IJCSC International Journal of Computer Science & Communication Vol. 1, No. 1, January-June 2010, pp. 153-159

Multi Step Motion Estimation Algorithm

Deepak J. Jayaswal[1] **& Mukesh A. Zaveri**[2]

[1]Electronics and Telecommunication Department, St. Francis Institute of Technology, Mumbai 400103, India.
[2]Computer Engineering Department, S.V.National Institute of Technology, Surat 395007, India.
Email: [1]djjayaswal_vcet@ yahoo.com, [2]mazaveri@coed.svnit.ac.in

ABSTRACT

We propose a multi step motion estimation algorithm (MSME) that encompasses techniques such as motion vector prediction, determination of search range and search patterns, and identification of termination criteria that suits to all type of video characteristic. This approach allows us to exploit random distribution of motion vector in successive video frames from which the initial candidate predictors are derived. The derived predictors are the most probable points in search window, which will assure that, the motion vectors in the vicinity of center point and at the edge of the search window does not miss out, as it does for earlier algorithms like Three step search(TSS), Four step search(FSS), Diamond(DS), etc and refinement stage used in the algorithm will allow us to extract true motion vector so that the picture quality is as good as Full search(FS) which is the optimal algorithm. The novelty of the proposed MSME algorithm is that the search pattern derived is not static but can dynamically shrink or enlarge to account for small and large motion and fixed threshold used improves speed without sacrificing the quality of video. The Simulation result shows that our proposed algorithm outperforms all sub-optimal algorithms in terms of quality and speed up performance and in many cases PSNR of proposed algorithm is comparable to Full Search.

Keywords: Motion Vector, Block Matching Algorithm, Three Step Search, Four Step Search, Diamond Search, Full Search.

1. INTRODUCTION

Motion estimation (ME) and compensation are the keys to high quality video coding [1]. ME is a process of estimating the motion of macroblock (MB) of some predefined size from the reference frame to generate the current frame. Block Matching based motion estimation is used in most video codecs, including MPEG-2, MPEG-4 and H.263 [2]. ME is also a key component in the digital restoration of achieved video and for post production and special effect in the movie industry. ME and the exploitation of the strong correlation among the successive frames allows us to encode and transmit the motion vectors along with the error frame obtained using the regenerated current frame and the actual current frame and hence, reduces the number of bits used to convey the information. To achieve this bit reduction, various approaches and algorithms have been proposed in the literature [1, 2, 3,]. The most accurate BMA is the exhaustive FS method. The only disadvantage of this method and perhaps the biggest flaw is the high computational cost associated with it. Other algorithms with reduced number of computations, for example, the successive elimination algorithm (SEA) [4], a new three step search [5], a novel four-step search (FSS) [6], efficient four step search [7], unrestricted center biased diamond search (UCBDS) [8], cross search [9], fast full search motion estimation [10], complexity bounded motion estimation [11], predictive coding based on interframe efficient motion estimation [12], Among these algorithms, the SEA is similar to the full search method except, the first one eliminates certain search points based on the Minkowiski's inequality. Further reduction in number of search points was achieved in TSS algorithm. The main drawback of TSS is the relatively large search pattern in the first step having a distance of 4, which renders it inefficient for finding blocks with small motions. In order to exploit the characteristics of the center-biased motion vector distribution, FSS algorithm was proposed to speed up the search mechanism. The main drawback of FSS is the relatively small search pattern in the first step having a distance of 2, which renders it inefficient for finding blocks with large motions. TSS, FSS algorithm are static that is they have fixed search pattern and restricted window size, and are inefficient for predicting true motion vector for low or high motion. Other algorithm like appeared in [13–18] uses motion vector prediction (MVP) technique and has shown significant performance improvement in terms of quality. These algorithms provide better peak to signal ratio (PSNR) as compared to TSS, FSS, DS, etc. and are not static but are very complex and needs memory to store motion vectors for prediction. This makes it difficult for multimedia application where porting of video codec for embedded processor is required as well as for video streaming application for video on demand.

