A Fast Sub-pel Motion Estimation Algorithm based on Best Position Calculation

Li Bo, Zhang Jinyin and Song Jianbin

(Digital Media Lab of School of Computer Science and Technology, Beihang University, Beijing, 100083)

Abstract—**As the currently prevalent sub-pel motion estimation algorithm for video compression is both time- and memory-consuming, this paper proposes a novel one, which directly calculates the best position in sub-pixel level and thus significantly promotes searching speed without additional memory cost. A DSP processor-based encoder using this algorithm is also introduced herein.**

Index Terms—**video compression, sub-pixel motion estimation, interpolation, best position calculation.**

I. INTRODUCTION

IN recent years hybrid video coding schemes based on DCT transform and motion estimation (ME) have been widely employed, such as H.263, MPEG-4, or JVT, where ME efficiently eliminates redundancy between different frames. ME algorithm consists of two stages, integer-pixel accuracy motion estimation (IME) and sub-pixel accuracy motion estimation (SME). While the former takes coded frames as references to search the best match block directly, the latter involves searching sub-sample interpolated position as well as integer sample positions, choosing the position that gives the best match. It is of great help to improve performance of compression.

The $1/2^K$ sample accuracy SME, adopted widely by current international standards including H.263, MPEG-4, JVT[10~12], works as follows. In the first stage, the reference region samples are interpolated to $1/2^K$ samples positions. Then the encoder immediately searches the half-sample positions next to the best integer match to determine whether the match can be improved. If required, the quarter-sample positions next to the best half-sample position are to be searched. The process, hereinafter referred to as TSME, continues until $1/2^K$ samples are searched as depicted in Figure 1. In view of both the need to check $8\times K$ blocks at sub-pel positions in addition to IME and the interpolation computation cost, SME takes largish weight among the whole ME process. Besides, as the interpolation image expands to be $2^{2K}-1$ times larger than the input image, additional memory has to increase by $2^{2K}-1$ times. Taking all these factors into account, an efficient fast sub-pel motion estimation algorithm should cut down both computational complexity and memory cost, which becomes especially important when encoders are to be implemented on a Digital Signal Processor (DSP) or some other particular Embedded Processors.

This work was supported by the NSFC, the Program for New Century Excellent Talents in University, the National Defense Basic Research Foundation, and the SRFDP. The research was made in the State Key Lab of Software Development Environment.

Li Bo,Professor,is with Digital Media Laboratory, School of Computer Science and Engineering , Beihang , University,Beijing , China (8610-82317608, 8610-82315113,boli@buaa.edu.cn , bhboli @vip.sina.com).

Zhang Jinyin,Master, is with Digital Media Laboratory, School of Computer Science and Engineering,Beihang,University,Beijing, China.

Song Jianbin,PhD.candidate, is with Digital Media Laboratory, School of Computer Science and Engineering,Beihang , University , Beijing , China(buaasjb@yahoo.com.cn).

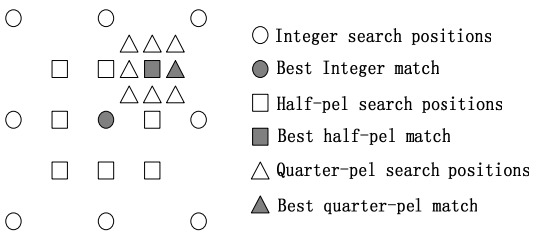

Figure 1. Quarter-pel motion estimate of TSME

This paper puts forth a novel fast algorithm, namely FSME*, which directly calculates the best position according to the neighboring integer sample value and the intermediate result of IME. Section 2 demonstrates the expression of best sub-pel position at any accuracy in special cases, and Section 3 presents a method to reach the approximate best sub-pel position with low computational complexity. Details of the algorithm FSME* and experiment results are described in Section 4, and an application example of this algorithm on DSP system is given in Section 5.

