An Adaptive Block Truncation Coding Algorithm Based on Data Hiding

Hengfu Yang and Jianping Yin
School of Computer, National University of Defense Technology, Changsha 410073, China
Department of Information Science and Engineering, Hunan First Normal University, Changsha 410205, China

Abstract: Generally, image compression using Block Truncation Coding (BTC) can’t provide high compression ratio and visual quality. A new adaptive BTC scheme is proposed by combining visual perception of original images and data hiding technique. The texture sensitivity is exploited to classify the types of image sub-block and reduce the bit rate while remaining good quality of reconstructed images. Moreover, the indicator bits of image sub-blocks are hidden into the quantizing levels of those blocks for decoding purpose. The simulation results show that the proposed BTC algorithm can achieve good balance between bit-rate and PSNR values. Especially, it obtained about 4.0 dB higher PSNR at approximate bit-rate.

Key words: Block truncation coding, data hiding, LSB, texture sensitivity

INTRODUCTION

Block truncation coding is a simple and effective method for image compression (Delp and Mitchell, 1979). It was also called the moment-preserving block truncation because it preserves the standard mean and standard deviation of each image block. Less computational complexity provided a widely applications for BTC (Lv and Lu, 2011; Mohammad et al., 2011). To reduce the bit-rate of BTC, various modifications to BTC have been proposed in the past few decades. Lena and Mitchell (1984) presented a simple and fast variant of BTC, named Absolute Moment BTC (AMBTC) that preserves the higher mean and lower mean of a block. In a comparative study, Kumar and Singh (2011) presented a BTC scheme called EBTC which has higher PSNR than BTC and AMBTC (Absolute moment BTC). Kamel et al. (1991) proposed a variable BTC algorithm with optimal threshold. It brought a reduction of the error in the reconstructed images by almost 40%. Wu (2002) presented probability based BTC to reduce the bit plane overhead. Hu (2004) employed two-dimensional prediction technique, the bit map omission technique and the bit map coding with edge patterns to cut down the bit-rate of moment preserving block truncation coding. Han et al. (2008) proposed a BTC based on the vector quantizer for the color image compression which can get high compression ratio and good visual quality. Wang and Chong (2010) proposed an adaptive multi-level BTC. The scheme adaptively selected 2-level or 4-level BTC according to the edge property of blocks. Guo (2010) proposed an Ordered Dither Block Truncation Coding (ODBTC) by using a dither array Look Up Table (LUT). Somasundaram and Vimala (2010) developed an efficient block truncation coding by exploiting the feature of inter-pixel correlation. Choi and Ko (2011) devised a novel DPCM-BTC. The scheme derived a bivariate quadratic function to represent the mean squared error (MSE) between the original block and the block reconstructed in the DPCM (differential pulse code modulation) framework and adopted a near-optimal quantizer to prevent the rapid increase of the quantization error. It improved peak signal-to-noise ratio performance compared with the common DPCM-BTC method without optimization. Natarajan and Rao (2011) proposed two new modified BTC algorithms by using the ratio of moments. It also coded the ratio values and the bitplane to reduce the bit-rates. Vimala et al. (2011) improved adaptive block truncation coding method by exploiting the feature of inter-pixel redundancy and it reduced the bit-rate further while retaining the quality of the reconstructed images.

However, many existing BTC schemes didn’t fully exploiting features of image sub-blocks and can’t obtain good trade-off between low bit-rate and high quality of reconstructed images. To solve this problem, we combine texture masking characteristics of image sub-blocks and Least Significant Bits substitution approach to design an adaptive BTC scheme. It can reach optimal balance between low bit-rate and high PSNR values.

Corresponding Author: Hengfu Yang, School of Computer, National University of Defense Technology, No. 1015, Section 3, Fenglin Road, Yuelu District, Changsha, Hunan 410205, China
Tel: 86-731-8284011 Fax: 86-731-8284011

647
DATA HIDING BASED ADAPTIVE BTC SCHEME

Encoding algorithm for adaptive BTC: In order to get an efficient trade-off between high visual quality and low bit-rate, visual perception of image sub-blocks should be considered during BTC. We mainly exploited the texture sensitivity of image sub-blocks. Generally, highly textured blocks contain more reach structural information than smooth blocks. This means smooth blocks can be encoded with fewer bits than texture blocks during BTC. Denotes the lower mean and higher mean of the current sub-block as \( x_i \) and \( x_h \) during BTC, respectively. For simplicity, the texture compression factor \( \lambda \) of an image sub-block can assumed to be directly proportional to the absolute of lower and higher mean. So \( \lambda \) can be calculated as follows:

\[
\lambda = |x_i - x_h| = x_h - x_i
\]  

