1997 IEEE International Symposium on Circuits and Systems, June 9-12, 1997, Hong Kong

# Block Motion Vector Estimation using Edge Matching : An Approach with Better Frame Quality as Compared to Full Search Algorithm

*Yui-Lam Chan and Wan-Chi Siu*

Department of Electronic Engineering
*The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong*
*Tel: (852)27666229 Fax: (852)23628439 email: enwcsiu@hkpucc.polyu.edu.hk*

## Abstract

Intensity based block motion estimation algorithms are widely used for exploiting temporal redundancies in video coding. However, blocks located on boundaries of moving objects are not estimated accurately, which are very sensitive to the human eyes. In this paper, we propose a novel approach which incorporates edge matching techniques to accurately predict the motion of moving objects such that the motion compensated frames are tied more closely to the physical features. Besides, accurate motion vectors of edge blocks can be used to develop an efficient block motion estimation algorithm. Experimental results show that our approach requires simple computational complexity, and it gives a significant improvement in accuracy on motion compensated frames as compared with the traditional intensity based methods, including the full search algorithm.

## 1. Introduction

In video coding applications such as high-definition television, video conferencing, etc., the block motion estimation is being widely used [1-3]. Traditionally, block matching algorithms are based on the matching of blocks between the intensities of two images. In such intensity based block matching algorithms, the present frame is divided into two-dimensional small blocks of $N \times N$ pixels. For each block in the current frame, we evaluate certain matching criterion over nearby blocks in the previous frame and select the block which yields the closest matching. There is a large choice of matching criteria [4], e.g., the mean square error (MSE), the mean absolute difference (MAD), etc. Among these criteria, the MAD is the most popular one because it does not require any multiplication and produces similar performance as the MSE. The MAD matching criterion for an $N \times N$ block is given by

$$MAD = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} |I_n(i,j) - I_{n-1}(i+x, j+y)| \quad \text{where} \quad -D \le x, y \le D \tag{1}$$

where $I_n(i,j)$ is the intensity of the pixel at location $(i,j)$ within the block in the $n^{th}$ frame. The motion vector $v$ of the block is $\arg_{(x,y)} \min MAD(x,y)$. $D$ is the maximum displacement of the motion vector. Finding an absolute minimum for the MAD matching criterion can only be guaranteed by performing an exhaustive search of a series of discrete candidate displacements within a maximum displacement range, and this exhaustive search method is generally referred to as the Full Search Algorithm (FSA). However, despite their successful applications, a common problem of the intensity based block motion estimation techniques is that blocks located on boundaries of moving objects are not estimated accurately. Blocks on these boundaries may contain objects moving in different manners, and the motion vector obtained from the traditional intensity matching might have serious errors. Fig. 1 illustrates a typical artifact caused by this problem in a portions of a frame containing a ball moving against a background. The motion of the ball can be obtained from Fig. 1(a) and 1(b). Fig. 1(c) is the motion compensated frame of the ball using motion vectors computed by the full search intensity based block matching algorithm with a block size of $16 \times 16$ pixels. As shown in Fig. 1(c), the upper side of the ball has been compensated incorrectly. Let us give a clearer account for this phenomenon. Eqn. (1) can be rewritten as follows.

$$\begin{aligned} MAD(x,y) &= \sum_{(i,j)\in\Re_B} |I_n(i,j) - I_{n-1}(i+x, j+y)| + \sum_{(i,j)\in\Re_M} |I_n(i,j) - I_{n-1}(i+x, j+y)| \\ &= MAD_B(x,y) + MAD_M(x,y) \end{aligned} \tag{2}$$

where $\Re_B$ and $\Re_M$ are the sets containing pixels in the background and the moving object respectively. For Fig. 1(b), the marked block in the current frame, which contains part of a moving object has motion vector $(x_1,y_1)$. But region B (the region bounded by the dotted line) in the reference frame, as depicted in Fig. 1(a), does not give the minimum value of the $MAD_B(x,y)$ due to a non-uniform pixel intensity. Instead, the group of pixels in the background region has motion vector $(0,0)$. In Fig. 1(b), the block at the upper part of the ball is mainly occupied by the background portion, thus the $MAD_B(x,y)$ dominates the matching criterion $MAD(x,y)$ in this situation. Hence the motion vector $(0,0)$ is selected to represent this block. This poor motion compensated prediction along the moving edges is very sensitive to the human visual system. Thus, in this paper, an edge oriented block motion estimation algorithm is proposed to obtain more accurate motion prediction along moving edges.

