Hindawi Publishing Corporation
EURASIP Journal on Image and Video Processing
Volume 2007, Article ID 65242, 11 pages
doi:10.1155/2007/65242

Research Article

Center of Mass-Based Adaptive Fast Block Motion Estimation

Hung-Ming Chen,[1] Po-Hung Chen,[1] Kuo-Liang Yeh,[2] Wen-Hsien Fang,[2] Mon-Chau Shie,[2] and Feipei Lai[1,3]

[1] *Department of Electrical Engineering, National Taiwan University, No. 1, Sec. 4, Roosevelt Road, Taipei 10617, Taiwan*
[2] *Department of Electronic Engineering, National Taiwan University of Science and Technology, No. 43, Sec. 4, Keelung Road, Taipei 106, Taiwan*
[3] *Department of Computer Science and Information Engineering, National Taiwan University, No. 1, Sec. 4, Roosevelt Road, Taipei 10617, Taiwan*

Received 13 August 2006; Revised 28 January 2007; Accepted 29 January 2007

Recommended by Yap-Peng Tan

This work presents an efficient adaptive algorithm based on center of mass (CEM) for fast block motion estimation. Binary transform, subsampling, and horizontal/vertical projection techniques are also proposed. As the conventional CEM calculation is computationally intensive, binary transform and subsampling approaches are proposed to simplify CEM calculation; the binary transform center of mass (BITCEM) is then derived. The BITCEM motion types are classified by percentage of $(0, 0)$ BITCEM motion vectors. Adaptive search patterns are allocated according to the BITCEM moving direction and the BITCEM motion type. Moreover, the BITCEM motion vector is utilized as the initial search point for near-still or slow BITCEM motion types. To support the variable block sizes, the horizontal/vertical projections of a binary transformed macroblock are utilized to determine whether the block requires segmentation. Experimental results indicate that the proposed algorithm is better than the five conventional algorithms, that is, three-step search (TSS), new three-step search (N3SS), four three-step search (4SS), block-based gradient decent search (BBGDS), and diamond search (DS), in terms of speed or picture quality for eight benchmark sequences.

Copyright © 2007 Hung-Ming Chen et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

1. INTRODUCTION

Motion estimation underlies the foundation of motion-compensated predictive coding of video sequences. Efficient block matching algorithms (BMAs) have received considerable attention and have been adopted in modern video compression standards such as MPEG4, H.264/AVC, and WMV9 [1, 2].

Several fast block matching algorithms, such as three-step search (TSS), new three-step search (N3SS) [3], four-step search (4SS) [4], diamond search (DS) [5], and block-based gradient decent search (BBGDS) [6], have been proposed to reduce computational complexity during the matching process by decreasing the number of search points. Based on the characteristic of center-biased motion vector (MV) distribution, the N3SS, 4SS, and DS algorithms were proposed in [3–5] for improving TSS algorithm performance when estimating small motions. These algorithms utilize the characteristic of center-biased MV distribution and use the halfway-stop approach to speed up stationary or quasistationary block matching. By employing the first step stop mechanism and the center-biased small square pattern, BBGDS [6] yields extremely small number of search points for zero motion.

On the other hand, some studies have applied one-bit transform (1BT) techniques for motion estimation. In [7, 8], 1BT was utilized to assess whether a pixel was an edge pixel. The benefit of such a representation is that distortion between the reference block and search block can be computed very efficiently using an exclusive-or (XOR) function. The 1BT markedly reduces arithmetic and hardware complexity, and power consumption, while retaining good compression performance.

As block-based motion compensation is commonly utilized in video coding to eliminate temporal redundancy, a blocking effect is generated that decreases video quality. Thus, using a fixed block size for block matching is inappropriate. Although utilizing large blocks decreases bitrate, blocking effect increases. This phenomenon is caused by ineffective matching of the blocks straddling the moving zone boundary. Conversely, a small block size increases the

number of MVs and, hence, requires additional bits to code the MVs. Therefore, numerous studies [9–12] have proposed quadtree-based variable block size segmentation approaches that utilize large blocks for the background to decrease the computational complexity, and small blocks for moving zone boundaries to improve prediction precision. However, considerable computations are required to obtain the difference, variance, or even MV from reference frames in top-down splitting or bottom-up merging approaches.

Moreover, some studies developed search techniques based on motion type to enhance speed and quality of BMAs. For example, Jiancong et al. [13] proposed the content adaptive search technique that clusters blocks within a frame into foreground and background regions based on video scene analysis. Parameters for motion characteristics for each region are extracted to identify a suitable search area and the initial search point.

This work proposes a novel adaptive fast block motion estimation algorithm based on center of mass (CEM), binary transform, subsampling, and horizontal/vertical projection techniques. A preliminary MV is computed based on the CEM difference between macroblocks, the CEM MV then classifies the moving direction and motion type to determine the initial search point and search patterns. As the conventional CEM calculation is computationally intensive, binary transform and subsampling techniques [15, 16] are utilized to simplify CEM MV calculations; the binary transform CEM (BITCEM) is then obtained. Since CEM properties do not hold for particular scenarios, horizontal and vertical projections are applied to segment the blocks when the variable block size option is enabled. The BITCEM MV is not applied when a block is segmented. After classifying motion type, different search patterns are employed to obtain the MVs.

The remainder of this paper is organized as follows. Section 2 describes the proposed BITCEM and techniques that decrease the computational complexity and define search patterns. Section 3 describes in detail the proposed CEM-based BMA algorithm. Sections 4 and 5 present experimental results and the discussion, respectively. Conclusions are reported in Section 6.

2. PROPOSED BINARY TRANSFORM CENTER OF MASS

The principle in the CEM scheme, which has been utilized in previous imaging applications [17], was first applied in motion estimation. The shortcoming of the CEM technique is that it requires a massive amount of computations. Therefore, this study redefines the CEM of a moving zone by transforming the gray-level image into a binary-level image, thereby decreasing the number of operations. Based on this BITCEM approach, the CEM of a moving zone within a block and its direction of movement can be obtained rapidly. Four additional techniques are employed in this stud y to decrease computational complexity and maintain picture quality. All approaches utilized in the proposed search scheme are described as follows.

2.1. Revised center of mass with binary transform

Center of mass

Motion of a CEM can represent rigid object motion. In this study, gray levels are regarded as the pixel mass. The definition of CEM is

$$\bar{i} = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} i \times I(i,j)}{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} I(i,j)},$$
$$\bar{j} = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} j \times I(i,j)}{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} I(i,j)}, \tag{1}$$

where $I(i,j)$ is the gray level of (i,j) of a block, $(\bar{i},\bar{j})$ is the coordinate of the CEM of a block, and (M,N) is the block dimension. Based on complexity, (1) require, considerable computation.