II. DETERMINE THE BEST POSITION AT ANY ACCURACY

As the random distribution of samples in an image makes IME results indeterminate, there is no choice but to check blocks one by one to find the best match[3][4]. Yet different from integer samples, sub-pel samples are not decided by captured image directly, but interpolated by adjacent integer samples. Thus it's possible to calculate the best position in SME by neighboring integer samples instead of traditionally monotonous search. The following is a derivation of the expression of the best position of SME at any accuracy. During the process, bilinear interpolation is used as in many international standards such as H.263 and MPEG-4.

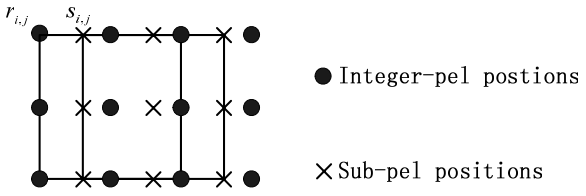

Figure 2. Horizontal 1-D motion at sub-pel level

First, the writer simplifies the problem to the case of only one-dimensional (1-D) motion appearing in the video for the sake of argumentation. As shown in ***Figure 2***, the object moves only horizontally. The following terms are employed to facilitate the description of the deduction:

$r_{i,j}$: Integer sample values belonging to the best match block of IME;

$s_{i,j}$: Sub sample values between $r_{i,j}$ and $r_{i+1,j}$ in the reference frame;

$c_{i,j}$: Integer sample values belonging to the current block, which will be coded immediately;

M : The horizontal block size;

N : The vertical block size;

$D_{d_{x,y}d'_{x,y}}$: ***MSE*** (mean square error) between two blocks containing top-left pixel $d_{x,y}$ or $d'_{x,y}$;

In terms of bilinear interpolation, a sub-pel sample value is given by

$$\begin{cases} s_{i,j} = mr_{i,j} + nr_{i+1,j} \\ m+n=1 \end{cases} \quad (1)$$

Where the sub-pel position is decided by m and n. ***MSE*** between the current block and integer-pel blocks in the reference frame can be expressed as

$$D_{c_{i,j}r_{i,j}} = \frac{1}{MN}\sum_{i=0}^{M}\sum_{j=0}^{N}(c_{i,j}-r_{i,j})^2 \quad (2)$$

$$D_{c_{i,j}r_{i+1,j}} = \frac{1}{MN}\sum_{i=0}^{M}\sum_{j=0}^{N}(c_{i,j}-r_{i+1,j})^2 \quad (3)$$

MSE between the current block and the sub-pel block can be expressed as

$$D_{c_{i,j}s_{i,j}} = \frac{1}{MN}\sum_{i=0}^{M}\sum_{j=0}^{N}(c_{i,j}-s_{i,j})^2$$

Then, from (1),(2),(3), ***MSE*** can be evaluated as follow:

$$\begin{aligned} D_{c_{i,j}s_{i,j}} &= \frac{1}{MN}\sum_{i=0}^{M}\sum_{j=0}^{N}[c_{i,j}-(mr_{i,j}+nr_{i+1,j})]^2 \\ &= mD_{c_{i,j}r_{i,j}} + (1-m)D_{c_{i,j}r_{i+1,j}} - \frac{m(1-m)}{MN}\sum_{i=0}^{M}\sum_{j=0}^{N}(r_{i,j}-r_{i+1,j})^2 \\ &= mD_{c_{i,j}r_{i,j}} + (1-m)D_{c_{i,j}r_{i+1,j}} - m(1-m)D_{r_{i,j}r_{i+1,j}} \\ &= D_{r_{i,j}r_{i+1,j}}m^2 + (D_{c_{i,j}r_{i,j}} - D_{c_{i,j}r_{i+1,j}} - D_{r_{i,j}r_{i+1,j}})m + D_{c_{i,j}r_{i+1,j}} \end{aligned} \quad (4)$$

To minimize the functional (4), calculus of variations is used and the following conclusion is drawn:

$$Min(D_{c_{i,j}s_{i,j}}) = \frac{4D_{r_{i,j}r_{i+1,j}} D_{c_{i,j}r_{i+1,j}} - (D_{c_{i,j}r_{i,j}} - D_{c_{i,j}r_{i+1,j}} - D_{r_{i,j}r_{i+1,j}})^2}{4D_{r_{i,j}r_{i+1,j}}} \quad (6)$$

where

$$m = \frac{D_{c_{i,j}r_{i+1,j}} - D_{c_{i,j}r_{i,j}} + D_{r_{i,j}r_{i+1,j}}}{2D_{r_{i,j}r_{i+1,j}}} = \frac{D_{c_{i,j}r_{i+1,j}} - D_{c_{i,j}r_{i,j}}}{2D_{r_{i,j}r_{i+1,j}}} + \frac{1}{2} \quad (5)$$

