IEEE Transactions on Consumer Electronics, Vol. 44, No. 3, AUGUST 1998

A FAST THREE-STEP SEARCH ALGORITHM WITH MINIMUM CHECKING POINTS USING UNIMODAL ERROR SURFACE ASSUMPTION

Jong-Nam Kim and Tae-Sun Choi

ABSTRACT---Fast motion estimation algorithms for compression of moving pictures are widely required for real-time video encoding instead of full search (FS) block matching algorithm. It provides optimal error performance but requires enormous computation for calculating motion vector. In many fast algorithms, the three-step search (TSS) has been used for real time video encoding and low bit-rate video communications because of reduced computations, simplicity and reasonable performance. Other modified algorithms of TSS have been studied for the speed of computation and the error performance for motion estimation. This paper proposes a new algorithm (Fast Three-Step Search) for further reduction in computational complexity using unimodal error surface assumption (UESA) without serious degradation of error performance. We'll show that computational complexity will be further reduced with additional subsampling of the matching block. It will be shown that the proposed algorithm (FTSS) is computationally efficient while keeping the same performance as one of TSS.

1. INTRODUCTION

Motion estimation plays a key role for the compression of moving picture data. For the motion vector in motion estimation, block matching algorithms (BMA) of many estimation techniques are most appropriate in the framework of generation coding [1]. A straightforward BMA is the full search (FS) which exhaustively searches for the accurate motion vector. This algorithm searches all locations in a specific search range and selects the position where the residual error of block matching is minimized.

The heavy computational load of the FS, however, can be a significant problem in real time video coding. Many fast algorithms for the reduction of computation have been developed in the last decade. We can classify these fast algorithms into several groups, which are the techniques based on UESA, the methods found on multiresolution, the means built on spatial/temporal correlation of the motion vectors, the fast algorithms with subsampling of matching block, etc.

Tae-Sun Choi(correnponding author) is with Dept. of Mechatronics, K-JIST, 572 Ssangam-dong, Kwangsan-ku, Kwangju 506-712, Korea
Tel: 82-62-970-2419, Fax: 82-62-970-2384
E-mail: tschoi@pia.kjist.ac.kr,
Jong-Nam Kim is with Dept. of Mechatronics, K-JIST, 572 Ssangam-dong, Kwangsan-ku, Kwangju 506-712, Korea
Tel: 82-62-970-2419, Fax: 82-62-970-2384
E-mail: jnkim@sipl.kjist.ac.kr

The fast motion estimation techniques based on UESA mainly constrain the number of checking points by applying this assumption to matching block. UESA means that the residual error of block matching increases monotonically when the checking point moves away from the location of the global minimum error. So many algorithms based on this concept have been reported for last decade. These techniques have one fault that the search could fall into a local minimum, which is not the optimal motion vector. It has been reported that these search algorithms could reduce the computational requirement significantly. Among these fast algorithms, TSS is the most popular algorithm and recommended by RM8 of CCITT and SM3 of MPEG due to its simplicity, regularity and reasonable performance [2]-[10].

Another approach of fast block matching algorithm uses coarse-to-fine strategy in terms of resolution. With this strategy, one can refine the obtained motion vector in a higher resolution after obtaining a coarse motion vector in a low resolution. We can further divide this method into two groups which are variable block size [11] and constant block size [12]-[15]. Methods under this strategy work relatively well and result in fast computation for motion vector search.

Another type for fast motion estimation is employing information from causal or noncausal adjacent blocks using spatial or temporal correlation of the motion vectors [16]-[17],[21]. The key idea of this approach is to choose the best one as the initial candidate for further refinement after selecting a set of initial candidate of motion vector from spatial/temporal neighboring blocks. This method must solve the problem of continuous motion and random motion. Recently, many researches have been reported for solving the problem [18]-[20].

Another efficient technique to reduce the computational complexity for motion vector is pixel subsampling of matching block. In this approach with the subsampling, all pixels of the matching block are examined instead of restricting the number of checking points using UESA. Several researches about subsampling of matching block have been presented. A popular algorithm for subsampling was proposed by Liu et. al [21], which is referred to as alternating subsampling. It is shown that using all four patterns in a specific alternating manner gives a better performance than simple 4:1 pixel

Manuscript received June 17, 1998

subsampling [21]. Besides the algorithm, other algorithms are also getting attention under this concept [22]-[24].

Many other algorithms which include bit-plane techniques, successive elimination search algorithm, its modified algorithm, simpler error criteria of matching block, variable search range methods based on the correlation of motion vector, variable block size according to image complexity, genetic motion search algorithm, the method using Kalman filtering, etc. were proposed [1],[25]-[30].

It is reported that TSS shows reasonable performance for local minimum in the real image sequences [7]. It is still exciting to further study the TSS to improve the estimation performance and further reduce the computational complexity. In [8], a new three-step search (NTSS) algorithm was proposed to improve the performance with the complexity similar to that of the TSS. The algorithm [7] was proposed to reduce more the computational complexity with the performance similar to TSS. We will decrease more the number of the redundant checking points with the proposed algorithm. Our algorithm makes use of the unimodal error surface assumption with constrained assumption and pixel decimation of matching block.