In this paper we suggest a probabilistic approach to determine the initial candidate predictor that is the most probable search point for first iteration and it is followed by refinement stage which allow us to extract true motion vector so that picture quality is as good as FS. The designed search pattern as mentioned earlier can dynamically shrink or enlarge depending on threshold criteria. The algorithm assures that 1) the search is not trapped in the local minima, 2) the search does not miss out on the motion vector found in the central region, 3) the search is terminated early by using fixed threshold without sacrificing PSNR, 4) computations are less 6) complexity is low.

The organization of the paper is as follows:

Section 2 presents proposed algorithm and explains the algorithm in detail considering the steps involved. Section 3 presents simulation results of our proposed algorithm in comparison with the algorithms like the FS, TSS, 4SS and DS.

2. PROPOSED MSME ALGORITHM

Our algorithm is based on stochastic approach which exploit random distribution of MVs in successive video frames for selection of search points within given search window, with the assumption that motion field varies slowly both spatially and temporally, therefore it is highly probable that MBs closer to the current MB, in time and space may show the same motion. If this assumption holds, instead of testing all possible MVs, as FS does, we can use the most probable MV that is the set of candidate predictors. Then the best candidate predictor, that is the MV with the lowest SSE, is sent to a refinement phase that allows obtaining the final MV. The algorithm described here operates in two phases. In first phase we determine the initial candidate predictor to begin the search and also ensure that search does not get trap in local minima and in second phase the refinement stage will look for true motion vector.

2.1. Initial Candidate Predictor

The driving idea is to reduce the cardinality of the candidate predictor set as much as possible and, at the same time, to put the "best" candidate in the set that is those MV that describe at best the motion. In order to find the best candidate for initial search, we propose the basic search pattern as shown in Fig.1. The search points in the central diamond region, at location (0, 0) (0, –1) (0, –2) (0, 1) (0, 2) (–1, –1) (–1, 0) (–1, 1) (–2, 0) (1, –1) (1, 0) (1, 1) (2, 0) are 13 most probable points which are derived by determining the motion vector distribution probability over a search window of $(2p + 1) \times (2p + 1)$.

Table I depicts the percentage motion vector distribution probability over search window of 15×15 ($p = 7$) for multiple frames of QCIF sequence using full search (FS) motion estimation algorithm. FS finds motion vectors by sequentially searching the whole 15×15 search window in the reference frame. A MB centered at each of the position in the window is compared to the MB in the target frame, pixel by pixel, and their respective sum of square (SSE) is then derived. The vector that offers the least MAD is designated the motion vector MV for the MB in the target frame. For each of the MV detected at location (i,j) in 15x15 search window a count is maintained for multiple frames of the sequence. The percentage MVP at a particular location (i, j) is % MVP (i, j) = 100 * (Number of MVs found at location (i, j) / Total number of MVs) = 100 * (Count(i, j) / Total number of MVs). For example if we consider 75 frames with frame size of 320×240 and MB size of 8×8, we have total $40 \times 30 \times 75$ numbers of MVs. If count at (i, j) location using FS equals 15 then

