Two-Bit Transform for Binary Block Motion Estimation

Alp Ertürk and Sarp Ertürk, *Member, IEEE*

***Abstract*—One-bit transforms (1BTs) have been proposed for low-complexity block-based motion estimation by reducing the representation order to a single bit, and employing binary matching criteria. However, as a single bit is used in the representation of image frames, bad motion vectors are likely to be resolved in 1BT-based motion estimation algorithms particularly for small block sizes. It is proposed in this paper to utilize a two-bit transform (2BT) for block-based motion estimation. Image frames are converted into two-bit representations by a simple block-by-block two bit transform based on multithresholding with mean and linearly approximated standard deviation values. In order to avoid blocking effects at block boundaries during the block-by-block transformation while enabling the two-bit representation to be constructed according to local detail, threshold values are computed within a larger window surrounding the transforming block. The 2BT makes use of lower bit-depth and binary matching criteria properties of 1BTs to achieve low-complexity block motion estimation. The 2BT improves motion estimation accuracy and seriously reduces the amount of bad motion vectors compared to 1BTs, particularly for small block sizes. It is shown that the proposed 2BT-based motion estimation technique improves motion estimation accuracy in terms of peak signal-to-noise ratio of reconstructed frames and also results in visually more accurate frames subsequent to motion compensation compared to the 1BT-based motion estimation approach.**

***Index Terms*—Block matching, Boolean block matching, motion estimation, video coding.**

I. Introduction

DIGITAL video compression is essential for the reduction of bandwidth for transmission or storage of video data in a wide range of applications including high-definition television (HDTV) and standard definition television (SDTV) broadcasting equipment, video conferencing transmitters, video cellular phones, digital video camcorders, and multimedia services for networking applications. Motion estimation and compensation play key roles in video coding systems due to the ability of realizing high compression rates achieved by removing temporal redundancies between successive image frames. Motion estimation is usually remarked as the computationally most intensive part of the video coding system, performing up to 50% of the computations encountered in the entire coding system [1].

The most popular technique for motion estimation is block matching [2]. In the block matching algorithm (BMA) the image frame is commonly divided into nonoverlapping rectangular blocks. The best match to the current block of pixels is searched for, in the previous frame of the sequence within a certain search area about the location of the current block. The optimal solution to BMA is the full search (FS) algorithm that exhaustively searches for the best matched block within all locations of the search window. The mean absolute difference (MAD) or mean square error (MSE) matching criteria are considered to be statistically optimal solutions to the matching process.

The FS-MAD motion estimation process of an image frame divided into $N \times N$ blocks, can be expressed in the form of

$$D(m,n) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |I^t(i,j) - I^{t-1}(i+m, j+n)|, \quad -s \le m, \quad n \le s-1 \tag{1}$$

where $D(m,n)$ is the MAD computed for the displacement of (m,n) pixels. $I^t(i,j)$ and $I^{t-1}(i,j)$ are luminance values of the pixels in the current and previous image frames, respectively, and s determines the maximum displacement allowed in both, horizontal and vertical directions, i.e., the search window range [3]. The motion vector (MV) for a corresponding block of pixels is obtained in the form of

$$v = \arg \min_{-s \le m,n \le s-1} D(m,n) \tag{2}$$

as the horizontal and vertical displacements resulting in the lowest absolute difference value.

Various methods have been proposed to reduce the high computational load of the FS minimum absolute difference BMA. Proposed approaches can be divided into three main categories [4]: fast search techniques that select a subset of the possible search candidate locations; techniques based on various forms of pixel pattern or motion field decimation that employ a certain subsampling of the pixel pattern or motion field; and techniques that exploit different matching criteria instead of the classical MAD.

Fast search techniques evaluate the matching criterion on a subset of possible search locations reducing the number of total matching calculations per block. The number of points and search locations can be fixed and set *a priori*, or can variably be determined according to image statistics. Examples include the three-step search (3SS) [5], 2-D logarithmic search (LOGS) [6], cross-search [7], conjugate direction search (CDS) [8], mean-pyramid search [9], new three-step search [10], four-step search (4SS) [11], block-based gradient descent search (BBGDS) [12], diamond search (DS) [13], and polynomial search (PS) [14].

The second category of techniques proposed to reduce the computational load of motion estimation comprises pixel pattern or motion vector subsampling methods. In [15] for example, it has been proposed to reduce the computational complexity of

Manuscript received January 27, 2004; revised October 14, 2004. This paper was recommended by Associate Editor I. Ahmad.

A. Ertürk is with the Department of Electrical and Electronics Engineering, Middle East Technical University, Ankara 06100, Turkey.

S. Ertürk is with the Department of Electronics and Telecommunications Engineering, University of Kocaeli, Kocaeli, 41040, Turkey.

Digital Object Identifier 10.1109/TCSVT.2005.848340

FS by subsampling of motion fields and image pixels. The motion field is subsampled by estimating vectors for a fraction of the blocks only, using a fraction of the pixels at any searched location. The pixel subsampling pattern is alternated during the search, and motion vectors of the subsampled fields are interpolated for the remaining blocks. Similarly, motion estimation by hexagonal subsampling has been presented in [16]. While memory bandwidth is reduced in these methods, only modest motion estimation complexity reduction can be obtained. Adaptive pixel decimation by selecting pixels that are found to have features important in determining a match has been proposed in [17]. Block-matching based on an adaptive matching scan and representative pixels obtained by Taylor series expansion has been proposed in [18].