This paper is organized as follows: We first explain briefly several conventional fast motion estimation algorithms using UESA in Section 2. Especially, TSS algorithm is dealt with in more detail, which is related to our main algorithm (FTSS) in Section 3. We propose the Fast Three-Step Search algorithm which has minimum checking points among modified TSS algorithms, by restricting candidate checking points based on UESA in Section 3. For additional computation reduction, we combine this algorithm with subsampling of matching block. In Section 4, simulation results and discussion are given. Finally, the conclusion is followed in Section 5.

2. FAST ALGORITHMS USING UESA

As described earlier, fast motion estimation algorithms using UESA limit candidates among all checking points in the matching block. This gives rise to inevitable problem of falling into the local minimum in the real image sequences. It is reported that TSS algorithm shows reasonable performance for computational reduction ratio. Before developing our main algorithm based on UESA, we refer to several popular algorithms using UESA. The popular algorithms based on this concept include the three-step search (TSS), the new three-step search (NTSS), the four step search (FSS), the conjugate directional search (CDS), one-at-a-time search (OTS), the 2-D logarithm search (2-DLOGS), 1-D full search (1-DFS), the parallel hierarchical one-dimensional search (PHODS), efficient-simple search (ESS), and their modified algorithms [2]-[10]. We explain to the modified TSS algorithms selectively among these algorithms.

The full search BMA requires the number of $(2P + 1)^2 * N^2$ computations where P presents search range and N shows the size of matching block. It has tremendous amount of computations and can be significant problem for the real-time video encoding process. To reduce the computational load for checking points, the TSS algorithm is used widely. Fig. 1 shows the TSS procedure for $P=7$. In Fig. 1, motion vector (MV) is (5, -7). As in Fig. 1, the TSS algorithm constraints the number of checking points. In general, the number (S) of the steps needed for fixed P *is*

$$S = \lceil \log_2(P + 1) \rceil \qquad (1)$$

where $\lceil a \rceil$ is the smallest integer which is larger than or equal to a. The pixel interval ($PI(n)$) of the nth step from minimum error point in the previous step is

$$PI(n)=2^{(S-n)} \qquad (2)$$

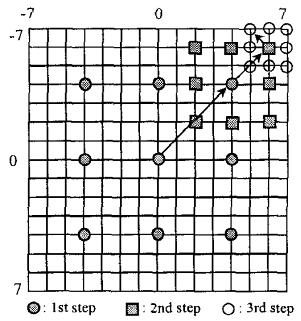

Fig. 1. The conventional TSS algorithm procedure.

The uniformly distributed search pattern of TSS algorithm results in simplicity and regularity. For $P=7$, the total checking points of TSS are 25. Meanwhile, the total number of checking points of FS is 225. Therefore, the speed ratio between TSS and FS is 225/25=9. For the speed up of computation of motion vector and accuracy of motion vector, many modified algorithms of the TSS algorithm have been studied [2],[3],[7],[8].

It is shown that TSS algorithm is inefficient in terms of computation for the images with inactive motion. That is, TSS has unnecessarily many checking points when the motion is small for the block. Therefore a new three-step search (NTSS) algorithm was proposed for small motioned images by R. Li et. al [8]. The features of NTSS are that it uses a center-biased checking pattern, which employs 17 checking points in the first step unlike TSS algorithm, and makes the search adaptive to the distribution of the motion vector. This adaptive search means that halfway-stop is possible in the first or second step when motion vector is in the range of (±1, ±1). However, the algorithm has unnecessary checking points for big motion images. Fig. 2 shows the search procedure of NTSS where the motion vector is (7,-3). It is reported that built on the center-biased motion vector distribution, NTSS algorithm results in improved performance over the TSS [8].

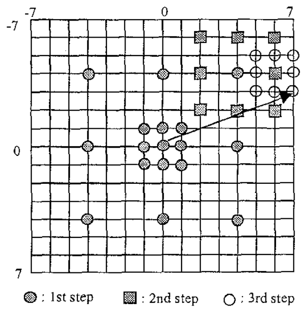

Fig. 2. The conventional NTSS algorithm procedure.

Computational complexity of NTSS algorithm is higher than the TSS in the worst case, and can be increased more than TSS even in terms of average computation. Meanwhile, for the real-rime video encoding the computational complexity even in the worst case should be considered as well as the average computational complexity. With this reason, a four-step search (FSS) algorithm based on the center-biased motion vector distribution characteristic is proposed. It was shown that the FSS results in better performance than the TSS and has similar performance to one of NTSS. In terms of checking points for the worst case, FSS requires 27 checking points instead of 33 candidates of NTSS. Fig. 3 shows the motion estimation of FSS pointing to the motion vector of (7, -3) [5],[6].

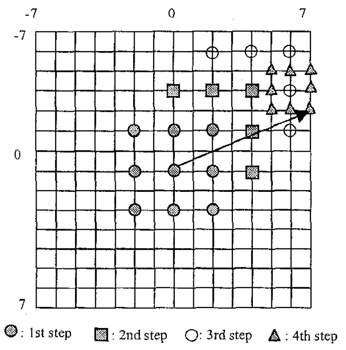

Fig. 3. The conventional FSS algorithm procedure.

Unlike the previous search algorithms, parallel hierarchical one-dimensional search is done independently along the two dimensions. For VLSI technology, this parallel processing is very important for speedup of computation. The feature of the search algorithm is that motion vector search is performed for horizontal and vertical axes independently. Therefore, PHODS algorithm has two distinct advantages over TSS algorithm. One is parallel calculation of MAE. The other is regular data flow of search process, because the search locations are along the horizontal axis and vertical axis.

