Chapter 3

Improvements on BTC

3.1 Introduction

Many applications require image and video communication processing capability and low-cost and high-performance multimedia systems. Several methods are proposed to meet these requirements. One such solution is System on a Chip (SoC) for reducing the system cost and better cost-effective performance due to its high integration of multiple processing units [63]. To store and transmit the images in SoC, image compression is needed. To realize a fast coding on SoC, a simple compression method such as BTC is demanded.

Sherif M. Saif, Hazem M. Abbas, and Salwa M. Nassar [52] have presented a Field Programmable Gate Array (FPGA) implementation for video compression using BTC image compression technique. The implementation exploits the inherent parallelism of the BTC algorithm to provide efficient algorithm-to-architecture mapping. The Xilinx VirrexE BTC [52] implementation has shown to provide about 205 x 106 of pixels per second which is about 3000 times faster than an Intel Pentium III 550 MHz processor.

Coding scheme with low complexity enables a fast codec system which flexibly cooperates with other real time application as required by the users. BTC and its existing variants are promising schemes for realizing a fast coding. But the bit rate of the original BTC is relatively high when compared to other still image compression techniques such as JPEG [41] or JPEG 2000.
Hence improvements on BTC have been proposed to further reduce the bit rate.

Arce and Gallagher [3] proposed a median filter roots method to code the bit plane and the bit rate was reduced to about 1.38 bpp. Udipikar and Raina [60] introduced a BTC image compression using Vector Quantization (VQ) and the bit rate achieved was in the range of 1.0 - 1.5 bpp. Zeng and Neuvo [68] proposed two BTC methods with VQ schemes. Chun-Hung Kuo et. al. [15] also presented a compression scheme based on BTC method with VQ scheme. The hybrid BTC / VQ techniques reduced the bit rate, but they need more computation for code book generation.

Ramana and Eswaran [47] developed a method by extending the BTC method. In their work, few bits were dropped in the bit plane thereby reducing the bit rate. In the reconstruction, they predict the dropped bit values by using the adjacent values. In that method they have achieved a compression of 1.5 bpp. Chung-Woei Chao et. al [14] presented a modified block truncation coding algorithm which used block clustering scheme and predefined binary edge patterns. Yung-Gi Wu [67] proposed a probability based block truncation image bit plane coding. Yu-Chen Hu [66] presented a modified BTC with predictive technique and bit plane coding with edge pattern.

Thus the improvements on BTC continue to reduce the bit rate and computational complexity by keeping the image quality to an acceptable limit. The improvements made in few key areas are discussed in detail in this chapter. The improvements on BTC and AMBTC are divided into three sections as i) Variants of BTC, ii) BTC with other popular methods and iii) multistage compression schemes based on BTC. In the following sections, the three categories are discussed.
3.2 Variants of BTC

The greatest deficiency of BTC is its high bit rate as compared to other coding schemes like vector quantization [37], DCT [41] and wavelet [44,29]. Several modifications and improvements of the basic BTC have been proposed by the research community. Their main aim is to reduce the bit rate of BTC. The BTC decoding has the statistical moments and the bit plane. Here the bit plane occupies 16 bits and assigning 8 bits each to the two statistical moments results in a data rate of 2 bpp. Delp and Mitchell [19] made a modification in BTC to reduce the bit rate less than 2 bits per pixel. The modification was proposed to use less than 8 bits to represent the values of mean and standard deviation. Experimental evidence [19] has indicated that coding the value of the mean with 6 bits and the value of the standard deviation with 4 bits introduces only a few perceivable errors. The bit rate after this modification is 1.63 bpp.

Generalized BTC

Halverson, Griswold, and Wise [23] developed a generalized BTC algorithm which preserves the nth and 2nth moments of each local block. That is, while the original method preserves the first and second moments of each block, the generalized version can preserve the second and fourth moments or the third and sixth moments, etc. While these alternative moment pairs can maintain or even improve the image quality, the use of very high moments is not recommended because of quantization overflow problems which reduce image quality. This modification improves the quality of the reconstructed image but the bpp is not lower than BTC.
Absolute Moment BTC

Absolute Moment BTC (AMBTC) is another variant of BTC proposed by Lema and Mitchell [30] which was already discussed in chapter 2.2. This method also improves the quality and reduces the computational complexity. But the bit rate remains the same as BTC.

Prediction Technique

A simple prediction technique proposed by Mitchell and Delp [34] reduces the volume of the bit plane by encoding only half of the bits as shown in the Figure 3.1. The remaining bits are predicted on the basis of coded ones by a set of logical expressions given in Fig. 3.1.