## 2. Characteristics of Block Motions

For an efficient block motion estimation, it is important to know the nature of the blocks. However, the characteristics of the blocks depend on many factors such as the contents of the scenes, the motion of the objects, and so on, and it is difficult to model the blocks analytically. Here most typical blocks arising from image sequence scenes are discussed qualitatively. Fig. 2 shows a pair of current frame and previous frame containing a moving object in a still background with intensity variation. Let us define four possible types of blocks.

**Type *a*:** Blocks in a still area or with a background of very slow motion. These blocks could have similar motion vectors with the previous frame.

**Type *b*:** Blocks on the boundary of moving objects and still area. For the block containing the right hand edge of the moving object in Fig. 2, the intensity based block motion

*Acknowledge: This work was supported by The Croucher Foundation PolyU340/055.*

0-7803-3583-X/97 $10.00 ©1997 IEEE

estimation works well. While, this is not the case for the block containing the left hand edge of this moving object. Since a large portion of pixels in this block contains still area, the motion vector dominated by the effect of the still area is unavoidably selected. Consequently, this left edge would be lost in the motion compensated frame. This example could bring our motivation of using an edge matching to replace the intensity based matching of block motion estimation such that a better moving edges could be predicted in the compensated frame.

**Type *c*:** Blocks within a moving object, for which the intensity based block motion compensation works well. But, we can see easily that the motion in these blocks is highly correlated with the surrounding edge blocks provided that the motion vectors in the edge blocks truly represent the motion objects. Thus, the motion vectors of edge blocks (type *b*) are significant to develop an efficient motion estimation of blocks which are classified as type *c*.

**Type *d*:** Blocks which have no appropriate corresponding blocks in the previous frame. These blocks could arise from the covered region of the previous frame, etc. This type of problem could be resolved by using the bi-directional block motion estimation which has been defined in MPEG standard [6]. However, this is not the issue focused in this paper.

## 3. The Proposed Edge Oriented Algorithm

In this section, we describe our new edge oriented algorithm for finding motion vectors of different types of blocks. From the consideration in Section 2, motion estimation of type *b* blocks is the most important factor to improve the subjective view of motion compensated frames, and we can also reduce the computational requirement for motion vector estimation of type *c* blocks. A new edge matching technique is proposed to compute the motion vector of type *b* blocks. Now, let us describe the realization procedure of our algorithm.

1) Block classifier: The basic task of the classifier unit is to discriminate the types of the blocks, as described in section 2, for the appropriate motion estimation methods to be used. We define the frame difference of two successive frames, $fd_n$, as shown in eqn. (3), to represent the activity level of moving objects:

$$fd_n = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} |I_n(i,j) - I_{n-1}(i,j)| \tag{3}$$

If $fd_n < T_{fd}$, where $T_{fd}$ is a pre-determined threshold, this indicates that the block has only little motion (type *a*). But, if $fd_n \geq T_{fd}$, this implies that the block contains a certain amount of motions.

An edge detector is used to separate blocks on the boundary of moving objects and still area. For choosing an edge detection algorithm, we consider its speed and precision. Before finding the directional derivatives, we have to smooth the image such that ripples, spikes or high frequency noises from the image could possibly be removed. A simple mean smoothing that performs equally weighted smoothing using a square window with the size of 5 has been employed in this paper. For simplicity, the sobel edge detector[5] is applied to the smoothed current frame and to generate the detailed edge frame $S_n(i,j)$. Apart from using detailed edge frame as the matching feature, a simple binary edge frame according to the eqn.(4), can also be used.

$$B_n(i,j) = \begin{cases} 1 & \text{if } S_n(i,j) > T_e \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where $T_e$ is a predefined threshold. The major advantage in using coarse edge matching technique is its simplicity which involves the use of only one bit edge information rather than eight bits intensity information; this simplicity can potentially have implications on both hardware and software implementation of block motion estimation.

The block is considered having an edge (type *b*) if $\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} B_n(i,j) \geq T_{count}$ is satisfied, where $T_{count}$ is the threshold of edge count. This is done in order to prevent false determination of edge blocks due to the noise and consequently it can guarantee that motion vectors obtained in the following steps are more reliable.