(1)

The detailed encoding process of BTC is described as below:

- **Input:** original image \( X \)
- **Output:** BTC code stream \( C \)

**Step 1:** Input the original image \( X \) and divide original image \( X \) into non-overlapping sub-blocks of size \( n \times n \)

**Step 2:** Calculate the mean \( \bar{x} \), lower mean \( x_l \) and higher mean \( x_h \) for the current \( n \times n \) block \( X \)

**Step 3:** Compute the texture factor \( \lambda \) using Eq. 1

**Step 4:** Given a threshold value \( T_i \), if \( \lambda < T_i \), then encode the current \( n \times n \) block \( X \), with the mean \( \bar{x} \), for decoding purpose, binary bit sequence '00' is used as the indicator bits of current block. At the same time, the indicator bits are hidden into the mean \( \bar{x} \) by applying LSB substitution as shown in Eq. 2 and go to step 14 else continue with step 5.

\[
\bar{x} = \bar{x} - \text{mod}(\bar{x}, 4) + 0
\]  

(2)

where mod (a, b) returns a modulo b and the number 0 is the decimal value of the binary bit sequence ‘00’.

**Step 5:** Divide the \( n \times n \) block into four non-overlapping blocks \( X \) with size \( n/2 \times n/2 \)

**Step 6:** Compute the mean \( \bar{x}_i \), lower mean \( x_{l,i} \), higher mean \( x_{h,i} \) and texture factor \( \lambda \) for the current block \( X_i \)

**Step 7:** Determine the second threshold value \( T_2 \), if \( \lambda \geq T_2 \), then encode the current block \( X_i \) with the mean \( \bar{x}_i \). Hide indicator bits ‘01’ into the mean \( \bar{x}_i \) for decoding purpose else go to step 9

**Step 8:** Repeat the steps 6 and 7 until all the four \( n/2 \times n/2 \) sub-blocks and go to step 14

**Step 9:** Further to divide current \( n/2 \times n/2 \) block \( X_i \) into four non-overlapping blocks \( X \) of size \( n/4 \times n/4 \)

**Step 10:** Compute the mean \( \bar{x}_i \), lower mean \( x_{l,i} \), higher mean \( x_{h,i} \) and texture factor \( \lambda \) for current block \( X_i \)

**Step 11:** If \( \lambda \geq T_3 \) then encode the current block \( X_i \) with the mean \( \bar{x}_i \) and embed the indicator bits ‘10’ into the mean \( \bar{x}_i \) using LSB method

**Step 12:** Else encode the current block \( X_i \), with the lower mean \( x_{l,i} \), higher mean \( x_{h,i} \) and bit plane \( B \) and denoted as a trio \( (x_{l,i}, x_{h,i}, B) \), use ‘11’ as the indicator bits for decoding purpose and hide them into the lower mean \( x_{l,i} \) with LSB scheme

**Step 13:** Repeat steps 10-12 until all the four \( n/4 \times n/4 \) sub-blocks and go to step 8

**Step 14:** Go to step 2 until all the image sub-blocks are processed and the BTC code stream \( C \) is generated

Decoding strategy for adaptive BTC: The decoding strategy is very simple, which consists of the following steps:

- **Input:** BTC code stream \( C \)
- **Output:** Reconstructed image \( I \)

**Step 1:** Read BTC code stream \( C \)

**Step 2:** Obtain the quantizing level \( x_i \) or mean value from compressed code stream \( C \)

**Step 3:** Extract the 2 LSBs from the quantizing level \( x_i \) or mean value as the indicator bits

**Step 4:** Reconstruct image blocks with the following strategy

If the indicator bits are ‘00’ then replace the \( n \times n \) block \( X_i \) with the block mean \( \bar{x}_i \).

If the indicator bits are ‘01’ then recover the \( n/2 \times n/2 \) block \( X_i \) with the block mean \( \bar{x}_i \).

If the indicator bits are ‘10’ generate the \( n/4 \times n/4 \) block \( X_i \) with only the block mean \( \bar{x}_i \).

If the indicator bits are ‘11’ then reconstruct the \( n/4 \times n/4 \) block \( X_i \) with the trio \( (x_{l,i}, x_{h,i}, B) \).

