Majority-Parity-Guided Watermarking for Block-Truncated Images

Jing-Ming Guo, Member, IEEE and Yun-Fu Liu

Department of Electrical Engineering
National Taiwan University of Science and Technology
Taipei, Taiwan
E-mail: jmguo@seed.net.tw, yunfuliu@gmail.com

ABSTRACT

In this paper, a watermarking scheme, called Majority-Parity-Guided Error-Diffused Block Truncation Coding (MPG-EDBTC), is proposed to achieve with high image quality and embedded capacity. The main problem of traditional BTC is its poor quality over configurations of high compression ratio. To overcome such problem, the extreme pixel values are employed to substitute both high and low means. The quantized error is also compensated by adjusting the neighboring pixels. With these strategies, the image quality and processing efficiency are improved. Moreover, the watermark is embedded by evaluating the parity value in a pre-defined Parity-Check Region (PCR). As seen in the experimental results, the proposed scheme can provide good robustness, image quality, and processing efficiency. Finally, the proposed MPG-EDBTC is extended to embed multiple watermarks and achieves excellent image quality, robustness, and capacity as well. Nowadays, most multimedia is stored in compressed format. It is more appropriate to embed information such as watermarks in compressed domain. The proposed method has been proved to solve effectively the inherent problems in traditional BTC, and provide excellent performance in watermark embedding.

Index terms: Block truncation coding, error diffusion, digital watermarking, digital halftoning

1. INTRODUCTION

Block Truncation Coding (BTC), which was proposed by Delp and Mitchell in 1979 [1], is a technique for image compression. Although BTC cannot provide comparable coding gain as other modern compression techniques, such as JPEG or JPEG2000, the complexity of BTC is much lower than that of these modern techniques, which makes it feasible for less powerful processing kernel, such as Arm-based applications. In the literature, many approaches have been proposed to improve BTC. Kamel et al. [2] proposed two modifications on BTC. The first one allows the partitioning of the image into variable block sizes rather than a fixed size. The second modification involves the use of an optimal threshold to quantize the blocks by minimizing the mean square error. Chen and Liu [3] proposed a Visual Pattern Block Truncation Coding (VPBTC), in which the bitmap is employed to compute the block gradient orientation and match the block pattern. Another refinement is the classification of blocks according to the properties of human visual perception. However, most of the improvements described above increase the complexity substantially.

Digital halftoning [4] is a technique which converts a grayscale image into a binary image. This halftone image can resemble the original grayscale image when viewed from a distance with the lowpass characteristic of Human Visual System (HVS). Various attempts including ordered dithering [4], error diffusion (EDF) [5]-[7], dot diffusion [8]-[9], and Direct Binary Search (DBS) [10] have been devoted to quality improvement and complexity reduction. On the other hand, the replacement of the two distinct values in a block of a BTC image is similar to the binary fashion in halftone image. Thus, the halftoning technique can be used for arranging the distribution of the two distinct values in a BTC image to subsequently improve the image quality [11]. Among the above-mentioned halftoning techniques, the EDF provides excellent image quality at reasonable complexity cost. More specifically, the average grayscale of an image can be maintained after it is transformed into binary fashion. Consequently, the EDF is combined with BTC in this work to achieve excellent image quality.

Digital watermarking is a value-added technique for providing copyright protection or authentication feature. Nowadays, it is impossible to store of transmit an image or a video sequence without prior compression. As mentioned above, BTC is a good solution for image/video compression with an extremely low complexity. Subsequently, the possibility of embedding watermarks in BTC-compressed images has been investigated. For example, Tu and Hsu [12] proposed an ownership share approach, which combines the image and its watermark in generating a secret key, while leaving the original image unmodified. The ownership share is then required in the process of decoding. Lin and Chang [13] proposed a data hiding scheme for images compressed by BTC. They embedded information into both high and low means by alternating one bit value, according to the value of the messages, as well as into the bitmap using the Minimum Distortion Algorithm (MDA).

2. PERFORMANCE EVALUATIONS

In this section, the performance evaluation approaches, PSNR and Correct Decoding Rate (CDR), employed in this work are defined. For an image of size $P \times Q$, the quality evaluation of an images is defined as

$$PSNR = 10 \log_{10} \frac{P \times Q \times 255^2}{\sum_{i=1}^{P} \sum_{j=1}^{Q} (f(i,j) - \hat{f}(i,j))^2}$$

where $f(i,j)$ denotes the original grayscale host image; $\hat{f}(i,j)$ denotes the watermarked host image; $\tilde{f}$ denotes a Gaussian filter for simulating the lowpass characteristic of human visual system (HVS), and $R$ denotes the support region of this filter. In this work, the standard deviation of the Gaussian filter is 1.3, and $R$ is fixed at 7x7.