2) To compute the motion vectors of still area (type *a*): Since blocks of type *a* have only little motion, only little motion search with a search range of *SW0* using intensity matching criterion is required to perform.

3) To search for the best match edge of moving edge blocks in the reference frame (type *b*): The distortion of the compensated frame using the intensity based block motion estimation between an object and the stationary background may introduce missing edge effects and it is very sensitive to human eyes. We reduce this distortion by modifying the criterion used in the matching procedure. The new matching criterion introduces some edge features $S_n(i,j)$ or $B_n(i,j)$ in the distortion measure. Hence the matching criterion using the detailed edge matching, EMAD, is defined as

$$EMAD = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} |S_n(i,j) - S_{n-1}(i+x, j+y)| \tag{5}$$

and for using the binary edge matching, it is given by

$$BEMAD = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} |B_n(i,j) - B_{n-1}(i+x, j+y)| \tag{6}$$

where $-SW1 \leq x, y \leq SW1$ and *SW1* is the maximum displacement of blocks of type *b*. These new edge matching criteria can improve the matching accuracy at the boundary in terms of subjective quality. True motion vectors of the moving objects are important, and there is a high correlation between their adjacent blocks. Thus an efficient motion estimation of blocks with type *c* can easily been developed.

4) To estimate the motion vectors of non-edge moving blocks (type *c*): The blocks inside moving objects are highly correlated with the moving edge blocks (type *b*). Thus, if there are adjacent blocks for which the motion vectors have already been computed, the current block can use these motion vectors as the initial centers and employ the intensity based block matching with smaller maximum displacement *SW2* (smaller than *SW1*) to compute its motion vector. But if there are no adjacent block having motion vectors been computed, the computation of the motion vector of the current block is postponed until the motion vectors of the required adjacent blocks are available. The advantage of this new motion

estimation technique is that it saves unnecessary computations so that the motion search can be conducted efficiently.

## 4. Experimental Results

We have tested the performance of the proposed algorithm on a variety of real image sequences. Only one well-known 80-frames image sequence, the "Table Tennis" with $352\times 240$ pixels is presented here. Results of the performance of the proposed algorithm and some conventional methods are compared. Two slightly different versions of our proposed edge oriented block matching algorithm have been realized, and let us call them EOBMA1 and EOBMA2. The only difference is that the EOBMA1 uses the detailed edge matching criterion as defined in eqn. (5), while the EOBMA2 refers to the use of binary edge matching criterion as given in eqn. (6). Also, a set of parameters as summarized in Table 1 has been selected. The conventional methods for performance comparison are the Full Search Algorithm (FSA) and the n-Step Hierarchical Search (n-SHS)[1]. The maximum allowable displacement in the $x$ and $y$ direction is set to $15$ and a block size of $16\times 16$ has been used.

In Fig. 3, it can be seen that the Mean Square Error (MSE) of the proposed EOBMA1 and EOBMA2 outperforms that of the n-SHS. As it is seen, our EOBMA1 has comparable MSE performance of the traditional intensity based FSA. Also, the EOBMA2 is still very close to the FSA, but it is inferior as compared with the EOBMA1. More importantly, the subjective quality of the proposed edge oriented algorithms is also shown in this paper. Fig. 4 shows the zoomed motion compensated frames of the ball and the racket in the "Table Tennis" sequence produced by different algorithms. Fig. 4(b) and Fig. 4(c) show the incorrect prediction of the ball and the racket along the edge produced by the traditional FSA and n-SHS , while the edge of the ball and the racket can be preserved by using the proposed EOBMA1 and EOBMA2 as shown in Fig. 4(d) and Fig. 4(e). This visual result indicates that even the EOBMA2 is better than the FSA in terms of subjective view, though the objective measure is inferior.

