Aggrandize Bit Plane Coding using Gray Code Method

Aditya Kumar & Pardeep Singh

Department Of Computer Science & Engineering
National Institute Of Technology
Hamirpur, India

ABSTRACT

With the increasing demand of technology, existing computer make use of graphics spaciously. Windows based operating system exposed to view the file disk file's directory graphically. Many applications provide graphical user interface , which it makes to easier to understand or interpret e.g. internet download manager shows the status of downloading a file graphically. Advancement of graphics, we can easy to understand that each information after converting it into image. So image is very important but sometimes it tends to be big. This is why image compression is so important. Many image compression algorithms (lossy or lossless) have already been devised adhering to their perspective point of view. In this paper we propose conceptually a new algorithm for image compression for minimizing a number of bits for storing an image into disk.

KEYWORDS

Bit plane, correlation gray code, PSNR, Quantization, spatial redundancy.

1. INTRODUCTION

A digital image is defined as a (pxq) rectangular array of dots, where p is number of rows and q is number of columns. In the technical term, pxq is called the resolution of the image and dots are called pixel (for image) or pels (for audio/video). Basically resolution of image represents number of pixel per unit area (dpi) [2].The purpose of compression is only reduce the no. of bits which used to store a data in our disk. Now before going detailed survey on image compression here we can distinguish many images on behalf of its pixel intensity value.

A Monochromatic image (bi level image).this type of image is simplest type of image because its pixel can have one of two value black and white and represented by one bit only.

A Gray scale image can have 2^n no. of gray shades [9]. Its pixel can have one of 0 to n-1 value. Each pixel of this type of image is represented by bytes. The group most significant bits of its pixel are most significant bit plane. Due to this it has n bit planes.

In a **continuous tone image**, when a neighborhood pixel (adjacent pixel) is not identical by just one unit of bits then it seems to be difficult for human eyes to distinguish. Basically this type of image have many similar color or represented by one area of colors and RGB [5] color pattern. Example of continuous tone image (natural image) is, taking a photograph through digital camera, scanning a photograph or painting.

Basically discrete tone image (**A synthetic image (discrete tone image or graphical image**) is non natural image [5] (artificial image) but every artificial image is not discrete tone image e.g. Computer generated image look like a continuous image spite of being artificially generated. In this type of image adjacent pixel are similar (identical) or vary significantly and also does not have noise or blurring. Image compression algorithm (lossy algorithm) which applied for continuous tone image often do not handle the sharp edge [9]. Compressing the text is an example of discrete tone image. As result of this, different compression methods to be apply for this type of image to removing the redundancy [4] .example of synthetic image, photograph of artificial object or machine, text etc.

There are many image compression algorithms which one applied for different type of image. Basically it's intuitively understandable that each image may have property of redundancy but they are redundant in different manner.

In image compression, information can be compression if it is redundant. It has been mentioned several times that data compression amounts to reducing or removing redundancy in the data. With lossy compression [9], however, we have a new concept, namely compressing by removing irrelevancy. An image can be lossy-compressed by removing irrelevant Information even if the original image does not have any redundancy.

The idea of losing image information becomes more delicious when we consider how digital images are created. Here are three examples:

(1) A real-life image may be scanned from a photograph or a painting and digitized [4] (converted to pixels).

(2) An image may be recorded by a video camera that creates pixels and stores them directly in memory [5].

(3) An image may be painted on the screen by means of a paint program.

In all these cases, some information is lost when the image is digitized. The fact that the viewer is willing to accept this loss suggests that further loss of information might be tolerable if done properly.

Digitizing an image involves two steps: sampling and quantization. Sampling an image is the process of dividing the two-dimensional original image into small regions: pixels. Quantization [3] is the process of assigning an integer value to each pixel. Notice that digitizing sound involves the same two steps, with the difference that sound is one-dimensional.