The other performance evaluation is the Correct Decode Rate (CDR), which determines the similarity between the original watermark and the corresponding decoded watermark. Suppose the watermark is of size $M \times N$.

$$CDR = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} w_{ij} \times 100\%,$$

where $w_{ij}$ denotes the encoded watermark, and $R_{ij}$ denotes the decoded watermark and the original watermark, respectively. The notation $\Theta$ denotes XNOR operation. In general, the greater the CDR, the better the decoded result will be.

3. MAJORITY-PARITY-GUIDED ERROR-DIFFUSED BLOCK

Truncation Coding

The proposed watermarking is introduced in this section. Figure 1 shows its system architecture. For simplicity, the selected watermarks are of identical size, each of which is permuted with a pseudo-random key to improve security and withstand some types of attacks. In the rest of this section, the two functional blocks, encoder and decoder of MPG-EDBTC, are fully discussed.

3.1 MPG-EDBTC Encoder

According to Fig. 1, two inputs, including host image and K watermarks, are directed to the MPG-EDBTC encoder. Figure 2 shows the fundamental embedding concept. Let the host image and watermark be of sizes $P \times Q$ and $M \times N$, respectively. A watermark is generally in binary fashion. As indicated, the host image is divided into many non-overlapped blocks, and the number of blocks is equal to the number of watermark pixels, meaning that a watermark bit is embedded in a block. The processing order is in raster scan path, which means from left to right and top to bottom. Figure 3 illustrates the processing algorithm. In Fig. 3(a), $h_{ij}$ denotes the grayscale
value of the host image at the current processing position \((i,j)\). \(h_{ij}\) denotes the diffused error sum added from the neighboring processed pixels, \(v_{ij}\) denotes the modified input value, and \(w_{ij}\) denotes the binary results by thresholding from \(v_{ij}\). The binary thresholded result is replaced by either maximum or minimum values in the current block, and the threshold exploits the mean of the current block. Error kernel is employed to diffuse the error caused by the difference \(e_{ij}\). Three well-known error kernels, Floyd [5], Jarvis et al. [6], and Stucki [7], are shown in Fig. 4, where the notation \(x\) denotes the current processing position. The relationship between these variables is organized as,

\[
v_{ij} = h_{ij} + k_{ij} + \epsilon, \quad \text{where} \quad k_{ij} = \sum_{x}^{e_{x}} e_{x} \times e_{x,n},
\]

(3)

where \(e\), \(h_{\text{min}}\), and \(h_{\text{max}}\) denote the mean, minimum, and maximum values, respectively of the current processed block. The variable \(\epsilon\) is an additive value for controlling the result of \(w_{ij}\), which will be discussed later. Comparing with traditional BTC, the high mean and low mean are replaced by the local maximum and minimum values. Consequently, the complexity can be significantly reduced by saving the effort in calculating the variance used in high and low means. Moreover, the image quality can be improved by diffusing the quantized error into the neighboring pixels. Notably, the error in the boundary pixels of a block should also diffuse to its neighboring blocks to eliminate the blocking effect.

A watermark bit is embedded by controlling the parity value in a divided block of the EDBTC image. The parity value is evaluated from the number of pixels with minimum values in a Parity-Check Region (PCR) as defined below,

\[
p_{\text{PCR}} = \sum_{\text{PCR}}^{\text{minimum}} \text{if } w_{\text{PCR}} = \text{minimum } \mod 2.
\]

(5)

Notably, the calculating area excludes the current processing position \((i,j)\). The additive value \(\epsilon\) as given in Eqs. 3-4 is employed to control parity of the current position \((i,j)\) in order to comply with the expected value. An example is shown in Fig. 3(b), where the size of PCR for each watermark embedding is different. If the PCR exceeds the boundary of the host image, as the bottom one with embedding watermark K (\(w_{K}\)) in Fig. 3(b), the exceeding area can be considered as an area with all minimum values. In this case, K watermarks are embedded in the same host image. In each time, each bit of the watermarks is considered simultaneously using a different PCR size.