Example 1. For a 16×16 block using (1) to identify the block CEM, the following computations are required:

addition: $2 \times 255 + 255 = 765$,
multiplication: $2 \times 256 = 512$,
division: 2.

Note that the number of additions in numerator is $16 \times 16 - 1$, whereas that in denominator is also $16 \times 16 - 1$. The number of additions for a horizontal or vertical component is the sum of that in the numerator and denominator—$255 + 255$. However, horizontal and vertical components can have a common denominator, indicating that the number of additions for both horizontal and vertical components is only $2 \times 255 + 255$ rather than $2 \times (255 + 255)$. To obtain the MV between two CEMs, calculations must be applied in colocated blocks in previous and current frames. Consequently, the total number of additions doubles to $2\times(2\times255+255) = 1530$.

When the mean absolute difference (MAD) is utilized as the criterion, then a search point requires 256 subtractions and 255 additions, implying that the computation of CEM is equivalent to approximately 11 search points, assuming that multiplication or division operations are four times the number of addition operations.

In the following section, the CEM is revised to decrease the computational complexity.

Revised center of mass with the binary transform

Notably, the additional effort required when calculating the CEM of a nonmoving zone within a block is unnecessary; consequently, the CEM of a moving zone is redefined to decrease computational effort. The binary transformation is applied to each block such that each pixel has a bi-level value and the bi-level image block is represented by P. The $P(i,j) = 1$ indicates that the (i,j) pixel is inside the moving zone, and $P(i,j) = 0$ indicates that the pixel is outside the

moving zone. The BITCEM is defined as

$$\bar{i} = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} i * P(i,j)}{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} P(i,j)} = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} i\big|_{P(i,j)=1}}{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} P(i,j)}, \quad (2)$$

$$\bar{j} = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} j * P(i,j)}{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} P(i,j)} = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} j\big|_{P(i,j)=1}}{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} P(i,j)}, \quad (3)$$

where $P(i,j)$ is the binary level of (i,j) of a block, $(\bar{i},\bar{j})$ is the coordinate of BITCEM of a block, and (M,N) is the block dimension.

Clearly, by utilizing (2) and (3), multiplication can be avoided when calculating BITCEM, and addition is only required when a pixel is located inside the moving zone, that is, when $P(i,j) = 1$. Take a 16×16 block as an example, the computations required in (2) and (3) are as follows:

additions in maximum: $2 \times 255 + 255 = 765$,
multiplications: 0,
divisions: 2.

Similarly, to acquire the MV between two BITCEMs, calculations must be performed for both colocated blocks in the previous and current frames. Consequently, the maximum number of additions doubles to $2 \times (2 \times 255 + 255) = 1530$, indicating that the computation of BITCEM is equivalent to roughly 3 search points in maximum, assuming that multiplication or division operations are four times the number of addition operations. Hence, this BITCEM formula markedly decreases the CEM computational complexity.

2.2. Definition of moving zone and BITCEM motion vector

In nature, an object may have uniformity or homogeneity of gray levels to some degree [18], suggesting that an object (the moving zone within a block) can be represented by a reference gray value. To eliminate false alarms or misdetection caused by noise prior to identifying a moving zone, the moving zone is assumed to be larger than a 5×5 pixel area. As movement of a moving zone generates gray-level differences, the current block B_k is subtracted from its colocated block B_{k-1} to obtain block difference. A moving zone should be located at the position at which a large pixel difference exists, of which there are two cases. One position is located in the path of a moving direction, and the other position is located in the path of the opposite moving direction. Hence, this work searches for the largest pixel difference with the outermost coordinates in the quadrant indicated by the motion vector MV_{k-1} of the colocated block in the reference frame. Those outermost coordinates with the largest pixel difference are most likely a moving zone edge. One must then identify a reference gray-level; it is best to adopt the pixel value inside the moving zone. Thus, according to the motion vector MV_{k-1}, pixel (i', j') is located at the farthest location along the moving direction among the candidates with the largest gray level difference. To obtain the pixel inside the moving zone with the reference gray level, 5 is added or subtracted from horizontal and vertical coordinates based on the reverse moving direction to derive $I_k(\hat{i},\hat{j})$ as the moving zone assumed larger than 5×5. Thus, Figure 1 shows the reference gray level of moving zone within the block k.

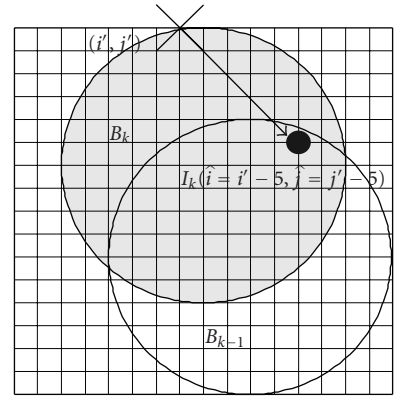

Figure 1: The reference gray levels of moving zone with moving directions to the top left.

After obtaining the reference gray level $I_k(\hat{i},\hat{j})$ for a moving zone, (2) and (3) are applied to locate the BITCEMs of moving zones within the current block B_k and the colocated block B_{k-1}. The following are the steps defining a moving zone and BITCEM of a block.

Step 1. If $I_k(\hat{i},\hat{j}) - \mathrm{TH} < I_k(i,j) < I_k(\hat{i},\hat{j}) + \mathrm{TH}$, let $P_k(i,j) = 1$; otherwise $P_k(i,j) = 0$. Hence, P_k represents the bi-level pixels of current block B_k.

Step 2. Use (2) and (3) to derive $(\bar{i}_k, \bar{j}_k)$, the BITCEM of current block B_k.

Step 3. If $I_k(\hat{i},\hat{j}) - \mathrm{TH} < I_{k-1}(i,j) < I_k(\hat{i},\hat{j}) + \mathrm{TH}$, let $P_{k-1}(i,j) = 1$; otherwise, $P_{k-1}(i,j) = 0$. Hence, P_{k-1} is the bi-level pixels of colocated block B_{k-1}.

Step 4. Use (2) and (3) to derive $(\bar{i}_{k-1}, \bar{j}_{k-1})$, the BITCEM of the colocated block B_{k-1}.