$$\%\text{MVP}(i, j) = \frac{15}{40 \times 30 \times 75} \times 100 = 0.11\%$$

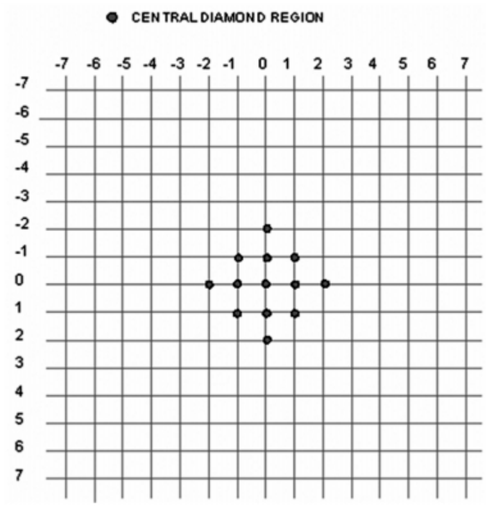

Fig 1: Basic Search Pattern of MSME

Table 1

Percentage Motion Vector Probability Distribution using CIF/QCIF Sequences

As observed from Table.1, the motion vector distribution is highly center biased with about 75% of the motion vectors being found in the central diamond region. To further justify our selection of 13 points in central diamond region, we analyze the shortcoming in earlier suboptimal algorithm like TSS, FSS, and DS. It is well known that the PSNR of these algorithms are low as compared to FS, the cause of failure of these algorithms is initial static search pattern which is unable to detect true motion vector that is TSS, FSS, and DS fails to find best match that FS has been able to find at particular location within 15x15 search window. To analyze the pitfalls of these algorithms we have generated error table using varieties of video sequences but we have shown result of akiyo sequence in Table. II where each entry in table indicates the percentage error that is number of times DS fail to find the true motion vector for multiple frames that FS has been able to find at particular location within 15 × 15 search window. The percentage error at location (i, j) is

$$\% \text{ error } (i, j) = 100 * \{1 - [(\text{number of MVs found by TSS at } (i, j) / \text{number of MVs found by FS at } (i, j)]\}$$

Hor / Ver	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
-7	0.07	0.03	0.03	0.04	0.03	0.03	0.04	0.09	0.03	0.03	0.03	0.03	0.02	0.02	0.06
-6	0.03	0.02	0.02	0.02	0.01	0.02	0.02	0.05	0.02	0.02	0.02	0.02	0.02	0.01	0.03
-5	0.03	0.02	0.02	0.02	0.01	0.02	0.03	0.06	0.03	0.02	0.02	0.02	0.02	0.01	0.03
-4	0.04	0.02	0.02	0.03	0.02	0.03	0.03	0.09	0.04	0.03	0.02	0.02	0.02	0.02	0.04
-3	0.37	0.03	0.02	0.02	0.03	0.03	0.04	0.12	0.05	0.04	0.03	0.02	0.03	0.03	0.07
-2	2.22	0.35	0.03	0.03	0.04	0.06	0.09	**0.24**	0.09	0.08	0.07	0.06	0.06	0.06	0.11
-1	0.53	1.37	0.58	0.60	0.40	0.16	**0.70**	**2.28**	**0.71**	0.24	0.14	0.11	0.10	0.09	0.21
0	0.05	1.61	2.03	2.34	1.42	**1.00**	**5.79**	**54.57**	**3.82**	**1.11**	0.57	0.43	0.37	0.28	0.66
1	0.03	0.61	0.67	0.59	0.30	0.19	**0.69**	**2.66**	**0.79**	0.21	0.13	0.12	0.10	0.08	0.16
2	0.05	0.04	0.04	0.05	0.05	0.05	0.08	**0.25**	0.21	0.10	0.05	0.06	0.06	0.04	0.08
3	0.03	0.02	0.03	0.03	0.03	0.03	0.03	0.13	0.07	0.06	0.04	0.03	0.05	0.03	0.05
4	0.03	0.02	0.02	0.02	0.02	0.02	0.03	0.08	0.04	0.05	0.04	0.05	0.05	0.05	0.06
5	0.03	0.02	0.02	0.02	0.02	0.02	0.02	0.06	0.03	0.05	0.04	0.02	0.04	0.06	0.07
6	0.03	0.02	0.02	0.02	0.02	0.02	0.02	0.06	0.02	0.02	0.02	0.01	0.02	0.03	0.04
7	0.05	0.03	0.03	0.03	0.03	0.02	0.02	0.08	0.03	0.02	0.02	0.02	0.02	0.02	0.06

The zero percent error at nine location in Table II is reflection of initial search points of DS but error accumulation in inner diamond region at location (–1, 0) (0, –1) (1, 0) (0, 1) is about 24% for DS, it is known that diamond search performs better than other two algorithms TSS and FSS, but still fails to account for MVs in inner diamond region. Therefore it is mandatory for any motion estimation algorithm, not to miss out on any MVs found in central diamond region.