Take a watermark embedded in a host image for example. Each bit in a watermark has two different values, 0 and 1. When the parity value is equal to the watermark value; for example, parity value=0 and watermarks=0, there is no need to modify the parity value to hide the watermark information. In this case, we select +noise (noise itself) as a positive value) to decrease the likelihood that the current processed value becomes a minimum value. Conversely, when the parity value does not comply with that of the watermark bit; for example, parity value=0 and watermarks=1, it implies a need to add a parity of 1 to change the parity value. In this case, we select –noise to increase the likelihood that current processed value becomes a minimum value. The previous case is for embedding a watermark in a host image. Nonetheless, it can be extended to embed multiple watermarks. As shown in Fig. 3(b), each watermark associated with a different PCR size can find a selection for noise operation. For a bit in each watermark, the voting condition can be obtained, which is called \(p_{\text{vote}}\) and \(n_{\text{vote}}\). The \(p_{\text{vote}}\) denotes the number of watermarks that need +noise, and \(n_{\text{vote}}\) denotes the number of watermarks that need –noise. Conducting this voting for all watermarks can determine the final selection of noise operation. Figure 3(a) shows the voting concept. Since the final selection is the result of majority voting. The variable \(\epsilon\) of Eqs. 3 and 4 is defined as follows,

\[
\epsilon = \begin{cases} 
N_{p} \text{ if } p_{\text{vote}} > n_{\text{vote}} & \text{,} \\
0 & \text{if } p_{\text{vote}} = n_{\text{vote}} \\
N_{n} \text{ if } p_{\text{vote}} < n_{\text{vote}} & \text{,}
\end{cases}
\]

(6)

where the variable \(N_{p}\) denotes noise increment.

3.2 MPG-EDBTC Decoder

In the encoder, the original host image is divided into many non-overlapped blocks, and a watermark bit is embedded in a block. In the decoder, the pixels inside a block are employed to conduct the majority voting to decode the watermark bit, and can thus achieve good robustness. First, the received watermarked image is divided into many non-overlapped blocks of size \((P/M) \times (Q/N)\), where \(P \times Q\) is the size of the watermarked image, and \(M \times N\) is the size of the embedded watermark. In practical application, all kinds of attacks may exist, which lead to alterations in maximum and minimum values. For this, instead of calculating the number of minimum values in a PCR directly, the mean in a block is calculated to threshold the pixels inside the block and then produces the temporal binary format image. The procedure described above is organized as below:

\[
wh = \frac{1}{M \times N} \sum_{n=1}^{N} \sum_{m=1}^{M} h_{i+m,j+n}. \quad \text{if } h_{i+m,j+n} < wh
\]

(7)

\[
wh_{i+m,j+n} < wh
\]

(8)

\[
wh_{i+m,j+n} \geq wh
\]

where \(w_{ij}\) and \(wh\) denote the watermarked image and the temporal binary image, respectively. Figure 5 shows the subsequent decoding procedure, where each watermark can be extracted independently using the corresponding PCR size. Watermark extraction involves parity values evaluation via the temporary binary image as given below:

\[
v_{n} = \sum_{m=0}^{N-1} \sum_{n=0}^{M-1} \begin{cases} 
1 & \text{if } th_{i+m,j+n} = 0 \\
0 & \text{if } th_{i+m,j+n} = 255, \\
\text{mod } 2, & \text{if } th_{i+m,j+n} = 255
\end{cases}
\]

(9)

\[
v_{n} = \begin{cases} 
0 & \text{if } \sum_{m=0}^{N-1} \sum_{n=0}^{M-1} \text{mod } 2 = 0, \text{otherwise}
\end{cases}
\]

(10)

where the \(v_{n}\) denotes the nth decoded watermark, and where 0 and 255 indicate black and white pixels, respectively.

4. Experimental Results

In this section, the proposed watermarking is adopted for quantitative performance evaluation. Figure 6 shows the performances with different number of watermarks. This experiment is obtained with parameters PCR=4, \(N_{b}=14\), and watermark of size 64x64. In terms of image quality, even number of watermarks embedding is superior to odd number of watermarks embedding. The reason behind this observation is that when even watermarks are embedded, there is no \(N_{b}\) added, which contributes to image quality. However, the benefit decreases as the number of watermarks is increased as shown in Fig. 6(a). In terms of decoded rate, the CDR decreases when the number of watermarks is increased as expected.