The decision regarding a threshold (TH) value is based on the human perceptual characteristic. Thus, the BITCEM MV (mx, my) can be obtained using the following equations,

$$mx = \bar{i}_k - \bar{i}_{k-1}, \quad (4)$$

$$my = \bar{j}_k - \bar{j}_{k-1}. \quad (5)$$

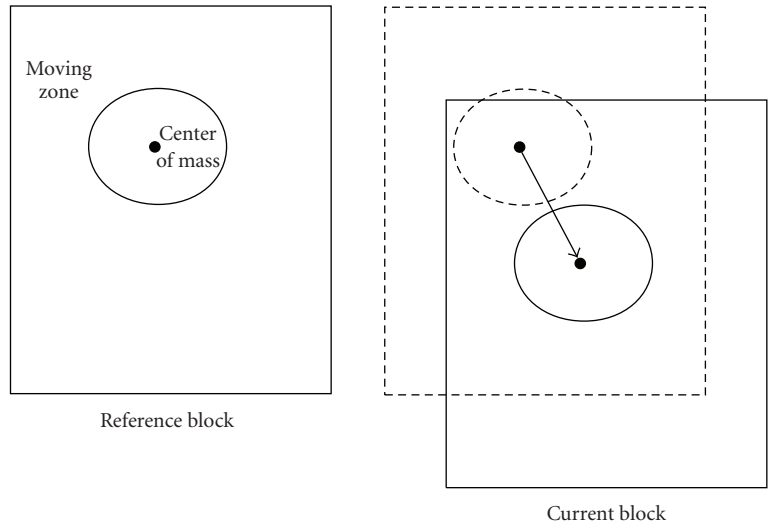

Figure 2: The relationship between block motion and BITCEM motion.

A BITCEM MV can be obtained from two colocated blocks between successive frames (Figure 2). The following simple proof verifies that the BITCEM MV represents the MV of a moving zone and is taken as the basis of the proposed algorithm.

Theorem 1. *Suppose that the moving zone will not move outside the block, the BITCEM MV then represents the MV of the moving zone.*

Proof. Let the BITCEM of moving zone within the current block P_k and the reference block P_{k-1} be $(\bar{i}_k, \bar{j}_k)$ and $(\bar{i}_{k-1}, \bar{j}_{k-1})$, respectively. The BITCEM MV $(m1, m2)$ is then defined by (4) and (5) as follows,

$$\begin{aligned} m1 &= \bar{i}_k - \bar{i}_{k-1}, \\ m2 &= \bar{j}_k - \bar{j}_{k-1}. \end{aligned} \tag{6}$$

Replace (4) by (2) to obtain

$$\begin{aligned} m1 &= \bar{i}_k - \bar{i}_{k-1} \\ &= \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} i_k \big|_{P_k(i,j)=1}}{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} P_k(i,j)} \\ &\quad - \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} i_{k-1} \big|_{P_{k-1}(i,j)=1}}{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} P_{k-1}(i,j)}, \end{aligned} \tag{7}$$

where $\sum_{i^*=0}^{M-1} \sum_{j^*=0}^{N-1} P_k(i,j) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} P_{k-1}(i,j) = \text{Area}$, representing the area of the moving zone within a block. Additionally, the motion quantity of all pixels within the moving zone is the same such that $i_k = i_{k-1} + \Delta i$, where Δi is the motion quantity of a moving zone. Equation (7) can therefore be rewritten as

$$\begin{aligned} & mx \\ &= \frac{\left(\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (i_{k-1} + \Delta i)\big|_{P_{k-1}(i,j)=1}\right) - \left(\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} i_{k-1}\big|_{P_{k-1}(i,j)=1}\right)}{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} P_{k-1}(i,j)} \\ &= \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left[(i_{k-1} + \Delta i) - i_{k-1}\right]\big|_{P_{k-1}(i,j)=1}}{\text{Area}} \\ &= \frac{\Delta i \times \left(\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} P_{k-1}(i,j)\right)}{\text{Area}} \\ &= \Delta i \times \frac{\text{Area}}{\text{Area}} = \Delta i. \end{aligned} \tag{8}$$

By the same reasoning, $my = \Delta j$. Clearly, the BITCEM MV is equivalent to the MV of the moving zone. □

2.3. Subsampling

To obtain the BITCEM for a 16×16 block, at least 2×256 subtractions and 2×256 comparisons are required. Hence, computations for at minimum two search points is required. Moreover, an additional computation is required to calculate the BITCEM which is dependent on moving zone size. Hence, under the assumption that each pixel in a block has the same MV, the subsampling approach can be utilized to simplify the BITCEM computation. In this approach, the subsampling of the bi-level frame is applied with subsampling rates of 1, 2, 4, or 8 causing a small reduction in precision. As a trade off between computational complexity and picture quality, the subsampling rate is set to 4 as an adequate subsampling rate. The following is the mathematical proof for the subsampling approach employed in the BITCEM algorithm.

Table 1: Still block percentage with different subsampling rate.

Still Block	Claire	Miss America	Salesman	Carphone	Flower	Football	Table tennis	Bike
Rate 8	96.30	98.30	95.90	93.90	55.20	69.70	82.60	93.80
Rate 4	96.80	96.40	85.10	85.10	50.00	59.30	74.80	76.60
Rate 2	93.70	94.10	79.80	79.80	44.50	60.70	60.70	82.00
Rate 1	90.90	91.20	71.90	71.90	38.20	53.40	53.40	76.60

Theorem 2. *Suppose that the sampling rate is R (i.e., to sample one pixel from all R pixels), and pixels within the same sampling range have identical attributes (i.e., pixels have the same motion type—moving or still) and the same bi-level pixel value, then the BITCEM MV is equivalent to the MV of a moving zone.*

Proof. Because $(i, j) = (R \times i^*, R \times j^*)$ and

$$\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} P_k(i,j) = R^2 \times \sum_{i=0}^{(M-1)/R}\sum_{j=0}^{(N-1)/R} P(i^*, j^*), \tag{9}$$

then,

$$\begin{aligned}\bar{i} &= \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} i \times P(i,j)}{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} P(i,j)} = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} i\big|_{P(i,j)=1}}{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} P(i,j)} \\ &= \frac{R^2 \times \sum_{i^*=0}^{(M-1)/R}\sum_{j^*=0}^{(N-1)/R} R \times i^*\big|_{P(i^*,j^*)=1}}{R^2 \times \sum_{i^*=0}^{(M-1)/R}\sum_{j^*=0}^{(N-1)/R} P(i^*, j^*)} = R \times \overline{i^*},\end{aligned} \tag{10}$$

where (i^*, j^*) is the pixel coordinate of the pixel after subsampling, $P(i^*, j^*)$ is the pixel bi-level value (i^*, j^*), and $(\bar{i}^*, \bar{j}^*)$ is the coordinate of BITCEM after subsampling. By the same reasoning, $\bar{j} = R \times \overline{j^*}$.