The two-bit transform (2BT)-based motion estimation technique proposed in this paper falls into the category of techniques that exploit different matching criteria to achieve reduction in computational complexity by simpler matching evaluation compared to the MAD. In [19] it has been proposed to utilize bit plane matching as a preprocessing step to exhaustive search to eliminate unlikely locations, and a classical pixel-based matching is carried out at the remainder locations. The block mean is used as threshold to accomplish a one–bit transform (1BT), and the bit plane of an image frame is constructed in the form of

$$B(i,j) = \begin{cases} 1, & \text{if } I(i,j) \geq t_{\mathrm{bm}} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where t_{bm} represents the threshold value that is set equal to the block mean. Hierarchical feature matching-motion estimation (HFM-ME) that employs sign truncated feature (STF) matching has been proposed in [20]. The STF extraction process is considered as zero-crossing phase detection with the mean as the bias and binary sign pattern as the phase deviation and block matching motion estimation is divided into mean matching and binary phase matching performed by Boolean logic operations. It has been proposed to accomplish binary block matching on the binary edge maps of image frames in [21], however it has been noted that the technique is inappropriate for blocks with inadequate edge information. In [22], motion estimation using the 1BT, where image frames are transformed into one-bit/pixel representations by comparing the original image frame against a multibandpass filtered version is proposed. A 17×17 multibandpass filter kernel in the form of

$$K(i,j) = \begin{cases} 1/25, & \text{if } i,j \in [0,4,8,12,16] \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

is used to filter the image frame, and the 1BT bit plane of an image is constructed as

$$B(i,j) = \begin{cases} 1, & \text{if } I(i,j) \geq I_F(i,j) \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where $I_F(i,j)$ represents the filtered version of the image frame $I(i,j)$, obtained by applying the convolution kernel K to I. The 1BT motion estimation proposed in [22] accomplishes a reduction in arithmetic and hardware complexity, however the reconstructed image is (sometimes seriously) degraded due to bad motion vectors resulting from the bit-depth being reduced to a single bit only (particularly for small block sizes). The addition of conditional local searches has been proposed in the modified 1BT to improve the predicted image at the expense of increased computational complexity in [23], however in this case the binary only matching characteristic of the motion estimation scheme is destroyed as additional local MAD matches have to be used. An all binary motion estimation approach using a hierarchical layer structure in the form of a binary pyramid has been proposed in [24] to incorporate a hierarchical scheme for binary matching motion estimation.

Earlier video compression standards such as MPEG1, MPEG2 and H.261 estimate displacement vectors on a macroblock level, and block matching based on 16×16 blocks is generally used, although no particular motion estimation method is specified in the standards [25]. It is shown in [22] that the 1BT can give a reasonable tradeoff between quality and speed for a 16×16 block size, and, therefore, 1BT can be regarded as being suitable for earlier video compression standards. Recent video compression schemes such as MPEG4 and H.264, however, also support smaller block sizes for motion compensation [26]. The 1BT-based motion estimation can result in bad motion vectors for small block sizes as image frames are reduced into two binary classes only and, therefore, 1BT-based motion estimation is not very appropriate for recent video compression schemes. This paper proposes to utilize a 2BT so as to enable the representation of four separate classes for improved motion estimation accuracy, especially at block sizes smaller than 16×16, while preserving the binary matching characteristic. A simple block-by-block 2BT is proposed based on multithresholding making use of mean and standard deviation measures. To avoid blocking effects during the transformation while enabling segmentation to be performed according to local detail, threshold values are computed within a larger threshold window surrounding the transforming block. It is shown that the proposed motion estimation technique improves accuracy in terms of peak signal-to-noise ratio (PSNR) as well as visual accuracy of reconstructed frames compared to 1BT-based motion estimation.

II. Block Motion Estimation Using Two-Bit Transform (2BT)

In the case of binary block matching motion estimation using a reduced number of bit planes it is important that as much information as possible is captured within the bit plane representation. As furthermore the search for motion vectors is typically carried out in a restricted local neighborhood of the corresponding block, local transformations (as proposed in [22]) result in more efficient and appropriate bit plane representations compared to global transformations. One way to realize local transformations is to make use of the well-known local histogram equalization approach [27], where a local window is constructed around the pixel to be processed. However if the transformation is carried out on a pixel-by-pixel basis, the computational load becomes extremely high. In the 2BT proposed in this paper, the transformation is carried out on a block-by-block basis to reduce the computational load of the local transformation.

