A BLOCK MOTION VECTOR ESTIMATION USING PATTERN BASED PIXEL DECIMATION

Yui-Lam Chan, Wai-Lam Hui and Wan-Chi Siu

Department of Electronic Engineering
The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
Tel: (852)27666250 Fax: (852)23628439 email: ra2980456@hkpu02.polyu.edu.hk

Abstract

A new pixel decimation technique based on a set of the pixel patterns for block motion vector estimation is presented. For uniform pixel decimation, regular patterns are used for computing the matching criterion to estimate the motion vector. The results can easily be misled by some image textures. Thus, in this paper, we define some "most representative pixel patterns" and make the selection according to the image content in each block for the matching criterion. Our approach can efficiently compensate the drawback in uniform pixel decimation. Computer simulations show that this technique is close to the performance of the full search, and has a significant reduction on computational complexity as compared with other pixel decimation algorithms in the literature. Also, it is more convenient for hardware realization as compared with the fully adaptive pixel decimation.

1. INTRODUCTION

Block motion estimation algorithms have been widely used in video coding [1] because of its simplicity and coding efficiency of motion vectors. For the block matching algorithm, the present frame is divided into two-dimensional small blocks of $N \times N$ pixels. Each block in the current frame estimates its motion vector by evaluating some matching criteria over the blocks in the previous frame and selecting the block which yields the closest matching. There are many choices [2] for the matching criterion, e.g., the mean square error (MSE), the mean absolute difference (MAD) and 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. Among all search algorithms, the Full Search Algorithm (FSA), for which all possible displaced candidates within the search area in the previous frame are searched, gives the best solution in the viewpoint of prediction error. However, the FSA requires extensive computations. Thus, many fast and efficient algorithms[3-5] have been developed to reduce the computational complexity by limiting the number of search locations. But these methods have the undesirable problem of local minima. Instead of limiting the number of locations to be searched, Koga et al.[3] used a pixel decimation technique to reduce the computational complexity. However, since only a regular fraction of the pixels enters into the matching computation, the use of these regular subsampling techniques can seriously affect the accuracy of motion vector detection. Liu and Zaccarin[6] have successfully improved the simple pixel decimation technique by using a scheme with alternating pixel decimation patterns. The subsampling patterns are alternated over the locations searched so that all pixels of a block contribute to the computation of the motion vector. In [7], an adaptive pixel decimation for block motion vector estimation is firstly introduced. This approach adaptively selects a set of representative pixels in each block for matching and shows an improvement as compared with Liu and Zaccarin's[6] algorithm. However, the drawback is that the selected pixels have large variation in each block and it is difficult to implement in practice. Thus, in this paper, a number of pixel patterns are predefined which are based on the edge content of possible blocks. Since the selected pixel patterns used here have to be predefined before the actual implementation, it is easy for hardware realization. Furthermore the prediction error as compared with the uniform pixel decimation is significantly reduced by devising the predefined set properly. The result is very close to the exhaustive search without pixel decimation. Furthermore, the performance of our algorithm is better than that of Liu and Zaccarin's[6] algorithm.

2. PROPOSED PATTERN BASED PIXEL DECIMAITON

For computing the matching criterion in regular pixel decimation, a regular pixel pattern is usually used. However, it could easily be misled by some image textures that would probably introduce excessive prediction errors. Thus, we have proposed an adaptive pixel decimation[7] which is to vary the number of selected pixels to be used for the computation of the matching criterion. We can use less pixels when the block has a uniform intensity. But for high activity blocks, more pixels can be employed for the MAD matching criterion. This adaptive approach can reduce the prediction error as compared with the regular pixel decimation. In order to resolve the difficulty for a practical realization of this adaptive technique, a predefined set of pixel patterns is employed in this paper.

Acknowledge: This work was supported by the Croucher Foundation & the Hong Kong Polytechnic University.