The position where the minimum of ***MSE*** is obtained, i.e. the best position of SME at any accuracy, can be calculated by (5). However, in general, performance gain tends to diminish as the interpolation accuracy increases, and when certain sufficient accuracy is reached, improvements can hardly be achieved. Therefore, in most cases, 1/2, 1/4 or 1/8 pixel ME is adopted. For instance, 1/2 pixel is used in MPEG-4 and 1/4 or 1/8 in JVT, which requires that m should be consistent with concrete standards (1/2, 1/4 or 1/8 level) instead of any value.

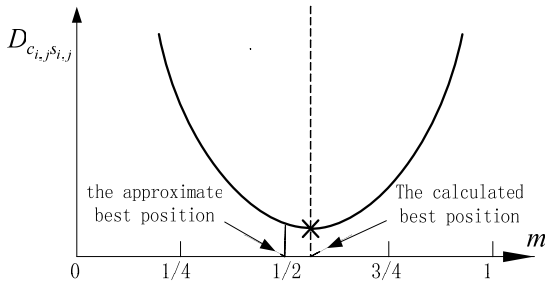

Figure. 3. calculate the approximate best position

As indicated in (4), $D_{c_{i,j}s_{i,j}}$ is a parabolic function, which suggests that the position, closest to the calculated best position, is the best one when sub-pel accuracy is given. For example, as shown in ***Figure 3***, while 1/4-pel accuracy is applied and m is 9/16, m' =1/2 should be the closest position to 9/16, so 1/2 is the best approximate position. Therefore, when $1/2^K$ accuracy is adopted, the best approximate position m' should be determined by the following expression:

$$m' = round(m \times 2^K) / 2^K \quad (7)$$

Where m is evaluated by (5), and $round()$ carries the function of round.

III. APPROXIMATION MEASURES USED IN BEST POSITION CALCULATION

As demonstrated in Section 2, the best position of SME can be directly calculated in 1-D motion in pictures. With regard to 2-D motion where sub-pel samples are interpolated by several neighboring integer samples, it is unlikely to advance an explicit expression of the accuracy best position. Thus some approximations must be taken into account.

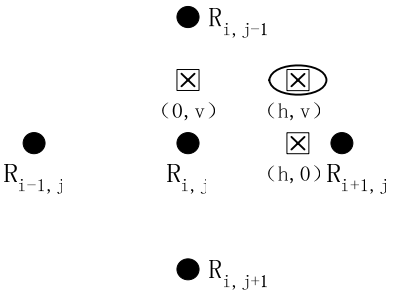

Figure 4. Best Position when considering 2-D motion

A 2-D motion can be taken as a synthesis of two orthogonal 1-D (horizontal and vertical direction) motions, and then the best position can be calculated respectively at the horizontal and vertical level. The corresponding 2-D position is taken as the best. As shown in Figure 4, $R_{i,j}$ represents the best match of IME, $R_{i+1,j}$, $R_{i-1,j}$, $R_{i,j+1}$ and $R_{i,j-1}$ represent blocks only one pixel apart from $R_{i,j}$. The horizontal best position h and the vertical best position v can be calculated respectively by (7). Then the block at (h,v) is regarded as the 2-D best candidate based on the hypothesis that 2-D motion is decomposable.

In some new video compression standards, filtering interpolation is introduced to replace bilinear interpolation, such as 6-tap or 8-tap interpolation in JVT [9]. In such interpolation schemes, a sub-pel sample is a weighted average of several integer samples around it, which provides better interpolation images at the expense of increased complexity. As

pixels farther from the interpolated sub sample take smaller weight compared with the nearer ones, they would contribute less to the best position calculation. Thus in order to reduce computational complexity, the previous conclusion, deduced in the case of bilinear interpolation, should still be applied to calculate the best position when 6-tap or 8-tap interpolation is adopted. The experiment results presented hereinafter demonstrate its high feasibility.