This method reduces the bit rate of the bit plane to 0.5 bpp. But this method naturally causes more distortion to the image. The result of the prediction technique is the weakest in the blocks with a large contrast. Therefore an adaptive bit plane coding method by [34] uses the prediction technique only when the standard deviation is small enough, say $1 \leq \sigma \leq 4$. Using this method the bit rate is reduced further for the images where the image blocks are smooth

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
</tr>
<tr>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
</tr>
<tr>
<td>M</td>
<td>N</td>
<td>O</td>
<td>P</td>
</tr>
</tbody>
</table>

$B = (A \text{ and } D) \text{ or } (F \text{ and } J)$
$C = (A \text{ and } D) \text{ or } (G \text{ and } K)$
$E = (A \text{ and } M) \text{ or } (F \text{ and } G)$
$H = (D \text{ and } P) \text{ or } (F \text{ and } G)$
$I = (A \text{ and } M) \text{ or } (J \text{ and } K)$
$L = (D \text{ and } P) \text{ or } (J \text{ and } K)$
$N = (M \text{ and } F) \text{ or } (F \text{ and } J)$
$O = (M \text{ and } F) \text{ or } (G \text{ and } K)$

Fig. 3.1 Bold faced bits are stored and the remaining are dropped in the bit plane and Prediction formula used to recover the dropped bits
Predictive **Bit plane Coding**

In 1995, Ramana and Eswaran [47] proposed a simple predictive scheme for BTC bit plane coding. It is similar to the previous method of Mitchell and Delp [34] but with different bit pattern. The idea of this scheme is based on the observation that a high correlation exists among neighboring pixels in most digital images. When a given image is compressed by the BTC method, the bit planes have a high degree of similarity among its adjacent bits. In this scheme, half of the bits in the bit plane are transmitted to the decoder while the other half is dropped. In the reconstruction the dropped bits are determined by prediction. Take the bit plane of 4X4 bits shown in Fig. 3.2. Only the alternate bits, i.e., the dark ones, are transmitted to the decoder. In the reconstruction the dependent bits, i.e., the light ones, are obtained by predicting the values of the surrounding dark bits.

![Fig. 3.2 4 x 4 Bit plane](image)

The prediction is performed according to the following rules:

- **B** = 1 if two or more of the surrounding bits, viz., A, F, and C are equal to 1.
- **D** = C.
- **E** = 1 if two or more of the surrounding bits, viz., A, F, and I are equal to 1.
- **G** = 1 if two or more of the surrounding bits, viz., F, K, H, and C are equal to 1.
- **J** = 1 if two or more of the surrounding bits, viz., F, I, K, and N are equal to 1.
- **L** = 1 if two or more of the surrounding bits, viz., H, K, and P are equal to 1.
- **M** = N.
- **O** = 1 if two or more of the surrounding bits, viz., N, K, and P are equal to 1.
Based on the prediction rules above, only 8 bits in the bit plane are transmitted to the decoder. The performance of this scheme is highly dependent on the block activities of image blocks. In fact, this scheme works well for smooth blocks. But it is not quite suitable for processing the complex image blocks because a lower degree of similarity exists within complex blocks.

Chapter 4 of this thesis proposes a modified compression method based on BTC.

### 3.3 BTC with other Methods

The BTC algorithm for image compression has the advantages of low computation load and less memory requirement. Even though it is easy to implement when compared to vector quantization and transform coding, it has a high bit rate. In order to reduce the bit rate of BTC coding, BTC is combined with other low bit rate methods like vector quantization and DCT. This section discusses how the combination BTC method with other image compression methods reduces the bit rate.

**BTC with Vector quantization**

A Vector quantization (VQ) is a system for mapping a sequence of discrete vectors into a finite number of representative vectors [22]. The aim of vector quantization is to reduce the number of bits necessary to represent the input vectors. VQ and BTC have been used for many years for compressing digital images for storage and transmission. BTC has the advantage of preserving the edge pattern. VQ has the ability of providing high compression ratios. Therefore an image compression technique that combines BTC and VQ produces still better performance.
The use of VQ for coding the bit plane of BTC was first proposed by Udipikar and Raina [60] in 1987. They used a variant of AMBTC that stores a compressed block in the form lower means $\bar{X}_L$ higher Mean $\bar{X}_H$ and the bit plane $B$. The decoder replaces each 0 in the bit plane by $\bar{X}_L$ and each 1 bit by $\bar{X}_H$

$$\bar{X} = \frac{1}{m} \sum_{i=1}^{m} x(i)$$  \hspace{1cm} (3.1)

$$\bar{X}_L = \frac{1}{m - q} \sum_{i=1 \& x(i) < \bar{X}}^{m} x(i)$$  \hspace{1cm} (3.2)

$$\bar{X}_H = \bar{X}_L + \frac{1}{q} (\bar{X} - \bar{X}_L)$$  \hspace{1cm} (3.3)

where $m$ is number of pixels in a block $q$ is number of pixels having value higher than $\bar{X}$.

