D DCT. The system involves no memory transposition, and is highly modular and utilizes a highly parallel structure to achieve high-speed performance. Due to its widely identical units, it will be relatively easy to implement and very suited to VLSI implementation. It uses two identical units for the computation of the row and column transforms and arrays of shift registers to perform the transposition operation. As compared to a pipelined regular architecture, the proposed architecture achieves the same throughput rate at much lower hardware cost and communication complexities. It is also worth mentioning that in the proposed design, the same architecture can be used for the computation of both the forward and the inverse 2D DCT.

The aforementioned attributes of the DCT have led to its widespread deployment in virtually every image/video processing standard of the last decade, for example, JPEG (classical), MPEG-1, MPEG-2, MPEG-4, MPEG-4 FGS, H.261, and H.263. Nevertheless, the DCT still offers new research directions that are being explored in the current and upcoming image/video coding standards.

References

[1] R. C. Gonzalez and R. E. Woods, *Digital Image Processing 2/E*. Upper Saddle River, NJ: Prentice-Hall, 2002.
[2] J. J. Ding and J. D. Huang, “Image Compression by Segmentation and Boundary Description,” June, 2008.
[3] G. K. Wallace, 'The JPEG Still Picture Compression Standard', Communications of the ACM, Vol. 34, Issue 4, pp.30-44.
[4] L. Agostini, S. Bampi, “Pipelined Fast 2-D DCT Architecture for JPEG Image Compression” Proceedings of the 14th Annual Symposium on Integrated Circuits and systems Design, Pirenopolis, Brazil. IEEE Computer Society 2001. pp 226-231.