In addition, SAD (Sum of Absolute Errors) is often employed in lieu of MSE to reduce computational complexity. In this case, SAD may be used instead of MSE in (5) and (6).

IV. FSME*: A NOVEL SME ALGORITHM

A. Description of FSME*

Based on the preceding study, a novel fast SME algorithm FSME* is proposed. In the algorithm, the best horizontal position h and the best vertical position v are calculated respectively by expression (7). Then given the approximations presented in Section 3, the block located at (h,v) is chosen as a candidate. Additionally, considering that (h,v) is an approximate position, the block with less ***MSE*** at position h or v is taken as another candidate to improve ME performance. Any of the two candidates will be checked if only it locates at sub-pel position. The process of FSME* is as follows:

Step1: Calculate the minimum of ME residuals between current block and blocks both left and right to the IME best match by (6). The smaller residual of this two is taken as the smallest horizontal residual and recorded as D_h .

Step2: Calculate the minimum of ME residuals between current block and blocks both above and below the IME best match by (6). The smaller residual of this two is taken as the smallest vertical residual and recorded as D_v .

Step3: Calculate the best position h , v corresponding to D_h and D_v by (7) respectively.***Step5:*** Compare D_h with D_v and choose the block corresponding to the smaller value. If it locates at sub-pel position, throw it into the set of candidate blocks.

Step4: If (h,v) is a sub-pel position, throw the block at this position into the set of candidate blocks.

Step6: Check each block in the candidate-block set, select the one with the smallest residual, and designate it as SAD_s. Compare SAD_s with SAD_I (the smallest residual of IME), and the smaller one is the smallest residual of the whole search, whose position is taken as the best one.

B. Experimental results

The experiment implemented the algorithm FSME* on JVT reference software JM6.0 and tested it by various sequences; the value of PSNR is regarded as a measure to evaluate image quality.

As to test sequences, some typical ones are selected from standard sequences, such as 'Mother & daughter' with comparatively low amount of movement, 'Coastguard' with medium spatial detail and movement, 'Basket' with high amount of movement, and 'Flower' with high spatial detail. In the experiment, quarter-pel and bilinear interpolation are used.

The experiment gathers PSNR values at different bits by regulating the value of the quantization coefficient QP to 20, 25, 30, 35, 40 and 45. Three algorithms, namely IME, FSME*, and the algorithm adopted in JVT reference software (TSME), are utilized respectively. Part of the experimental results is shown in Table1.

Table 1. Results of three algorithms

Sequence	JvtSME		FSME*		Decrease of check points (%)	IME	
	PSNR (dB)	Coding length (bit/frame)	PSNR (dB)	Coding length (bit/frame)		PSNR (dB)	Coding length (bit/frame)
mother	40.57	5857	40.56	5926	95	40.29	10954

	37.45	2285	37.42	2300	94	37.15	4079
	32.06	391	32.07	393	94	31.88	546
basket	40.64	126491	40.65	126646	93	40.32	206470
	36.43	75120	36.43	75353	93	36.10	130512
	32.00	38362	32.00	38392	92	31.71	70764
flower	41.79	144165	41.78	144046	95	41.39	268715
	37.57	91628	37.55	91612	95	37.08	196620
	32.97	48998	32.98	49058	95	32.43	129702

Rate distortion curves in Figure 5 are drawn based on the experiment data. In the coordinate, the abscissa represents average coding bits of a frame (Kb/frame), and the ordinate represents PSNR of reconstructed image (dB):