Based on this deduction, the BITCEM of a block is the BITCEM of a block following subsampling multiplied by R. In the same manner, the BITCEM MV following pixel subsampling is equivalent to the MV of a moving zone:

$$\begin{aligned}mx &= \left[\frac{R^2 \times \sum_{i^*=0}^{(M-1)/R}\sum_{j^*=0}^{(N-1)/R} R \times (i^*_{k-1} + \Delta i^*)\big|_{P_{k-1}(i^*,j^*)=1}}{R^2 \times \sum_{i^*=0}^{(M-1)/R}\sum_{j^*=0}^{(N-1)/R} P(i^*, j^*)}\right] \\ &\quad - \left[\frac{R^2 \times \sum_{i^*=0}^{(M-1)/R}\sum_{j^*=0}^{(N-1)/R} R \times i^*_{k-1}\big|_{P_{k-1}(i^*,j^*)=1}}{R^2 \times \sum_{i^*=0}^{(M-1)/R}\sum_{j^*=0}^{(N-1)/R} P(i^*, j^*)}\right] \\ &= \frac{\sum_{i^*=0}^{(M-1)/R}\sum_{j^*=0}^{(N-1)/R} R \times [(i^*_{k-1} + \Delta i^*) - i^*_{k-1}]\big|_{P_{k-1}(i^*,j^*)=1}}{\sum_{i^*=0}^{(M-1)/R}\sum_{j^*=0}^{(N-1)/R} P(i^*, j^*)} \\ &= \frac{\Delta i \times \left(\sum_{i^*=0}^{(M-1)/R}\sum_{j^*=0}^{(N-1)/R} P_{k-1}(i^*, j^*)\right)}{\text{Area}} \\ &= \Delta i \times \frac{\text{Area}}{\text{Area}} = \Delta i.\end{aligned} \tag{11}$$

By the same reason, $my = \Delta j$. □

Table 2: Classification of video motion type.

BITCEM motion type	Still block percentage
Still	100% ∼ 93.75% (15/16)
Slow	93.75% ∼ 75% (12/16)
Fast	75% ∼ 0% (0/16)

2.4. Classification of video motion types

To utilize computational resources efficiently, different search patterns are allocated to different video motion types. The (0, 0) BITCEM MV implies a still block. Table 1 lists the percentage of still blocks in each sequence using different subsampling rates. The still block percentage for the previous frame is utilized to classify the video into three BITCEM motion types: near-still motion, slow motion, and fast motion (Table 2).

Table 2 is utilized as a reference for classifying BITCEM motion types when the percentage of still blocks (Table 1) is given. The three classification types of video motion are not arbitrary. First, the still block percentage in Table 1 is calculated. Each frame in an image sequence is then classified dynamically according to the classification rule in Table 2.

Notably, the still block percentage ranges (Table 2) are empirical values. As the background blocks always dominate a full scene, background blocks account for more than 75% of all video motion types.

2.5. Estimation of initial search point

The spatial and temporal correlations between blocks are significant characteristics for increasing the speed of the block matching algorithm [19].

(1) In consecutive frames, the moving zones are almost at the same velocity; consequently, the MVs of colocated blocks at consecutive frames are strongly correlated.
(2) The MVs of neighboring blocks within the same frame are almost the same.

Consequently, when the MVs of certain blocks are identified, the linear prediction model MV [20] can be applied to predict the initial search point of the related block.

Let $\text{MV}(i, j, k)$ be the MV of block (i, j) in the kth frame; then

$$\text{MV}(i,j,k) = E[\text{MV}(i,j,k)] + d\,\text{MV}(i,j,k), \tag{12}$$

where $d\,\text{MV}(i, j, k)$ is the MV difference between the MV and the estimation of the initial search point, and $E[\text{MV}(i, j, k)]$

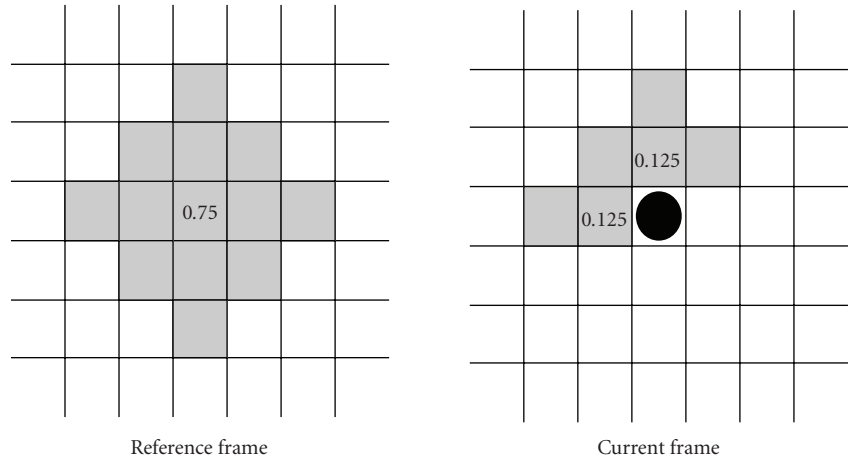

Figure 3: Estimation of initial search point.

can be represented as

$$E[\,\mathrm{MV}(i,j,k)] = \sum_{p,q\in W1} \lambda_{p,q,k}\,\mathrm{MV}(i-p, j-q, k) + \sum_{p,q\in W2} \lambda_{p,q,k-1}\,\mathrm{MV}(i-p, j-q, k-1), \tag{13}$$

where (p,q) is the coordinate difference between neighboring blocks and the current block; W1 and W2 are the ranges of weighted MVs in the current and previous frames, respectively; $\lambda_{p,q,k}$ and $\lambda_{p,q,k-1}$ are weighted coefficients; $\lambda_{p,q,k}$ is the spatial correlation of $\mathrm{MV}(i,j,k)$; and $\lambda_{p,q,k-1}$ is the temporal correlation of $\mathrm{MV}(i,j,k)$ (Figure 3).

