HALF-PIXEL MOTION ESTIMATION BYPASS BASED ON A LINEAR MODEL

Keman Yu, Shan Lu, Jiang Li and Shipeng Li

Microsoft Research Asia

ABSTRACT

In hybrid video coding schemes, motion estimation at sub-pixel accuracy, such as 1/2 pixel and 1/4 pixel obviously possesses higher coding efficiency than that at integer-pixel accuracy only. However, it requires more computational overhead for additional processes such as interpolation and search on sub-pixels. We proposed a novel half-pixel motion estimation bypass algorithm based on a linear model. The key idea is to skip over blocks that do not benefit from half-pixel search, therefore we not only reduce the search points but also the interpolation process. Experimental results show that significant computation reduction is achieved while the degradation in video quality is negligible. The proposed algorithm is very suitable for scenarios where low-complexity computing is required, such as mobile video coding applications.

1. INTRODUCTION

In video coding system, motion estimation is efficient in eliminating temporal redundancy between adjacent frames. At the same time, motion estimation is also regarded as a vital component in a video encoder as it consumes the largest amount of computation resources. Block matching algorithm is the most widely used technique in motion estimation due to its simplicity. In Block matching algorithm, a picture frame is divided into blocks, such as 16x16 pixels size. Motion estimation of the current block is performed by searching a similar block in the reference frame. Since the real motion of a moving object is not always in a multiple of integer-pixel, sub-pixel search may give more accurate results.

Taking complexity into consideration, video coding standards such as H.263 [1] and MPEG-4 [2] adopt half-pixel accuracy motion estimation. In general, a motion estimation procedure consists of two steps. First, an integer-pixel motion vector is determined by searching the best matching block within a defined area in the reference frame. Second, 8 half-pixel points surround the selected integer-pixel motion vector need to be examined to obtain the final motion vector.

There has been significant advance in fast integer-pixel motion estimation techniques in recent years. In Diamond Search (DS) [3], Small-Cross-Diamond Search (SCSD) [4] and Predictive Algorithm (PA) [5], an integer-pixel motion vector usually can be found by searching less than 10 points. Consequently, the computational overhead required by half-pixel motion estimation that includes interpolation and search processes becomes significant and comparable to that of integer-pixel motion estimation. Therefore, it is meaningful to reduce the computational cost of half-pixel motion estimation.

To speed up the computation of half-pixel motion estimation, several fast algorithms have been developed. In a method that uses horizontal and vertical directions as reference [6], the number of search points on half-pixel is decreased to 5. Another half-pixel accuracy fast search method proposed in [7] reduces the amount of computation to 50%. In the parabolic prediction based fast half-pixel search [8], there is a 59% saving in computation for block searching. A fast method based on directional search and a linear model [9] even reduces the number of search points to 2.2 in average, while the image quality of reconstructed sequences is similar to that of conventional methods. In summary, these methods reduce the complexity of half-pixel motion estimation by decreasing the number of search points on half-pixel. However, the calculation for 1/2 pixel interpolation, which is an important portion of half-pixel motion estimation, still has not been reduced in these methods.

In this paper, we propose a half-pixel motion estimation bypass algorithm based on a linear model. The key idea is to skip over the blocks that do not benefit from half-pixel search. Therefore, we not only reduce the search points but also the interpolation process. The proposed method can also be combined with any of the aforementioned algorithms.

The rest of this paper is organized as follows. Section 2 describes the proposed half-pixel motion estimation bypass algorithm. Experimental results are shown in Section 3. Finally, we conclude this paper and give future directions in Section 4.

2. HALF-PIXEL MOTION ESTIMATION BYPASS ALGORITHM

In the proposed algorithm, we assume that a video frame is divided into macroblocks with size of 16×16 integer pixels. Sum of Absolute Difference (SAD) is used as the cost measure to select the best matching block in motion estimation.

2.1. Observation and motivation

The motivation of our proposed algorithm comes from the observation that, for most sequences in relatively low motion scenes, a significant number of macroblocks, typically 50% to 90% of all macroblocks, have their final motion vectors on integer-pixels. In 8 MPEG-4 test sequences,we performed a conventional half-pixel search method on each macroblock, i.e. searching 8 half-pixel points surround the integer-pixel motion vector. If the minimum SAD obtained at half-pixel accuracy is larger than that at integer-pixel accuracy, the motion vector on the integer-pixel is selected as the final result and the half-pixel

search for this macroblock is regarded as wasted. Otherwise, the half-pixel motion estimation is regarded as effective. Table 1 shows the Effective half-pixel Search Ratio (ESR) for each sequence. For relatively low motion scenes, such as Akiyo, Salesman, News and Miss America sequences, the majority of half-pixel search is wasted, even for the Foreman sequence, which possesses larger facial motion and camera panning, only about 47% of macroblocks possesses effective half-pixel motion estimation.