Here is a simple process to determine qualitatively the amount of data loss in a compressed image. Given an image *A*, (1) compress it to *B*, (2) decompress *B* to *C*, and (3) subtract *D* = *C* − *A*. If *A* was compressed without any loss and decompressed

Properly, then *C* should be identical to *A* and image *D* should be uniformly white. The more data was lost in the compression, the farther will *D* be from uniformly white.

Certain compression methods are lossy. They achieve better compression by losing some information. When the compressed stream is decompressed, the result is not identical to the original data stream. Such a method makes sense especially in compressing images, movies, or sounds. If the loss of data is small, we may not be able to tell the difference.

The main objective of this paper is to provide a new image compression algorithm for gray level image or color images.

After this introduction we have given related work in section 2, proposed compression algorithm in section 3, performance analysis in section 4 and finally the conclusion in last section 5.

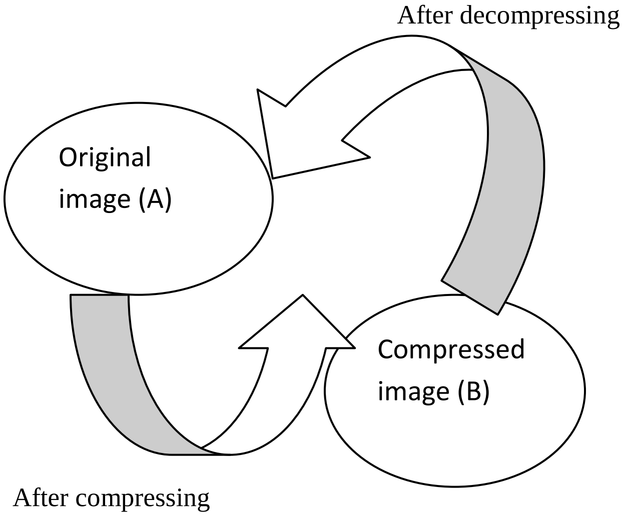


Figure.1

1. RELATED WORK

In this section we present various image compression algorithms.

There are several image compression algorithms already implemented for gray level [1] [6] or color images. It is intuitively clear that an image compression algorithm has been developed for particular image can be used for another type of image. A bi-level image [7] compression algorithm can be used for compressing gray level image by taking a gray scale image, separating into bit plane then compressing by bi-level image. Now suppose, for an image with 16 gray scale value then each pixel can be represented by four bit only. As a result image can be separated into four bi-level images. But difficulty arise when suppose $3=0110_2$ and $4=0100_2$ are two adjacent bit (difference of just one) but putting both into different bit planes. It's against the image compression approaches. For utilize any bi level image compression scheme to gray level image compression, binary representation of chronological (consecutive) integers are differentiate by just one bit only. It's called reflected gray code (RGC) scheme. This scheme is simple to implement by following way. For computing or constructing a 4-bit reflected gray code .first we produce 3-bit two set (000 001 011 010) and (100 101 111 110) then appending 0 and 1 alternatively for producing input variables, after that calculate grey code and reflected grey code of that respective input.

For elevating (constructing) a grey code , we first generate a logical right shift of that integer then ex-OR with its corresponding integer(binary code).e.g.0010 ex-OR(LRS of 0010) =>0010 ex-OR 0001=>0011.

Table 1 shows that how are bits going to change when we displace i_k.bit to i_{k+1} through binary code of the first 16 integer [1] [6]. In the above table bits changes between chronological integers are displayed in boldfaces. It is simple to recognize that least significant bit alter significantly while most significant bits are remain identical (sometimes unchanged).this method can be applied to compressing color images, but the color image should first be separated into three color components [7], and each component compressed individually as a grayscale image. Color images but sometimes its performance would suffer as result.