Table 2
Percentage of Times Diamond Search have Failed to Find Best Match that Full Search has Found at Particular Point within 15 × 15 Search Window for 75 Frames of Akiyo Sequence.

	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
-7	81.91	81.72	80.53	74.09	77.36	77.09	76.44	66.96	73.21	78.86	81.52	83.38	84.87	87.06	80.2
-6	89.16	86.19	84.8	81.22	80.93	82.57	83.96	68.2	81.27	77.47	86.69	87.83	91.94	90.63	85.86
-5	86.27	91.13	77.91	80.83	75.69	74.06	76.31	64.5	69.47	75.89	83.48	82.94	87.41	88.89	84.54
-4	81.76	84.47	87.24	65.42	78.93	66.5	65.16	43.35	60.81	63.39	75.98	75.68	82.79	87.4	84.19
-3	76.61	81.85	76.64	77.33	47.88	62.05	43.96	26.1	41.86	50.96	59.83	79.53	79.08	79.28	79.36
-2	76.04	73.67	69.9	56.95	58.82	24.27	38.31	**0**	23.56	20.83	44.86	50.74	63.21	66.6	77.35
-1	75.26	69.81	61.56	51.43	32.88	36.53	**0**	7.14	**0**	23.12	32.02	53.58	58.27	65.37	65.11
0	69.31	67.57	53.83	42.42	18.11	**0**	4.17	**0**	5.79	**0**	24.44	38.94	46.05	51.59	54.67
1	70.53	70.19	62.81	50.16	35.41	35.66	**0**	6.7	**0**	26.37	34.23	49.81	54.52	64.95	64.82
2	83.71	82.75	80.92	66.75	64.36	33.41	29.75	**0**	16.6	14.8	48.87	37.2	53.76	58.24	61.92
3	82.08	90.93	83.17	77.13	64.91	59.91	43.02	24.17	25.43	40.73	45.67	63.15	57.6	72.03	72.25
4	87.1	91.69	88.29	80.83	77.54	64.35	65.74	41.93	56.89	44.46	62.5	54.05	65.78	63.02	72.15
5	86.03	90.81	87.2	87.42	84.64	76.95	70.99	56	72.71	51.46	54.1	72.39	63.46	72.15	70.04
6	88.19	89.34	91.21	88.02	83.23	85.11	82.09	70.9	81.56	71.55	78.44	86.28	85.28	71.73	77.18
7	80.59	87.34	89.9	85.69	76.54	79.64	77.81	65.91	79.73	79.43	85.9	82.53	85.4	88.33	79.35

Based on the probability distribution of motion vector and error table, our next logical step was to decide on number of search points in proximity of central point that is $mv(0, 0)$, for this we selected maximum 13 most probable points in central diamond region. We found that this 13 most probable points as shown in Fig.1. gives better PSNR result than diamond search at cost of few additional points. However, from simulation result of Table 8-9, it is observed, this algorithm, just like diamond search, gets trapped in local minima and often fails to find motion vectors which are closer to edge of the search window. In order to solve the problem of local minima trap and also look for the motion vector located at edge of search window. We propose the search pattern of Fig.2 as initial search for first iteration with most probable points in central diamond region and next probable 8 points in outer diamond region located at (–6, 0) (–3, –3) (–3, 3) (0, –6) (0, 6) (3, –3) (3, 3) (6, 0) to take care of motion vector located at edge of search window.

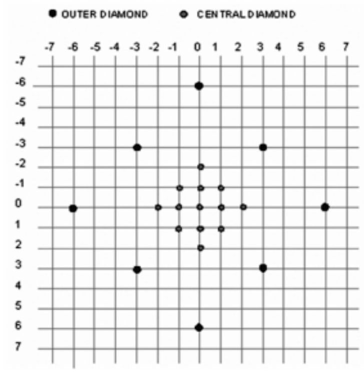

Fig 2: Search Pattern of MSME

This pattern has ability to shrink or enlarge depending on criteria mentioned, the shrinking can be as small as five points and enlarge may be as big as twenty one points. After initial iteration, if minimum cost function is found in the central diamond region the refinement stage uses eight point search (EPS) described in section IV-B and if minimum cost function is found in outer diamond the search algorithm uses DS to track true motion vector. Simulation result will show that this pattern do not get trap in local minima as well as look for true motion vector.