2.6. Variable block size option

In addition to fixed block size (FBS) mode, the variable block size (VBS) option, including 8×8 and 16×16 block sizes, is proposed in this work. As the projection of the binary image retains considerable information, the projection can be widely utilized for object shape recognition [21]. Horizontal projections (HP) and vertical projections (VP) that project a binary image in horizontal and vertical directions, respectively, are the two simplest projection methods. Blocks that produce zeros within 2 pixels from the middle of the current block will be horizontally or vertically segmented after horizontal or vertical projection; the block motion can then be estimated using small blocks. Horizontal projection is applied to the binary value block, resulting in a zero value in the horizontal direction (Figure 4). In the proposed algorithm, segmentation is applied in accordance with the horizontal projection $\mathrm{HP}(i)$ or the vertical projection $\mathrm{VP}(j)$. Almost no additional computations are required for binary image projections when obtaining the BITCEM. $\mathrm{HP}(i)$ and $\mathrm{VP}(j)$ are

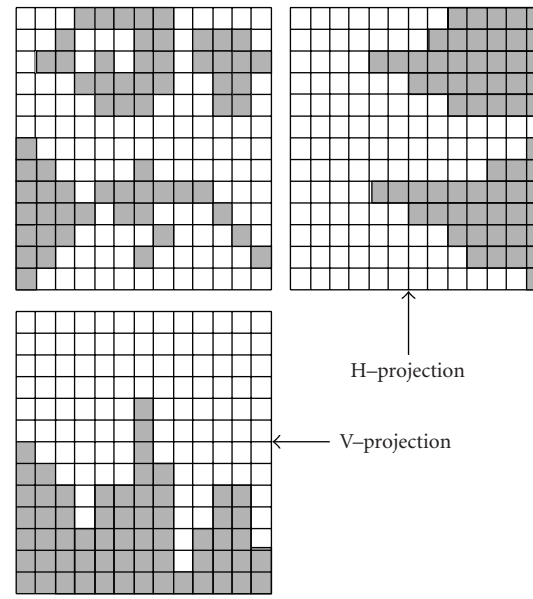

Figure 4: Binary projection.

defined as

$$\mathrm{HP}(i) = \sum_{j=0}^{N-1} P(i,j), \qquad \mathrm{VP}(j) = \sum_{i=0}^{M-1} P(i,j), \tag{14}$$

where $P(i,j)$ is the binary value of pixel (i,j) of a block and (M,N) is the block dimension.

Based on the assumption of a rigid object, the translation of a moving zone is $\sum_{i=0}^{M-1}\mathrm{HP}(i) = \sum_{j=0}^{N-1}\mathrm{VP}(j) = \mathrm{Area}$, where Area is the area of a moving zone. Then, the BITCEM

$(\bar{i}, \bar{j})$ can be rewritten as

$$\bar{i} = \frac{\sum_{i=0}^{M-1} i \times \mathrm{HP}(i)}{\mathrm{Area}},$$
$$\bar{j} = \frac{\sum_{j=0}^{N-1} j \times \mathrm{VP}(j)}{\mathrm{Area}}. \quad (15)$$

Based on this analysis, the BITCEM can be derived using the HP and the VP of a block with a binary value. Thus, only 64 multiplication operations are needed to obtain the BITCEMs of the current and reference blocks. The computation that has 256 additions is equivalent to a negligible 0.5 search point, assuming that multiplication operations are four times the number of addition operations.

3. THE PROPOSED CENTER OF MASS-BASED ADAPTIVE MOTION ESTIMATION SCHEME

Initial search point

In the proposed scheme, the current and reference blocks are first input to acquire the BITCEM MV; the percentage of the $(0, 0)$ BITCEM MVs in the previous frame is then utilized to classify the three BITCEM motion types. This study alternated the conventional linear prediction model MV (as described in Section 2.5) with the proposed BITCEM MV as the initial search point, based on the BITCEM motion type. The BITCEM MV is applied in near-still and slow BITCEM motion types to acquire a precise initial search point, whereas for the fast BITCEM motion type, the linear prediction model in (13) is adopted instead.

Segmentation

When the VBS option is enabled (as described in Section 2.6), the proposed scheme determines whether segmentation is required after identifying the initial search point. Both HP and VP employ the derivatives of BITCEM calculation to determine whether the block requires segmentation.

The BITCEM of the original 16×16 block is not used as the original block has been segmented. This BITCEM fails to represent the BITCEMs of multiple moving zones within the block. For simplification, the BITCEMs of the subblocks after horizontal and vertical segmentations are not calculated. The BITCEM MV calculated prior to segmentation is replaced by $(0, 0)$ as the initial search point.

Search patterns

Based on BITCEM motion directions and motion types, different search patterns with different search strategies are proposed (Figures 5(a) and 5(b)) to estimate a motion vector with increased precision. For near-still and slow BITCEM motion types, concentrated search patterns are applied, whereas for fast BITCEM motion type, dispersed search patterns are applied for fast BITCEM motion types. Additionally, alternative search patterns are introduced into the scheme to further decrease the number of search points when attempting to retain picture quality.

When the BITCEM MV is not $(0, 0)$, some additional points, in addition to the points close to the center, are added along the BITCEM moving direction (horizontal, vertical, sloped, inverse-sloped) to improve search precision. For a BITCEM moving horizontally or vertically, additional search points, such as SP3H/SP4H or SP3V/SP4V, are allocated to horizontal or vertical directions, respectively. Regardless of the direction in which the BITCEM is moving, the search patterns contain points close to the center to employ the characteristic of center-biased distribution for near-still and slow BITCEM motion types. For the fast BITCEM motion type, points in a circular shape are added to locations far from the center. To accommodate all directions with a slope other than straight horizontal or straight vertical BITCEM moving direction, defined as sloped or inverse-sloped, concentrated and dispersed search patterns, such as SP5S or SP5IS, are combined for all BITCEM motion types. During the next search step, when the frame is the near-still or slow BITCEM motion type, SP6 or SP1 is allocated alternatively around the best match candidate of the first search step to acquire the final MV.

When the BITCEM MV is $(0, 0)$, no directionally biased search points are allocated. The SP1 search pattern is applied for near-still or slow BITCEM motion types and SP2 is applied for fast BITCEM motion type. When the block requires segmentation, a single search pattern, SP7, is utilized as the initial search pattern.

The proposed algorithmic process is summarized as follows.

Step 1. Input the current block.

Step 2. Calculate the BITCEM, BITCEM MV, HP, and VP using (2)–(5) and (14)-(15).

Step 3 (VBS option). If any zero value is located in the middle of the $\mathrm{HP}(i)$ or $\mathrm{VP}(j)$, then the block is segmented.

Step 4 (VBS option). When the block is segmented, the initial MV is then assigned to be $(0, 0)$; go to Step 7.

Step 5. Classify the BITCEM motion types according to the percentage of the $(0, 0)$ BITCEM MV.

Step 6. Assign the initial search point ($(0, 0)$ is suitable for segmented blocks, BITCEM MV for near-still and slow BITCEM motion types, and linear prediction model MV for fast video motion type) and allocate the search pattern based on the BITCEM motion type and direction of BITCEM motion.

Step 7. Begin searching in accordance with the initial search pattern.