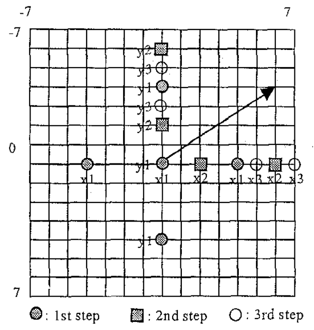

Fig. 4. The conventional PHODS algorithm procedure.

Experimental results show that the TSS provides a good performance even when the UESA does not exactly hold [7]. Meanwhile, it is still of interest to study TSS in order to improve the performance or further reduce the computational complexity. Under the purpose, efficient and simple search (ESS) algorithm further reduces the computational complexity. It was shown that the ESS can further speed up the TSS by a factor of two and keep its regularity and good performance comparable with the TSS [7].

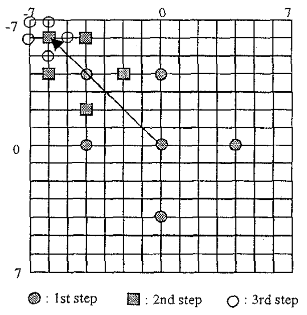

Fig. 5. The conventional ESS algorithm procedure.

We summarize the above described algorithms in terms of the computation reduction, which is expressed in checking points for the matching block. TSS and PHODS have fixed checking points and the others have variable candidates.

Table 1. Checking Points for the Conventional Algorithms

TSS	NTSS	FSS	PHODS	ESS
25	17 ~ 33	17~27	13	10~16

3. PROPOSED ALGORITHM

As described earlier, TSS shows reasonable performance for local minimum in real image sequences [7]. In [7], we can see that many checking points for the motion estimation are reduced compared with the original TSS algorithm. We will decrease more the number of unnecessary checking points with the proposed algorithm. It makes use of the unimodal error surface assumption with applying the assumption more tightly and pixel decimation of matching block. Therefore, in the proposed algorithm, we reduce the amount of the computation for checking the candidate motion vector using the unimodal error surface assumption (UESA) and subsampling of matching block. As described previously, the UESA means that matching error increases monotonically while getting away from the global minimum of matching error. It is reported that reasonable performance is obtained with many TSS algorithms experimentally even though this assumption is always not exact in the real image sequences. Proposed algorithm focuses on the reduction of computation without serious degradation of error performance.

3.1 Computational Reduction Based on Unimodal Error Surface Assumption

We'll derive the proposed algorithm-Fast Three-Step Search (FTSS) from the conventional TSS algorithm. We'll keep the structure of three-step, while reducing checking points among the candidates. We check only three points at first instead of checking nine points for each step. Then we decide the direction of additional checking points based on the relationship of examined three points. Fig. 6 shows search direction and additional checking points based on compared three points. In this figure, shaded points *x, y* and *z* are examined points at first and unshaded points are the points to be compared in the next procedure.

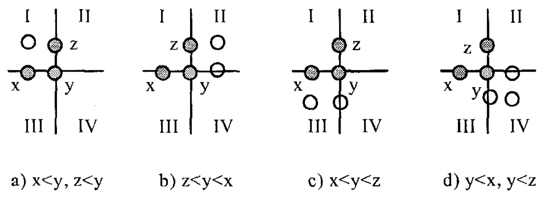

Fig. 6. Next search direction and additional checking points based on the relationship of the first three points.

With the direction determined from the first three points, we can search the next checking points as shown in Fig. 7. However, we'll not check all unshaded points for the determined direction as shown in Fig. 6. For b), c) and d) of Fig. 6, We'll reduce one or two more checking points based on constrained UESA. Fig. 7 shows next procedure after checking three points at first. In this figure, shaded rectangular points in d) and g) are compared after checking shaded circular points and unshaded circular point in the proper condition.

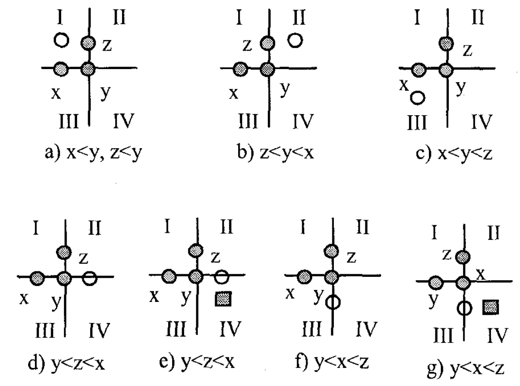

Fig. 7. Next checking points for the *MAD( )* at the locations *x, y* and *z*.

In Fig. 7, a) of the figure is the same as a) of Fig. 6. But b) of Fig. 7 is different from b) of Fig. 6. b) of Fig. 7 was resulted from b) of Fig. 6 with more constrained UESA. Therefore we examine additionally only one more point to the shaded points based on UESA. According to the same principle, we check only one point for c) of Fig. 6. In Fig. 6, d) can be divided into four cases-d) ~ g) of Fig. 7. So, we can reduce more checking points by dividing d) of Fig. 6 into four cases as d) of Fig. 7 instead of checking three points unconditionally. In the condition of d) of Fig. 7, we check additional one point based on the first three points. If y is still minimum MAD, there is no next search at the step. If newly checked point is minimal MAD, another point is examined like shaded rectangular point of e) of Fig. 7. In a similar manner, f) and g) of Fig. 7 can be explained as d) and e) of Fig. 7. This consideration for dividing d) of Fig. 6 into four cases like Fig. 7 is more efficient in small motion-images compared with [7]. We can obtain motion vector with the amount of computation equal to 2/5 of that of the conventional TSS algorithm by this algorithm in the images of small movement. We can calculate motion vector with the amount of computation of 2/3 of one of [7] by using this algorithm for inactive images. The number of checking points in experiments was less than theoretical value without serious degradation of error performance.