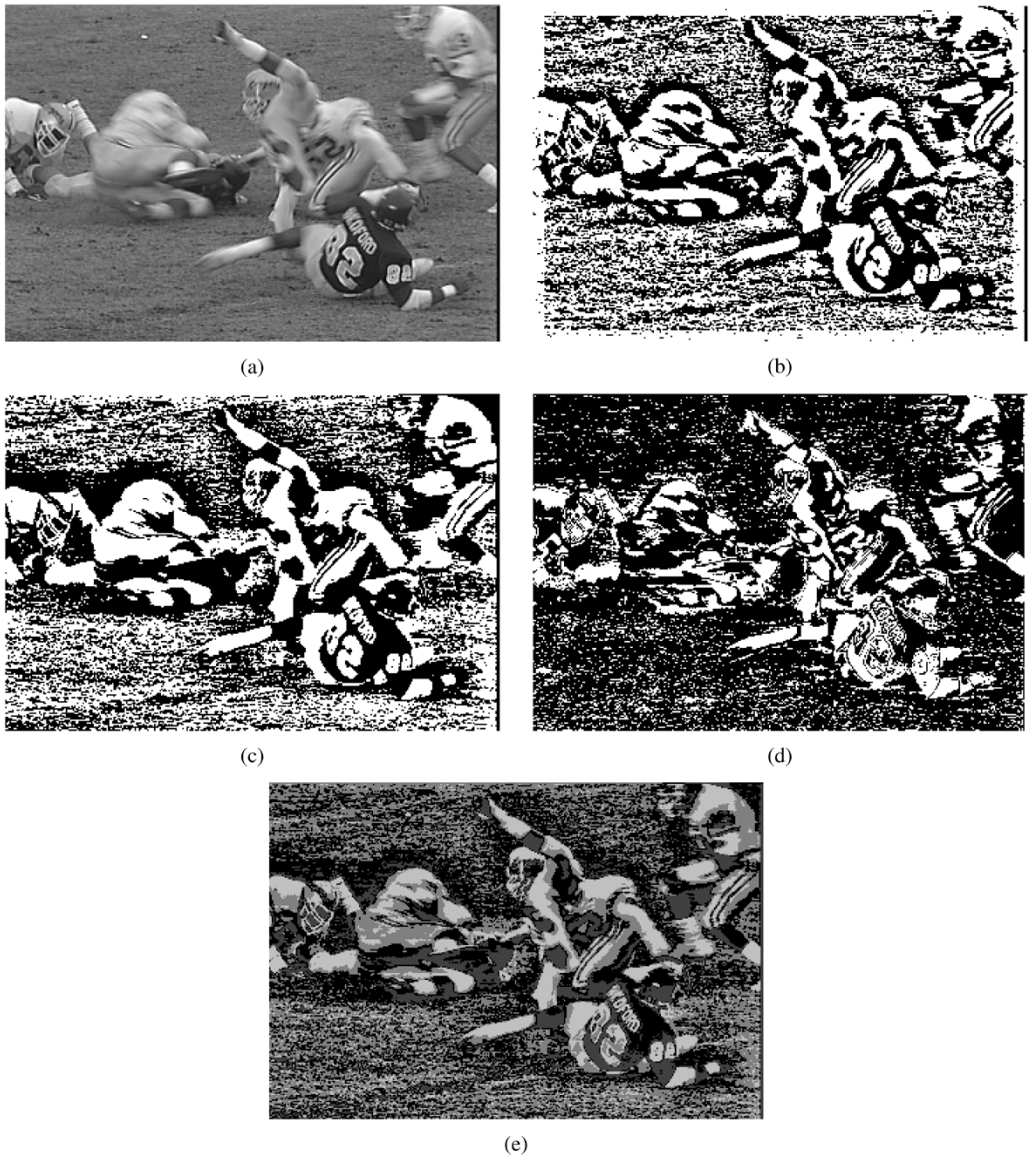

Fig. 1. 1BT and 2BTs for a sample frame of the "football" sequence. (a) Original frame #15. (b) One-bit transform of frame #15. (c) First bit plane $(B_1(i,j))$ of the 2BT of frame #15. (d) Second bit plane $(B_2(i,j))$ of the 2BT of frame #15. (e) The two bit planes of the 2BT superimposed to show the four separate sets.

In the proposed 2BT-based block motion estimation algorithm, image frames are initially divided into nonoverlapping blocks, which constitute the local transformation core. Although the transformation is carried out on a nonoverlapping block-by-block basis, transforming thresholds are evaluated in a larger window surrounding the transforming block, to prevent blocking effects during the local transformation. It is already proposed in [19] to make use of the block mean as a threshold for the 1BT. As a 2BT is targeted in the proposed approach, the standard deviation is used as additional information for constructing reasonable local thresholds. Image thresholding using a local threshold that is computed in the form of

$$T = \mu + k\sigma \tag{6}$$

where μ and σ are the mean and standard deviation values of a local area and k is an empirical constant has already been utilized in the literature [28]. This approach is exploited in the technique proposed in this paper to construct the 2BT. For the proposed 2BT, the mean (μ) and variance (σ^2) values of pixels in the threshold window surrounding each transforming block are computed to obtain statistical information about the distribution of pixel values within the window. If $E[x]$ denotes the expected value obtained by averaging the values of x, the mean and variance is computes as

$$\begin{aligned} \mu &= E[I_{\mathrm{tw}}] \\ \sigma^2 &= E[I_{\mathrm{tw}}^2] - E^2[I_{\mathrm{tw}}] \end{aligned} \tag{7}$$

where I_{tw} shows the pixel values within the threshold window. As the computation of the standard deviation from the variance requires a complex square root operation, a linear approximation is used instead, to reduce the computational complexity. The tangent line-based linear approximation states that any function $f(x)$ can be approximated by

$$f(x) \approx f(a) + (x-a)f'(a) \tag{8}$$

TABLE I
AVERAGE PSNR (dB) OF SEVERAL SEQUENCES RECONSTRUCTED BY VARIOUS MOTION ESTIMATION TECHNIQUES, WITH FS AND A BLOCK SIZE OF 16 × 16 PIXELS WITH A MOTION VECTOR SEARCH RANGE OF 16 PIXELS

Method	Video Sequences (Frame Size, Sequence Length)					
	Football (352×240) (125 frames)	Foreman (352×288) (299 frames)	Tennis (352×240) (112 frames)	Flowergarden (352×240) (115 frames)	Mobile (352×240) (140 frames)	Coastguard (352×288) (299 frames)
MAD	22.88	32.10	29.87	23.79	22.99	30.47
1BT [22]	21.83	30.36	28.77	23.32	22.71	29.84
2BT	22.08	30.71	28.89	23.43	22.72	29.93

TABLE II
AVERAGE PSNR (DECIBELS) OF SEVERAL SEQUENCES RECONSTRUCTED BY VARIOUS MOTION ESTIMATION TECHNIQUES, WITH FS AND A BLOCK SIZE OF 8 × 8 PIXELS WITH A MOTION VECTOR SEARCH RANGE OF 8 PIXELS

Method	Video Sequences (Frame Size, Sequence Length)					
	Football (352×240) (125 frames)	Foreman (352×288) (299 frames)	Tennis (352×240) (112 frames)	Flowergarden (352×240) (115 frames)	Mobile (352×240) (140 frames)	Coastguard (352×288) (299 frames)
MAD	24.73	32.89	31.25	25.22	23.88	31.58
1BT [22]	22.72	29.82	29.23	24.10	22.71	29.20
2BT	23.36	30.64	29.91	24.55	22.99	30.50

for any x near to the value a, where $f'(a)$ denotes the derivative of the function being computed at a. Therefore, a linear approximation to the square-root operation of the variance is used to obtain the approximate standard deviation as

$$\sigma_a = 15 + 0.0125\sigma^2 \tag{9}$$

where σ_a denotes the approximate standard deviation. This corresponds to the linear tangent line approximation of the square root function for $a = 900$ (for a standard deviation of thirty), with the scale factor being slightly reduced to compensate for the large range of values. This linear approximation gives close approximations for large standard deviations, and while greater approximation values are obtained for small standard deviations it is still reasonable to use this approximation as it will facilitate to discriminate between distinct features that have rather different pixel values.

The proposed two bit transform then converts image frames block-by-block into a two-bit/pixel representation, i.e., constructs two bit planes, using the mean and approximate standard deviation values computed in the surrounding threshold window of each block. Because threshold windows will be overlapping, the thresholds computed for each block will be contingent on surrounding pixels so that blocking effects are avoided during the transformation. The 2BT can be expressed as

$$B_1(i,j) = \begin{cases} 1, & \text{if } I(i,j) \geq \mu \\ 0, & \text{otherwise} \end{cases}$$
$$B_2(i,j) = \begin{cases} 1, & \text{if } I(i,j) \geq \mu + \sigma_a \quad \text{or} \quad I(i,j) \leq \mu - \sigma_a \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where $B_1(i,j)$ and $B_2(i,j)$ represent the resulting two bit planes. The 2BT enables the representation of four separate classes resulting in improved motion estimation accuracy compared to one bit/pixel representations. The utilization of local mean and standard deviation measures in the computation of the thresholds of the 2BT enables the computation of reasonable local thresholds.