Step 8. Continue searching from the best match point via Step 7 using the next search pattern.

Step 9. When the best match point is $(0, 0)$ during a search iteration, stop the search and go to Step 1 (when the block is segmented, continue searching other subblocks); otherwise, continue searching based on the next search pattern.

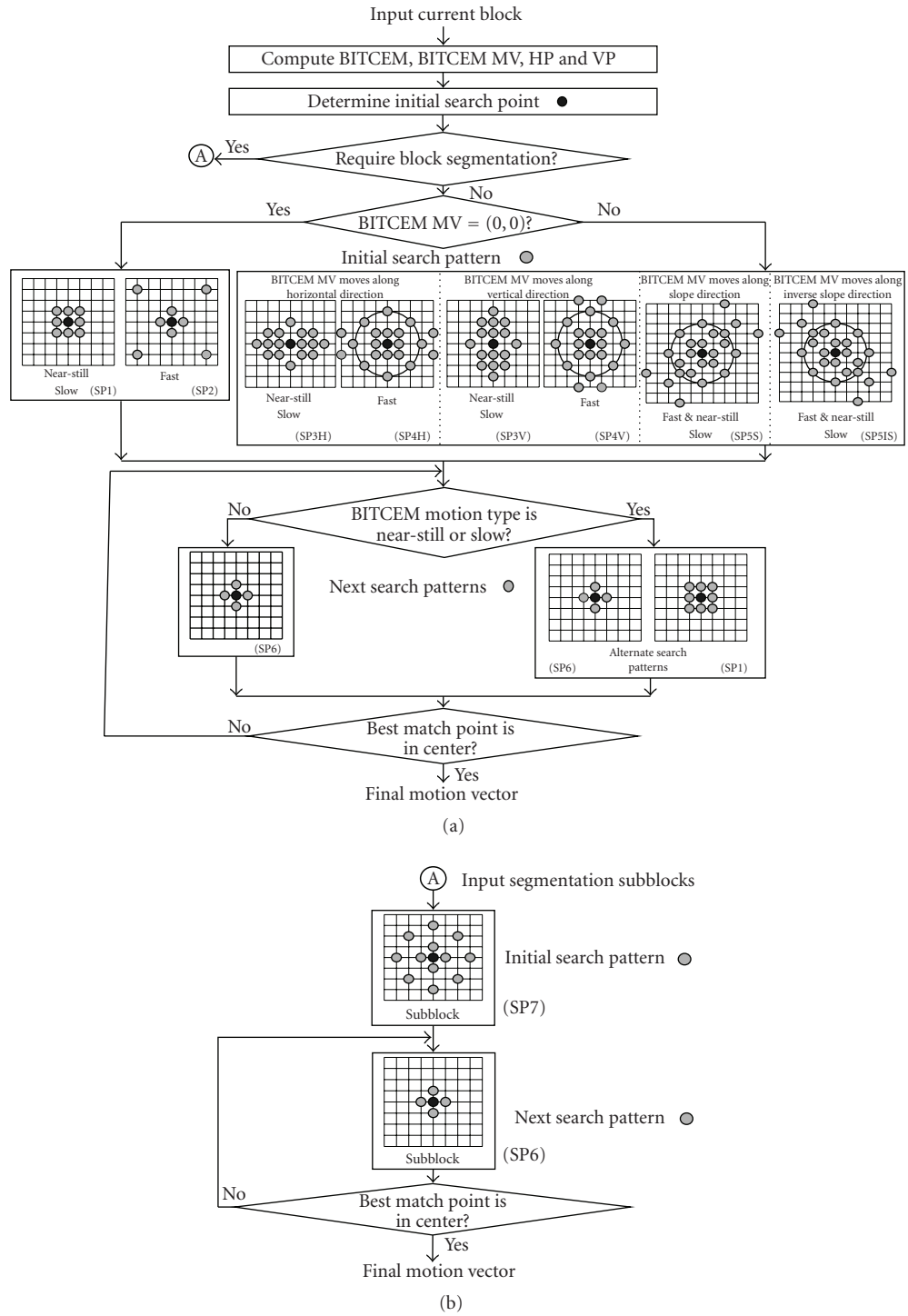

Figure 5: The proposed scheme: (a) flow chart for nonsegmented block, (b) flow chart for segmented block.

4. EXPERIMENTAL RESULTS

In this experiment, TH is set at 40. Four is chosen as the sampling rate, a compromise between complexity and precision for defining a moving zone, performing HPs and VPs, and calculating the BITCEM. The proposed algorithm is compared with the full search (FS), TSS, N3SS, 4SS, BBGDS, and DS. The VBS mode is optional. The search area is 15×15 pixels. The frame sizes of test sequences are 352×288, 352×240, and 176×144 pixels. The following criteria are applied to measure the performance of each algorithm.

(1) Average mean square error: since the focus of this work is on motion estimation rather than the whole coding scheme, only the difference between the reconstructed frame via motion compensation and the original frame is compared. That is, the residual frame is not added to the reconstructed frame to clarify the comparison of each BMA. Notably, MSE is inversely correlated with picture quality.

(2) Picture deterioration percentage: this criterion measures the difference in MSE between each algorithm and the FS algorithm divided by the MSE of the FS algorithm. Deterioration percentage is inversely correlated with picture quality.

(3) Complexity/block: complexity is a measure of the number of search points for each algorithm. Take BITCEM for example, since each search point requires 256 subtractions and 255 additions, the complexity of BITCEM is calculated as follows:

$$\text{complexity} = \text{search points} + \text{BITCEM computation}/511\ (\text{search points}). \tag{16}$$

Complexity is inversely correlated with coding speed.

(4) Speedup: speedup represents the complexity of the FS algorithm divided by that of each algorithm.

The FBS mode (Table 3) demonstrates that the proposed algorithm decreases computational complexity significantly. The algorithm is 13–20 times faster than the FS algorithm. Additionally, based on MSE/pixel comparison results, the proposed algorithm renders the best picture quality and fewest search points compared with those of TSS, N3SS, 4SS, and DS except for the Football and Carphone sequences. Although BBGDS requires the fewer search points than the other algorithms, it is likely to be trapped into a local minimum for video sequences with large motion content. The proposed algorithm requires slightly more search points than BBGDS and retains superior MSE performance.