Now, the computational complexities of our techniques based on edge matching to that of intensity based block matching algorithms are evaluated. In order to make a comparison, we assume that the cost of an $8$ bit addition is $16$ times than that of one bit matching [8]. Both of the intensity and the detailed edge matching criteria require two-dimensional operations; $255(16\times 16\text{-}1)$ additions, $256(16\times 16)$ subtractions, and $256(16\times 16)$ absolute conversions per search point are needed. Therefore, the number of operations of the FSA and the n-SHS are shown in Table 2. For our proposed EOBMA1, it requires $391$ search locations. While the EOBMA2 requires only $58$ search locations using the intensity matching criteria. Besides, it needs $333$ search locations using one bit matching. Thus, its low cost is the major payoff in using this binary edge matching. Note that the binary edge matching criterion as defined in eqn. (6) can be easily implemented by a simple circuitry containing an 'XNOR' logic gate and a counter.

Apart from calculating the matching criteria, the edge detection is the major overhead of our proposed edge oriented algorithm. The edge detection algorithm consists of two parts: smoothing, and taking directional derivatives. For smoothing, it simply takes the average of the input with the $5\times 5$ window. By exploiting the maximal data reusage, the computation for smoothing can be calculated more efficiently. The input image is smoothed in the j-direction and a sum of image pixels along $j$-direction $V(i,j)$ is calculated, which is defined as:

$$V(i,j)=\sum_{h=-2}^{2} I(i,j+h) \tag{5}$$

where $I(i,j)$ is the input image at spatial location $(i,j)$. It is obvious that there is only a difference of two pixels for computing $V(i,j)$ and $V(i,j+1)$. From the above considerations, the total number of additions required per $N\times N$ block for the smoothing process in an image is:

$$A_s = N^2 \times \frac{W[4+2(H-5)]+H[4+2(W-5)]}{WH} \tag{6}$$

To detect edges, eleven additions, two absolute conversions and two multiplications by $2$ are required to compute the $S_n(i,j)$ as defined in eqn. (5). Note that multiplication by $2$ can be done with shifting, and therefore their costs could be neglected. Putting all of these together, we have found that computing the edge extraction requires $3{,}829$ additions and $512$ absolute conversions per $16\times 16$ block. Thus, table 2 shows a comparison of the computational requirement of the proposed methods, the FSA and the n-SHS. From table 2, the proposed EOBMA1 and EOBMA2 have a speed-up of $2.5$ and $13$ as compared with the FSA respectively.

## 5. Conclusions

In this paper, a block motion estimation algorithm that uses edge matching has been proposed. Considering the characteristics of motions of some typical image sequences, we have pointed out that accurate motion information of moving edge blocks is important to improve the performance of block motion estimation. In those blocks which contain boundaries, the edge matching is used to compute the motion vectors such that better motion compensation prediction along moving edges is obtained. They give a high correlation between their adjacent moving blocks which have uniform pixel intensity. Consequently, a fast motion estimation technique of these types of moving blocks can be easily developed. Experimental results show that our edge oriented algorithm has comparable MSE performance as compared to the traditional intensity based methods. Also, the poor prediction along the moving edges, which is very annoying around moving objects is effectively reduced by our proposed algorithm. Comparing to intensity matching, the major advantage in using binary edge matching technique is its simplicity, which involves one bit operations rather than eight bits operations; this simplicity can potentially have implications on both hardware and software implementation of block motion estimation.

## References


[1] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," Proceedings of the Nat. Telecomm. Conf., pp.G.5.3.1-5.3.5, 1981.

[2] Y.L. Chan and W.C. Siu, "Adaptive multiple-candidate hierarchical search for block matching algorithm," Electronics Letters, vol. 31, no. 19, pp.1637-39, Sept. 1995.

[3] Y.L. Chan and W.C. Siu, "New adaptive pixel decimation for block motion vector estimation", IEEE Trans. on Circuits and Systems for Video Technology, vol.6, no.1, pp.113 -8, Feb. 1996.

[4] K.R. Rao and R. Yip, *Discrete Cosine Transform - Algorithms, Advantages and Applications*. New York: Academic Press, 1990.

[5] R.C. Gonzalez and R.E. Woods, *Digital Image Processing*. Addison-Wesley Publishing Company, 1992.

[6] ISO/IEC JTC1 CD 11172, "Information technology - coding of moving pictures and associated audio for digital storage media up to about 1.5Mbit/s-part 2: coding of moving pictures information," Technical Report, 1991.

[7] D. Marr and E.C. Hildreth, "Theory of edge detection," Pro. of Royal Soc., London, vol. 207, pp. 187-217, 1980.

