A MULTI-STAGE MULTI-CANDIDATE ALGORITHM FOR MOTION ESTIMATION

Liao Ta-Chang[†], *See-May Phoong*[†], *and Yuan-Pei Lin*[‡]

[†]Dept. of EE & Grad. Inst. of Comm Engr., National Taiwan Univ.,Taipei, Taiwan, R. O. C.
[‡]Dept. Elec. and Control Engr., National Chiao Tung Univ., Hsinchu, Taiwan, R. O. C.

ABSTRACT

Motion compensation using the optimal full search algorithm is often too computational heavy for real time implementation. Many suboptimal fast search algorithms have been proposed. In particular, Liu and Zaccarin proposed the Alternating Subsampling Search Algorithm (ASSA). The ASSA reduces the computation by subsampling the pixels instead of limiting the search locations. It was shown that ASSA has nearly the same MSE performance as the full search but its complexity is only 1/4 of the full search. In this paper, we generalize the idea to the multi-stage case. Simulation results show that the proposed algorithm has a comparable performance to the ASSA but it has a much lower computational cost.

1. INTRODUCTION

Motion estimation plays an important role in the coding of video signals. Both block matching algorithm and pixel recursive algorithm have been developed. Block matching algorithms have been widely used in video coding standards due to their simplicity for both software and hardware implementation. The optimal block matching algorithm that yields the best motion vector (according to some predetermined criteria) is the full search (FS) algorithm. It explores all the possible motion displacements over a defined search window. However, the computational complexity of the FS algorithm is often too high for many applications. A number of fast motion estimation algorithms that yield suboptimal (or near optimal) motion vector at a much lower cost have been developed. Most of the proposed fast algorithms reduce the complexity by limiting the number of search locations. Some of these examples include the three-step search (3SS) algorithm [1], the new three-step search (N3SS) [2], the four-step search (4SS) algorithm [3], the cross search (CS) algorithm [4], the block-based gradient descent search (BBGDS) [5], the unrestricted center-biased diamond search (UCBDS) [6], etc. The performance of these algorithms rely on the monotonically increasing property of the distortion function when one moves away from the optimal solution. For some image sequences, these algorithms can be trapped into local minimum points, and thus, can produce a much higher matching error compared with the FS algorithm.

Liu and Zaccarin proposed an Alternating Subsampling Search Algorithm (ASSA) [7] which reduces the number of pixels used in each block distortion measure instead of reducing the number of checking points. This algorithm uses alternating subsampling patterns in calculating different location's distortion. Experimental results [7] show that the MSE performance of ASSA with downsampling factor of 4 is very close to that of FS. However it can only achieve computation reduction of 4 times compare to the FS. This is in general still too costly for most applications. Though ASSA with higher downsampling factor can have more computational saving, its performance degrades substantially. Therefore, to further reduce the complexity, the authors limit the number of search location by downsampling the motion field. In this paper, we generalize the algorithm in [7] to multiple stages. The original block of image is partitioned into multiple groups. In the first stage, one of the groups of pixels are used for the evaluation of distortion at all search locations, and the best few candidate search locations are selected. In each of latter stages, we continue to eliminate a number of candidates by evaluating the distortion at one new group of pixels, until the last stage when the best motion vector is obtained by using the all the groups. By retaining more candidates, we can afford to have a larger downsampling factor at earlier stages and hence can achieve more computational saving than the ASSA. Our experiments show that the proposed algorithm compares favorably with the 3SS, 4SS, ASSA, BBGDS and UCBDS.

2. THE PROPOSED MULTI-STAGE MULTI-CANDIDATE ALGORITHM

The pixel value at the coordinate (x_0, x_1) of the frame n will be represented as $f_n(x_0, x_1)$. Block-matching motion estimation is done between current frame n and previous frame $n-1$. Each frame is divided into nonoverlapped N by N blocks. Each block will be referred to by the coordi-

THIS WORK WAS SUPPORTED BY NSC 89-2213-E-002-122 AND 89-2213-E-009-118, TAIWAN, R.O.C.