2.2. Refinement Stage

In this phase the refinement vectors try to correct the best candidate predictor searched in first phase to achieve the true motion vector. The refinement is done on the vectors obtain in first iteration to ensure that algorithm does not get trapped in local minima. Refinement phase let the motion field evolve and provide the required resolution. Since there is not a preferred direction, the refinement vectors are placed as a grid, centered on the best predictor, made of four points on cross direction and four on diagonal directions for motion vector found in central diamond region. These correction vectors of the refinement grid have a very limited range, one pixel, but they are able to give a good correction in most cases. However, in some circumstances, for example the MV found at the edge of the search window, it is convenient to enlarge the grid to recover the correct motion, in other words use diamond search to locate true motion vector. In our algorithm refinement grid shrinks if minimum SSE is found in central diamond that is standard EPS algorithm is used. It enlarges if minimum SSE is found in outer diamond that is DS algorithm used.

2.3. Early Termination of Algorithm

To reduce computation in first iteration and iteration to follow without sacrificing the quality of video the algorithm uses technique of early termination using fixed threshold. In all of the video sequences that we analyzed using the FS, the average PSNR, was in the range of 22 dB to 45 dB. For early termination of search, selecting PSNR of 22dB will degrade the quality and choosing PSNR much above 45 dB increases the number of search points. Considering this fact, we can safely assume that for all practical purposes, a PSNR of 45 can be considered to be a "good PSNR".

Now,

$PSNR = 10 * \log (255^2/MSE)$

Therefore, $45 = 10 * \log (255^2/MSE)$

Therefore, $MSE = 2.056$

Therefore, for an 8×8 macroblock,

$SSE = 64 * 2.056 = 131$ (approx.)

Thus, we can say that a match is a good match if the SSE is less than or equal to 131, the threshold SSE. The graph of Fig.3 justifies our selection of fixed threshold (45 dB) and effect of selecting lower and higher threshold on picture quality and number of search points (NSP).

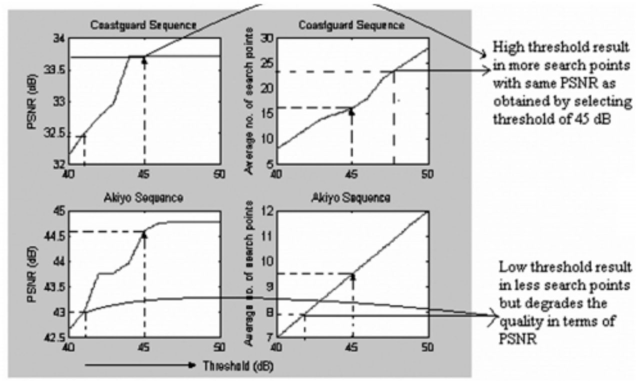

Fig 3: Effect of Choice of Threshold on Picture Quality and on NSP

2.4. Search Strategy

The flow chart for the aforementioned MSME is depicted in Fig.4.

1. Initially search inner five points of central diamond.
   a. Obtain the best match SSEmin(S1) from this five points
   b. If SSEmin (S1) < Threshold, the best match is S1 transmit vector location of S1 and go to step 4.
   c. If S1 > Threshold go to step 2.
2. Consider all 13 search points of central diamond for the pattern shown in Fig.2. Calculate the SSE at every point proceeding sequentially from the central diamond point to outer points.

   Case i: If any of these points has SSE value less than or equal to the threshold transmit vector location of that point and go to step 4.

Case ii: If match is found in the central diamond region, continue the search by using EPS and find the best match (threshold condition applies) and transmit vector location of that point and go to step 4.