Fig. 1 shows the 1BT, obtained using (5), and the 2BT, obtained using (10) (with an 8×8 transformation block and a 40×40 surrounding threshold window size), of a sample frame of the "football" test sequence. The two bit planes of the 2BT are shown separately as well as superimposed so as to display the four different sets obtained by 2BT using different gray levels. It is clearly seen that blocking effects are avoided during the 2BT as a result of using larger threshold windows surrounding the transforming blocks. It can further be observed that the 1BT result resembles the first bit plane of the 2BT, i.e., $B_1(i,j)$ obtained by window mean thresholding, and the second bit plane of the 2BT enables additional features to be represented.

The motion vector of a block is decided based on the number of nonmatching points measure, which can be expressed in the form of

$$\mathrm{NNMP}(m,n) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} \left\{B_1^t(i,j) \oplus B_1^{t-1}(i+m, j+n)\right\} \Big\| \left\{B_2^t(i,j) \oplus B_2^{t-1}(i+m, j+n)\right\}, \quad -s \leq m, \quad n \leq s-1 \tag{11}$$

where $\mathrm{NNMP}(m,n)$ is the number of nonmatching points computed for the displacement of (m,n) pixels. $B_{1,2}^t(i,j)$ and $B_{1,2}^{t-1}(i,j)$ are the 2BTs of the current and previous image frames, respectively, s determines the search range, N determines the block size, $\oplus$ represents Boolean exclusive-or (XOR) and $\|$ represents Boolean-Or operation. The matching criterion is defined using Boolean operators only to enable fast execution and simple implementation. It is possible to make XOR comparisons for the 2BT bit planes $B_1(i,j)$ and $B_2(i,j)$ of the current and previous frame separately, in parallel to speed up the computation process, and then obtain the number of nonmatching points (NNMP) measure by the Boolean OR result of the two bit plane correlations, which could again be executed in parallel. Note that an NNMP measure is introduced if the corresponding pixels differ in any of the 2BT bit planes, hence only pixels that have exactly the same 2BT are counted as a match. As the motion vector search strategy is unrelated to

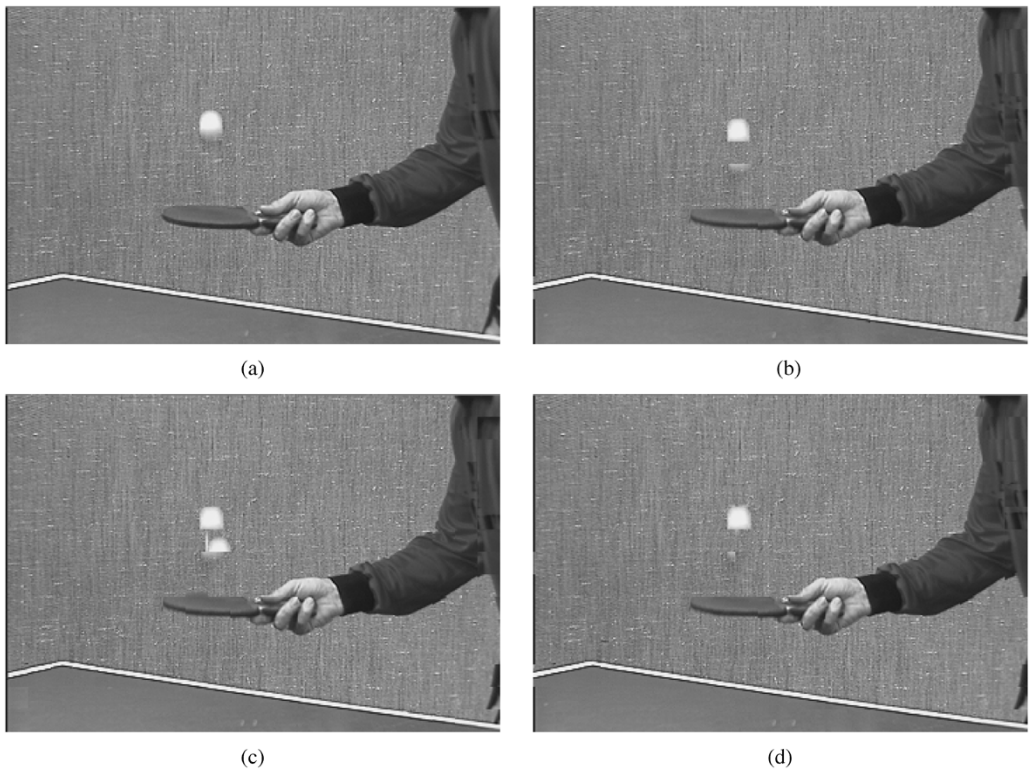

Fig. 2. Sample results for the "tennis" test sequence, with a block size of 16×16 and a search range of 16 pixels. (a) Original frame #29. (b) Frame #29 reconstructed from the previous frame with motion vectors of MAD (PSNR: 27.51 dB). (c) Frame #29 reconstructed from the previous frame with motion vectors of 1BT (PSNR: 26.73 dB). (d) Frame #29 reconstructed from the previous frame with motion vectors of 2BT (PSNR: 27.04 dB).

the matching criteria, it is possible to utilize any search strategy in conjunction with the proposed 2BT-based matching process.

Compared to the 1BT proposed in [22] the 2BT uses one additional bit for the representation of pixels as two bit planes are utilized. The matching evaluation of the 1BT requires a single XOR operation per pixel, while two XOR operations and one Boolean OR operation are needed for the 2BT, however as well as the 1BT, the 2BT can also make use of parallel comparisons to speed up the process.