We'll reconsider above algorithm with pseudo-code for one step. Fig. 8 represents the candidate checking points in the first step. The algorithm for reducing the checking points using the UESA is as follows. In this algorithm, *MAD(x)* is Mean Absolute Difference at the location *x*.

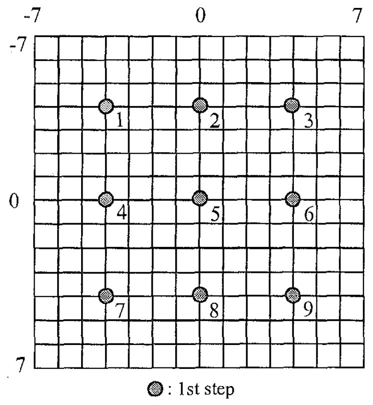

Fig. 8. Candidate checking points in first step.

First of all, we will compare MAD(5), MAD(6), and MAD(8) and sort these value in the order of magnitude. Table 2 shows pseudo code for the proposed algorithm.

Table 2. Pseudo-code for the Proposed Algorithm

```
• If MAD(6) < MAD(5) & MAD(8) < MAD(5)
                 check 9 & select min{MAD(x)}        (4 pts)
  End

• If MAD(6) < MAD(5) < MAD(8)
      then check 3 & select min{MAD(x)}              (4 pts)
  End

• If MAD(8)< MAD(5)< MAD(6)
      then check 7 & select min{MAD(x)}              (4 pts)
  End

• If MAD(5) < MAD(6) & MAD(5) < MAD(8)
    If MAD(5)< MAD(6)< MAD(8)
      If MAD(2)< MAD(5)
        then check 1 & select min{MAD(x)}            (5 pts)
      else  min{MAD(x)}= MAD(5)                      (4 pts)
      end
    end
    If MAD(5)< MAD(8)< MAD(6)
      If MAD(4)< MAD(5)
        then check 1 & select min{MAD(x)}            (5 pts)
      else  min{MAD(x)}= MAD(5)                      (4 pts)
      end
    end
End
```

The proposed algorithm using the unimodal error surface assumption ignores the unnecessary checking points. In the proposed algorithm, we can see that the case of five points checking occurs twice and one of four point checking occurs twice in the one direction. In the other directions, four points are always checked. Therefore, the number of average checking points for the first step is

$$(4 + 4 + 4 + 4.5)/4=4.125 \text{ points.}$$

The number of average checking points for the second step and third step is

$$(4 + 4 + 4 + 4.5)/4 -1= 3.125 \text{ points.}$$

The number of the checking points in the second and third step is reduced by one point compared with the first step because checking of the second and third step is performed with 8 points instead of 9 points. Therefore, the total checking points are

$$3.125*2 +4.125=10.375 \text{ points.}$$

The number of computed checking points is less than half of 25 points of TSS. The improvement with respect to computation reduction is

$$25/10.375=2.4096 \text{ times.}$$

The characteristic of the proposed algorithm keeps 4 checking points for each step when minimum MAD of the step is center. But [7] has maximum checking points as motion vector is (± 7, ± 7). This fact is important for the application of small motion images such as videophone, teleconferencing, etc.

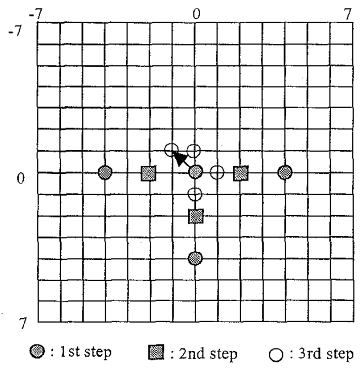

Fig. 8. The FTSS algorithm procedure.

Fig. 8 shows the procedure of the FTSS for calculating motion vector. As mentioned previously, we can make sure that the algorithm has four checking points for each step when minimum MAD for the step is center. The motion vector is (-1,-1) and total checking points are eleven points.

We'll summarize the proposed algorithm with flowchart of Fig. 9. This flowchart shows searching procedure for each step of FTSS. 'Setp_MV' in Fig. 9 points to motion vector for the step.