0-7803-7041-4/01/$10.00 ©2001 IEEE

nate (k_0, k_1) of its top left corner. The matching criterion used in this paper is the mean absolute difference (MAD). The MAD is chosen because it does not need any multiplications and gives similar solution as the mean square error (MSE). The MAD between block (k_0, k_1) of current frame and block $(k_0 + v_0, k_1 + v_1)$ of previous frame is given by:

$$MAD_{(v_0,v_1)}(k_0,k_1) = \sum_{i_0=0}^{N-1}\sum_{i_1=0}^{N-1} |f_n(k_0+i_0,k_1+i_1) - f_{n-1}(k_0+v_0+i_0,k_1+v_1+i_1)|.$$

The range of the search window is $-p \leq v_0 \leq p$ and $-p \leq v_1 \leq p$. From the search window, the vector (v_0, v_1) that gives the smallest MAD is chosen as the motion vector. For each block, there are a total of $(2p+1)^2$ search locations. For each search location, the computation of the MAD takes $2N^2$ additions. Therefore the FS algorithm needs $2(2p+1)^2N^2$ additions for the computation of each motion vector. This complexity is often too high for many applications. From the expression $2(2p+1)^2N^2$, we see that complexity can be lowered by limiting the number of search locations or reducing the number of pixels in evaluation of the MAD. Our proposed multi-stage multi-candidate (MSMC) algorithm belongs to the latter, and the algorithm is described below.

The MSMC Algorithm

The MSMC algorithm matches all the locations inside the search window as the FS algorithm. However it uses a subset of pixels to evaluate the MAD. To explain the method, we will use a 4-stage MSMC algorithm as an example. It is not difficult to generalize the method to more stages. The block size is assumed to be 16 by 16. The pixels in each block are partitioned into 4 groups as shown in Fig. 1. Note that the number of groups is equal to the number of stages. The algorithm is as follows:

Stage 1. Evaluate the MAD using the 16 pixels labeled as 1. The double summation in the MAD expression has only 16 terms. After evaluating the MAD for all the locations in the search window, we keep only the K_1 candidate motion vectors that have the smallest K_1 MAD.

Stage 2. For the K_1 motion vectors obtained from Stage 1, evaluate the MAD using the 64 pixels labeled as 1 and 2. Note that in this case, there are only 48 new terms in the MAD expression due to those pixels in Group 2. We keep only the K_2 candidate motion vectors that have the smallest K_2 MAD.

Stage 3. For the K_2 motion vectors, evaluate the MAD using the 128 pixels labeled as 1, 2 and 3. There are 64 new terms. We keep only the K_3 candidate motion vectors that have the smallest K_3 MAD.

1	4	2	4	1	4	2	4	1	4	2	4	1	4	2	4
4	3	4	3	4	3	4	3	4	3	4	3	4	3	4	3
2	4	2	4	2	4	2	4	2	4	2	4	2	4	2	4
4	3	4	3	4	3	4	3	4	3	4	3	4	3	4	3
2	4	1	4	2	4	1	4	2	4	1	4	2	4	1	4
4	3	4	3	4	3	4	3	4	3	4	3	4	3	4	3
2	4	2	4	2	4	2	4	2	4	2	4	2	4	2	4
4	3	4	3	4	3	4	3	4	3	4	3	4	3	4	3
1	4	2	4	1	4	2	4	1	4	2	4	1	4	2	4
4	3	4	3	4	3	4	3	4	3	4	3	4	3	4	3
2	4	2	4	2	4	2	4	2	4	2	4	2	4	2	4
4	3	4	3	4	3	4	3	4	3	4	3	4	3	4	3
2	4	1	4	2	4	1	4	2	4	1	4	2	4	1	4
4	3	4	3	4	3	4	3	4	3	4	3	4	3	4	3
2	4	2	4	2	4	2	4	2	4	2	4	2	4	2	4
4	3	4	3	4	3	4	3	4	3	4	3	4	3	4	3

Figure 1: Partition of an 16 by 16 block into 4 groups. Groups 1, 2, 3, 4 are respectively the new pixels used in the evaluation of the MAD.

Stage 4 For the K_3 motion vectors, evaluate the MAD using all the pixels in the block. There are 128 new terms. We keep only the best motion vector that have the smallest MAD.