The computational complexity of the 1BT itself is comprised of twenty-five addition operations, one multiplication operation and one comparison per pixel (for the 1BT proposed in [22]). The computational complexity of the proposed 2BT changes depending on the transformation and threshold window sizes. It is possible to achieve a fast computation of the 2BT if the threshold window size is an integer multiple of the transformation block size, as in this case mean and variances can independently be computed for nonoverlapping blocks of size equal to the transformation block size, and these values can then be combined to obtain the mean and approximate standard deviation of surrounding threshold windows. For comparison purposes the computational load of the 2BT is evaluated for 8×8 transformation block size and 40×40 surrounding threshold window size, which is also used in the demonstration of experimental results. In this case, the entire image frame is initially divided into nonoverlapping blocks of size 8×8. For any block, the sum of all pixel values within that block can be computed using one addition operation per pixel and the sum of all pixel values squared can be computed using one multiplication and one addition operations per pixel (Note that this is equivalent to computing $E[X]$ and $E[X^2]$ except for the averaging that is left to the combination stage, where X denotes the pixel value within any block). Once the sum of all pixel values and all pixel values squared is computed for the nonoverlapping blocks, the sum of all pixel values within a threshold window can be computed using twenty five addition operations and the sum of all pixel values squared requires another twenty five additions, as any 40×40 threshold window will comprise a total of twenty five nonoverlapping 8×8 blocks. Afterwards, two multiplication operations are required for averaging to compute the expected values, after which the mean value is already obtained. In order to compute the variance of the threshold window, one multiplication is required to square the expected value of pixels and one subtraction operation is required to subtract this value from the expected value of pixel values squared, as shown in (7). The approximate standard deviation can then be computed from the variance using one multiplication and one addition operations, as shown in (9). Furthermore one addition operation is required to add the approximate standard deviation to the mean, and one subtraction operation is required to subtract the approximate standard deviation from the mean so as to obtain the necessary thresholds for the 2BT. Hence, once the sum of all pixel values and all pixel values squared is computed for the nonoverlapping blocks, the computation of the thresholds for any 8×8 transformation block, i.e., 64 pixels, requires 52 addition operations, 4 multiplication operations and 2 subtraction operations; which is equivalent to 0.8125 addition operations, 0.0625 multiplication operations and 0.031 25 subtraction operations per pixel. Therefore, the entire threshold computation process requires 2.8125 addition, 1.0625 multiplication and 0.031 25 sub-

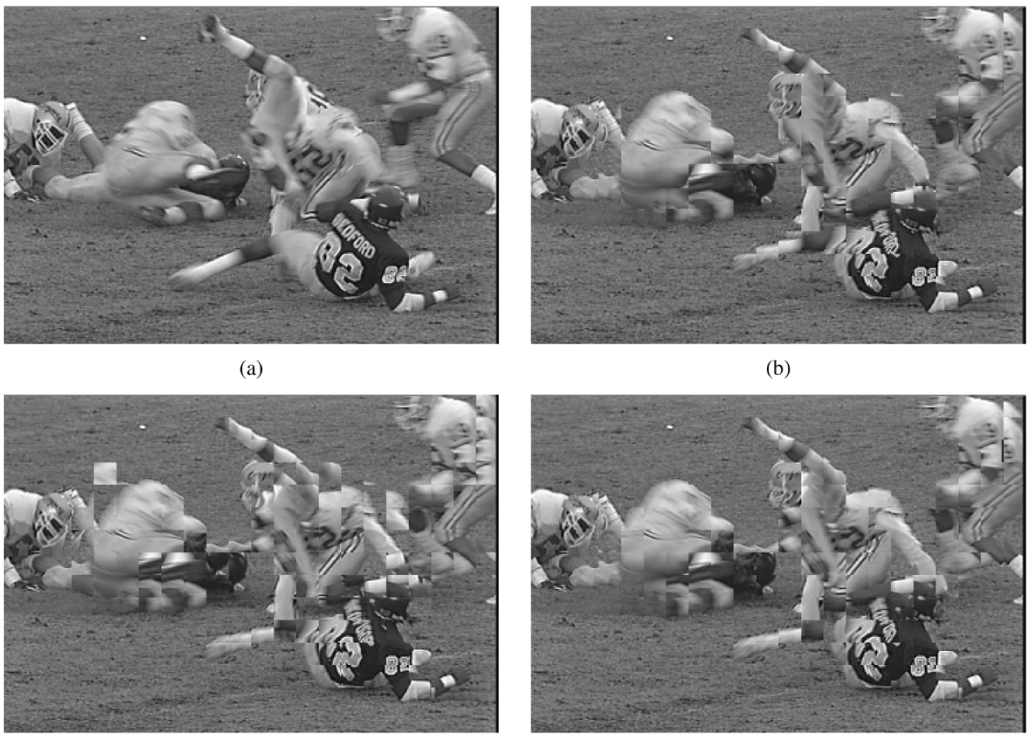

Fig. 3. Sample results for the "football" test sequence, with a block size of 16×16 and a search range of 16 pixels. (a) Original frame #15. (b) Frame #15 reconstructed from the previous frame with motion vectors of MAD (PSNR: 22.97 dB). (c) Frame #15 reconstructed from the previous frame with motion vectors of 1BT (PSNR: 21.25 dB). (d) Frame #15 reconstructed from the previous frame with motion vectors of 2BT (PSNR: 22.08 dB).

traction operations per pixel, while another three comparison operations and one Boolean OR operation per pixel are required to construct the final 2BT.

III. Experimental Results

Motion estimation has been performed for various video sequences using FS with MAD matching, the 1BT proposed in [22], and the 2BT proposed in this paper. Exhaustive FS, that evaluates every pixel location within the search range to determine the best suitable motion vector within the search window, is utilized to asses the performance of the matching criteria so that any possible search strategy influence is avoided. Note that other 1BTs proposed are not evaluated for comparison as the procedure of [22] is known as the most accurate 1BT process. Transforming blocks of size 8×8 pixels with 40×40 surrounding threshold windows are utilized for the 2BT process. PSNR values for the video sequences with image frames reconstructed from previous frames are utilized for statistical evaluation of motion estimation accuracy. For the PSNR measure, image frames are reconstructed from previous frames using motion vectors obtained using FS with the corresponding motion estimation procedure. These reconstructed image frames are furthermore used for visual evaluation of motion estimation accuracy.