By applying vector quantization to the pair $(\bar{X}_L, \bar{X}_H)$, the bit rate can be reduced to 8 bits for $(\bar{X}_L, \bar{X}_H)$ with the cost of increased MSE value. Statistical properties of the output of BTC coder are found to be useful in their vector quantization. VQ’s designed using these properties improves the coding efficiency of the BTC coder. Suitable training sequence-based VQ’s for a typical monochrome image reduces the bit rate up to 1.0 bpp. The method however, does not consider inter block correlations i.e. the dependencies between neighboring blocks.

Another approach was proposed by Alcaim and Oliveira in 1992. [2]. They encode $\bar{X}$ and $\sigma$ separately, making two sub sample images one from the $x$ values and another from the $\sigma$ values of the blocks. These sub sample images are decomposed into 2X2 blocks which are then coded by VQ. In this way, one can reduce the interlock and intrablock redundancy.
Sherif A. Mohamed and Moustafa M. Fahmy [51] proposed a compression method using VQ and BTC in 1995 called VQ-BTC. In VQ-BTC method, a low-detailed block is encoded using VQ. For a high detail block, a modification of BTC is used to determine the locations of the relatively lighter and relatively darker pixels inside the block and VQ is then used to encode each. If the difference between higher Mean $\overline{X}_H$ and lower mean $\overline{X}_L$ is less than a threshold then the block is considered as low detail block otherwise it is considered as high detail block.

The low detail blocks can be coded efficiently using one moderate size codebook. In the high detail blocks, the pattern of each 4 x 4 edge block is specified by assigning a binary “1” at the locations of the pixels whose intensity is closer to the higher mean than the lower mean, otherwise a binary “0” is assigned. The high detail block is then divided into two vectors. One vector represents the light pixels inside the block and the other vector represents the dark pixels. The light (dark) vector is then encoded within a codebook that is trained only with light (dark) vectors. Thus the VQ-BTC method had both BTC and VQ technique. This method achieves a bit rate of 0.715 bpp on a standard Lena image.

Even though the above method reduces the bit rate of BTC to 0.715 bpp, it has high encoding complexity and needs large amount of memory to store 31 code books at both the encoder and decoder. Chou-Chen Wang, Chin-hsing Chen and I-Hong Chen [13] have developed a modified VQ-BTC algorithm for image compression which achieves a performance close to that of VQ-BTC, but needs only three code books and requires less computation time than VQ-BTC.

The hybrid BTC / VQ techniques reduce the bit rate, but they need more computation for encoding the code book generation. The main advantage of BTC is low complexity computational compression, but VQ has
high encoding complexity. Therefore the hybrid BTCA/Q techniques are not attractive in real time low computational applications.

**BTC with Arithmetic coding**

Arithmetic coding is known to be an optimal coding method. Moreover, it is extremely suitable for dynamic modeling, since there is no actual code table like in Huffman coding to be updated. The deficiency of Huffman coding is emphasized in the case of binary images. Binary images having the probability distribution of 0.99 for a white pixel and 0.01 for a black pixel may be considered. The entropy of the white pixel is $-\log_{2} 0.99 = 0.015$ bits. However, in Huffman coding, the code length is bound to be 1 bit per symbol at minimum. In fact, the only possible code for binary alphabet would be one bit for each symbol, thus no compression would be possible.

The basic idea of arithmetic coding is to represent the entire input file as a small interval between the range \([0,1]\). The coding is done by taking $-\log_{2} A$ bits, where $A$ is the length of the interval.

QM-coder is an implementation of arithmetic coding which has been specially tailored for binary data. One important factor considered in its design is the speed.

Franti and Nevalanien\[40\] have proposed the method of BTC with arithmetic coding. In their method the bit plane is processed in a row major order from left to right. A high degree Markov model is applied for each pixel. The value of a pixel is predicted by the 7 bit context template and then coded by arithmetic coding according to its probability. The QM coder is used as the arithmetic coding component.

The size and shape of the template have major effect on the coding efficiency when compressing binary images and better results can be
achieved with larger templates. However this does not necessarily apply to the BTC. The problem here is the block wise quantization according to the mean of each individual block. This destroys the neighboring bits belonging to two different blocks. The benefit of the entropy coding remains relatively small.