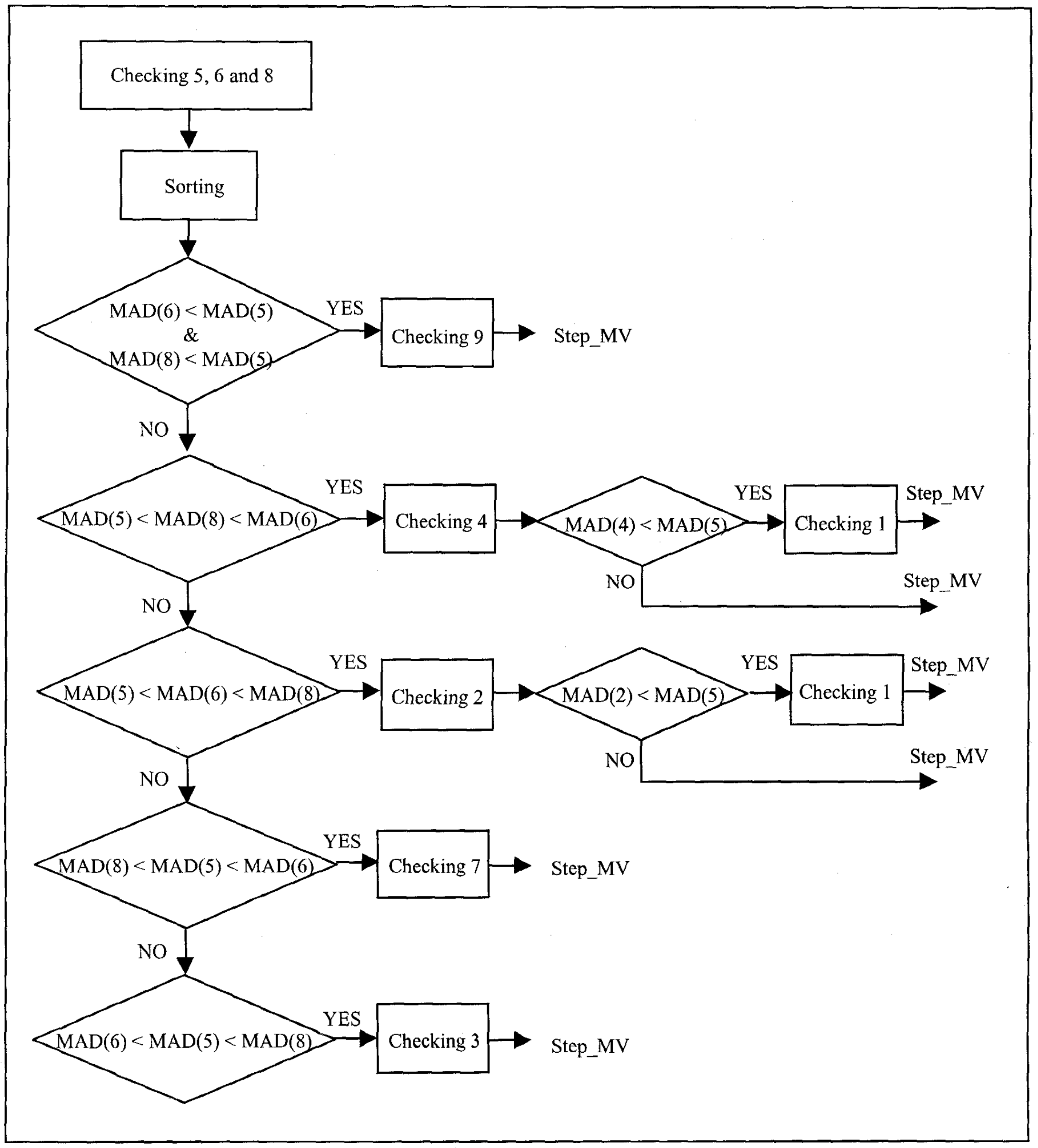

Fig. 9. Flowchart of the proposed algorithm (FTSS).

3.2 Computational Reduction Based on Subsampling of Matching Block

Another good technique to reduce the computation of motion estimation is pixel subsampling of block matching. Many researches about pixel subsampling of block matching have been reported for a decade. By this technique, we can reduce the number of operations by a factor of four without serious degradation of error performance. among these researches, Liu and et al's alternate subsampling [21] is very popular for reducing computation of motion estimation. It was reported that using four patterns in alternating pattern results in a better error performance than employing only subsampling pattern. Besides, [22] proposed fast motion estimation algorithm using partial distortion measure, which is obtained by calculating the distortion between the decimated block of current frame and that of previous frame. While these two algorithms use regular subsampling pattern, next two algorithms use irregular subsampling pattern. [23] proposed gradient-assisted pixel decimation which selects pixels with larger gradient magnitude for subsampling. It is reported that motion compensation error is proportional to the gradient magnitude. [24] announced check-board algorithm for subsampling. He utilized the fact that the random sampling pattern reduces the probability of systematically passing over a region of pixels and decreasing the accuracy of the resultant motion vector. In our work, we employed simple one sampling pattern because FTSS is not full search algorithm. Because searching points are irregular, error performance is less dependent on the subsampling patterns. Experimental results showed reasonable error performance for computational reduction ratio between whole matching block algorithm and subsampled matching block algorithm. Fig. 10 displays simple subsampling pattern

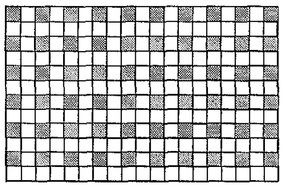
Fig. 10. Subsampling Pattern of Matching Block.