Table 1

Binary Number	Grey code
0000	0000
000**1**	0001
00**10**	0011
001**1**	0010
0**100**	0110
010**1**	0111
01**10**	0101
011**1**	0100
1000	1100
100**1**	1101
10**10**	1111
101**1**	1110
1**100**	1010
110**1**	1011
11**10**	1001
111**1**	1000

Another next method, Subsampling [9] is perhaps the simplest way to compress an image. One approach to sub sampling is simply to ignore some of the pixels. The encoder [2] may, for example, ignore every other row and every other column of the image, and write the remaining pixels (which constitute 25% of the image) on the compressed stream. The decoder inputs the compressed data and uses each pixel to generate four identical pixels [9] of the reconstructed image. This, of course, involves the loss of much image detail and is rarely acceptable. Notice that the compression ratio is known in advance. A slight improvement is obtained.

Quantization [10] as another example of compressing an image, in this scheme image partition into equal size of block of pixel and the encoder has a list of blocks of the same size. Each image lock *B* is compared to all the blocks of the codebook and is matched with the "closest" one. If *B* is matched with codebook block *C*, then the encoder writes a pointer to *C* on the compressed stream. If the pointer is smaller than the block size, compression is achieved.

Image transform, the mathematical concept of a transform is a powerful tool in many areas and can also serve as an approach to image compression. As well as the Fourier transform [9]. An image can be compressed by transforming its pixels (which are correlated) to a representation where they are decorrelated [9]. Compression is achieved if the new values are smaller, on average, than the original ones. Lossy compression can be achieved by quantizing the transformed values. The decoder inputs the transformed values from the compressed stream and reconstructs the (precise or approximate) original data by applying the opposite transform [5].

In next section we propose an algorithm for minimizing the constraints of above deal in algorithms.

2. PROPOSED COMPRESSION ALGO

The main objective is to compress the image with efficiently and after all applying reversed process (decompression algorithm) to get original image which have nearest PSNR ratio with that image.

Image A is the original image whereas B is marks as a reconstructed image. We use firstly gray code method to convert a binary image into reflected gray code image. Using this method we divide an image into n (four bit planes for more than 16 bit image) bit planes. If image is color image then cardinally divides it into three RGB bit planes then applying gray code method. It is intuitively clear that gray code method basically remove the correlation between pixels of an image.

Algorithm (M, n)

```
1. Take a gray scale image M.
2. Separate the bit planes of that gray scale image ($2^n$ shade image) into n bit plane where n is the number of bits to represent a pixel.
3. Apply the gray code method 1 to $n^{th}$ bit plane.
4. Compression_Algorithm(Gray_code_applied_image($M_n$))
```

Gray_Code (M_n)

```
1. Open the file (image M) for binary read access.
2. Read matrix binary image intensity value.
3. Assigning bit plane=n
4. Apply right shift on image($M_n$) and its ex-OR with 1.
```

```
filename=’aditya_kumar.tif’;
fid=fopen(filename,'r');
image=fread(fid);
bitplane=n;
for=1: n
a=bitshift(img,-1);
b=bitxor(img, a);
Gray_Image=bitget (b, n);
Imagesc (Gray_Image),
end
```

Compression _Algorithm (Image (M_n))

1. An image is divided into nxn block (usually 2x2, 4x4, and 8x8).
2. Assuming that a block is containing n pixel with intensity value p_1 top_n.
3. Calculate the mean and variance of each block.

$$\text{Mean p}= \frac{1}{n}\sum_{i=0}^{n} p_i$$

$$\text{Variance } p^2 \quad = \frac{1}{n}\sum_{i=0}^{n} p_i^2$$

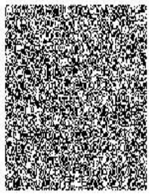
$$\text{Deviation } \alpha = \sqrt{\text{mean} - \text{varinace}}$$

4. Calculate three values for each blockp_{max}, p_{min} andp_{base}. If ($p_{max} \geq p_{base}$) then p_i=p_{max} Otherwise p_i=p_{min}
5. It is intuitively clear that n^+ pixels are greater than base value and n^- pixel less than its base value (p_{base}).
6. Now update the 4 pixel intensity value using following formula.