Computational Complexity

Let the range of the search window be $-p \leq v_0 \leq p$ and $-p \leq v_1 \leq p$ and block size of 16 by 16. The FS algorithm takes $512*(2p+1)^2$ additions to find the best motion vector for each block. For the above 4-stage MSMC algorithm, the number of additions needed to compute the motion vector for each block is given by:

$$2(16*(2p+1)^2 + 48*K_1 + 64*K_2 + 128*K_3). \quad (1)$$

When $(2p+1)^2$ is much larger than K_i, the MSMC algorithm has a considerable saving over the FS algorithm. However when K_i are too small, the mean square error (MSE) performance of the MSMC algorithm will be degraded. There is a tradeoff between the computational complexity and performance. After carrying out the simulations on a number of test sequences, we found that the choice of $K_1 = 8$, $K_2 = 4$ and $K_3 = 2$ is a good compromise. For this choice of K_i, we compare its complexity with the FS, ASSA, 3SS algorithms for the cases of $p = 7$ and 15. The complexity of FS is normalized to 1. For the ASSA algorithm [7], it has a downsampling factor of 4. The results are shown in Table 1. As we can see, for $p = 7$, the MSMC has a considerable saving over the other 2 methods. For $p = 15$, the 3SS has the lowest complexity.

In (1), we have not included the computation needed for selecting the candidate. To check each search location if it is a candidate, we need to do at least one and at most K_i comparisons. This overhead can be significant when K_i is

	FS	ASSA	3SS	MSMC
p = 7	1	0.264	0.111	0.078
p =15	1	0.253	0.034	0.066

Table 1: Comparison of Complexity.

large. To reduce this overhead, we exploit the center-biased motion vector distribution characteristics of the real world sequences [2]. The search of the motion vector starts at the origin of the search window and then moves outwards with a spiral scanning path, as shown in Fig. 2. By doing so, the overhead can be reduced to one comparison for most search locations.

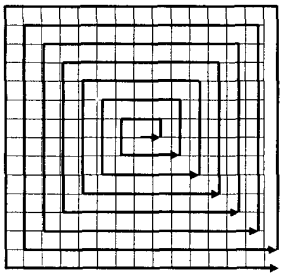

Figure 2: Spiral scanning path.

3. SIMULATION RESULTS

We compare the performance of the FS, ASSA, 3SS, 4SS, BBGDS, UCBDS and the proposed MSMC algorithms. Only the luminance component of the video sequences is used for the calculation of motion vector. The video sequences used are the first 150 frames of the sequences *Suzie, Trevor, Foreman,* and *Carphone* (frame size = 176 by 144) and the first 100 frames of the sequences *Tennis, Football,* and *Flower* (frame size = 352 by 240). Block size is 16 by 16. The 4-stage MSMC algorithm is used and $K_1 = 8$, $K_2 = 4$, and $K_3 = 2$. The parameter $p = 7$. The experiment is done for two different frame rates of 30 fps and 10 fps.

The MSE performances of all the methods are shown in Tables 2 and 3 for the two different frame rates. From the average values in the last column of the tables, we see that the ASSA has nearly the same MSE performance as the FS. The MSE performance of MSMC algorithm is slightly worse than that of ASSA but is much better than the MSE performances of 3SS, 4SS, BBGDS and UCBDS. We also compare the execution time taken for these methods. We normalize the time taken for the FS algorithm as 100%. The results are shown in Tables 4 and 5 respectively for the frame rate of 30 fps and 10 fps. From the average values, we see that the BBGDS is the fastest and the MSMC is only slightly slower. However the MSE of BBGDS is substantially larger than the MSMC. In summary, the ASSA can achieve an MSE performance that is very close to the FS but it is only 3.3 times faster than the FS. The MSMC has a comparable MSE performance to ASSA but it is 12 times faster the FS.

4. CONCLUSIONS

In this paper, we have proposed a new fast and efficient algorithm for motion compensation. The proposed MSMC algorithm compares favorably with the 3SS, 4SS, ASSA, BBGDS and UCBDS. Moreover, MSMC algorithm can be combined with the predictive search algorithm and can further reduce the computational load without sacrificing much MSE performance [8].

5. REFERENCES

[1] T. Koga; K. Ilinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe image coding," in Proc. NTC 81, NOV./Dec. 1981, pp. C9.6.1-C9.6.5.

[2] Li, R.; Zeng, B; Liou, M.L. , "A new three-step search algorithm for block motion estimation ," IEEE Trans. on CSVT, Aug. 1994, pp. 438 -442.