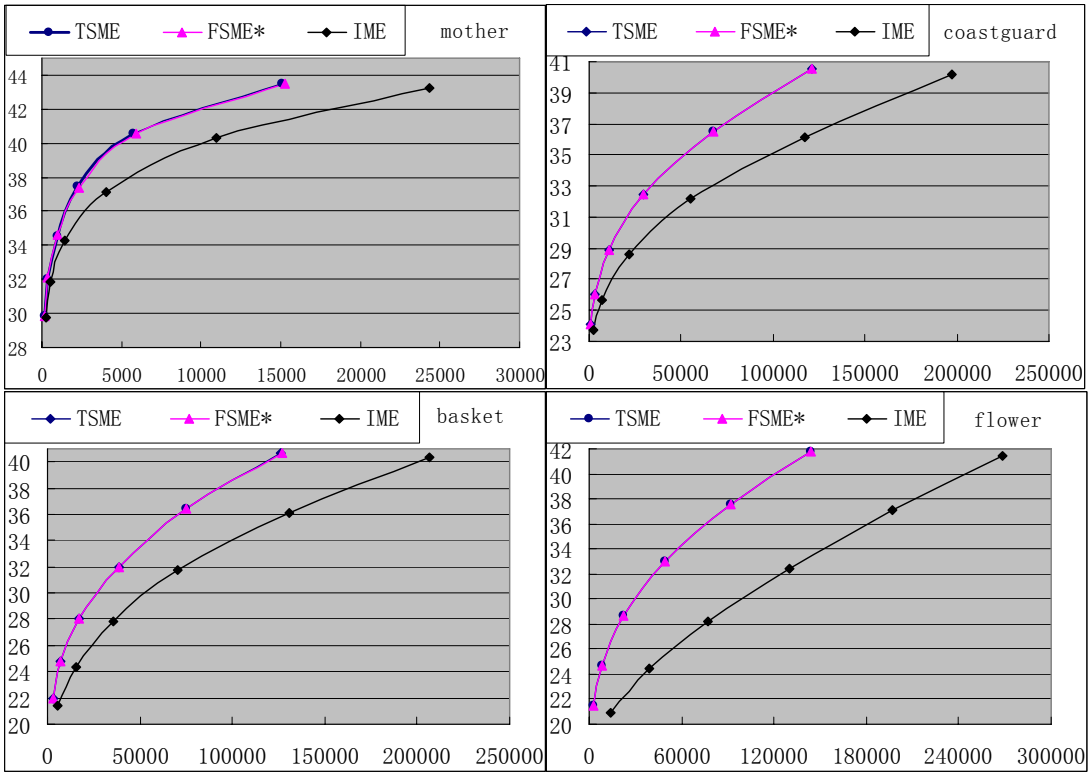

Figure 5. rate distortion curves of three algorithms

Several conclusions can be drawn from the aforementioned experimental results. As can be easily seen, compared with IME, TSME considerably improves the compression performance, yet the check points increase to a great degree. FSME* proposed in this paper not only performs similarly to TSME with only 0.01dB quality decrease on average, but enhances the search speed more than 90% measured by check points.

In conclusion, FSME* calculates the best position directly as opposed to the traditional way of checking points one by one, and the total check points will decrease to 2 at most. In comparison with TSME, the check points decrease by over 87.5% when quarter-pel is used and over 91.6% when eighth-pel is followed. Besides, TSME interpolates the whole picture before search to avoid repeatedly interpolating the same position. Consequently, in $1/2^K$ sub-pel SME, the interpolation image will extend to be 2^{2K} times as large as input image, necessitating $2^{2K}-1$ times additional memory during the coding process. Yet since the total check points are less than 2 in FSME*,

the probability of interpolating the same position is still very little even if sub-pel samples are interpolated during search. Hence, it is feasible to interpolate samples during the searching process, which saves mass of memory and becomes indispensable to such memory limited application as DSP-based systems.

In expression (6) and (7), $D_{a_{i,j}b_{i,j}}$ and $D_{a_{i,j}b_{i+1,j}}$ are intermediate results of IME; thus only $D_{b_{i,j}b_{i+1,j}}$ needs to be calculate in addition. In FSME*, the total additional computation of each block includes: ***SAD*** between best integer-pel match block with its four neighboring blocks; 4 calculations by (7); 2 calculations by (6). Compared with the computation of check and interpolation of $8\times K$ blocks in $1/2^K$ sub-pel TSME, the additional computation is considerably reduced in FSME*. Advantage of this algorithm tends to aggrandize as the value of K increases.