Figures 7 and 8 show the performances with different number of watermarks and noise increments. The objective of this experiment is to find a recommended \(N_{b}\) for a corresponding number of watermarks to achieve acceptable image quality (around PSNR=40 dB) and high CDR. According to Fig. 7, the extremely high-capacity case of embedding 10 watermarks, it still obtains CDR=89.59% and PSNR=39.95 dB. Figure 8 shows the watermarked images and the corresponding decoded watermarks using the recommended \(N_{b}\) from Fig. 7 when even numbers of watermarks are embedded. Figure 8(a) shows the 10 employed watermarks. The results demonstrate that the proposed method can achieve good image quality and decoded rate under huge embedding capacity.

So far, there have been few former approaches addressing the issue of embedding watermarks in BTC images. To Hu’s [12] method is different from the proposed MPG-EDBTC, and the comparison is thus made with Lin-Chang’s method only [13]. Table I shows the robustness of the proposed method is superior to that of Lin-Chang’s method for five types of attacks, two types of attacks have similar performance, and inferior to their method by one type of attack.

5. Conclusions
Nowadays, most images are compressed before they are transmitted or stored, and thus watermarking is highly suggested to be catered into compressed domain of an image. In this work, a high-capacity watermarking technique for Block Truncation Coding (BTC) images is proposed. This technique improves image quality of traditional BTC for configurations of high coding gain, where the energy-preserving property of EDF is exploited to effectively remove the blocking effect inherent in BTC images. Moreover, the efficiency can also be improved by replacing the high mean and low mean with the maximum and minimum values in a block. When watermarks are embedded, the proposed Majority-Parity-Guided Error-Diffused BTC (MPG-EDBTC) can achieve good image quality and decoded rates under a huge embedded capacity. The robustness of the proposed method is superior to that of Lin-Chang’s method for many types of attacks, and thus proves that the proposed watermarking is effective in addressing the security issues in compressed images.

**REFERENCE**


Fig. 6. Average performance with different number of watermarks. (PCR=4 and \( N_t = 14 \)) (a) Average PSNR. (b) Average CDR.

Fig. 7. Average performance with different number of watermarks and noise increments. (PCR=4) (a) Average PSNR. (b) Average CDR.

Fig. 8. Practical embedded images and the corresponding decoded watermarks with different sizes of watermarks (PCR=4). (a) 10 watermarks. (b) Watermarked image and the two decoded watermarks \( (N_t = 32) \), and (c) Four watermarks embedding \( (N_t = 27) \), (d) Six watermarks embedding \( (N_t = 25) \), (e) Eight watermarks embedding \( (N_t = 24) \), and (f) 10 watermarks embedding \( (N_t = 24) \). (Watermarked images printed at 400 dpi, decode watermarks printed at 200 dpi)

| Table I: Performance comparison with Lin-Chang method [13] |
|----------------------------------|-----------------|-----------------|------------------|
| Attack                           | Lightening      | Darkening       | Pulse Noise      | Gaussian Noise   |
| Methods                          | -30             | -50             | -50              | 0.2              | 0.3              |
| MPG-EDBTC                        | 98.9            | 97.6            | 97.2             | 90.6             | 91.7             | 67.7             | 99.2             | 94.5             |
| Lin-Chang Lowmean                | 99.7            | 98.9            | 98.9             | 97.9             | 96.2             | 49.5             | 61.1             | 55.2             |
| Lin-Chang Highmean               | 98.9            | 96.8            | 96.9             | 99.2             | 64.2             | 50.1             | 60.8             | 54.0             |
| Lin-Chang MDA                    | 99.2            | 97.7            | 96.6             | 88.9             | 64.1             | 50.3             | 98.4             | 83.1             |
| JPEG                            | Inter           | Cropping        | PEPo             | JPEG2000         |
| Methods                          | 1/12            | 1/256           | 10%              | 30%              | 85               | 55               | 8                | 5                |
| MPG-EDBTC                        | 98.5            | 97.6            | 92.4             | 79.0             | 88.2             | 72.0             | 84.6             | 70.1             |
| Lin-Chang Lowmean                | 68.4            | 64.0            | 92.2             | 79.4             | 75.9             | 50.4             | 80.0             | 62.3             |
| Lin-Chang Highmean               | 68.6            | 63.9            | 92.4             | 79.7             | 73.7             | 49.8             | 79.9             | 61.3             |
| Lin-Chang MDA                    | 54.9            | 54.0            | 92.3             | 79.1             | 72.4             | 55.6             | 82.8             | 71.3             |