The motion estimation accuracy is evaluated for a block size of 16×16 pixels with a search range of 16 pixels, and a block size of 8×8 pixels with a search range of 8 pixels. The PSNR results for various test sequences are given in Tables I and II, respectively. Table I shows that for a block size of 16×16 pixels with a search range of 16 pixels, 1BT is usually within 1 dB of MAD matching, hence the 1BT performance is reasonably close to MAD. This is the main reason why 1BT has been proposed for blocks of size 16×16 pixels, and is, therefore, suitable for earlier video coding standards such as MPEG1 and MPEG2. Table I shows that the performance improvement of 2BT over 1BT is minor for a block size of 16×16 pixels if 1BT is already close to MAD. It is seen from the 1BT bit plane given in Fig. 1 that in the case of larger block sizes sufficient information is captured within any block so that a reasonable match can be achieved. It is furthermore observed from Fig. 1 that the first bit plane of the 2BT is similar to the 1BT bit plane. Noting that no match is counted in the 2BT unless a match in both bit planes is achieved, it is possible to say that for larger block sizes the improvement obtained by the second bit plane is limited because ubiquitous matching of the comparatively large binary pattern is already rare. It is observed from Table I that for a block size of 16×16 pixels the improvement obtained by 2BT over 1BT depends on the relative performance of 1BT. If 1BT shows a good performance and is within about 0.2 dB of MAD nearly no improvement is achieved by 2BT, while 2BT achieves an improvement of about 0.1 dB if 1BT is within about 0.5 dB of MAD, an improvement of about 0.2 dB is achieved if 1BT is within only about 1 dB of MAD, and an even larger improvement is achieved using 2BT if 1BT performance is yet inferior. Table II shows that for a block size of 8×8 pixels with a search range of 8 pixels, 1BT can result in a PSNR of up to 3 dB lower than MAD matching and in this case 2BT can result in a PSNR more than 1dB better than 1BT. It is observed that 1BT is not suitable for smaller block sizes as the performance degrades seriously. The main reason for the performance degradation of 1BT is that

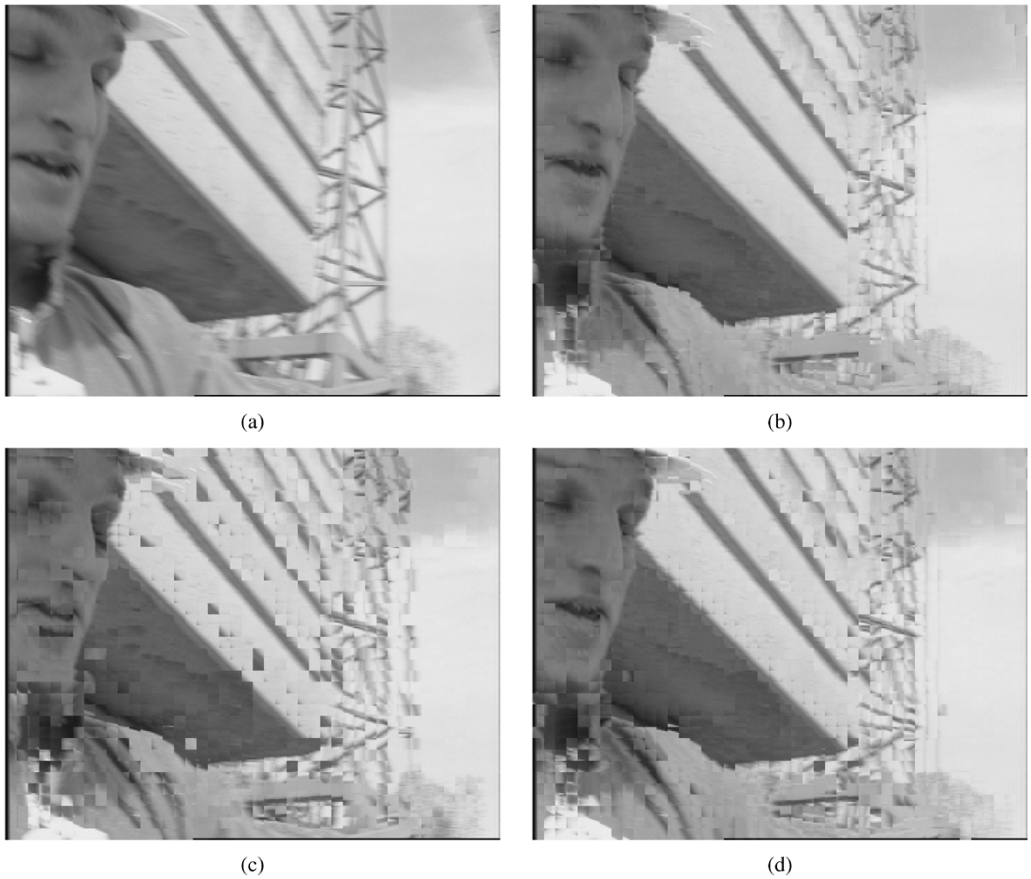

Fig. 4. Sample results for the "foreman" test sequence, with a block size of 8×8 and search range of 8 pixels. (a) Original frame #185. (b) Frame #185 reconstructed from the previous frame with motion vectors of MAD (PSNR: 26.08 dB). (c) Frame #185 reconstructed from the previous frame with motion vectors of 1BT (PSNR: 22.05 dB). (d) Frame #185 reconstructed from the previous frame with motion vectors of 2BT (PSNR: 23.52 dB).

small blocks can contain insufficient detail resulting in ubiquitous matching of the binary pattern. The second bit plane of the 2BT reduces incorrect matches and significantly improves the matching performance. It is observed from Table II that 1BT is not very appropriate for recent video coding standards, such as MPEG 4, that employ smaller block sizes for motion compensation, while 2BT can still be utilized as a reasonable low-complexity motion estimation approach. Another important point is that recent video compression schemes propose variable size block motion estimation, where the block size can be reduced to improve the performance. It is naturally expected that the reconstructed frame should be superior if smaller block sizes are utilized. However this is not always the case for 1BT, as observed from Tables I and II, the PSNR can drop significantly although the block size is reduced which is the case for the "coastguard" and "foreman" test sequences, and it is, therefore, not very feasible to use 1BT for variable block size motion estimation with small block sizes. The proposed 2BT actually results in considerably improved PSNR results when the block size is reduced, except for the "foreman" sequence for which the performance stays about the same, and is, therefore, more suitable for recent video compression schemes that utilize a variable block size motion estimation approach.