Case iii: If match is found in the outer 8 points continue the search by using the DS and find the best match (threshold condition applies) and transmit vector location of that point and go to step 4.

Case iv: If SSEmin is greater than Threshold go to step 3.

3. Compute best match for all points and transmit SSEmin vector location
4. End.

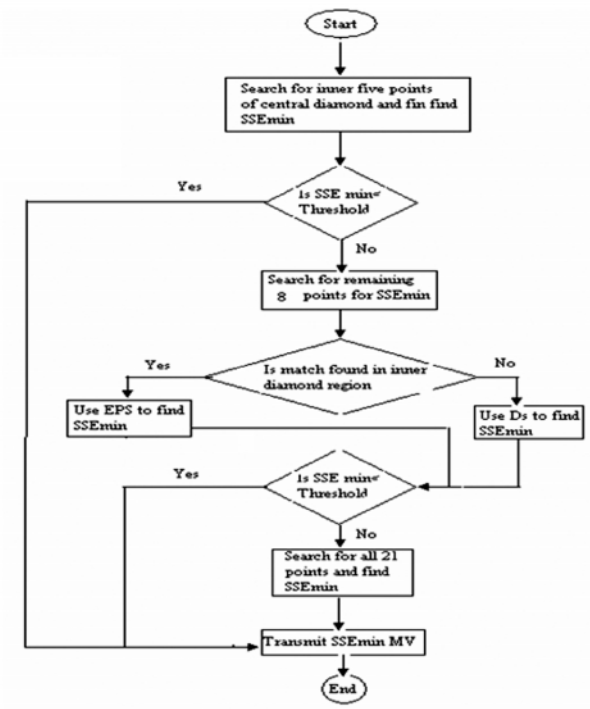

Fig 4: Flowchart of MSME Algorithm

3. SIMULATION AND EXPERIMENTAL RESULT

The MSME algorithm is simulated with threshold using the luminance component of fifteen CIF/QCIF sequence with variable frame numbers. Out of fifteen sequences, we have shown two video sequences which present different kinds of motion to test the algorithm behaviour under different conditions.

1. Miss America: Typical videoconference sequence Fig.5 with head and shoulder movement with fixed background.
2. Garden: Sequence Fig.6 consists mainly of stationary objects, but with a fast camera panning motion.

Fig 5: Miss America Sequence

Fig 6: Garden Sequence

The average peak signal to noise ratio PSNR of all encoded pictures is used as a measure of objective quality. The sum of square error (SSE) distortion function is used as the block distortion measure (BDM). We compared the MSME against four other block based motion estimation methods—FS, TSS, FSS, and DS using the following test criteria.

- Average peak signal to noise ratio (PSNR).
- Average number of search points per block (ASP).
- PSNR versus variable bitrate.

Table 3
Comparison of the Average psnr (db) Value of FS, MSME, DS, FSS AND TSS

Sequence	FS	MSME	DS	FSS	TSS
Miss America	41.866	41.863	41.78	41.618	41.587
Suzie	37.422	37.419	37.157	37.006	36.89
Akiyo	44.776	44.776	44.763	44.665	44.647
Carphone	33.654	33.652	33.106	32.971	32.99
Coastguard	33.804	33.799	33.688	33.59	33.173
Foreman	33.919	33.884	33.068	32.998	32.989
Mobile	26.332	26.308	26.302	26.304	26.296
Mthr & Daughter	42.185	42.181	42.083	42.022	41.807
News	37.923	37.913	37.652	37.571	37.451
Salesman	41.248	41.239	41.192	41.13	41.064
Claire	43.362	43.349	43.342	43.307	43.27
Grandma	43.673	43.647	43.645	43.632	43.632
Stefan	27.321	26.977	25.055	25.128	25.907
Bus	26.808	25.916	22.85	22.758	24.062
Flower	27.204	27.208	26.909	26.53	24.038

Tables 3, 4 and 5 summarize the PSNR value, average number of search points and average PSNR versus varying bitrate for algorithms like FS, MSME DS, FSS, and TSS. Simulation result shows considerable improvement in PSNR of sequences like suzie, akiyo, mother and daughter, and carphone. The PSNR value of this sequences are comparable to full search with less number of search points, this is because of fixed threshold. MSME for other sequences like coastguard, Stefan etc have inferior match compared to FS but is far better than TSS, FSS, and DS. The varying bitrate also do not affect the performance of MSME which makes it suitable for rate constrained ME.

