TWO-BIT TRANSFORM BASED BLOCK MOTION ESTIMATION USING SECOND DERIVATIVES

Nam-Joon Kim 1, Sarp Ertürk 2, Member, IEEE, and Hyuk-Jae Lee 1

1 School of Electrical Engineering and Computer Science, Seoul National University, Seoul, South Korea
2 Department of Electronics and Telecommunications Engineering, University of Kocaeli, Kocaeli, Turkey

ABSTRACT
A novel two-bit transform based block motion estimation (ME) algorithm is presented in this paper. The proposed approach achieves more effective binarization of image frames than the previous 2BT approach by making use of the positive and negative derivative values separately, which are computed from the second derivatives of a local area as the threshold value for the second bit plane. The second derivatives are also used to find the most accurate motion vectors (MVs) and to reduce computational complexity. Experimental results show that the proposed binary motion estimation algorithm improves motion estimation accuracy and furthermore provides faster processing time in flat or background regions with an acceptable bit-rate increase. In applying the proposed 2BT-SD approach in a real video compression standard, a further reduction of ME processing time with reasonably good compression efficiency is achieved by 2BT-SD based integer ME (IME) followed by full resolution fractional ME.

1. INTRODUCTION
Motion estimation is the process by which elements in a frame (typically blocks in the current frame) are correlated to best fitting elements in other frames (typically the previous frames) by the estimated amount of motion [1]. A widely used technique for motion estimation is the full search (FS) block matching algorithm (BMA), and the best criterion for this matching process is the mean absolute difference (MAD) or mean square error (MSE). However, motion estimation using FS with MAD or MSE criterion requires very high computation power. To reduce the computation requirements, many motion estimation algorithms have been proposed. These can mainly be grouped as the following three categories [2] fast search techniques that select a subset of possible search candidate locations, techniques based on various forms of pixel pattern or motion field decimation that employs a certain sub-sampling of the pixel pattern or motion field, and techniques that exploit different matching criteria instead of the classical MAD.

Among the techniques exploiting a new matching criterion, an early achievement is the one-bit transform (1BT) [3], where image frames are transformed into one-bit/pixel representations by comparing the original image frame against a multi band-pass filtered version of the image. This provides a simple and low computation approach, and gives a reasonable tradeoff between quality and speed. However the reconstructed image by 1BT is often degraded due to inaccurate MVs as a result of the bit-depth being reduced to only a single bit [4]. A two-bit transform (2BT) using two bit-planes for block matching is proposed in [4]. This approach uses local mean and variances computed for a larger threshold window, surrounding the transforming block, so as to obtain the regional thresholds for a two-bit/pixel representation. The 2BT improves motion estimation accuracy compared with 1BT, especially for block sizes smaller than 16x16. Recently, a multiplication-free one-bit transform (MF-1BT) approach has been proposed in [5] to further reduce the computational complexity by eliminating floating point multiplications of the 1BT.

In this paper, a new two-bit transform based block motion estimation algorithm using the second derivatives is proposed. The proposed binary representation is called a two-bit transform with second derivatives and denoted by 2BT-SD in short hereafter in this paper. The proposed approach uses local mean values and second derivatives to obtain regional thresholds for a two-bit representation. Because blocks with inadequate edge information are inappropriate for binary block matching, 2BT-SD solves this problem by using the correlation between MVs in spatial domain, and achieves a reduction of computation. The simulation results show that the proposed motion estimation technique improves the peak signal-to-noise ratio (PSNR), compared with 1BT, MF-1BT and 2BT-based motion estimation techniques. It is also shown by simulation that 2BT-SD can provide an effective tradeoff between accuracy and computing time of motion estimation.

2. TWO-BIT TRANSFORM BASED MOTION ESTIMATION USING SECOND DERIVATIVES

2.1. The two-bit transform using second derivatives
Image frames are partitioned into nonoverlapping blocks, each of which is independently transformed to a two-bit/pixel representation. The first bit-plane of 2BT-SD is
constructed in the same method as 2BT [4] by using the mean of the window surrounding the block to be transformed.

\[ B_A(i,j) = \begin{cases} 1, & \text{if } I(i,j) \geq E[I_m] \\ 0, & \text{otherwise} \end{cases} \]  

(1)

where \( B_A(i,j) \) represents the first plane transformed from the image frame \( I(i,j) \), \( E[I_m] \) is the mean of the pixel values in the surrounding window and \( I_m \) represents the pixel values within the surrounding window. The surrounding window is called the threshold window and its size is larger than the block size in order to avoid blocking effects caused by slightly different transformations applied to different blocks [4].