0-7803-3583-X/97 $10.00 ©1997 IEEE

Since each pixel pattern represents a certain local activity of an image block, the number of possible patterns increases exponentially with the size of the image block. Unfortunately, recent standards [1] often use *16 × 16* block size. Our strategy to be adopted for devising the pixel patterns is that, the pattern size is chosen such that the corresponding image part of which could at most have one edge. It is reasonable to assume that a block of *4 × 4* pixels would only contain a single edge, and let us refer it to as a "sub-block". In our design, we firstly divide the *16 × 16* image block into sixteen *4 × 4* sub-blocks respectively. Then, each *4 × 4* sub-block selects its corresponding predefined pixel pattern. Finally, these sixteen selected pixel patterns will give the most representative pixels for the image block. And also, these representative pixels are to be involved in the MAD matching criterion to compute the motion vector.

2.1. Pixel selection in uniform sub-block and edge sub-block

The goal of designing a predefined set of pixel patterns is to find a set of meaningful patterns. Based on the nature of sub-blocks, they can be classified as uniform sub-block and edge sub-block. For uniform sub-block, only the simplest comparison is required. Since the sub-block has uniform intensity, only two pixels as shown in Figure 1 could contribute to the computation of the matching criterion and this could be sufficiently enough. Blocks having sufficient intensity variation are classified as edge sub-block. For this type of sub-blocks, more pixels are to be involved in the matching criterion to reduce the prediction error for block motion vector estimation.

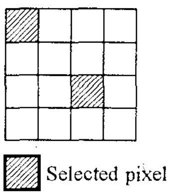

Figure 1: Predefined pixel pattern in uniform sub-block.

There is an excessively large number of possible physical situations for a sub-block that contains an edge. Since small blocks are used, the edge within the block is confined to a resolution of 45°. Eight basic edge patterns (c_i: where i is the edge orientation) with orientations 0°, 45°, 90°, 135°, 180°, 225°, 270° and 315° are defined. Figure 2 shows the 0°, 45°, 90° and 135° edge patterns. The remaining orientations are 180°, 225°, 270° and 315° which have contrast in the opposite direction. These are defined in order to devise a set of pixel patterns to be used as edge sub-blocks for the matching criterion.

A small set of predefined pixel patterns containing important edge information is constructed. It is according to a single edge for each sub-block. From Figure 3, we define the normal lines to different edge orientations. Since these lines contain significant intensity variations, a pair of pixels is selected along a normal line for the case that two adjacent pixels are in different intensity regions. As a result, different pixel patterns have been designed as shown in Figure 3.

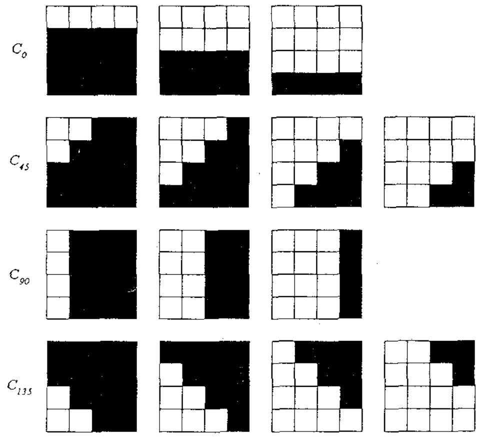

Figure 2: Edge patterns with different orientations.

2.2. Edge pattern matching

An efficient matching scheme must be provided, that maps each *4 × 4* sub-block into their corresponding pixel pattern. We firstly denote the intensity of the *4 × 4* sub-block as $I(i,j)$ where $0 \le i, j \le 3$. A natural measure of the edge is the discrete gradient $\nabla I(i,j) = \{\Delta_x I(i,j), \Delta_y I(i,j)\}$ where the directional derivative can be approximated as [8]:

$$\Delta_x I(i,j) = \left(\sum_{j=0}^{3}\sum_{i=2}^{3} I(i,j) - \sum_{j=0}^{3}\sum_{i=0}^{1} I(i,j) \right)$$

and

$$\Delta_y I(i,j) = \left(\sum_{i=0}^{3}\sum_{j=2}^{3} I(i,j) - \sum_{i=0}^{3}\sum_{j=0}^{1} I(i,j) \right) \quad (1)$$

The gradient magnitude and gradient orientation with each image sub-block are given, respectively, by

$$|\nabla I(i,j)| = \sqrt{(\Delta_x I(i,j))^2 + (\Delta_y I(i,j))^2}$$

and

$$\angle \nabla I(i,j) = \tan^{-1}\left(\frac{\Delta_y I(i,j)}{\Delta_x I(i,j)} \right) \quad (2)$$