Table 4
Comparison of the Average Number of Search Points of FS, MSME, DS, FSS and TSS

Sequence	FS	MSME	DS	FSS	TSS
Miss America.	204.28	9.66	13.69	16.756	23.268
Suzie	204.28	12.51	13.49	16.579	23.256
Akiyo	204.28	10.31	12.22	15.816	23.212
Carphone	204.28	17.31	14.71	17.131	23.322
Coastguard	204.28	16.44	14.16	17.03	23.286
Foreman	204.28	18.32	15.71	17.71	23.316
Mobile	204.28	14.55	12.33	15.869	23.216
Mthr & Daughter.	204.28	8.08	13.15	16.493	23.261
News	204.28	6.01	12.57	16.017	23.212
Salesman	204.28	6.41	12.29	15.858	23.214
Claire	204.28	8.78	12.42	15.926	23.231
Grandma	204.28	7.87	12.84	16.276	23.266
Stefan	214.52	22.34	17.11	18.733	24.141
Bus	214.52	23.5	20.08	20.388	24.312
Flower	214.52	16.07	16.7	19.172	24.157

Table 5
Comparison of the Average psnr (dB) Versus Varying Bitrates of FS, MSME, DS, FSS and TSS

		Bitrates - kbps QP-16				Bitrates - kbps QP-20			
Sequence	Algorithm	128k	256k	512k	1024k	128k	256k	512k	1024k
Akiyo	FS	38.164	41.01	42.861	44.776	35.965	37.801	40.992	42.813
	TSS	38.01	40.31	42.103	44.647	34.899	36.962	39.905	42.557
	FSS	38.55	41.01	42.661	44.665	35.25	37.21	40.655	42.675
	DS	38.82	41.21	42.673	44.763	35.338	37.41	40.74	42.786
	FPSME	38.162	41.22	42.861	44.776	35.562	37.555	40.844	42.808
Coast-guard	FS	27.901	29.671	31.886	33.804	25.871	26.998	29.97	31.457
	TSS	27.201	29.152	31.112	33.173	25.001	25.972	29.102	31.199
	FSS	27.325	29.537	31.444	33.59	25.151	26.337	29.281	31.259
	DS	27.331	29.535	31.447	33.688	25.431	26.442	29.482	31.288
	FPSME	27.652	29.651	31.881	33.799	25.551	26.481	29.681	31.431

4. CONCLUSION

Our proposed algorithm, MSME, has been presented in this paper for motion estimation. The proposed algorithm uses probabilistic approach to locate the true motion vector. Simulation results show that MSME achieves better estimate accuracy as compared to earlier proposed algorithms. Due to its efficient search pattern it track motion vector in vicinity of central point as well as at edge of the search window which makes it more applicable search algorithm for video with small and large motion. The early termination of algorithm with reasonable PSNR is possible using fixed threshold. The computations involved in this technique are considerably low as compared to FS, TSS, FSS, DS, and MSME with PSNR comparable to MSME and much better than other sub optimal algorithm. In addition proposed algorithm is more robust as compared to earlier algorithms because it is flexible enough to work well, for any search range and window size which will be useful in rate constrained environment. Even the performance of MSME is consistent for the image sequence that contains complex movements such as camera panning and zooming. The simulation result demonstrates that the proposed algorithm is very suitable for high quality video encoding.

REFERENCES

[1] B. Haskell, A. Puri, A.Netravali, "Digital Video: An Introduction to MPEG-2", *Kluwer Academic Publisher,* 2000.