The second bit-plane is constructed with the information obtained from the second derivatives as follows:

\[ B_B(i,j) = \begin{cases} 1, & \text{if } I(i,j) \geq E[I_m] + PM \text{ or } I(i,j) \leq E[I_m] - NM \\ 0, & \text{otherwise} \end{cases} \]  

(2)

where \( B_B(i,j) \) represents the second bit plane and the positive mask (PM) and negative mask (NM) are obtained from positive and negative second derivatives, respectively, as follows.

\[ PM = \frac{S_{\text{pos}}(I_m)}{p\text{pixs}[I_m]} + \alpha, \quad NM = \frac{S_{\text{neg}}(I_m)}{n\text{pixs}[I_m]} - \alpha \]  

(3)

where \( S_{\text{pos}}(I_m) \) and \( S_{\text{neg}}(I_m) \) are the sums of positive and negative second derivatives, respectively, with their absolute values larger than a pre-set threshold \( T_\alpha \). \( p\text{pixs}[I_m] \) and \( n\text{pixs}[I_m] \) are the total numbers of pixels with their absolute value larger than \( T_\alpha \) for positive and negative second derivatives, respectively. In other words, \( PM[I_m] \) and \( NM[I_m] \) are the arithmetic averages of the positive and negative second derivatives, respectively, with their absolute values larger than \( T_\alpha \). The second derivatives are computed using a Laplacian mask and constant \( \alpha \) in (3) is an offset value pre-set experimentally to 9.

2.2. Motion estimation in the two bit planes
In order to derive an MV with the two-bit/pixel representation, the suitable matching criterion is necessary. Thus, an NNMP (Number of Non-Matching Points) is defined as the measure of the correlation between two blocks.

\[ \text{NNMP}(m,n) = \sum_{i+j=0}^{m+1} \{ B'_l(i,j) \oplus B_{l}^{-1}(i+m,j+n) \} \oplus \]  

(4)

\[ \{ B'_l(i,j) \oplus B_{l}^{-1}(i+m,j+n) \}, \quad s \leq m, \quad n \leq s \cdot 1 \]

where \((m,n)\) denotes the candidate displacement and \( B'_l(i,j) \) and \( B_{l}^{-1}(i,j) \) are the two bit-planes for the current and previous image frames, respectively. \( N \) represents the block size, \( s \) represents the search range, \( \oplus \) denotes the bitwise exclusive-or (XOR). The operation \( \oplus \) denotes a binary operation which generates a two-bit binary number.

\[ \text{MB-type} = \begin{cases} 1 - \text{type}, & \text{if } (p\text{pixs} + n\text{pixs}) < \text{VALID\_PIXEL\_NUM} \\ S - \text{type}, & \text{otherwise} \end{cases} \]  

(6)

Fig. 1. The current macroblock with its neighboring macroblocks used for spatial prediction of the MV of the current macroblock

Let a and b be two inputs of operation \( \odot \). Then, the output of \( \odot \) is \((a \text{ AND } b) + 2 \text{ + (a OR b)}\).

For the derivation of the MV, two candidate displacements that have the two smallest NNMP values are derived. If the difference between the two NNMP values is greater than the smallest NNMP value is chosen as the final MV. If the difference of the two smallest NNMP values is less than or equal to 1, then the MV closer to the predicted motion vector (PMV) is chosen as the final MV. The PMV is derived from the neighboring blocks as follows:

\[ PMV = MV_A + MV_B + MV_C - \min(MV_A, MV_B, MV_C) \]  

\[ - \max(MV_A, MV_B, MV_C) \]  

(5)

where \( MV_A, MV_B, MV_C \) are the MVs of the macroblocks, \( MB_A, MB_B \) and \( MB_C \) in Fig 1, respectively. \( \min(MV_A, MV_B, MV_C) \) and \( \max(MV_A, MV_B, MV_C) \) are operations selecting the smallest and largest values among the horizontal and vertical components of \( MV_A, MV_B, MV_C \), respectively. The above motion vector prediction is one of the widely used MV prediction techniques.