The VBS mode (Table 3) demonstrates that the speed of the proposed algorithm remains high, and generates better picture quality than the FS algorithm with fixed block size, such that the deterioration percentage comparison results in negative values. That is, the algorithm takes advantage of the calculation for BITCEM MV to further increase picture quality without excessive computations in the projection technique utilized when determining whether to segment the block or not. The proposed algorithm costs only 5.38% to 8.22% of the computation cost required by FS to enhance 0.21%–9.67% of the picture quality generated by FS. Thus, the proposed BITCEM-based adaptive BMA with variable block size technique effectively eliminates the blocking effect, thereby improving the precision of motion estimation. Experimental results justify the motivation and robustness of the proposed scheme.

5. DISCUSSION

The threshold TH is inversely correlated with the degree of uniformity of the moving zone gray level, and positively correlated with the moving zone size. Thus, when the TH is set as a large value, then an increased number of pixels fall within the range, in which $P_k(i, j) = 1$. That is, the moving zone area estimated by the number of $P_k(i, j) = 1$ or $P_{k-1}(i, j) = 1$ enlarges. Considering the uniformity of the moving zone gray level, 40 is the empirical optimal threshold that attains a satisfactory result for all video sequence types suggesting that 40 as the threshold generates the most likely moving zone and the most accurate BITCEM. In "Football" and "Carphone" fast motion sequences, many blocks break the assumption in Theorem 1; consequently, the moving direction of the BITCEM cannot be accurately estimated, and, hence, a correct search pattern for successive block matching cannot be applied.

Furthermore, like all BMAs, three assumptions are required: (1) no object distortion while moving; (2) a single moving object within a block; and (3) the object will not move outside a block. The following discusses the impact on BITCEM robustness when any one of the three assumptions does not hold.

(1) No object distortion while moving (rigid object translation): the nonrigid object translation problem cannot be solved using BMAs. When this assumption is violated, any BMA fails to find a similar block as the best match. This typically results in a large prediction error.

(2) A single moving object within a block: when there is more than one moving zone in a block, only one reference pixel will fail to represent the gray levels for multiple moving zones. This issue may be solved using the VBS option with the proposed H/V projection segmentation.

(3) The moving zone stops outside a block. When the moving zone moves out of a block, then the reference pixel cannot be located to perform a binary transform and successive CEM operation. Since a moving zone moving out of a block is prone to happen in fast video motion, the mechanism of detecting the assumption break can be performed by classifying the BITCEM motion type. As this assumption breaks, the moving zone does not exist in the current block, which leads to an inaccurate BITCEM MV. The proposed algorithm applies a linear prediction model MV rather than CEM MV for an initial search point.

Moreover, the approach for performing block size variation is computationally efficient, as HP and VP are derived with the BITCEM calculation, for which no overhead is required. On the other hand, by enabling the VBS option, a specific block size can be determined. Although this study only simulated the block sizes of 16×16 and 8×8, which are adopted in MPEG-4 and H.263, the proposed scheme can be extended to block sizes of $16 \times 8/8 \times 16/8 \times 4/4 \times 8/4 \times 4$ for H.264 via further horizontal and/or vertical segmentations. Consequently, the same motion search algorithm is unnecessary for each block size to decrease significantly the number of search steps [14].

Considering the conditional branches, there are 5 conditional branches in the proposed scheme when the segmentation branch condition is not taken: (1) to check if the block is required for segmentation (one conditional branch); (2) to

Table 3: Performance comparison of BITCEM, FS, TSS, N3SS, 4SS, DS, and BBGDS.

Sequences	Search algorithm	FS	TSS	N3SS	4SS	DS	BBGDS	BITCEM (FBS)	BITCEM (VBS)
Claire (352 ∗ 288 ∗ 91)	MSE/pixel	9.14	9.35	9.31	9.31	9.29	9.29	9.21	8.91
	Deterioration (%)	0.00	2.25	1.79	1.81	1.66	1.66	0.71	−2.51
	Complexity/block	204.28	23.28	20.28	17.59	14.99	11.3	11.75	12.63
	Speedup	1.00	8.77	10.07	11.61	13.62	18.07	17.39	16.18
MissAmerica (352 ∗ 288 ∗ 91)	MSE/pixel	10.11	10.57	10.24	10.50	10.26	10.23	10.22	9.98
	Deterioration (%)	0.00	4.58	1.34	3.94	1.51	1.18	1.17	-1.21
	Complexity/block	204.28	23.44	21.78	18.83	16.60	12.92	12.91	13.18
	Speedup	1.00	8.72	9.38	10.85	12.31	15.81	15.82	15.49
Saleman (352 ∗ 288 ∗ 91)	MSE/pixel	27.60	28.29	27.92	28.16	28.13	28.11	27.84	27.09
	Deterioration (%)	0.00	2.52	1.18	2.03	1.95	1.85	0.89	−1.84
	Complexity/block	204.28	23.23	16.85	16.24	12.92	9.45	10.41	10.98
	Speedup	1.00	8.79	12.13	12.58	15.82	21.61	19.62	18.60
Flower Garden (352 ∗ 240 ∗ 91)	MSE/pixel	277.00	320.33	285.01	299.69	287.03	292.6	284.93	250.23
	Deterioration (%)	0.00	15.64	2.89	8.19	3.62	5.63	2.86	−9.67
	Complexity/block	202.05	23.25	21.58	18.90	17.02	14.67	15.53	16.60
	Speedup	1.00	8.69	9.36	10.69	11.87	13.77	13.01	12.17
Football (352 ∗ 240 ∗ 91)	MSE/pixel	384.88	416.43	412.54	428.89	433.32	445.21	419.91	363.82
	Deterioration (%)	0.00	8.20	7.19	11.43	12.59	15.67	9.10	−5.47
	Complexity/block	202.05	23.09	20.56	18.04	16.06	14.66	15.65	15.85
	Speedup	1.00	8.75	9.82	11.20	12.58	13.78	12.91	12.75
Table Tennis (352 ∗ 240 ∗ 91)	MSE/pixel	184.86	240.07	217.46	213.28	205.94	221.04	203.66	184.47
	Deterioration (%)	0.00	29.87	17.64	15.37	11.40	19.57	10.17	−0.21
	Complexity/block	202.05	23.32	21.57	19.03	16.87	15.54	14.63	15.27
	Speedup	1.00	8.67	9.37	10.62	11.98	13.01	13.81	13.24
Bike (352 ∗ 240 ∗ 91)	MSE/pixel	40.67	44.01	42.21	42.92	42.38	43.3	42.09	37.64
	Deterioration (%)	0.00	8.21	3.78	5.54	4.21	6.47	3.49	−7.44
	Complexity/block	202.05	23.18	22.24	19.22	17.51	15.12	14.73	15.91
	Speedup	1.00	8.72	9.09	10.51	11.54	13.36	13.72	12.70
Carphone (176 ∗ 144 ∗ 91)	MSE/pixel	41.44	43.19	41.87	43.27	42.41	42.1	42.42	39.08
	Deterioration (%)	0.00	4.22	1.04	4.41	2.33	1.59	2.37	−5.70
	Complexity/block	184.56	21.60	17.56	15.95	13.55	10.72	11.41	12.76
	Speedup	1.00	8.54	10.51	11.57	13.62	17.21	16.18	14.47

determine the initial search pattern and next search pattern based on the value of the BITCEM MV and BITCEM motion types (two conditional branches); and (3) to determine successive search patterns by the BITCEM motion type and the termination condition whether the best match point is in the center (two conditional branches). Conversely, when the segmentation branch condition is taken, there are only 2 conditional branches: (1) to check if the block is required for segmentation (one conditional branch); and (2) to determine the successive search patterns by the termination condition whether the best match point is in the center (one conditional branch). Note that the penalty is very different for each conditional branch in each BMA and the penalty varies with the way how a BMA is implemented in software or hardware.

