Proceedings of the Fifth International Conference on Computer Science and Informatics, March 2000, Atlantic City, New Jersey

A Center-Biased Hybrid Search (CBHS) Method for Block Motion Estimation

Sung-Chul Shin[1], Hyunki Baik[1], Myong-Soon Park[1] and Dong Sam Ha[2]
[1]Computer Systems Laboratory, Department of Computer Science and Engineering Korea University
{scshin, gibson, myongsp}@cslab5.korea.ac.kr
[2]Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University
ha@vt.edu

Abstract

This paper proposes a new method for block motion estimation based on center-biased hybrid search (CBHS) that employ a hybrid of a compact X-shaped search and a diamond search. It has only 9 search points per block for the best case and, on average, 10.2 search points. CBHS is about 22.2 times faster than the full-search algorithm. This paper compares the popular suboptimal block-matching technique with CBHS in which both the processing speed and the accuracy of motion compensation are tested over widely used H.263 test video sequences.

1. Introduction

Motion estimation is the most time consuming process for a video encoder. so a fast motion estimation algorithm is crucial for real-time applications. Block-based motion estimations are widely used for video compression such as MPEG[2][3] and H.263 standard[1] owing to its effectiveness and simple implementation. However, the high computational complexity of the full-search algorithm has motivated a host of suboptimal but faster search strategies. A good example is the three-step search(TSS) algorithm[5]. However, the uniformly spaced search pattern in TSS does not match well to real-world video sequences in which the motion vector distribution is mostly nonuniformly biased toward the zero vector. Such an observation inspired the four-step search(4SS) algorithm[7] which has a center-biased search pattern. 4SS algorithm exploit the center-biased motion vector distribution characteristic by utilizing a nine-point search pattern on a 5 × 5 grid. As a result of starting with a smaller search grid pattern, 4SS is, on the average, faster than TSS and yields a better motion estimation. Recently, Tham et al. proposed a novel method called unrestricted center-biased diamond search(UCBDS) algorithm[9]. The average number of search points is reduced to 13.7 for UCBDS when compared with 17.4 points for 4SS, while the performance of UCBDS is better than 4SS in motion compensation error.

In this paper, we propose a center-biased hybrid search (CBHS) algorithm which is a hybrid of a compact X-shaped search and a diamond search. CBHS is much faster than the previous techniques such as 4SS and UCBDS. The performance of the proposed CBHS algorithm is similar to that of UCBDS in motion estimation error. Our method is based on the observation that about 90%, of motion vectors are concentrated in the region of (0,0), (0,-1), (0,+1), (+1,0), and (-1,0) as shown Figure.1. The CBHS aims to reduce the number of search points in the region to speed up.

	-3	-2	-1	0	+1	+2	+3
-3	0.01	0.01	0.04	0.03	0	0.01	0
-2	0.01	0	0.05	0.07	0.02	0	0.01
-1	0	0.06	0.63	0.39	0.15	0.04	0.02
0	0.06	0.18	5.04	88.71	2.00	0.05	0.01
+1	0.03	0.09	0.74	0.58	0.21	0.01	0
+2	0.01	0	0.04	0.05	0.04	0.01	0
+3	0.01	0	0	0.02	0	0.01	0

Figure 1. Center-biased motion vector distribution characteristics of Miss America sequence : 96.72%

2. Proposed Center-Biased Hybrid Search Algorithm

In this section, we explain our center biased hybrid search(CBHS) algorithm, this consist of two phase for the purpose of utilizing a center-biased characteristic of video sequences. First phase is compact X-shaped search pattern for the center's 9 pixels, and second phase is diamond search pattern which is very suitable pattern by its compact structure.

2.1 Algorithm of CBHS

A frame is divided into blocks of size N ✕ N pixels, and a block-based motion estimation algorithm is applied to individual blocks. For example, the block size of 16 ✕ 16 is used for MPEG-1[2], MPEG-2[3], H.261[4], and H.263[1]. The search for a match in a block is usually performed over a 15 ✕ 15 search area for low and very low bit-rate video applications. Hence, it requires to 225 search per block for the full search. This is computationally too expensive. Hence, the objective of our algorithm like any suboptimal center-biased search algorithm is to choose an appropriate subset of the 225 points.

As a metric for the estimation, a block distortion measure (BDM) is used as the objective function for each block's evaluation. For simplicity, we employ the sum of absolute difference(SAD) in our algorithm CBHS. Fig. 2.(a) depicts the basic X-shaped search point configuration used in our algorithm CBHS. The X-pattern is placed at (0,0), which is the center of the search window. It consists of five candidate search points.