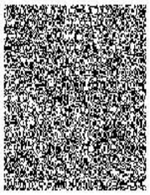
$$p_{max}=p_{max}+ \alpha\sqrt{\frac{n^+}{n^-}}\ , \ p_{min}=p_{min}+ \alpha\sqrt{\frac{n^+}{n^-}}$$

original image

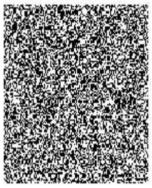
binary image #1



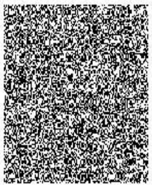
gray code applied image #1



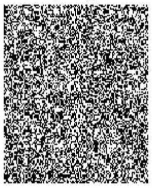
binary image #2



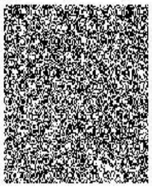
gray code applied image #2



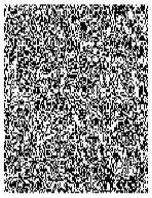
binary image #3



gray code applied image #3

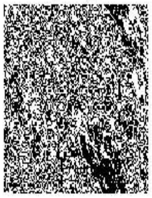


binary image #4



gray code applied image #4

binary image #5



gray code applied image #5

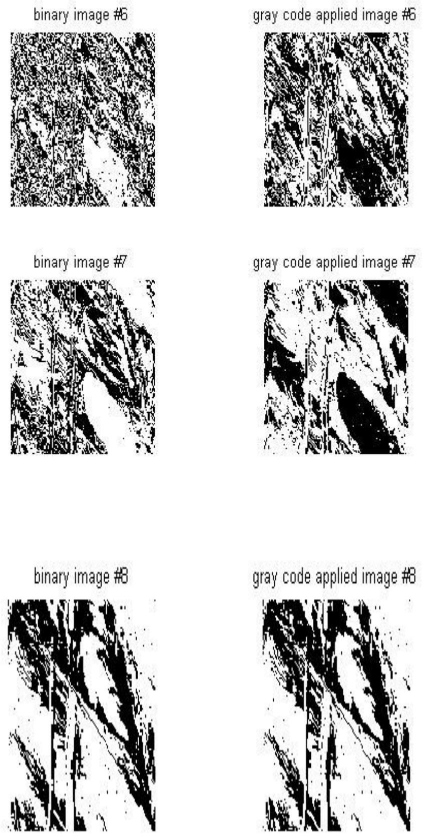

binary image #6
gray code applied image #6
binary image #7
gray code applied image #7
binary image #8
gray code applied image #8

Fig 1, Fig 2

Left side Bit plane without using gray code method and right side bit plane using gray code method

4. PERFORMANCE ANALYSIS

The above discussed algorithm has been implemented in MATLAB-7.0.In this image compression algorithm; we take adi32x32.tif gray scale image, it is intuitively understandable that this image have only 8 bit plane. Figure1, figure2 shows 8 bit plane with gray code method or without gray code method. After applying a gray code method, correlation between pixel of an image from 1 to 8 decrease mean to say spatial redundancy reduce because most significant bit have less pixel correlation and redundancy compare than least significant bit of pixels of image.

Here we take each bit plane image separately and compressed its one by one individually and for reconstructing an image we decompress each bit plane separately using decompression algorithm.

Now simulating the above proposed algorithm on matlab tool, gray scale image is divided firstly into 8 bit plane then compressed each bit plane separately. In above depicted diagram following steps are followed.adi32x32.tif image is divided into M_n(where n=1….n-1) bit plane.

Gray code method is applied to each bit plane. In the above figure 4 is last bit plane applied image (only one bit plane compression procedure mention here).