6. CONCLUSION

This study presented a novel adaptive motion estimation based on CEM. The proposed scheme primarily focuses on accurately predicting the moving direction and motion quantity of a block to increase matching process efficiency (including speed and precision). The principal approaches applied are the CEM via binary transform, subsampling, predictive search, classification of video motion types, arrangement of search patterns, and variable block size. To decrease computational complexity, a binary transform approach with colocated measures (e.g., reference pixel estimation and empirical threshold finding) is utilized. Subsampling is applied to further decrease the number of computations, which is the best method of descreasing overhead

generated when calculating a binary transform and CEM. When the VBS option is enabled, the horizontal/vertical projections of a binary transformed macroblock are employed to determine whether the block requires segmentation. Experimental results show that the VBS mode generates the best picture quality with a slight increase in overhead complexity. When the FBS mode is adopted, its speed is close to the first step-stop BBGDS algorithm with the fewest search points (complexity/block), and picture quality, with the exception of Football and Carphone sequences, remains the highest among FS, TSS, N3SS, 4SS, DS, and BBGDS algorithms. Experimental findings demonstrate that the proposed algorithm is an efficient BMA that is robust in prediction quality and descreasing the computational complexity to fit all benchmark sequences.

REFERENCES

[1] A. Puri, X. Chen, and A. Luthra, "Video coding using the H.264/MPEG-4 AVC compression standard," *Signal Processing: Image Communication*, vol. 19, no. 9, pp. 793–849, 2004.

[2] S. Srinivasan, P. Hsu, T. Holcomb, et al., "Windows media video 9: overview and applications," *Signal Processing: Image Communication*, vol. 19, no. 9, pp. 851–875, 2004.

[3] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 438–442, 1994.

[4] L.-M. Po and W.-C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 313–317, 1996.

[5] J. Y. Tham, S. Ranganath, M. Ranganath, and A. A. Kassim, "A novel unrestricted center-biased diamond search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 4, pp. 369–377, 1998.

[6] L.-K. Liu and E. Feig, "A block-based gradient descent search algorithm for block motion estimation in video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 4, pp. 419–422, 1996.

[7] O. Akbulut, O. Urhan, and S. Erturk, "Fast sub-pixel motion estimation via one-bit transform," in *14th IEEE Signal Processing and Communications Applications*, pp. 1–4, Antalya, Turkey, April 2006.

[8] A. A. Yeni and S. Ertürk, "Fast digital image stabilization using one bit transform based sub-image motion estimation," *IEEE Transactions on Consumer Electronics*, vol. 51, no. 3, pp. 917–921, 2005.

[9] P. Strobach, "Quadtree-structured recursive plane decomposition coding of images," *IEEE Transactions on Signal Processing*, vol. 39, no. 6, pp. 1380–1397, 1991.

[10] V. Seferidis and M. Ghanbari, "Generalised block-matching motion estimation using quad-tree structured spatial decomposition," *IEE Proceedings - Vision, Image, and Signal Processing*, vol. 141, no. 6, pp. 446–452, 1994.

[11] J. Lee, "Optimal quadtree for variable block size motion estimation," in *Proceedings of the IEEE International Conference on Image Processing*, vol. 3, pp. 480–483, Washington, DC, USA, October 1995.

[12] M. Silveira and M. Piedade, "Variable block sized motion segmentation for video coding," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS '97)*, vol. 2, pp. 1293–1296, Hong Kong, June 1997.

[13] L. Jiancong, I. Ahmad, L. Yongfang, and S. Yu, "Motion estimation for content adaptive video compression," in *Proceedings of International Conference on Multimedia and Expo (ICME '04)*, vol. 2, pp. 1427–1430, Taiwan, June 2004.

[14] T. Shimizu, A. Yoneyama, H. Yanagihara, and Y. Nakajima, "A two-stage variable block size motion search algorithm for H.264 encoder," in *Proceedings of International Conference on Image Processing (ICIP '04)*, vol. 3, pp. 1481–1484, Singapore, October 2004.

[15] P.-H. Chen, H.-M. Chen, K.-L. Yeh, M.-C. Shie, F. Lai, and C.-W. Yu, "BITCEM: an adaptive block motion estimation based on center of mass object tracking via binary transform," in *Proceedings of the IEEE International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS '01)*, pp. 185–188, Nashville, Tenn, USA, November 2001.

[16] P.-H. Chen, K.-L. Yeh, M.-C. Shie, F. Lai, and C.-W. Yu, "Fast block matching algorithm based on video motion type using BITCEM object tracking technique," in *National Workshop on Safety Critical Systems and Software (SCS '01)*, pp. 465–468, Brisbane, Australia, July 2001.

[17] B. Feng, P. P. Bruyant, P. H. Pretorius, et al., "Estimation of the rigid-body motion from images using a generalized center-of-mass points approach," in *IEEE Nuclear Science Symposium Conference Record*, vol. 4, pp. 2173–2178, San Juan, Puerto Rico, USA, October 2005.

[18] S. Chen and D. Li, "Image binarization focusing on objects," *Neurocomputing*, vol. 69, pp. 2411–2415, 2006.

[19] J. Wang, D. Wang, and W. Zhang, "Temporal compensated motion estimation with simple block-based prediction," *IEEE Transactions on Broadcasting*, vol. 49, no. 3, pp. 241–248, 2003.

[20] S. V. Vaseghi, *Advanced Digital Signal Processing and Noise Reduction*, John Wiley & Sons, New York, NY, USA, 3rd edition, 2006.

[21] R. Jain, R. Kasturi, and B. G. Schunck, *Machine Vision*, McGraw-Hill, New York, NY, USA, 1995.