This method is fine tuned by hierarchical decomposition. The hierarchical decomposition causes some problems in the practical implementation, but the use of variable block size is important both for the image and bit rate. The results of the overall compression out-performed the BTC variant with VQ. But the benefit of the above is relatively small. For a “Lena” image approximately 10% can be saved.

Another method proposed by Yung-Gi Wu [67] in 2002 saves 26% overhead for the “Lena” image by the exploiting of probability distribution of each bit plane pattern. In this method, each bit plane pattern is expressed as a state. For 4X4 pixel block of AMBTC, there are 216 possible states. However, the occurrence of each state is different. This property can be used to reduce the overhead of bit plane coding in this method. Fig. 3.3a and 3.3b present statistical data of the occurrence frequency of each state.

The occurrence of each state is different. An intuitive method is to code the most frequently occurring states in fewer bits and least frequently occurring states in more bits so as to reduce the bit rate in transmission or storage. This method calculates the state occurrences from the training set to achieve the suitable statistical model at first. The frequency of occurrence of each state is shown in Figs. 3.3a and 3.3b resulting from seven different images in their experiment.
Fig. 3.3 State of occurrences a) Lower range b) Higher range

The complete algorithm of this method includes two major procedures. The first stage calculates the occurrences from the training images. Then, the method determines the probability of each state. The second stage involves encoding each state by arithmetic coding according to the probability value and recording the codes in a table. In the practical coding, however, encoding the bit plane serves to retrieve the codes from the code table.

In his experiments, he has used seven different images to acquire the occurrence data shown in Fig.3.3 in the training stage to determine the statistical data for the arithmetic coder. Then, arithmetic coding has generated
a code table for all the states. The bits used to encode the five test images are provided in the Table 3.1.

Table 3.1 Bpp for bit plane of BTC used by Yung-Gi Wu [67] for different Images

<table>
<thead>
<tr>
<th>Test Image</th>
<th>Bit Plane Overhead (bpp)</th>
<th>Saved percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>0.734</td>
<td>26.4</td>
</tr>
<tr>
<td>Lenna</td>
<td>0.712</td>
<td>28.8</td>
</tr>
<tr>
<td>F16</td>
<td>0.701</td>
<td>29.9</td>
</tr>
<tr>
<td>F18</td>
<td>0.661</td>
<td>33.9</td>
</tr>
<tr>
<td>Children</td>
<td>0.744</td>
<td>25.5</td>
</tr>
</tbody>
</table>

The range of saved percentages is from 34% to 25% with the same reconstructed quality. This method out-performs the result published in Ref. [16], which gives a 12% savings for the “Lena” image. In addition, the speed is faster than that achieved in Ref. [40] due to the fact that this method does not require complicated computation. The time to yield statistical data for arithmetic coder is huge. However, it does greatly increase the time for practical compression. This method uses only the statistical results. Reference [47] describes a bit plane coding strategy to save the bit plane burden as well. However, it scarifies reconstructed image quality, while the said method does not have this drawback.

BTC is a simple and fast algorithm for image compression. But the hitherto discussed methods based on arithmetic coding add complexity to the algorithm.

Filtering Technique

Median filtering is a signal processing technique where the input signal is processed by replacing each input signal with the median of the original
values within a given window. The signal can be filtered over and over again until it reaches its roots i.e. it is not affected by the filtering anymore. The root signal set includes all the signals that are roots. It has been shown that the block signals have a very high probability of belonging to the root signal set.

Arce and Gallagher [3] proposed a 1-D median filtering for the BTC bit plane. The window is set to three pixels including the current pixel and two of its neighboring pixels in the same row. For the border pixels, the outer pixels are presumed to be of the same intensity as the border pixel. Each row of a block is median filtered to its roots. In the case of 4x4 pixel blocks, roots can be reached by a single filtering pass. Compression is achieved because the number of possible patterns has decreased from 16 down to 10, as shown in Fig. 3.4 and Table 3.2.

All the combinations that include isolated bits have been eliminated. To realize the benefits of filtering, only the eight common combinations (marked *) shown in Table 3.2 are allowed to occur. The remaining patterns can now be coded by 3 bits each, giving a total of 0.75 bits per pixel for the bit plane. The scheme is extremely simple and can be implemented very efficiently by a look-up-table (LUT).
This method has been improved further so that the vertical correlations are also considered. Here the first row of each block is coded by 3 bits as proposed above, but for the other three rows only 2 bits are used. This is done by reducing the number of transitions to four discarding the least likely patterns and taking into consideration the pattern of the previous row. This scheme reduces the bit rate of the bit plane to $(3+2+2+2)/16 = 0.56$ bits per pixel. Thus median filter with BTC reduces the bit per pixel compared to the basic BTC.