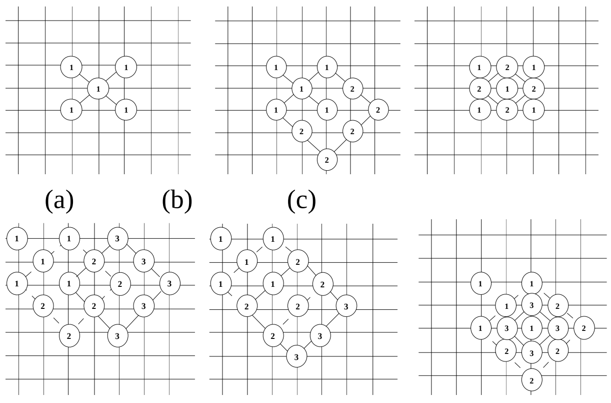


Figure 2. CBHS search pattern. (a) Original CBHS search-point configuration. (b) Next step along a X's vertex. (c) Final step in X. (d) Next step along a diamond's vertex (e) Next step along a diamond's face (f) Final step in UCBDS

Unrestricted search path strategy using CBHS algorithm can be summarized as follows.

• **Starting** : The original X pattern [Fig. 2.(a)] is placed at (0,0), the center of the search window. The SAD is evaluated for each of the five candidate search points. If the minium SAD point is found to be at the center(c,c) of the X, proceed to **X End**. Otherwise, proceed to **Searching**.

• **Searching** : If the minimum SAD point in the starting step is located at one of the outer, then proceed to **Outer Search**. Else, if the minimum SAD point in the previous search step is located at one of the four vertices, then proceed to **Vertex Search**. Else if it is located at one of the four possible faces of the previous diamond, then proceed to **Face Search**.

- **Outer Search** : The diamond pattern of Fig. 2.(b) is used with the center of the new diamond coinciding with lowest SAD point. Five new candidate search points are evaluated.
- **Vertex Search** : The diamond pattern of Fig. 2.(d) is used with the center of the new diamond coinciding with the lowest SAD point. Five new candidate search points are evaluated.
- **Face Search** : The diamond pattern of Fig. 2.(e) is used with the center of the new diamond coinciding with the lowest SAD point. Three new candiate search points are evaluated.

Note that any candidate point that extends beyond the search window is ignored, The minimun SAD is again identified. If the minimun SAD is found at (c,c), then proceed to **Diamond End**; otherwise, proceed to **Searching** to continue the next search step.

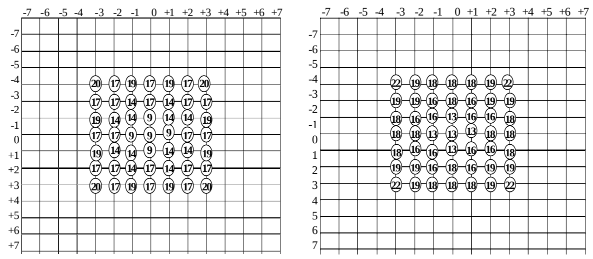
• **Ending** :

- **X End** : The shrunk X pattern of Fig. 2.(c) is used with the same center (c,c). the final four internal points of the X pattern are evaluated. The candidate point that give the lowest SAD is chosen as the estimated motion vector(m_x ,m_y). The current block's search process is completed. Proceed to Starting for the next block.
- **Diamond End** : The shrumk diamond pattern of Fig. 2.(f) is used with the same center(c,c). The final four internal points of the previous diamond are evaluated.

2.2 Theoretical Analysis of CBHS

This subsection aims to investigate theoretically why CBHS is truly center biased, and how speed improvement can be obtained over search algorithm. In particular, we are comparing CBHS with UCBDS.

Our main argument in this analysis is based heavily on the observed center-biased motion vector distributions. To begin, we first analyze the minimum number of search points N_s within a region of ±3 pixels about the motion vector (0,0). This is depicted in Fig. 3. It can easily be observed that, within this region, CBHS gives lower value of N_s as compared UCBDS. To get a better picture of the gain in N_s ,we subtract the corresponding candidate points of CBHS from UCBDS over this region, By doing so, we can obtain a saving of as high as block matches per block.



(a) CBHS (b)UCBDS

Figure 3. Minimum possible number of search points for search points for each motion vector location using

To further quantify this gain in N_s for block estimation we define the following probabilities of occurrence:

- P_0: probability of stationary block
- P_1: probability of quasi-stationary blocks within ±1, but excluding (0,0)
- P_2: probability of quasi-stationary blocks within ±2, but excluding the ±1 region at the center
- P_3: probability of quasi-stationary blocks within ±3, but excluding the ±2 region at the center
- P’: probability of blocks in the region where $4 \le |m_x|, |m_y| \le W$

By taking the average of the difference in N_s between UCBDS and CBHS over each of the above regions, the statistical average gain of CBHS over UCBDS can be represented as

Gain in $N_g = 4P_0 + (\frac{24}{8})P_1 + (\frac{28}{16})P_2 + (\frac{20}{24})P_3 + nP'$, (1)

where $P' = 1 - (P_0 + P_1 + P_2 + P_3)$, and n is some positive number. Suppose further that we assume a uniform probability distribution over the ±3 region at the center, and that no motion vector lies outside of this region. Then from (1), we will have a uniformly distributed average gain

uniform gain in $N_s = 0.25 \times (4 + 3 + 1.75 + 0.83) = 2.40$ search points per block

However, observations from most real-world sequences show very peaked probabilities around P_0 and P_1, as depicted Fig. 1. This means that an average gain of more than 2.40 search points per block can be expected. More experiment results later will justify this statement.