3. APPLICATIONS OF 2BT-SD TO H.264 ENCODING

3.1. Complexity reduction using edge information
The side information obtained during the derivation of 2BT-SD can be used to further reduce the complexity of video compression. For example, the second derivatives computed to obtain 2BT-SD include edge information. Note that motion estimation may not yield an accurate MV for the blocks with insufficient edge information. In this case, this section proposes to use the PMV obtained from its neighboring blocks as (5) instead of the MV derived by finding the smallest NNMP among the candidates in the search range. This PMV may require a less amount of computation yet to provide reasonable accuracy when compared with the MV derived by finding the best-matched vector in the search range.

In order to determine which MV to be selected between the PMV and the best-matched MV, the state of each block is defined. This decision is made for every \( 16 \times 16 \) block (MB). The state information of the current MB denoted by MB-type is classified as either an S-type or I-type as follows:

\[ MB - \text{type} = \begin{cases} 1 - \text{type}, & \text{if } (p\text{pixs} + n\text{pixs}) < \text{VALID\_PIXEL\_NUM} \\ S - \text{type}, & \text{otherwise} \end{cases} \]  

1616
where $p^{\text{pix}}$ and $n^{\text{pix}}$ are the numbers of pixels in the MB with the positive and negative second derivatives, respectively, with their value larger than the pre-defined $T_d$ and VALID PIXEL_NUM is a constant pre-set threshold experimentally chosen for the best motion estimation. Equation (6) implies that an MB is labeled as I-type if it has inadequate edge information and it is labeled S-type otherwise. For an I-type MB, the PMV is chosen as the MV. For an S-type MB, the MV predicted from its neighboring MBs is, in general, more accurate than the MV derived by finding the best matched displacement in a search area, because it does not have sufficient edge information. However, it is observed that the PMV is often not very accurate even for an I-type MB if it is the S-type in the previous frame. To avoid the misprediction of an MV in this case, the MV is predicted only in the case when the MB is the I-type for both the previous and current frames. Note that $p^{\text{pix}}$ and $n^{\text{pix}}$ can be obtained during the two-bit transform so that very little additional computation is required to compute (6). Thus, computational complexity for motion estimation is significantly reduced for an I-type MB.

For S-type MBs, another short-cut derivation of an MV is proposed. The idea comes from the fact that the MVs of adjacent blocks in a background region may have the same values. For the current MB in Fig. 1, if the MVs of adjacent macroblocks (MB_A, MB_B, MB_C, and MB_D as shown in Fig. 1) are all the same, the current MB may be located in a background region so that the current MV is chosen to be the same MV as its four neighboring MBs.

### 3.2. Integer motion estimation with 2BT-SD

In general, a video compression standard has a motion estimation decomposed into two steps: integer motion estimation and fractional motion estimation. The result of integer motion estimation is used as the starting point of fractional motion estimation. In general, integer motion estimation requires larger computation time than fractional motion estimation while the accuracy of motion estimation is determined finally by fractional motion estimation. This paper proposes to use 2BT-SD for integer motion estimation while a standard 8-bit representation per pixel is used for fractional motion estimation. With the use of 2BT-SD for integer motion estimation, the computation time can be significantly reduced while the accuracy of the final MV is maintained with the fractional motion estimation based on 8-bit representation.

The accuracy of the MV derived by 2BT-SD decreases as the block size is less than 8x8. Therefore, the integer motion estimation with 2BT-SD is performed with the block sizes greater than or equal to 8x8. For the block size smaller than 8x8, the MV derived for the 8x8 block including the small block is used as the starting point of the fractional motion estimation of the small block. In addition, another starting point is also used to run the fractional motion estimation one more time. This starting point is the PMV from the neighboring blocks as in (5). As a result, the fractional motion estimation is performed twice for the block with its size less than 8x8 pixels. The double executions increase the computing time of fractional motion estimation. However, this increase is small compared with the decrease of the computation time by employing 2BT-SD for integer motion estimation.

### 4. EXPERIMENTAL RESULTS

2BT-SD is compared with a variety of motion estimation algorithms which are MAD, 1BT proposed in [3], MF-1BT proposed in [5], and 2BT proposed in [4]. For fair comparisons, exhaustive full search operations are performed for all algorithms. For the evaluation of the motion estimation accuracy, PSNR values obtained from the two frames are reconstructed and the reconstructed image frames are used for comparative visual evaluation.