In 2000, Zhe-Minh Lu et al. [69] proposed an algorithm with median filtering to lower the bit rate of the AMBTC algorithm. The main idea of this algorithm is to encode to the higher mean and lower mean together as a pair by simple look-up-table methods and to encode the bit plane using the root signal set of 1-D median filtering. The experimental results of this method showed that it had lower bit rate with better performance than AMBTC.

<table>
<thead>
<tr>
<th>Root</th>
<th>Pattern</th>
<th>Experimental frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0 0 0 0</td>
<td>0.20</td>
</tr>
<tr>
<td>2</td>
<td>0 0 0 1</td>
<td>0.08</td>
</tr>
<tr>
<td>3</td>
<td>0 0 1 1</td>
<td>0.11</td>
</tr>
<tr>
<td>4</td>
<td>0 1 1 0</td>
<td>0.02</td>
</tr>
<tr>
<td>5</td>
<td>0 1 1 1</td>
<td>0.07</td>
</tr>
<tr>
<td>6</td>
<td>1 0 0 0</td>
<td>0.09</td>
</tr>
<tr>
<td>7</td>
<td>1 0 0 1</td>
<td>0.02</td>
</tr>
<tr>
<td>8</td>
<td>1 1 0 0</td>
<td>0.12</td>
</tr>
<tr>
<td>9</td>
<td>1 1 1 0</td>
<td>0.09</td>
</tr>
<tr>
<td>10</td>
<td>1 1 1 1</td>
<td>0.20</td>
</tr>
</tbody>
</table>
Discrete Cosine Transform

A hybrid of BTC and DCT is proposed by Delp and Mitchell [19]. First the given compressed image by BTC is compressed by taking the two-dimensional discrete cosine transform over 16x16 pixel blocks. Only the eight non-dc coefficients in the low frequency section of each block are retained. A difference image is constructed by subtracting the transform coded image from the original. The base BTC algorithm is then applied to this difference image. Both BTC and transform codes are sent to the decoder with bit rate of $1.63 \times 10^2 = 1.88$ bpp.

The hybrid method has a certain advantage compared to basic BTC. While BTC preserves edges, they tend to be ragged (the staircase effect). On the other hand, transform coding usually produces smooth visual quality of the image. At the same time the MSE value is reported to decrease.

Yiyan Wu and David C. Coll [62] presented a hybrid BTC-VQ-DCT image coding algorithm which combines the simple computation and edge preservation properties of BTC and the high fidelity and high-compression ratio of adaptive DCT with the high compression ratio and good subjective performance of VQ, and which may be implemented with significantly lower coding delays than either VQ or DCT alone. BTC-VQ-DCT coding has three phases.

**Phase I:** BTC coding of the input image compresses blocks of the original $N \times N$ image into a “bit-map” and two reduced-sized gray-level images which are composed of the “high-mean” and “low-mean” pixel values of each BTC block. The primary compression ratio of Phase I is 4:1 for 4 x 4 blocks.

**Phase II:** The VQ is implemented on the bit-map to further remove the intrablock redundancy. A universal 64 K codebook is built to quickly vector quantize the bit-map through the use of a LUT. The compression ratio in this phase is between 1.6:1 and 2.67:1.
**Phase III:** An adaptive DCT is implemented to compress the $N/4$ by $N/4$ high and low mean sub images which remove both inter and intra block correlations among high and low mean pixel values. Residual error feedback is used to reduce the MSE and bit rate. The compression ratio is about 3:1.

The overall computational complexity of BTC-VQ-DCT coding is much less than either DCT or VQ, while the fidelity performance is competitive. The algorithm has strong edge-preserving ability because of the implementation of BTC as pre-compress decimation. The total compression ratio is about 10:1.

**Bit Plane Coding Using Frequent Patterns**

Panos Nasiopoulos, et al. [39] introduced a simple scheme to reduce the storage cost of the BTC bit plane. They design a set of image-independent edge patterns to represent the bit plane. The motivation of this scheme is that the probability of each possible bit plane is not equal. If we can find a set of representative bit planes, a great deal of storage cost can be saved.

For example, if the image to be compressed is partitioned into a set of non overlapped 4X4 image blocks, the size of each bit plane is 16 bits. There are a total of $2^{16}$ combinations of BTC bit planes. However, the majority of the $2^{16}$ patterns are rarely encountered in practice. The reduction of bit rate can be achieved by selecting only the frequent patterns to represent all possible bit planes. Whenever a rare pattern is encountered, it is approximated to the closest frequent pattern using the Hamming distance measurement.