[8] M.I. Sezan and R.L. Lagendijk, *Motion analysis and image sequence processing*. London: Kluwer Academic Publishers, 1990.


**Table 1: Summary of chosen parameters.**

| Parameters | $SW0$ | $SW1$ | $SW2$ | $T_{fd}$ | $T_e$ | $T_{count}$ |
|---|---|---|---|---|---|---|
| Value | 1 | 15 | 3 | *1000* | *40* | *16* |

**Table 2: Comparison of computaional complexity.**

| | Additions/ subtractions | Absolute conversions | One bit matching (equivalent to 8 bits additions) |
|---|---|---|---|
| FSA | 491,071 | 246,016 | — |
| n-SHS | 16,863 | 8,448 | — |
| EOBMA1 | 203,630 | 100,608 | — |
| EOBMA2 | 33,467 | 15,360 | 5,328 (85,248/16) |

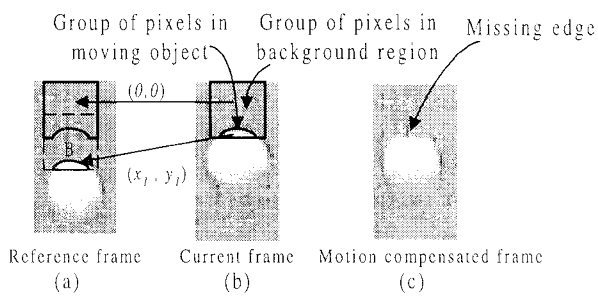



**Figure 1: Boundary problems of intensity based block motion compensated frames in the ball of the table tennis sequence.**

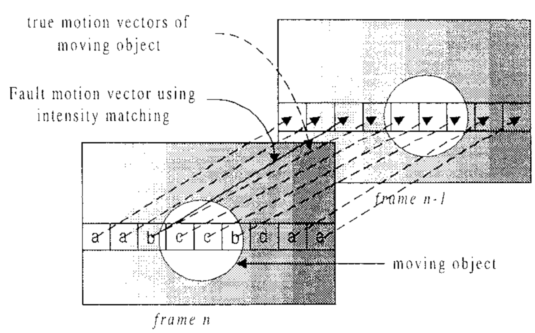



**Figure 2: Typical blocks in the image sequence scenes.**

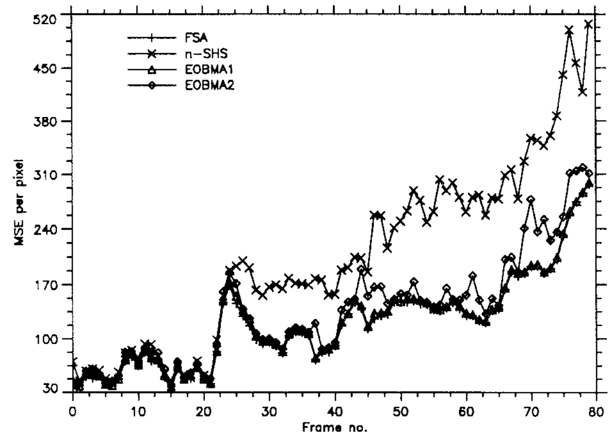
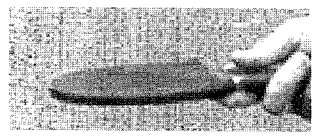
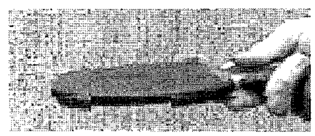


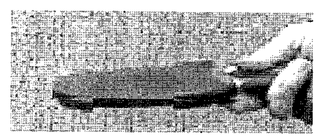

**Figure 3: MSE produced by different algorithms for "Table Tennis" sequence.**



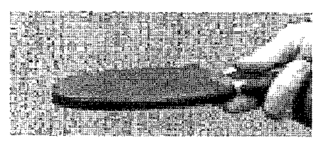
**(a) Original frame.**



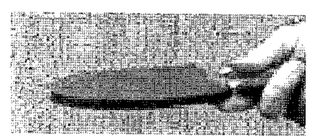
**(b) the FSA.**



**( c) the n-SHS.**



**(d) the EOBMA1.**



**(e) the EOBMA2.**

**Figure 4: Motion compensated frame of the ball and the racket in "Table Tennis".**