Table 1: Effective half-pixel search ratio

Sequence	ESR
Akiyo	2.95 %
Salesman	4.21 %
News	10.45 %
Miss America	13.01 %
Trevor	26.70 %
Coastguard	35.77 %
Carphone	38.46 %
Foreman	46.80 %

If we can predict the macroblocks that do not benefit from half-pixel search, we can eliminate the computation of interpolation and search associated with those macroblocks. Therefore we can achieve a substantial saving.

SAD as a cost measure is commonly used in selecting the best matching block on integer-pixel and half-pixel. A motion vector with smaller SAD is regarded as highly approximate to the real motion. If the minimum SAD of an integer-pixel motion vector is small enough, it is highly probable that half-pixel search is unnecessary. This inspires us to find a threshold. Only those macroblocks that possess minimum SAD larger than the threshold need half-pixel search. A hidden assumption is that ESR increases along with SAD. We will examine this assumption in the following experiments.

We divide the range of the minimum integer-pixel SAD for each macroblock into segments with wide of 100. We calculate the ESR for each segment. and obtain the ESR-SAD curves. Here we show the results of three sequences that represent few, moderate and large motion scenes respectively. As shown in Figure 1, the trends of the three curves are basically consistent, i.e. ESR increases along with SAD. This trend also matches the results of 11 MPEG-4 test sequences. It verifies our assumption and indicates that SAD of a macroblock can be used to determine whether we need to perform half-pixel search for the macroblock.

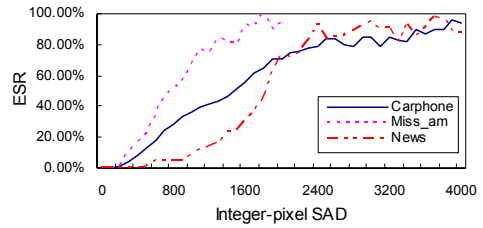

Figure 1: Effective half-pixel search ratio vs. integer-pixel SAD.

2.2. Finding the optimal SAD threshold

The ESR curves shown in Figure 1 do not sharply jump from 0 to 100%. This indicates that there is not a SAD threshold in the curves, above which effective half-pixel search is worth performing. Since a smaller threshold preserves more effective half-pixel search, gives more accurate motion vectors, but expenses more computation power, we need to find an optimal SAD threshold which reaches good tradeoff between compression efficiency and complexity.

To obtain the optimal threshold for a sequence, we encode the entire sequence with various SAD values. The coding efficiency, which is measured in Peak Signal-to-Noise Ratio (PSNR), decreases along with the raise of the threshold. Our goal is to find a point where PSNR starts sharply decreasing. As shown in Figure 2, this kind of point is also the point with the largest curvature in the curve. Table 2 shows the optimal threshold values of 7 sequences obtained by this method.

Figure 2: The decreasing of PSNR vs. the SAD threshold.

Table 2: Optimal SAD thresholds

Sequence	SAD threshold
Miss America	300
Akiyo	400
Suzie	600
Carphone	700
Salesman	800
Foreman	1000
Coastguard	1300

As these optimal thresholds are manually selected through many times of trials they cannot be applied to run-time encoding. We need to find a method that can automatically calculate the optimal threshold. After analyzing various factors such as PSNR, ESR and the average integer-pixel SAD that may be used to determine the optimal threshold, we found that there is not an equation that can directly determine the optimal threshold using these factors. However, we find that we can know whether the current threshold is appropriate by examining the compression results in run-time. In this way, we can increase or decrease the threshold to approach an appropriate value.

2.3. The "effective search ratio - optimal search ratio" model

We define Optimal half-pixel Search Ratio (OSR) as the half-pixel search ratio that is calculated when the SAD threshold is set at the optimal SAD threshold described in Section 2.2. We calculate OSR and ESR respectively for 6 sequences and show in Table 3. Although OSR and ESR vastly vary for different sequences, OSR basically increases along with ESR.