In this scheme, the authors suggested that 128 visual patterns are used to encode the bit plane. The result was that only 7 bits were needed to transmit the index of the closest frequent pattern. The idea of using frequent
patterns to reduce the storage and transmission cost is very simple. However, the selection of a set of representative frequent patterns is the most important issue of this scheme. This is so because the image quality of the scheme heavily relies on the performance of the selected frequent patterns.

Compression ratios as low as 0.5-0.8 bite per pixel are possible. The high-quality edge preservation characteristics at these compression ratios are still maintained, however a post filter is needed to improve the quality of the smooth and textured regions.

The methods discussed above have aimed at reducing the bit rate of BTC compression. The methods BTC with VQ and DCT have reduced the bit rate with little distortion but failed to achieve the low computational complexity of BTC. Arithmetic coding methods with BTC also achieve low bpp compared to BTC but again these methods also add complexity to the BTC method. The next chapter discusses BTC in multi stage compression schemes which are aimed at obtaining high quality compressed images with high compression ratio and low computational complexity.

3.4 Multistage image compression scheme based on BTC

BTC method combined with other compression methods has the advantage of BTC and the efficiency of other methods. So we can get low bit rate, acceptable quality and also the low computational complexity. BTC method is block wise compression technique, so according to the nature of the current block, one can adopt different methods with BTC. It can be called as adaptive or multistage compression schemes. Hence this section discusses how the BTC method reduces bit rate with other image compression methods in multistage compression schemes.
BTC with Hamming codes and DPCM

Chih Shoung Huang and Yinyi Lin in 1997 [12] have presented a hybrid block truncation coding (BTC). In the hybrid BTC, a universal codebook using Hamming codes and a differential pulse code modulation (DPCM) are employed, respectively, to the bit plane and the side information of BTC to reduce coding rate. Experimental results reveal that the performance of this hybrid BTC algorithm is only slightly worse than that of the hybrid BTC using vector quantization (VQ) techniques, but with much lower computational or hardware complexity.

In this scheme, a universal book using a Hamming code is provided to reduce the rate of the bit plane of BTC. Hamming codes are originally used as data transmission codes to combat random errors occurring in communications channel. A Hamming code is a perfect code (i.e., can correct exactly single error) with the following parameters. (m > 3)

- Code length: \( n = 2^m - 1 \)
- Number of information digits: \( k = 2^m - m - 1 \)
- Number of parity check digits: \( n - k = m \)
- Error correcting capability: \( t = 1 \) (i.e., minimum Hamming distance \( d_{\text{min}} = 3 \))

When a \((n,K)\) Hamming code is used as a data compression code and applied to the bit plane of BTC, a data sequence of length \( n \) in the bit plane of BTC is represented by (or encoded to) an index of length \( K \). The decoder and encoder of the Hamming transmission code are used as an encoder and a decoder of the code, respectively, when it is used as a compression code. As a result, no learning procedure and no codebook are required during encoding and decoding since the Hamming codes are linear codes. A compression ratio of \((2^m-1)/2^{m-rm-1}\) can be achieved and the bit error in each data block is at the most one bit since a Hamming code is a perfect code. As demonstrated, the performance of using Hamming codes as a universal
codebook is only slightly inferior to that of using block scan VQ technique with a comparable code rate.

The experimental results show that the hybrid BTC-Hamming-DPCM algorithm has only 0.97 dB degradation compared to the BTC-VQ algorithm. However, this hybrid BTC algorithm has much lower computational or hardware complexity.

BTC with Mean pair codebook and Visual Pattern

In 2000, LU Zhe -ming et al. [33] have developed an efficient BTC image compression scheme with mean pair codebook and visual patterns. This scheme presented a simple look-up-table method that codes the transformed mean pair of the higher mean and the lower mean. In order to reduce the number of bits required to code the bit plane, they have introduced 24 visual patterns.

The encoding process is done as follows:

1. Input a block x and compute the mean x, the lower mean ‘a’ and the higher mean ‘b’ for it
2. if b - a > 10 then the block x is considered as a high detail block and go to step 4. Otherwise x is a low detail block and go to step 3.
3. x is encoded using a 12-bit string whose first four bits are “1111” followed by the 8 bit string of the mean value x and the bit string is transmitted to the decoder by the channel. Go to step 5.
4. First calculate the transformed mean pair A and B to look up the corresponding bit string in the codebook. It needs 14 bits to encode the mean pair. Secondly, compute the bit plane BP for block x to find the best match visual pattern and use its index which needs five bits in
24 patterns to encode the BP. Finally transmit the bit string of the transformed mean pair followed by the index of the bit plane.