Table 2

121	**114**	**56**	**47**
37	**200**	**247**	**255**
16	**0**	**12**	**169**
43	**5**	**7**	**251**

Compression algorithm is used for figure 4and figure 5 is image after compression. In this compression following task is done for simulation point of view. Suppose we select 4x4 block of pixel. In this above block can calculate three parameter value mean, variance and standard deviation. The calculated value of mean and standard deviation is 98.75 and 92.75 respectively. Here n^+ and n^- have 7 and 9 respective value. Now we can find out p_{max} and p_{min}using the formula which mention in our propose algorithm.

$$\mathrm{p_{max}}=98.75+ 92.75\ \sqrt{\frac{7}{9}}=204.14$$

$$\mathrm{p_{min}}=98.75+ 92.75\ \sqrt{\frac{7}{9}}=16.78$$

The rounded value of $\mathrm{p_{max}}$ and $\mathrm{p_{min}}$ is 204 and 17 respectively. Put these value in table 2 the modified table is

Table 3

204	**17**	**17**	**17**
37	**204**	**204**	**204**
17	**17**	**17**	**204**
17	**17**	**17**	**204**

It is intuitively clear that 4x4 block is compressed into 16 bit only. Now again altered block is

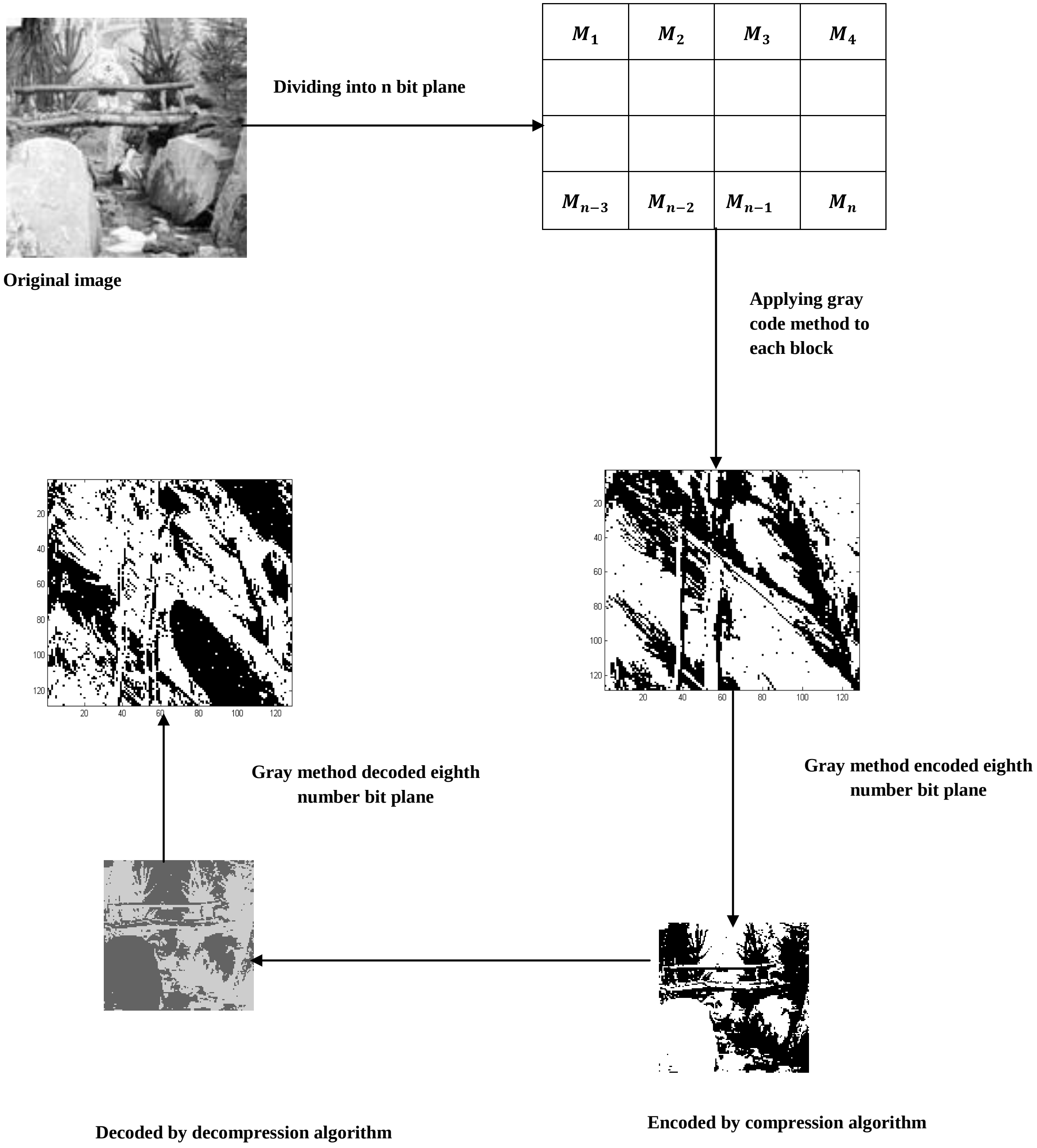
Table 4

1	**0**	**0**	**0**
0	**1**	**1**	**1**
0	**0**	**0**	**1**
0	**0**	**0**	**1**

Now respective bit plane is compressed like table 4. This procedure is applied to all bit planes then our original image is compressed.

5. CONCLUSION

This paper presents an improved compression algorithm for gray scale image to reduce the correlation and spatial redundancy between pixels of an image. Our proposed algorithm is useful to maintain the compression ratio and quality of an image. This algorithm was test on different gray scale images with different sizes. Our future works is update or extend this proposed algorithm for color images.


Original image
Dividing into n bit plane
M_1
M_2
M_3
M_4
M_{n-3}
M_{n-2}
M_{n-1}
M_n
Applying gray code method to each block
20
40
60
80
100
120