**Step 5:** Repeat step 2-step 4 until all the BTC code bits are processed and finally the reconstructed image \( I \) is generated
EXPERIMENTS AND ANALYSIS

The proposed algorithm has been implemented in MATLAB-7.0. In our experiments, some standard images of size 256×256 are used to test our BTC algorithm. Figure 1a-d show four test images as examples and the corresponding BTC compressed images are shown in Fig. 2 under n = 8 and threshold T1, T2, T3, value 0.05, 0.05, 0.10, respectively.

In Fig. 2a-d, it is observed that compressed images generated by the proposed BTC method have satisfactory visual quality with PSNR values over 30.56 dB.

We have applied our image compression schemes with different combination of threshold values and image sub-block size. Table 1 lists the tested results of the proposed scheme under different threshold values for n = 8. From Table 1, we can know that our BTC scheme can obtain high PSNR values and low bit-rate. Especially it achieves 27.9527 dB PSNR at 0.7912 bpp and 31.3547 dB at 1.5029 bpp on average. Moreover, the reconstructed images have high weighted PSNR values over 39.24 dB at low bitrate with 0.7912 bpp.

Finally, The PSNR values and the bit rate obtained with the proposed scheme are compared with that of the

![Fig. 2(a-d): BTC compressed images generated by our scheme (a) Opera, (b) Lena, (c) Boast and (d) Peppers](image)

![Fig. 3: PSNR values and bit-rate obtained with different BTC schemes](image)

---

**Table 1: PSNR and bpp values under different threshold values for n = 8**

<table>
<thead>
<tr>
<th>Images</th>
<th>(T1, T2, T3): (0.05, 0.05, 0.1)</th>
<th>(T1, T2, T3): (0.05, 0.1, 0.15)</th>
<th>(T1, T2, T3): (0.1, 0.15)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>1.1887 32.1072 43.5769 0.9603 31.2401 42.4129 0.8294 30.3502 42.1575 0.5927 28.5512 38.7000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Boats</td>
<td>1.2009 30.9116 45.9065 1.4536 29.5169 42.7564 1.2745 28.7453 42.4548 0.9526 27.1932 38.6752</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peppers</td>
<td>1.2923 31.83 44.2545 1.0051 30.5765 42.1052 0.8615 29.9815 42.2519 0.6891 28.3843 38.6644</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cameraman</td>
<td>1.615 30.598 46.2638 1.3985 29.9111 44.6006 1.2187 28.0089 44.2297 0.9305 27.6819 40.8966</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Airplane</td>
<td>1.4115 32.5517 44.745 1.2214 31.6064 43.3474 1.0843 30.6703 43.6545 0.9127 29.2231 40.2901</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Couple</td>
<td>0.9852 33.2661 44.0233 0.6912 31.7544 41.825 0.6185 31.1980 41.6652 0.4066 29.2806 38.0641</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Opera</td>
<td>1.3344 32.8466 45.1001 0.9655 31.1875 42.4403 0.8488 30.4193 42.1880 0.5789 28.5483 38.2871</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>1.5029 31.3547 44.9997 1.2026 30.3097 43.0438 1.0535 29.5458 42.7735 0.7912 27.9527 39.2441</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
existing EBTC algorithm (Kumar and Singh, 2011) and the comparison results are shown in Fig. 3. Figure 3 show that the PSNR value achieved by the presented algorithm is about 4.0 dB higher than that by the EBTC method when the bit-rate is roughly the same.

**CONCLUSION**

BTC is one of simple and fast image compression algorithms but common BTC schemes have high bit-rate. In this paper an adaptive BTC algorithm is developed by fully considering texture masking characteristics of original image blocks. During BTC, the texture sensitivity is exploited to recognize the image sub-blocks in terms of perception feature. At the same time, the LSB substitution method is employed to hide the indicator bits of each image sub-block. The proposed BTC algorithm outperforms other existing BTC schemes in terms of bit-rate and PSNR values. It can be applied to real-time image communications via the Internet.

**ACKNOWLEDGMENT**

This study was supported by National Natural Science Foundation of China (61073191, 611710287), Human Provincial Natural Science Foundation of China (10J16090), Scientific Research Fund of Human Provincial Science and Technology Department, China (2011GK3140, 2010GK3049, 2011GK3139), Science and Technology Innovative Research Team in Higher Educational Institutions of Hunan Province ([2010]212).

**REFERENCES**