V. FSME* IMPLEMENTED ON DSP SYSTEM

With the development of Large Scale Integrated Circuits, video coding systems on DSP are increasingly adopted. Computational resources and chip memory in DSP are so limited that reducing computation and memory cost has turned out to be very important to the design of a real-time system based on DSP.

The research group to which the writer belongs have designed and implemented a real-time video coding system, using TI TMS320C6416 DSP as its main chip. It requires a MPEG-4 encoder with a frame rate of 25 fps (frames per second) for CCIR (4Cif 720*576) image over a 1.8 Mbps constant bitrate channel. Internal memory of C6416DSP is only 1Mb, while an input 4Cif frame is nearly 4.98Mb. Together with the program data and intermediate data of the coding process, necessary memory cost to encode a frame will be far more than 1Mb. Thus only part of the data can be loaded into the internal memory of DSP, and others must be stored in external SDRAM. As internal memory access is much faster than external SDRAM, it is highly recommendable to modify the encoding framework and take full advantage of the internal memory. The scheme utilized by the writer's group divides a frame into several slices and reads only one slice into the internal memory to finish its coding at a time. If only IME is used, a frame will be divided into 6 slices at least, and thus one slice consists of 6 rows of macroblocks. When Half-pel ME is adopted, interpolated image will expand to be 4 times as large as the input one. If the whole frame is interpolated before search, less than 2 rows of macroblocks will be allowed into the internal memory at a time, which frequently leads to mass data transfer between internal memory and external SDRAM. As a result, coding efficiency drops sharply. Therefore, interpolating the whole frame before search is an impossible scheme in the said system. In case of TSME, blocks must be interpolated where they are needed during the search, even though it will cause repeated interpolation at the same position.

Table 2 shows the performance of this encoder, when only IME is used together with half-pel TSME. Three 4Cif (720*576) standard test sequences, 'Cheer', 'Bus', and 'Stef', are tested here. In all of the following tables the column "***Time***" denotes average time of SME for every frame, "***Coding length***" denotes average bits of each coded frame, and "***Frame rate***" denotes the number of frames the system encodes per second.

Table 2. Experimental results of IME and TSME in an encoder based on DSP

sequences	IME				TSME			
	PSNR (dB)	Coding length (bit/frame)	Time (ms)	Frame rate (fps)	PSNR (db)	Coding length (bit/frame)	Time (ms)	Frame rate (fps)
Cheer	27.37	111066	0	31.743	28.34	111151	1189	22.937
Bus	27.95	111092	0	35.945	29.31	111387	1151	24.621

Stef	29.06	110966	0	36.026	30.85	110920	1231	24.801

From the table above, it can be easily found that, when TSME is used, the PSNR value enhances obviously, but the frame rate of the encoder decreases sharply to less than 25 fps. In order to resolve the problem, FSME* herein proposed is implemented in the system. In so doing, the total check points are fixed no matter what sub-pel accuracy is adopted. Therefore, in half-pel SME, the speed advantage seems not as outstanding as in 1/4-pel, 1/8-pel and the sequent ones. Nevertheless, it still helps to improve image quality without excessive computational cost.

In addition, it should be noted that in this encoder, the algorithm amvFast [9] is used to finish IME, which usually does not check all of the four blocks around the best integer-pel match block. Yet in FSME*, these intermediate values are necessary to calculate the best position. In this case, two choices are available. One way is to calculate these values in addition, which will increase the computational complexity to some extent. The other is to omit calculating these additional values and exclude the positions involved, which is timesaving yet at the cost of a possible loss of some good matches. Correspondingly, FSME* is modified into FSME*-1 and FSME*-2, both of which are implemented in the said system. Experimental results are shown as follows. In the following tables the column '**Compare with TSME**' compares the algorithms herein adopted with TSME, '**ΔPSNR(dB)'** denotes difference of PSNR value, '**ΔCoding length'** denotes the rate of coding length change, '**ΔTime**' denotes the rate of time change, and the sign '+' means increase while '-' means decrease.