4. SIMULATION AND DISCUSSION

To test the proposed algorithm, we used 150 frames of "Salesman", "Missamerica, "Susie", and 'Football" images. Experiments were performed with the Three-Step Search (TSS), the Parallel Hierarchical One-Dimensional Search (PHODS), the Efficient and Simple Search (ESS) and our proposed Fast Three-Step Search (FTSS) algorithms. The block size was 16 by 16 pixels and the search range was ±7 pixels. Image size was CIF (288 by 352) for each sequence and only forward prediction was used. MAD as error criterion for finding optimal motion vector was used. The simulation results are presented in terms of PSNR, MSE and average checking points per block for each frame. These criteria for error and performance are described as follows. *M, N* are the parameters about size of the matching block and *i, j* are related to position information in the matching block. *f(i,j)* points to original pixel of current frame and *f'(i,j)* shows predicted pixel from past frame.

$$MAD = \frac{1}{M*N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left|f(i,j)-f'(i,j)\right| \tag{3}$$

$$PSNR = 10\log_{10}\frac{255^2}{\frac{1}{M*N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(f(i,j)-f'(i,j))^2} \tag{4}$$

$$MSE=\frac{1}{M*N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(f(i,j)-f'(i,j))^2 \tag{5}$$

Average Checking Points =

(Total Checking Points) / (Number of Blocks) (6)

To compare the performance of our algorithm, we present various results with several conventional algorithms. Fig. 11 ~ Fig. 14 show PSNR performance for 'Salesman' sequence and Fig. 15 presents average checking points per block for 'Salesman' sequence. Table 3 shows average PSNR performance, Table 4 average MSE performance, Table 5 average checking points for each block and Table 6 computational reduction ratio for previously mentioned algorithms with listed sequence above. As described previously, our proposed algorithm focuses on reducing the amount of computation without serious degradation of predicted image rather than on quality of predicted image. Two (TSS and PHODS) of the algorithms for experiment have fixed checking points for each block and the other algorithms have variable checking points dependent on each block. As expected, the TSS results in the best quality of predicted image among the experimented algorithms. But this algorithm takes the most time for calculating motion vector. The ESS shows reasonable error performance for saved computation. The PHODS gets the lowest error performance compared with the conventional algorithms employing minimum checking points. Our proposed algorithm (FTSS and FTSS+Subsampling) shows good performance with minimum checking points among the experimented algorithms. Specifically, even though our algorithm uses

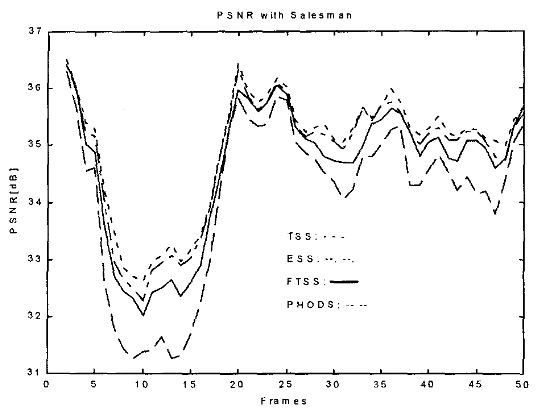

Fig. 11. PSNR performance with 'Salesman' sequence (2 : 50).

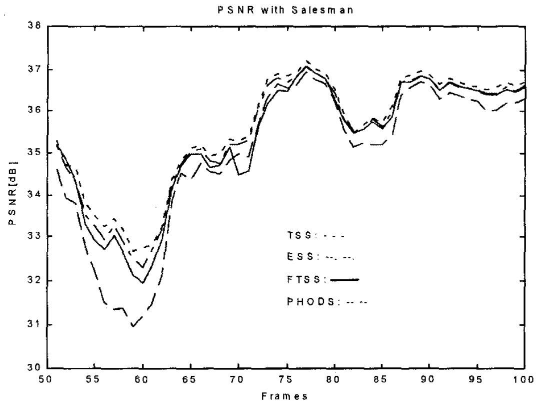

Fig. 12. PSNR performance with 'Salesman' sequence (51 : 100).

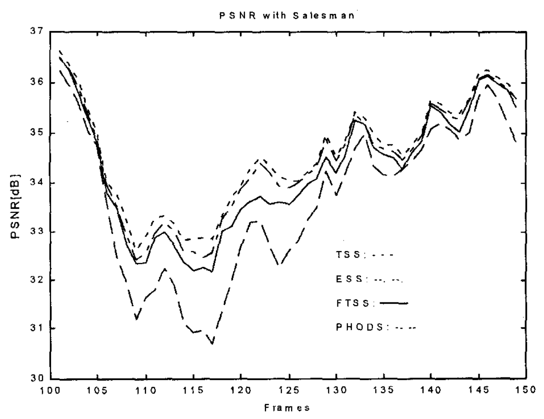

Fig. 13. PSNR performance with 'Salesman' sequence (101 : 150).

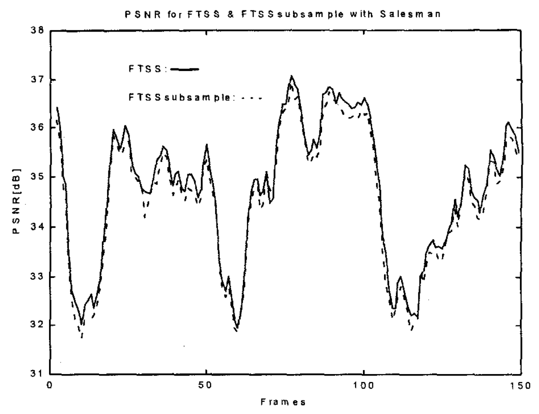

Fig. 14. PSNR performance for Algorithms FTSS and FTSS+Subsampling with 'Salesman' sequence.