[3] Po, L.; Ma, W. "A novel four-step search algorithm for fast block motion estimation," IEEE Trans. on CSVT, June 1996, pp. 313 -317.

[4] Ghanbari, M. ,"The cross-search algorithm for motion estimation ," IEEE Trans. on Communications, July 1990, pp. 950 - 953.

[5] Liu, L.; Feig, E. , "A block-based gradient descent search algorithm for block motion estimation in video coding ," IEEE Trans. on CSVT, Aug. 1996, pp. 419 -422.

[6] Tham, J.; Ranganath, S.; Ranganath, M.; Kassim, A.A. , "A novel unrestricted center-biased diamond search algorithm for block motion estimation," IEEE Transactions on CSVT, Aug. 1998, pp. 369 -377.

[7] Liu, B.; Zaccarin, A., " New fast algorithms for the estimation of block motion vectors ," IEEE Trans. on CSVT, April 1993 , pp. 148 -157.

[8] Liao, T., "New motion estimation method and improved video codec," *Master Thesis,* National Taiwan Univ., June 2000.

Table 2: Comparison of MSE performance for frame rate = 30 fps.

MSE	Suzie	Trevor	Foreman	Carphone	Tennis	Football	Flower	Average
FS	12.99	27.23	26.24	17.49	100.27	260.90	185.64	136.19
ASSA	13.03	27.28	26.26	17.49	101.20	261.87	186.33	136.81
3SS	13.55	27.79	28.97	18.37	122.46	285.61	218.51	155.52
4SS	13.50	27.97	27.94	18.32	113.42	290.66	201.73	150.51
BBGDS	13.26	28.09	26.95	17.77	118.93	306.18	197.28	154.34
DS	13.28	27.88	27.50	17.91	110.05	293.48	192.74	148.15
MSMC	13.09	27.71	26.58	18.08	104.73	271.07	190.41	140.91

Table 3: Comparison of MSE performance for frame rate = 10 fps.

MSE	Suzie	Trevor	Foreman	Carphone	Tennis	Football	Flower	Average
FS	41.37	85.38	59.07	32.42	254.67	587.33	706.61	384.31
ASSA	41.44	85.52	59.18	32.36	256.85	590.04	712.49	386.89
3SS	44.53	87.74	67.62	33.98	282.80	626.87	861.75	438.47
4SS	45.86	88.41	65.87	33.98	303.03	648.46	1040.59	491.03
BBGDS	46.19	89.74	64.86	33.33	342.52	686.88	1274.62	565.30
DS	44.91	88.61	64.21	33.60	306.96	655.81	1113.73	510.93
MSMC	42.14	87.70	60.34	33.67	266.49	612.32	746.59	402.99

Table 4: Comparison of complexity for frame rate = 30 fps.

%	Suzie	Trevor	Foreman	Carphone	Tennis	Football	Flower	Average
FS	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00%
ASSA	30.78	30.68	30.87	30.62	30.68	30.71	30.73	30.72%
3SS	11.83	11.85	11.87	11.71	11.72	11.82	11.81	11.80%
4SS	8.24	8.95	8.50	9.30	8.09	9.61	8.73	8.77%
BBGDS	5.14	5.63	5.33	6.32	5.54	7.21	5.78	5.85%
DS	6.68	7.27	7.07	7.97	6.52	8.86	7.34	7.39%
MSMC	8.27	8.23	8.25	8.25	8.24	8.20	8.25	8.24%

Table 5: Comparison of complexity for frame rate = 10 fps.

%	Suzie	Trevor	Foreman	Carphone	Tennis	Football	Flower	Average
FS	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00%
ASSA	30.68	30.73	30.80	30.59	30.62	30.66	30.70	30.68%
3SS	11.84	12.00	11.91	12.04	11.73	11.86	11.90	11.90%
4SS	8.79	10.02	9.37	9.63	10.06	10.91	9.73	9.79%
BBGDS	6.06	8.06	6.44	7.51	9.52	9.72	7.77	7.87%
DS	7.51	9.10	7.86	8.89	9.65	11.02	8.94	9.00%
MSMC	8.25	8.29	8.28	8.25	8.22	8.24	8.26	8.26%