The PSNR for various test sequences are given in Table 1 and Table 2. The block size of 16x16 pixels with a search range of 16 pixels is used for Table 1 while 8x8 blocks with a search range of 8 pixels are used for Table 2. For the block size of 16x16 pixels, the threshold $T_d$ necessary to obtain $S_{2BT}[I_{n}]$, $S_{1BT}[I_{n}]$, $p^{\text{pix}}[I_{n}]$, and $n^{\text{pix}}[I_{n}]$ of (3) is set to 8, and the VALID PIXEL_NUM of (6) is set to 6. For the block size of 8x8 pixels, the threshold $T_d$ is set to 5 and VALID PIXEL_NUM is set to 4. These values are determined experimentally to yield the best performance.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Football (16x16)</th>
<th>Foreman (16x16)</th>
<th>Table Tennis (16x16)</th>
<th>Flowergarden (16x16)</th>
<th>Mobile (16x16)</th>
<th>Coastguard (16x16)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAD</td>
<td>22.053</td>
<td>32.1189</td>
<td>32.2594</td>
<td>23.7397</td>
<td>24.5894</td>
<td>30.4768</td>
</tr>
<tr>
<td>1BT (2)</td>
<td>21.8157</td>
<td>30.6316</td>
<td>30.7517</td>
<td>23.3144</td>
<td>24.2858</td>
<td>29.6927</td>
</tr>
<tr>
<td>2BT (5)</td>
<td>20.9220</td>
<td>30.7220</td>
<td>31.2041</td>
<td>23.0271</td>
<td>24.2965</td>
<td>29.9347</td>
</tr>
</tbody>
</table>

For various test sequences, the block size of 8x8 pixels with a search range of 8 pixels is used for Table 2 while 8x8 blocks with a search range of 8 pixels are used for Table 2. For the block size of 8x8 pixels, the threshold $T_d$ necessary to obtain $S_{2BT}[I_{n}]$, $S_{1BT}[I_{n}]$, $p^{\text{pix}}[I_{n}]$, and $n^{\text{pix}}[I_{n}]$ of (3) is set to 8, and the VALID PIXEL_NUM of (6) is set to 6. For the block size of 8x8 pixels, the threshold $T_d$ is set to 5 and VALID PIXEL_NUM is set to 4. These values are determined experimentally to yield the best performance.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Football (8x8)</th>
<th>Foreman (8x8)</th>
<th>Table Tennis (8x8)</th>
<th>Flowergarden (8x8)</th>
<th>Mobile (8x8)</th>
<th>Coastguard (8x8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAD</td>
<td>21.017</td>
<td>30.7291</td>
<td>31.3043</td>
<td>23.5546</td>
<td>24.3897</td>
<td>30.1514</td>
</tr>
<tr>
<td>1BT (2)</td>
<td>20.7765</td>
<td>29.7019</td>
<td>31.2704</td>
<td>24.5303</td>
<td>24.4107</td>
<td>29.7330</td>
</tr>
<tr>
<td>2BT-SD</td>
<td>22.8565</td>
<td>30.2075</td>
<td>31.5205</td>
<td>24.3014</td>
<td>24.4107</td>
<td>29.7330</td>
</tr>
<tr>
<td>2BT (5)</td>
<td>23.3405</td>
<td>30.6403</td>
<td>32.2775</td>
<td>24.5887</td>
<td>24.7142</td>
<td>30.5807</td>
</tr>
</tbody>
</table>

Table 1 Average PSNR (dB) of reconstructed frames with the block size of 16x16 pixels and the search range of 16 pixels

Table 2 Average PSNR (dB) of reconstructed frames with the block size of 8x8 pixels and the search range of 8 pixels
Fig. 2. 31\textsuperscript{st} frame of the “Table Tennis” test sequence reconstructed from the previous frame with the MVs derived by various motion estimation algorithms.

Fig. 3. Foreman sequence

Fig. 4. Coastguard sequence

Fig. 5. Mobile sequence

5. CONCLUSIONS

The proposed 2BT-SD makes use of lower bit-depth and binary matching criteria characteristic in a similar way to the 2BT and achieves the reduction of motion estimation complexity. According to the experimental results, the proposed 2BT-SD based motion estimation algorithm achieves improved motion estimation accuracy in terms of PSNR performance for a reconstructed frame and also results in visually more accurate frames compared with other binary motion estimation algorithms. 2BT-SD provides various possibilities to reduce computing time of motion estimation, so that it is particularly suitable for applications that require portable and mobile video encoding.

ACKNOWLEDGMENT

This work was sponsored by ETRI SoC Industry Promotion Center, Human Resource Development Project for IT SoC Architect.

REFERENCES