The gradient magnitude is used to determine the type of the sub-block. If $\left(|\nabla I(i,j)|\right)^2 \leq \left(|\nabla I|_{\min}\right)^2$, where $|\nabla I|_{\min}$ is the minimum value of the discrete gradients containing edge feature, the block is determined to be a uniform sub-block; otherwise it is an edge sub-block. The gradient orientation $\angle\nabla I(i,j)$ is quantized by an increment of 45° and it maps the sub-block into their corresponding edge pattern subspace, c_i. A unique mapping is achieved by selecting the distribution of shaded and unshaded pixels in the edge pattern, as depicted in Figure 2, to be the closest to that of the sub-block. This can easily be accomplished by simply counting the number of shaded and unshaded pixels in the sub-block, and choosing the edge patterns having the best match. And then, an appropriate pixel pattern according to Figure 3 is assigned for the sub-block.

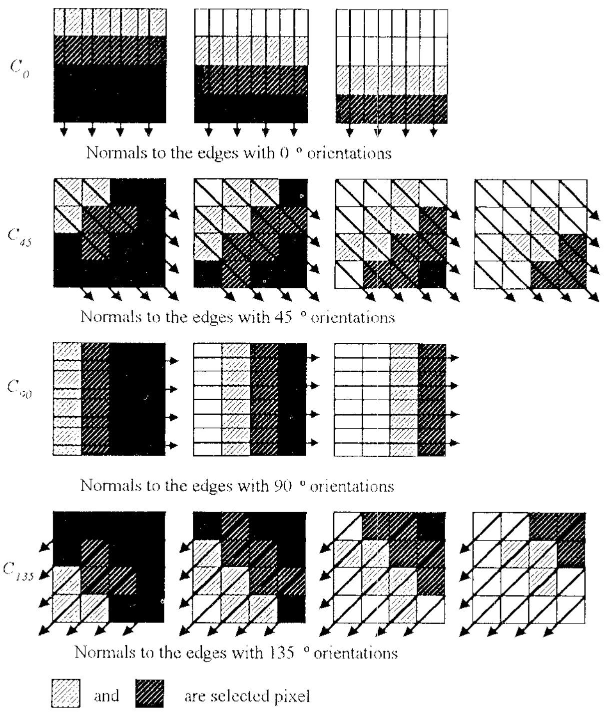

Figure 3: Possible edge patterns with its corresponding pixel pattern.

After we have assigned appropriate pixel patterns to a number of 4 × 4 sub-blocks within each *16* × *16* block, the most representative pixels are obtained. Only these pixels contribute to the computation of the matching criterion (MAD).

We select k motion vectors which are the ones to give the smallest MAD for further comparison. Then, we compute the MAD matching criterion for each of the k motion vectors using all pixels. The one that has the minimum MAD among k motion vectors is selected as the final motion vector.

3. RESULTS

In order to evaluate the performance of the pattern based pixel decimation algorithm, many sequences of motion pictures have been tested; these include *80* frames of the "football", "tennis" and "salesman" sequences with *352 × 240* pixels. The maximum allowable displacement is set to *8* with a block size of *16 × 16*. We compare different algorithms using the prediction errors (MSE) of the motion-compensated frames. Although the simulations were run using many sequences, we only present the results on "football" sequence since they are typical ones among all results from the sequences.

The prediction errors (MSE) of a regular 4 to 1 pixel decimation[3] and that of the pattern based pixel decimation algorithm with $|\nabla I|_{\min} = 20$ and $k=4$ are shown in Figure 4. It is seen that our pattern based algorithm is significantly better than that of the regular 4 to 1 pixel decimation[3]. Note that, the performance of the FSA without pixel decimation, the Liu and Zaccarian's[6] pixel decimation and adaptive pixel decimation[7] are also shown in Figure 4. It is seen that the pattern based pixel decimation algorithm has an MSE very close to these famous pixel decimation approaches. In Table 1, the average numbers of selected pixels for different pixel decimation algorithms are shown. The average numbers of pixels used for our pattern based pixel decimation are 1.6 less than that of the regular 4 to 1 pixel decimation[3] and the Liu and Zaccarin's[6] approach. The results also show that the pattern based pixel decimation algorithms are very effective.