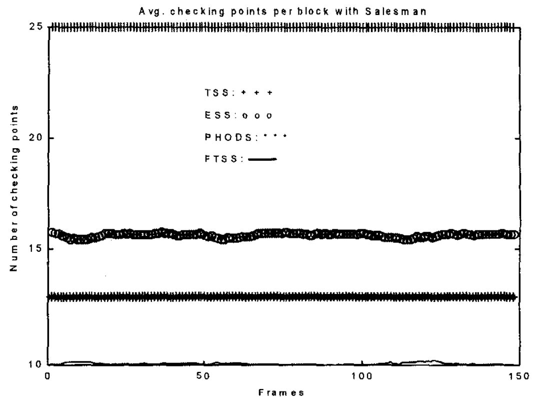

Fig. 15. Average checking Points for each block with 'Salesman' sequence.

less checking points than the PHODS algorithm, error performance of our algorithm is better than that of PHODS algorithm. We can see from presented PSNR performance that the error performance of our algorithm is close to that of TSS and ESS algorithms. Compared with computational reduction, the Algorithm FTSS+Subsampling shows very reasonable quality of predicted image and close error performance to FTSS algorithm.

In the view of checking points, ESS and our proposed algorithms have variable checking points for each block of frames. Theoretical maximum checking points for ESS algorithm are 16. Maximum points of theoretical value for FTSS are 13. Therefore, we can expect that the average difference of checking points between ESS and FTSS algorithms is 3. Fig. 15 and Table 5 tell us that the guess is wrong and average difference of checking points is larger than one of our forecast. This larger difference of checking points results from the characteristic of two algorithms.

The Algorithm ESS has maximum checking points when motion of the sequence is very large or the sequence has small motion. So, as well as motion vectors are (± 7, ±7), when motion vector is (0,0), the algorithm has maximum checking points. Meanwhile, the FTSS algorithm takes maximum checking points only when motion is large in the sequence. Therefore, the FTSS algorithm employs minimum checking points for the motion vector of (0,0). Table 5 well explains this fact.

Table 6 presents the computational reduction ratio for motion estimation in terms of employed checking points. We use checking points of the TSS as reference. At this condition, FTSS uses about 2/5 of checking points in the TSS. FTSS+Subsampling algorithm has 1/10 of checking points of the TSS. We can identify that good performance was obtained with these small numbers of checking points.

Table 3. Average PSNR Results

Images	Algorithms				
	TSS	SES	PHODS	FTSS	FTSS+Subsampling
Salesman	35.03	34.90	34.22	34.72	34.52
Missamerica	36.95	36.74	34.62	36.38	36.13
Susie	39.11	38.37	34.69	37.94	37.88
Football	26.78	26.23	24.40	25.75	25.65

Table 4. Average MSE Results

Images	Algorithms				
	TSS	SES	PHODS	FTSS	FTSS+Subsampling
Salesman	21.87	22.01	26.58	23.07	24.18
Missamerica	13.24	13.92	23.10	15.25	16.13
Susie	9.86	11.89	31.53	13.64	13.78
Football	144.48	161.30	244.41	179.29	182.89

Table 5. Average Checking Points for Each Block

Images	Algorithms				
	TSS	SES	PHODS	FTSS	FTSS+Subsampling
Salesman	25	15.64	13	10.06	10.06
Missamerica	25	14.92	13	10.21	10.21
Susie	25	15.28	13	10.26	10.26
Football	25	15.13	13	10.37	10.37

Table 6. Computational Reduction Ratio

Images	Algorithms				
	TSS	SES	PHODS	FTSS	FTSS+Subsampling
Salesman	1	0.626	0.52	0.402	0.100
Missamerica	1	0.597	0.52	0.408	0.102
Susie	1	0.611	0.52	0.410	0.103
Football	1	0.605	0.52	0.414	0.104

5. CONCLUSIONS

In this paper, several approaches for fast motion estimation algorithms and algorithms using UESA were discussed. Our proposed algorithm reduces the computational complexity using the UESA and simple 4:1 subsampling, while keeping the error performance similar to one of the TSS algorithm. With the proposed algorithm, we can calculate the motion vector with about 2/5 of the checking points of the TSS algorithm for search range P=$\pm$7. The performance combined with 4:1 subsampling shows almost the same performance as only FTSS algorithm. In terms of computational reduction, only 1/10 of computation for the TSS algorithm is required to do motion estimation.

The computational complexity can be further reduced by employing the similar algorithms as proposed in [28]-[30]. Because these algorithms are independent to the proposed algorithm, the combination of these techniques can reduce more the computation for motion estimation. Therefore, the proposed algorithm will be very useful in software based real time video coding field requiring fast motion estimation.

ACKNOWLEDGMENT

The authors would like to thank Y.K. Seong for his help in the experiment for testing the proposed algorithm and for his discussions in the early stages of this work. Also, his revision for the final version of the paper was very useful. M. Asif, B.S. Baek, and Tanzeem are acknowledged for their help in revising the paper carefully.

REFERENCES

[1] F. Dufaus and F. Moscheni, "Motion estimation techniques for digital TV: A review and a new contribution," *Proc. IEEE*, vol. 83, pp. 858-876, Jun. 1995.

[2] J.R. Jain and A.K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Trans. commun.*, vol. COM-29, pp. 1799-1808, Dec. 1981.

[3] R. Srinivasan and K.R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. commun.*, vol. COM-33, pp. 888-896, Aug. 1985.

[4] L.G. Chen, W.T. Chen, Y. Sjehng, and T.D. Chiueh, "An efficient parallel motional algorithm for digital image processing," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 1, pp. 378-385, Sept. 1991.