Visual assessment of motion estimation accuracy clearly shows that the proposed 2BT results in visually more appropriate frames compared to the 1BT. The 1BT can result in bad motion vectors, particularly for small block sizes, while the amount of bad motion vectors is seriously reduced in the 2BT. Fig. 2 shows an original frame of the "tennis" test sequence, and frames reconstructed from the previous frame using MAD, 1BT, and 2BT motion vectors. It can be seen that even at a block size of 16×16 pixels, 1BT can result in bad motion vectors as observed from the two ping-pong balls present in the reconstructed frame. In this case, bad motion vectors are clearly avoided in the proposed 2BT that even outperforms MAD matching in terms of visual appearance for the presented frame, as the lower half of the second ball displayed incorrectly in the MAD reconstructed frame is not present in the 2BT reconstructed frame. Fig. 3 shows sample results for the "football" sequence, with a block size of 16×16 pixels and a search range of 16 pixels. It can be seen that 1BT again results in bad motion vectors that seriously degrade visual appearance of the reconstructed frame, while bad motion vectors are avoided in 2BT that performs about the same as MAD in terms of visual appearance. In Fig. 4 it is particularly seen that 1BT can result in a substantial amount of bad motion vectors for a block size of 8×8 pixels, making it mostly infeasible for motion estimation with small block sizes. Although some adverse motion vectors are also encountered for the 2BT approach compared to MAD results, the amount of bad motion vectors and their visual impact is significantly lower than 1BT, showing that the proposed 2BT provides a reasonable low-complexity motion estimation approach.

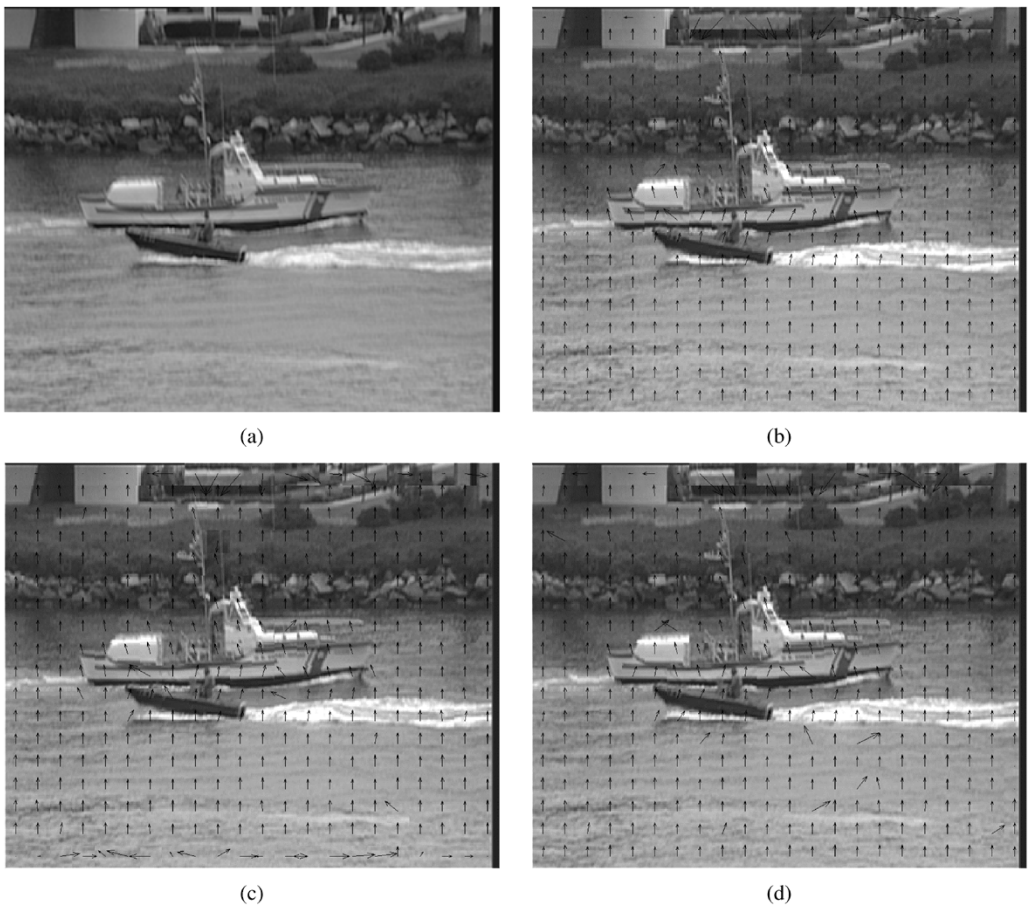

Fig. 5. Sample results for the "coastguard" test sequence, with a block size of 16×16 and search range of 16 pixels with estimated motion vectors overlaid. (a) Original frame #70. (b) Frame #70 reconstructed from the previous frame with motion vectors of MAD (PSNR: 28.47 dB). (c) Frame #70 reconstructed from the previous frame with motion vectors of 1BT (PSNR: 27.29 dB). (d) Frame #70 reconstructed from the previous frame with motion vectors of 2BT (PSNR: 27.53 dB).

A uniform motion field is important to reduce the number of bits required to encode motion vectors so as to increase coding efficiency. As observed from Fig. 5 that shows the motion vectors estimated by MAD, 1BT and 2BT for a sample frame of the "coastguard" sequence overlaid on the reconstructed image frames, 1BT, and 2BT result in about similar motion fields slightly less uniform compared to MAD, for a block size of 16×16 pixels. For a block size of 8×8 pixels on the other hand, the motion field obtained by the 2BT is fairly more uniform than that obtained by 1BT as the 1BT results in a considerably higher number of bad motion vectors, however 2BT still results in a less uniform motion field compared to MAD.

Linear approximation of the standard deviation as given in (9) is observed to have an effect of only ± 0.05 dB on the PSNR performance of the proposed 2BT. For some cases the linear approximation results in an insignificant reduction in the PSNR, and for some cases it even results in an insignificant increase in the PSNR. Therefore, it is realistic to use the linear approximation of the standard deviation to avoid the complexity of the square root operation.