5. Go to step 1 to encode the next block.

This method is a hybrid method of BTC with look up table for mean pair and 24 visual patterns for bit plane. This method totally needs 19 bits to code a block where BTC needs 32 bits for a block. So the bit rate is reduced from 2.0 bpp to a bit rate less than 1.0 bpp. At the time of decoding it has to look for the mean pair in the look up table. The results of this method show that it has lower bit rate than the AMBTC algorithm but with extra distortion in image quality.

**BTC with search order and prediction**

In 2003 Yu Chen Hu [64] introduced a novel image compression scheme based on BTC. He refers BTC as Moment Preserving BTC (MPBTC). To reduce the bit rate of the traditional BTC scheme, the block search order coding (BSOC) technique is employed to exploit the similarity among neighboring image blocks. In addition, smooth blocks and complex blocks are processed using different methods. Experimental results show that this scheme provides good image quality at a low bit rate.

The flow chart of the BSOC scheme is shown in Fig.3.5. The motivation of the scheme is based on the observation that neighboring image blocks tend to be quite similar in most digital images.
The goal of BSOC is to find a suitable neighbouring encoded block to represent the current block \( x \) under process. Since the image blocks are processed sequentially in the order of left-to-right and top-to-bottom using MPBTC, one of these already encoded image blocks can be used to encode the current processing block \( x \) if it is quite similar to \( x \). The block searching order of BSOC is shown in Fig. 3.6

![Block search order of current block in BSOC](image-url)
By choosing suitable searching range of the encoded neighbors and the controlling threshold \( T_{hsbd} \), the block similarity among neighboring image blocks can be fully used in BSOC. Here, the squared Euclidean distance [61] is used in BSOC to measure the degree of similarity between two given image blocks.

If a similar encoded neighbor of block \( x \) is found in the predefined searching range, it will be encoded by this similar neighbor. Otherwise, the block characteristic of \( x \) is determined. Then the absolute difference can be employed between these two quantization levels \( a \) and \( b \), i.e. \( |a - b| \) of \( x \) generated by BTC as the controlling factor to determine the block characteristic. By choosing a suitable threshold \( T_{habd} \), the block characteristic of \( x \) can be determined. If \( x \) is classified as a smooth block, \( x \) will be encoded by its block mean. Otherwise, \( x \) will be encoded by an improved version of BTC, called MPBTC64, by coding block mean and variance with 6 and 4 bits, respectively.

Thus three different encoding methods are used in the this scheme. First, one given block may be encoded by BSOC. Secondly, it may be encoded by its block mean. Thirdly, it may be compressed by MPBTC64. This scheme provides higher image quality at a lower bit rate than the other similar schemes. The experimental results show that this scheme provides a good image quality at a low bit rate. It consumes very little computational cost compared to that of VQ and requires no extra codebook. But at the time of encoding, it has to find the near similar block using BSOC technique thereby increasing the encoding time compared to BTC.

BTC with Quad tree segmentation and 64 edge patterns

In 2003 YU Chen hu [65] introduced an image compression scheme based on absolute moment block truncation coding. The quad tree
segmentation technique is employed in this scheme to exploit the variable block-sized segmentation. In addition, the concept of using visual patterns to reduce the storage cost of the BTC bit plane is incorporated into this scheme. Finally, the bit plane omission technique is employed if these two quantization levels in each image block are quite similar. From the experimental results, it is shown that this scheme requires a very little computational complexity. Besides, a good image quality of the compressed image is obtained at a lower bit rate. In other words, this scheme indeed provides a good approach to compress digital images with little computational cost at a low bit rate.

This section deals with the quad tree-segmentation AMBTC scheme with bit plane omission and bit plane coding using edge patterns. It is named as AMBTC-QBOC. In the quad tree segmentation process, the given image is first partitioned into a set of non-overlapped image blocks of size 16X16 pixels. Then, the block mean and the AMBTC quantization levels ‘a’ and ‘b’ are calculated. In the quad tree segmentation process, an absolute distance between these two quantization levels is used to determine the block characteristic. It assumes that one given image block is a complex block if a large absolute distance between quantization levels a and b is found. The same statistical measurement is used in the successive quad tree segmentation process for image blocks of sizes 8x8 pixel and 4x4 pixel. In the quad tree segmentation process, two predefined thresholds \( \beta \) and \( \phi \) are used to control whether each block is to be further subdivided for 16x16 pixel blocks and 8x8 pixel blocks, respectively. In addition, another predefined threshold \( s \) is utilized to determine the block characteristic for 4x4 pixel image blocks.