[5] L.M. Po and W.C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, Jun. 1996.

[6] D.H. Lee, and S.H. Cho, "An efficient VLSI architecture for motion estimation with 4-step search algorithm," *Proc. ITC-CSCC'97 Okinawa, Japan,* pp.153-155, 1997.

[7] J. Lu, and M. L. Liou, "A simple and efficient search algorithm for block-matching motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 429-433, Apr. 1997.

[8] R. Li, B. Aeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, pp. 438-442, Aug. 1994.

[9] B. Zeng, R. Li, and M. L. Liou, "Optimization of fast block motion estimation algorithms," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 833-844, Dec. 1997.

[10] M. R. Pickering, J. F. Arnold, and M. R. Frater, "An adaptive search length algorithm for block matching motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 906-912, Dec. 1997.

[11] K. M. Uz, M. Vetterli, and D. LeGalll, "Interpolative multiresolution coding of advanced television with compatible subchannels," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 1, pp. 86-99, Mar. 1991.

[12] J. Chalidabhongse, and C.-C. Jay Kuo, "Fast motion vector estimation using multiresolution-spatio-temporal correlations," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 477-488, Jun. 1997.

[13] J. Wei, and Z.N. Li, "An enhancement to MRMC scheme in video compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 564-568, Jun. 1997.

[14] Y. Q. Shi, and X. Xia, "A thresholding multiresolution block matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 437-440, Feb. 1997.

[15] A. Wang, and M. Ghanbari, "Scalable coding of very high resolution video using the virtual zerotree," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 719-726, Feb. 1997.

[16] J.b. Xu, L.M. Po, and C.K. Cheung "A new prediction model search algorithm for fast block motion estimation," *Proc. ICIP*, pp. 610-613, 1997.

[17] L. Luo, C. Zou, and Z. He, "A new prediction search algorithm of block motion estimation for video coding," *Proc. ICASSP*, pp. 1175-1178, 1997.

[18] K.W. Lim, B.C. Song, and J.B. Ra, "Fast hierarchical block matching algorithm utilizing spatial motion vector correlation," *Proc. SPIE*, vol. 3024, pp. 284-292, 1997.

[19] B.C. Song, and J.B. Ra, "A hierarchical block matching algorithm using partial distortion criterion," *Proc. SPIE*, vol. 3309, pp. 88-95, 1997.

[20] B.C. Song, K.W. Lim, and J.B. Ra, "A fast motion estimation algorithm using spatial correlation of motion field and hierarchical search," *Proc. SPIE*, vol. 2952, pp. 308-315, 1996.

[21] B. Liu, and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, pp. 148-157, Apr. 1991.

[22] C.K. Cheung, and L.M. Po, "A hierarchical block matching algorithm using partial distortion measure," *Proc. ISCAS*, pp. 1237-1240, 1997.

[23] B. Tao, and M.T. Orchard, "Feature-accelerated block matching," *Proc. SPIE*, vol. 3309, pp. 469-476, 1997.

[24] N. Cao and B.W.Y. Wei, "A 4:1 checker-board algorithm for motion estimation," *Proc. SPIE*, vol. 2847, pp. 408-412, 1997.

[25] W. Li and E. Salari, "Successive elimination algorithm for motion estimation," *IEEE Trans. Image Processing*, vol. 4, pp. 105-107, Jan. 1995.

[26] K.H.K. Chow and M.L. Liou, "Genetic motion search algorithm for video compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, pp. 440-445 Dec. 1993.

[27] C. M. Kuo, C. H. Hsieh, Y. d. Jou, H. C. Lin, and P. C. Lu, "Motion estimation for video compression using kalman filtering," *IEEE Trans. Broadcast.*, vol. 42, pp. 110-116, Jun. 1996.

[28] B. Natarajan, V. Bhaskaran, and K. Konstantinides, "Low-complexity block-based motion estimation via one-bit transforms," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 702-706 Aug. 1997.

[29] L.W. Lee, J.F. Wang, J.Y. Lee, and J.D. Shie, "Dynamic search-window adjustment and interlaced search for block-matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 1, pp. 378-385, Feb. 1993.

[30] M.J. Chen, L.G. Chen, T.D. Chiueh, and Y.P, Lee, "A new block-matching criterion for motion estimation and its implementation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, pp. 231-236, Jun. 1995.

Jong-Nam Kim received the B.S. degree in electronic engineering from Kum-Oh National University of Technology, Korea, in 1995. He received the M.S. degree in information and communications engineering from Kwanju Institute of Science and Technology (K-JIST), Korea, in 1997. He is currently working towards the Ph.D. degree in the Department of Mechatronics at Kwangju Institute of Science and Technology, Korea. His research interests include image processing, scalable coding, fast motion estimation, post video processing, software-based video codec, and VLSI design for video coding.

Tae-Sun Choi received the B.S.degree in electrical engineering from Seoul National University, Korea, in 1976, the M.S. degree in electrical engineering from Korea Advanced Institute of Science and Technology in 1979, and the Ph.D. degree in electrical engineering from the State University of New York at Stony Brook in 1993. He is currently an assistant professor in the Department of Mechatronics at Kwangju Institute of Science and Technology in Korea. His research interests include image processing, machine/robot vision, and visual communications. He is a member of IEEE, OSA, SPIE, and Sigma Xi.