IV. Conclusion

A novel 2BT-based low-complexity motion estimation scheme is proposed in this paper. Initially image frames are converted into two-bit representations by a simple block-by-block two bit transform based on multithresholding with mean and linearly approximated standard deviation values, with threshold values being computed in a larger surrounding window of the transforming block to avoid blockiness. The 2BT makes use of lower bit-depth and binary matching criteria characteristic of 1BTs for reduced block motion estimation complexity, but improves motion estimation accuracy and reduces the amount of bad motion vectors particularly for small block sizes compared to 1BTs. It is shown that the proposed 2BT-based motion estimation technique improves motion estimation accuracy in terms of PSNR of reconstructed frames and also results in visually more accurate frames subsequent to motion compensation compared to the 1BT-based motion estimation approach, and in general performs reasonably well to be used as an alternative low-complexity motion estimation approach.

The 2BT provides a low complexity motion estimation approach that is in between the 1BT and the MAD approach in terms of speed against quality tradeoff, very close to the 1BT. It might be possible to further increase the number of bit planes used during matching for additional quality improvement, however the more bit planes are utilized the more difficult it will become to construct reasonable representation techniques for the bit planes and also the complexity of the system will increase. It is, therefore, reasonable to use the proposed 2BT strategy, as construction of the two bit planes is simple and fast. In future

studies it is planned to investigate possible threshold prediction strategies to further improve the performance.

REFERENCES

[1] Z.-L. He, C.-Y. Tsui, K.-K. Chan, and M. L. Liou, "Low-power VLSI design for motion estimation using adaptive pixel truncation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 8, pp. 669–678, Aug. 2000.

[2] J. M. Jou, P.-Y. Chen, and J.-M. Sun, "The gray prediction search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 6, pp. 843–848, Sep. 1999.

[3] V. G. Moshnyaga, "A new computationally adaptive formulation of block-matching motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 1, pp. 118–124, Jan. 2001.

[4] M. Mattavelli and G. Zoia, "Vector-tracing algorithm for motion estimation in large search windows," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 12, pp. 1426–1437, Dec. 2000.

[5] T. Koga, K. Linuma, A. Hirano, Y. Lijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in *Proc. NTC'81*, 1981, pp. G5.3.1–G5.3.5.

[6] J. Jain and A. Jain, "Displacement measurement and its application in internal image coding," *IEEE Trans. Commun.*, vol. 29, no. COM–12, pp. 1799–1808, Dec. 1981.

[7] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Trans. Commun.*, vol. 38, no. 7, pp. 950–953, Jul. 1990.

[8] R. Srinivasan and K. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. Commun.*, vol. 33, no. 8, pp. 888–896, Aug. 1985.

[9] K. M. Nam, J.-S. Kim, R.-H. Park, and Y. S. Shim, "A fast hierarchical motion vector estimation algorithm using mean pyramid," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, no. 4, pp. 344–351, Aug. 1995.

[10] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, no. 4, pp. 438–442, Aug. 1994.

[11] L. M. Po and W. C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, pp. 313–317, Jun. 1996.

[12] L. K. Liu and E. Faig, "A block-based gradient descent search algorithm for block motion estimation in video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 4, pp. 419–422, Aug. 1996.

[13] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast block-matching motion estimation," *IEEE Trans. Image Process.*, vol. 9, no. 2, pp. 287–290, Feb. 2000.

[14] C. J. Kuo, C. H. Yeh, and S. F. Odeh, "Polynomial search algorithm for motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 5, pp. 813–818, Aug. 2000.

[15] B. Liu and A. Zaccarin, "New fast algorithm for motion estimation of block motion vectors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, no. 2, pp. 148–157, Apr. 1993.

[16] K. T. Choi, S. C. Chan, and T. S. Ng, "A new fast motion estimation algorithm using hexagonal subsampling pattern and multiple candidates search," in *Proc. ICIP*, vol. 2, 1996, pp. 497–500.

[17] Y. Wang, Y. Wang, and H. Kuroda, "A globally adaptive pixel-decimation algorithm for block-motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 6, pp. 1006–1011, Sep. 2000.

[18] J. Kim and T. Choi, "A fast full-search motion-estimation algorithm using representative pixels and adaptive matching scan," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 7, pp. 1040–1048, Oct. 2000.

[19] J. Feng, K.-T. Lo, H. Mehrpour, and A. E. Karbowiak, "Adaptive block matching motion estimation algorithm using bit plane matching," in *Proc. ICIP'*, 1995, pp. 496–499.

[20] X. Lee and Y. Zhang, "A fast hierarchical motion-compensation scheme for video coding using block feature matching," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 6, pp. 627–635, Dec. 1996.

[21] M. M. Mizuki, U. Y. Desai, I. Masaki, and A. Chandrakasan, "A binary block matching architecture with reduced power consumption and silicon area requirement," in *Proc. IEEE ICASSP*, vol. 6, Atlanta, GA, 1996, pp. 3248–3251.

[22] B. Natarajan, V. Bhaskaran, and K. Konstantinides, "Low-complexity block-based motion estimation via one-bit transforms," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 4, pp. 702–706, Aug. 1997.

[23] P. H. W. Wong and O. C. Au, "Modified one-bit transform for motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 7, pp. 1020–1024, Oct. 1999.

[24] J.-H. Luo, C.-N. Wang, and T. Chiang, "A novel all-binary motion estimation (ABME) with optimized hardware architectures," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 8, pp. 700–712, Aug. 2002.

[25] A. M. Tekalp, *Digital Video Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1995, pp. 432–455.

[26] R. Xiong, F. Wu, S. Li, Z. Xiong, and Y.-Q. Zhang, "Exploiting temporal correlation with adaptive block-size motion alignment for 3 D wavelet coding," in *Proc. SPIE Visual Communications and Image Processing Conf.*, vol. 5308, pp. 144–155.

[27] R. C. Gonzales and R. E. Woods, *Digital Image Processing*. Reading, MA: Addison-Wesley, 2002, pp. 182–183.

[28] W. Niblack, *An Introduction to Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1986, pp. 115–116.