Gray method encoded eighth number bit plane
Gray method decoded eighth number bit plane
Encoded by compression algorithm
Decoded by decompression algorithm

6. REFERENCES

[1] Jun Wang, Lin-bo Luo, Kyeong-yuk Min, Yeun-Cheul Jeung, Jong-wha Chong, "Single bit plane based block truncation coding for color image compression in LCD overdrive ,"2010 Digest of Technical Papers International Conference on IEEE INFOCOM.

[2] E.M. Rhoma, A.M Abobaker," Mean square error minimization using interpolative block truncation coding algorithms ,"2nd International Conference on Education Technology and Computer (ICETC), 2010.

[3] Yu, Luo H,"Colour image retrieval using pattern co-occurrence matrices based on BTC and VQ, "Publication Year: 2011, Page(s): 100 – 101.

[4] G. Chopra, A.K Pal," An Improved Image Compression Algorithm Using Binary Space Partition Scheme and Geometric Wavelets". IEEE Transactions on Image, 2011.

[5] Nak-Kwon Lim, Dae-Young Kim, Haeyoung Lee," Interactive progressive image transmission for real time applications," IEEE Transactions on Consumer Electronics,2010.

[6] Minjie Chen, P Franti, Mantao Xu, "Lossless bit-plane compression of images with context tree modeling," International Conference on Green Circuits and Systems (ICGCS),2010 .

[7] Pham, D. Premkumar, A. Madhukumar, " Error Detection and Correction in Communication Channels Using Inverse Gray RSNS Codes,", IEEE Transactions on Communications .

[8] Ichigaya, A. Kurozumi, M. Hara, N.Nishida, Y. Nakasu, "A method of estimating coding PSNR using quantized DCT coefficients," IEEE Transactions on Circuits and Systems for Video Technology.

[9] J. Shukla, M. Alwani, A.K.Tiwari," A survey on lossless image compression methods", 2010 2nd International Conference on Computer Engineering and Technology (ICCET).

[10] Nam-Joon Kim, Erturk, S, Hyuk-Jae Lee; "Two-bit transform based block motion estimation using second derivatives," IEEE Transactions on VLSI design, 2010.