Table 3: ESR and OSR

Sequence	ESR	OSR
Akiyo	2.90%	23.88%

Salesman	4.19%	20.85%
Miss America	12.52%	40.97%
Suzie	34.10%	61.19%
Carphone	34.49%	65.66%
Foreman	40.22%	70.73%

After we draw these OSR and ESR values into a figure (Figure 3), we find that they basically obey a linear relationship (the solid line in Figure 3). This inspires us to find an adjustment method to approach OSR in real-time encoding. Given the current SAD threshold, we can calculate the Actual half-pixel Search Ratio (ASR) and the ESR. Using the linear relationship, we can calculate the OSR. After that, we can exam whether the current SAD threshold is appropriate by comparing the calculated OSR with the current ASR. For example, If ASR is larger than OSR, the current SAD threshold is regarded as too small and needs to be increased. This can be completed in real-time.

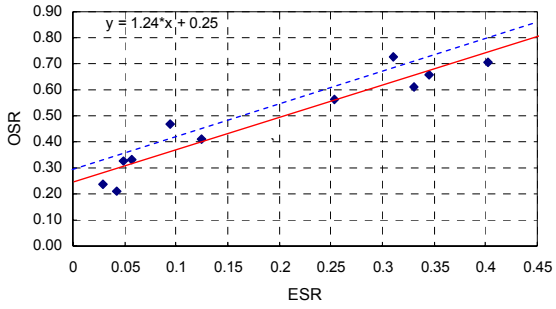

Figure 3: A linear model of ESR and OSR.

In practice, the prevention of loss in video quality may be more important than a little bit increase in complexity of half-pixel motion estimation. Therefore, we slightly moved up the solid line to the position of the dashed line shown in Figure 3. It actually increases the value of OSR, and preserves more effective half-pixel search. Finally, we define the linear model of ESR and OSR as follows.

$$OSR = 1.24 \times ESR + 0.3 \quad (1)$$

2.4. Dynamically adjusting the SAD threshold

Here we describe a programming procedure to dynamically adjust the SAD threshold. The following is the pseudo-code.

Step 1: Let *Block* = 0, *Block_Search* = 0, *Block_Useful* = 0,
SAD_Threshold = SAD_0
Step 2: For each macroblock in a frame
Block++
Perform integer-pixel search, obtain *IntSAD*
If *IntSAD* > *SAD_Threshold* then
Perform half-pixel search, *Block_Search*++
If the half-pixel search is effective
Block_Useful++
Step 3: Let *ESR* = *Block_Useful* / *Block*,
ASR = *Block_Search* / *Block*
Calculate *OSR* using Eq. (1)
Step 4: Adjust *SAD_Threshold* for next frame

In practice, we use a moderate SAD value 600 as the initial threshold, namely SAD_0.

The last issue is how to adjust *SAD_Threshold* in Step 4. A simple and straightforward approach is to increase or decrease the threshold with a fixed step size. However, the result is very sensive to the selected step size. If the step size is too small, the approaching speed will be very slow. On the other hand, if the step size is too large, it is possible to incur oscillation. A desirable method should adaptively change the step size in terms of the difference between *ASR* and *OSR*.

The requirement is similar to that of the rate-control of a video codec. In H.263 TMN5 [10], the quantization parameter (QP) is adjusted by comparing the consumed bits length B_{i-1} and the target length $\overline{B}$. Referring to the idea of TMN5 rate-control, we defined our SAD threshold adjustment equation as follows. It should be noted that the computational overhead for the SAD threshold is negligible.

$$SAD\_Threshold_{new} = SAD\_Threshold_{i-1}\left(1 + \frac{ASR - OSR}{2 \times OSR}\right) \quad (2)$$

3. EXPERIMENTAL RESULTS

To examine the effectiveness of the aforementioned algorithm, we chose 7 sequences with vastly varying content from the standard MPEG-4 test video clips and implement in an H.263-compatible encoder [5]. These sequences are classified into 4 categories, which represent different kinds of motion:

1) Few motion scenes: Salesman and Akiyo sequences, which possess little movements and a still background.
2) Moderate motion scenes: News and Silent sequences.
3) Large motion scenes: Carphone which possesses large facial motion and a fast moving background, and Coastguard which possess large but uniform motion.

In order to better evaluate the half-pixel motion estimation algorithm, we use full search method in the integer-pixel motion estimation stage. All sequences are in QCIF format and encoded at 15 frames per second and 56 Kbps. Moreover, all frames except the first frame is encoded as P-frames.

Table 4 shows the comparison of the coding efficiency of the integer-pixel accuracy method, the conventional Full Half-Pixel Search (FHPS) method and the proposed method. Obviously, FHPS outperforms the integer-pixel accuracy method by 1.49 dB in average. It shows that half-pixel accuracy motion estimation is really necessary in reaching higher visual quality. In Table 4, we can see that the proposed method only has 0.04 dB degradation in average. The difference is negligible while the computational savings of the proposed method will be shown in the next table.