After the quad tree segmentation process, the variable sized blocks will be compressed according to the following rules. For smooth 16X16 pixel and 8X8 pixel blocks, they are encoded by their block mean. For smooth blocks of 4X4 pixels, the bit plane omission technique is employed to encode. For the
complex blocks of 4×4 pixels, they are encoded by AMBTC with bit plane coding using 64 edge patterns. These 64 edge patterns are shown in Fig. 3.7.

![The 64 edge patterns of 4 X 4](image)

Fig. 3.7 The 64 edge patterns of 4 X 4

The Experimental results indicated that AMBTC-QBOC achieves good image quality at a low bit rate while requiring a very low computational cost. By choosing adequate threshold values, this scheme achieves good image quality while requiring low computational complexity at a low bit rate. It solves the major drawback of the BTC scheme: the high bit rate requirement. In other words, the major contribution of this scheme is that it puts the BTC scheme in practice at a low bit rate.

Yu Chen Hu [65] has also compared this scheme with JPEG 2000. Although the image quality of the proposed scheme by him is worse than that of JPEG 2000 or some other similar schemes at the same bit rate, his scheme is guaranteed to be the most effective in terms of the execution time. Therefore, this scheme provides a quality approach to improve the coding performance of the conventional BTC schemes at low bit rate.
One of our schemes which is presented in chapter 5 is based on this scheme.

**BTC with predictive and Bit map coding**

In 2004, Yu Chen Hu[66] proposed a novel image compression scheme based on the block truncation coding (BTC) scheme. In his scheme, three techniques are employed to cut down the bit rate of moment preserving block truncation coding. They are two-dimensional prediction technique, the bit map omission technique, and bit map coding using edge patterns. The experimental results of this scheme not only achieves good image quality at low bit rate, but also requires little computational cost in the encoding/decoding procedures. In other words, this scheme indeed improves the performance of the BTC scheme and it is quite suitable for multimedia applications with low computational cost requirement.

In his scheme, each image to be compressed is first partitioned into a set of nonoverlapped image blocks of size 4X4 pixels. Then each image block is sequentially encoded in a left-to-right and then top-to-bottom order. In other words, the image blocks to the left and the image block directly above current encoding block x should have been encoded. Here, / and \ denote the encoded blocks to the left of x and directly above x respectively. Figure 3.8 shows the relative position of the encoding block x, the block immediately to its left and the block directly above it. Every image block has an adjacent left block and a block above it unless it is either located in the first row or the first column of the given image.
The motivation behind the two-dimensional prediction technique is that if the current block under process \( x \) has a high degree of similarity to either its encoded left neighbor or the encoded block directly above it, then the similar block is employed to represent \( x \) instead of performing the BTC encoding procedure. The squared Euclidean distance between two given image blocks is used to measure the degree of similarity.

A pre-defined threshold \( \delta \) is used in the two-dimensional prediction technique to determine the degree of similarity between two given image blocks. If the squared Euclidean distance between two given image blocks is less than or equal to the threshold \( \delta \), it assumes that these two image blocks are similar to each other. Otherwise, these two image blocks are assumed to be different in terms of block characteristics.

If no similar block is found in the neighborhood of an image block \( x \), another test is then performed to determine whether the bit map omission technique can be applied to this image block. Here, the block variance \( \sigma \) of the image block \( x \) is computed. The block variance \( \sigma \) is used to decide the block characteristics of image block \( x \). If \( \sigma \) is less than or equal to the pre-defined threshold \( \phi \), this image block is classified as a smooth block. Then, the bit map omission technique will be used to encode it. Otherwise, the image block \( x \) is classified as a complex block. Each complex block is then
encoded by BTC with bit map coding using 64 edge patterns. In general, if a set of representative visual patterns can be derived, the required bit rate of BTC can be reduced with a slight loss of image quality.

Thus this scheme employs three different techniques along with BTC to reduce the bit rate without losing the complexity advantage of BTC. The experimental results of this scheme show that by choosing adequate threshold values, this scheme achieves good image quality at low bit rate while requiring low computational cost. It solves the major weakness of the BTC scheme: the high bit rate requirement. In other words, the major contribution of this scheme is that it puts the BTC scheme in practice at low bit rate. Although the image quality of the proposed scheme is inferior to that of JPEG 2000 or some other similar schemes at the same bit rate, the proposed scheme is guaranteed to be the most effective in terms of execution time.

The methods and schemes discussed in this chapter are an improvement on the basic BTC method. Several modifications of BTC have been proposed to further reduce the bit rate. The improved methods also maintained the desirable quality of the reconstructed image after decompression. This thesis proposes eight compression schemes based on the Block Truncation Coding. Chapter 4 gives the details of the two compression methods based on BTC proposed in this thesis.