Table 3. Experimental results of FSME*-1 on DSP

sequences	PSNR (dB)	Coding length (bit/frame)	Time (ms)	Frame rate (fps)	Compare with TSME		
					ΔPSNR(dB)	ΔCoding length(%)	ΔTime(%)
Cheer	28.27	111172	624	26.016	-0.07	+0.00	-47.52
Bus	29.24	110862	599	29.008	-0.07	-0.47	-47.96
Stef	30.83	111621	625	28.816	-0.02	+0.63	-49.23

Table 4. Experimental results of FSME*-2 on DSP

sequences	PSNR (dB)	Coding length (bit/frame)	Time (ms)	Frame rate (fps)	Compare with TSME		
					ΔPSNR(dB)	ΔCoding length(%)	ΔTime(%)
Cheer	28.02	111168	342	28.326	-0.32	+0.01	-71.24
Bus	28.87	111231	354	31.486	-0.44	-0.14	-70.24
Stef	30.31	110996	338	31.739	-0.54	+0.06	-72.54

Experiments show that FSME*-1 obtains similar PSNR to TSME with nearly 50% time cost drop. With this algorithm, the encoder's frame rate can reach over 25 fps. As to FSME*-2, although it is obviously timesaving, the image distortion increases. However, compared with IME, it still enjoys a good quality improvement. Thus given limited computation and storage resources, FSME*-2 is still an excellent alternative scheme.

VI. CONCLUSION

This paper proposes a novel fast sub-pel motion estimation algorithm named FSME*, which directly calculates the best position in sub-pixel level as opposed to the traditional way of checking points one by one and two at most based on the calculated position. Computation analysis reveals that in $1/2^K$ sub-pel SME, the check points decrease by more than $(8\times K-2)/8\times K$. Experiments on JVT reference software show that when 1/4 sub-pel SME is applied, the check points decrease by over 90% without significant change of image quality and coding bits. Meanwhile, as no additional memory cost is needed for interpolation image, it performs well on a real-time

encoder based on DSP.

REFERENCES

[1] Lifeng Zhao, C.-C. Jay Kuo. Fast predictive integer- and half-pel motion search for interlaced video coding. Circuits and Systems, 2003. ISCAS '03. Proceedings of the 2003 International Symposium on, 2003, 2(2):848~851.

[2] Chen WQ, Gao W. Experimental study on fast half pixel search methods of motion estimation. Chinese Journal of Image and Graph, 1998, 3(7):557~561

[3] Bemd Cirod. Motion-compensating prediction with fractional-pel accuracy. IEEE Transactions on Communications, 1993, 41(4): 604~612.

[4] Krit Panusopone, David M. Baylon. An analysis and efficient implementation of half-pel motion estimation. IEEE Transactions on Circuits and Systems for Video Technology, 2002, 12(8):724~729.

[5] Cheng Du, Yun He, and Junli Zheng. A parabolic prediction-based, fast half-pixel search algorithm for very low bit-rate moving-picture coding. IEEE Transactions on Circuits and Systems for Video Technology, 2003,13(6):514~518.

[6] Siu-Leong lu.Comparison of motion compensation using different degrees of sub-pixel accuracy for interfield/interframe hybrid coding of HDTV image sequences.Acoustics, Speech, and Signal Processing, IEEE International Conference On, 1992, 3(3):465~468.

[7] Wang WD, Yao QD. Quick Algorithm of Sub-pixel Accuracy Motion Estimation. Chinese Signal Processing, 2002, 18(1):49~51.

[8] Wang WD, Yao QD. Fast algorithm of fractional pixel accuracy motion estimation. Journal of China Institute of communication, 2003, 24(4):128~132.

[9] Li B, Li W, Tu Y M. A Fast Block Matching Algorithm Using Smooth Motion Vector Field Adaptive Search Technique. Journal of Computer Science and Technology, 2003, 18(1):168~173.

[10] Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, 4th Meeting: Klagenfurt, Austria, 22-26 July, 2002, Document JVT-D157

[11] Moving Picture Coding for Low Bit Rate Communication, ITU-T Recommendation H.263, Jan. 1998

[12] ISO/IEC JTC1/SC29/WG11 N3093 Coding of moving pictures and audio, Maui, Dec. 1999.