Table 4: Performance comparison in terms of PSNR (dB)

Sequence	Integer	FHPS	Half-pixel bypass
Salesman	36.91	37.72	37.69
Akiyo	41.30	42.16	42.09
News	33.59	34.99	34.96
Silent	33.93	35.52	35.50
Carphone	30.49	32.84	32.82
Coastguard	28.08	30.01	29.95

The complexity reduction of the proposed method is clearly shown in Table 5. Comparing to FHPS, there is an average 49.0% saving in both half-pixel search and interpolation, thus the average number of the half-pixel SAD computation (namely the number of search points) for each macroblock is only about 4.01. It is also shown in the table that scenes such as Carphone sequence and Coastguard sequence with relatively large motion need more half-pixel search than those with few motion.

Table 5: Complexity reduction

Sequence	Bypassed	SAD computation
Salesman	64.11%	2.87
Akiyo	66.40%	2.69
News	58.34%	3.33
Silent	51.88%	3.85
Carphone	25.96%	5.92
Coastguard	32.70%	5.38

As mentioned in Section 1, many fast integer-pixel motion estimation algorithms have been developed in recent years. They are commonly preferred in real-time video coding application. In the following experiments, we combine the proposed half-pixel accuracy method with a fast integer-pixel search method in [5] (i.e. PA) to show how important the complexity reduction in half-pixel motion estimation is and what the overall performance is. As shown in Table 6, only 5 search points in average are needed in finding an integer-pixel motion vector, and only 4 points in average are needed in half-pixel refinement. Therefore, the total number of search points for determining the final motion vector is only 9 in average while the PSNR degradation is only 0.07 dB in average comparing to the combination of integer-pixel full search and half-pixel full search.

Table 6: Average number of search points and PSNR loss (dB)

Sequence	Integer-search	Half-search	PSNR loss
Salesman	4.63	2.88	0.03
Akiyo	4.60	2.69	0.00
News	4.78	3.33	0.06
Silent	4.87	3.83	0.10
Carphone	5.29	5.90	0.12
Coastguard	5.77	5.31	0.12

We also examined the proposed method on video clips captured from real scenes using PC cameras. The proposed method works well in different video clips with low and high motion activities.

4. CONCLUSIONS

We propose a novel half-pixel motion estimation bypass algorithm for video coding. In the proposed algorithm, we built a linear model to determine the optimal SAD threshold, and then used the threshold to predict the macroblocks that do not need half-pixel search. As a result, the computation of search and interpolation associated with those macroblocks are saved. Experimental results show that significant computation reduction is achieved by using the proposed method without leading to visible loss in video fidelity.

Future directions may include further decreasing the half-pixel search ratio, which is still much larger than the effective half-pixel search ratio, whereas preserve the prediction accuracy at the same time.

5. REFERENCES

[1] ITU-T Recommendation H.263 Video coding for low bit rate communication, 02/98.

[2] ISO/IEC JTC1/SC29/WG11 N3312 Coding of moving pictures and audio March 2000/Noordwijkerhout.

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

[4] C.H. Cheung, L.M. Po, "A Novel Small-Cross-Diamond Search Algorithm for Fast Video Coding and Videoconferencing Applications," *Proc. ICIP 2002*, pp.681-684.

[5] K.M. Yu, J.B. Lv, J. Li and S.P. Li, "Practical Real-time Video Codec for Mobile Devices," *Proc. ICME 2003*, pp.509-512, Jul 2003.

[6] K.H. Lee, J.H. Choi, B.K. Lee and D.G. Kim, "Fast two-step half-pixel accuracy motion vector prediction," *Electronics Letters*, vol. 36, pp.625-627, Mar. 2000.

[7] D.N. Kwon, "Half-pixel accuracy fast search in video coding," *Proc. ISSPA 2003*, pp.73-76, Jul 2003.

[8] C. Du, Y. He and J.L. Zheng, "PPHPS: A Parabolic Prediction-Based, Fast Half-Pixel Search Algorithm for Very Low Bit-Rate Moving-Picture Coding," *IEEE Trans. On CSVT*, vol. 13, no.4, pp.514-518, Jun 2003.

[9] Y.G. Lee, J.H. Lee and J.B. Ra, "Fast half-pixel motion estimation based on directional search and a linear model," *SPIE Proc. VCIP 2003*, vol. 5150, pp.1513-1520, Jul 2003.

[10] ITU - Telecommunications Standardization Sector Q.15/16, Portland, 24-27 June, 1997.