[2] K.Jack, "Video Demystified: A Handbook for the Digital Engineer", *Elsevier Publisher,* Linacre House, Oxford, 2005.

[3] T.Koga,K.Iinuma,A.Hirano,Y.Iijima, and T.Ishiguro, "Motion Compensated Interframe Coding for Video Conferencing," *in Proc. National Telecommunications Conf.*, Nov. 29–Dec. 3, 1981, pp. G.5.3.1–G.5.3.5.

[4] W. Li and E Salari, "Successive Elimination Algorithm for Motion Estimation", *IEEE Transactions on Image Processing*, **4**, No. 1, pp. 105-107, Jan.1995.

[5] B.Zeng, and M.Liou, "A New Three Step Search Algorithm for Block Motion Estimation," *IEEE Trans. Circuits Syst. Video Technol.*, **4**, No.4, pp. 438–442, Aug. 1994.

[6] Lai-Man Po and Wing-Chung Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, **6**, No. 3, pp. 313-317, June 1996.

[7] Kuan-Tsang wang, "Efficient Four-Step Search'', *IEEE international symposium on Circuits and Systems,* 1998, ISCAS apos,'98, **4**, pp. 217-220, June 1998.

[8] Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath and Ashraf Ali Kassim, "A Novel Unrestricted Centre Biased Diamond Search Algorithm for Motion Estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, **8**, No. 4, pp. 369-377, Aug. 1998.

[9] M.Ghanbari, "The Cross Search Algorithm for Motion Estimation," *IEEE Trans.Commun.*, **38**, No.7, pp.950-953, July 1990.

[10] Jong-Nam Kim, Sung-Cheal Byun, Yong-Hoon Kim, and Byung-Ha Ahn, "Fast Full Search Motion Estimation Algorithm using Early Detection of Impossible Candidate Vectors", *IEEE Transaction on Signal Processing*, **50**, No. 9, pp.2355-2365, Sept. 2002.

[11] A.Chimienti, Claudia Ferraris, and Danilo Pau,"A Complexity-bounded Motion Estimation Algorithm", *IEEE Transactions on Image Processing*, **11**, No. 4, pp.387-392, Apr. 2002.

[12] R. Srinivasan and K.R. Rao, "Predictive Coding Based on Efficient Motion Estimation," *IEEE Trans. Circuits Syst.Video Technol.,* Vol.Comm-33, No.8, pp.888-896, Aug.1985.

[13] Ishfaq Ahmad, Weiguo Zheng, Jiancong Luo and Ming Liou "A Fast Adaptive Motion Estimation Algorithm" *IEEE Trans. Circuits Syst. Video Technol.,* **16**, No.3, pp.420-438, Mar.2006.

[14] Tourapis, O. C. Au, and M. L. Liou, "Fast Block-matching Motion Estimation using Advanced Predictive Diamond Zonal Search(APDZS),"ISO/IEC JTC1/SC29/WG11 MPEG2000/M5865, Noordwijkerhout, The Netherlands, Mar. 2000.

[15] Ce-Zhu, Wei-Song Qi, and Ser-W, "Predictive Fine Granularity Successive Elimimination for Fast Optimal Block- Matching Motion Estimation," *IEEE Transaction on Image Processing,* **14**, No.2, pp.213-221 Feb.2005.

[16] Shen-Chen Tai, Ying-Ru Chen, Yu-Hung Chen, "Small-diamond-based Search Algorithm for Fast Block Motion Estimation," *Elsevier Publisher, Signal Processing: Image Communication,* **22**, No. 10, pp. 877- 890, Nov. 2007.

[17] J.-J. Tsai and H.-M. Hang, "Modeling of Pattern-based Block Motion Estimation a nd its Application", *IEEE Trans. Circuits and Systems for Video Technology*, **19**, No. 1, pp.108-113, Jan 2009.

[18] L.-M. Po et al, "Novel Directional Gradient Descent Searches for Fast Block Motion Estimation", *IEEE Trans. Circuits and Systems for Video Technology*, **19**, No. 8, pp.1189-1195, Aug 2009.