4. CONCLUSIONS

A new pixel decimation block matching algorithm is proposed to compensate the drawback in regular pixel decimation techniques. The proposed algorithm consists of the definition of a small set of representative pixel patterns. Then, the discrete gradient measure is used to determine the corresponding pixel patterns which contribute to the computation of the matching criterion for the estimation of motion vectors. This pattern based algorithm can reduce the heavy computational burden of the FSA without significantly increasing the prediction error of the predicted frames. The results show that it is significantly better than that of regular pixel decimation, and about 1.6 times faster than that of the famous approach given by Liu and Zaccarin's[6]. Also, it is a very practical approach, especially when it is compared with the fully adaptive approach[7].

References

[1] 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.
[2] K.R. Rao and R. Yip, *Discrete Cosine Transform - Algorithms, Advantages and Applications*. New York:Academic Press, 1990.
[3] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," Proceedings of the National Telecommunication Conference, pp.G.5.3.1-5.3.5, Nov. 29-Dec. 3, 1981.
[4] J. R. Jain, and A. K. Jain, "Displacement measurement and it's applications in interframe coding," IEEE Transactions on Comm., vol.29, no.12, pp. 1799-1808, July 1981.
[5] 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.
[6] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," IEEE Transactions on Circuits and Systems for Video Technology, pp.148-157, vol.3, no.2, April 1993.
[7] Y. L. Chan and W. C. Siu, "New adaptive pixel decimation for block motion vector estimation," IEEE Transactions on Circuits and Systems for Video Technology, vol.6, no.1, pp.113 -118, Feb. 1996.
[8] B. K. P. Horn, *Robot Vision*. Cambridge, MA: M.I.T. Press 1986.

Table 1: Comparison of average selected pixels for different algorithms.

	Average pixel selected
Without pixel decimation	256
Regular 4 to 1 pixel decimation	64
Liu and Zaccarian's pixel decimation	64
Adaptive pixel decimation based on neighbour pixels	44
Pattern based pixel decimation	40

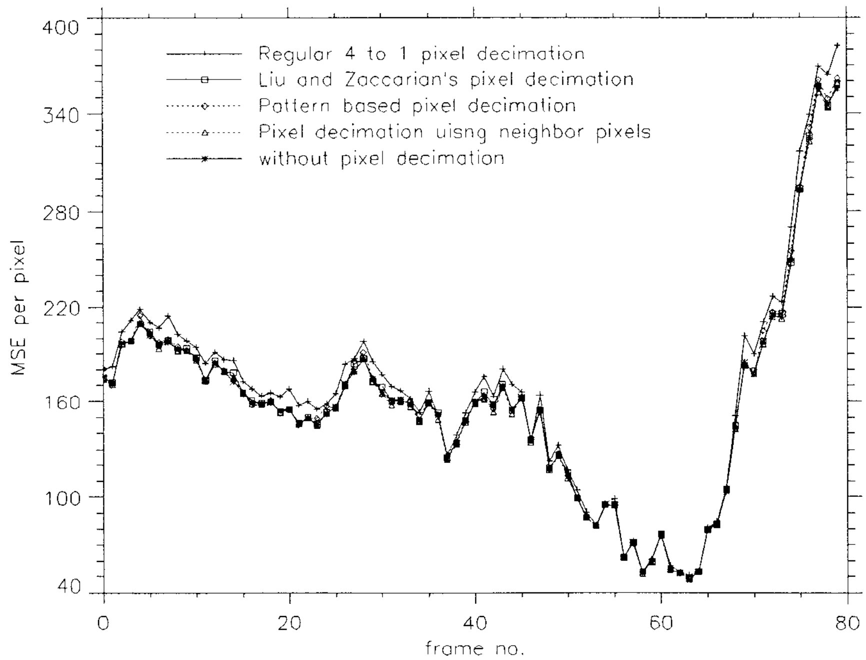

Figure 4: MSE produced by different pixel decimation algorithms for the "Football" sequence.