3. Experimental Results

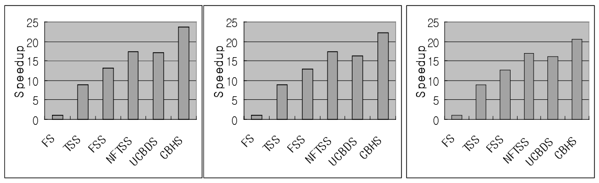
This section aims to investigate the actual experimental performance of CBHS. we will present three representative sequences. First, we use 100 frames of the “Miss America”, which is a typical videoconferencing scene with limited object motion and a stationary background. Second, we chose 100 frames of the “Trevor”. Third, we selected 100 frames of the “Suzie” sequence.

We compared the CBHS against five other block-based motion estimation methods using the following four test criteria

1) **Average number of search points N_s per block** : This provides an equivalent measure of the actual CPU run time.
2) **Speedup** : This indicate how many times faster than FS.
3) **Probability of finding true motion vector per block** : This gives the likelihood of the suboptimal predicted block motion vectors to be the same as those found using the optimum FS;
4) **Average SAD per block :** This shows the magnitude of distortion per block .

	Using “Miss America” Sequence				Using “Trevor” Sequence				Using “Suzie” Sequence			
	Avg. Search points	Speedup	Avg. prob.	Avg.SAD	Avg. Search points	Speedup	Avg. prob.	Avg.SAD	Avg. Search points	Speedup	Avg. prob.	Avg.SAD
FS	225.0	1.0	100.00	493.78	225.0	1.0	100.00	1264.03	225.0	1.0	100.00	968.00
TSS	25.0	9.0	96.87	497.85	25.0	9.0	90.98	1346.94	25.0	9.0	89.60	1012.24
4SS	17.0	13.2	97.07	494.87	17.5	12.9	96.86	1273.61	17.7	12.7	94.01	975.51
NFTSS	13.0	17.3	87.95	518.79	13.0	17.3	81.69	1380.57	13.2	17.0	68.68	1034.14
UCBDS	13.1	17.2	98.10	493.95	13.8	16.3	98.42	1269.15	14.1	16.0	96.70	971.67
CBHS	9.5	23.7	97.82	494.04	10.1	22.3	97.22	1272.44	10.9	20.6	94.95	973.50

TABLE 1. Performance Comparisons of FS, TSS, 4SS, NFTSS, UCBDS, and CBHS



(a) Miss America (b)Trevor (c)Suzie

Figure 4. Performance comparisons of FS, TSS, FSS, NFTSS, UCBDS and CBHS over Speedup criteria

Table 1. summarizes the experimental performance of each search technique over the four test criteria using three representative sequences and Fig. 4. illustrate the speedup factor. The first column tabulates the average number of search points, The second column tabulate speedup factor with respect to FS is reported. This factor shows that how many times faster than FS. The speedup with CBHS > NFTSS > UCBDS > 4SS > TSS

> FS; such observations were true for all of the test sequences we used. This shows that CBHS is generally more efficient than the other schemes in the low bit-rate sequence, such as video conference used for H.263.

A question now is: How much does CBHS trade off block distortion for higher search speed? From column 4 of Table 1., it can be observed that CBHS actually performs very competitively in term of low block distortion, even though it has the lowest average number of search points. It can be seen that CBHS has little worse SAD performance and Probability of finding true motion vector compared to the UCBDS and FS technique, However, the speed improvements are quite substantial, and thus justify its use over the techniques. A possible explanation for the good performance of CBHS is that it has a very compact search configuration which speeds up the search, while reduce the risk of being trapped into a local minimum. From the third columns of Table 1., it can be concluded that CBHS generally gives a higher average probability of finding the true motion vectors, when compared with the other suboptimal search techniques. Third column shows the percentage of the Probability of finding true motion vector per block.

4. Conclusions

In this paper, we proposed a Center-Biased Hybrid Search (CBHS) algorithm for fast suboptimal block-based motion estimation. Motivated by the center-biased motion vector distribution characteristics of real-world video sequences. CBHS was consist of two phase, first phase was designed with very compact X-shaped search point configuration, and second phase was designed with the compact diamond search point configuration, We then explained the algorithm development of CBHS, and performed a theoretical analysis of its efficiency. Experimental results were presented to show that CBHS is more efficient and robust as compared to some other popular block-matching algorithm such as FS, TSS, 4SS, NFTSS and UCBDS. In short, CBHS has the following advantages over the other search algorithm.

• Efficiency : CBHS is highly center biased, and it has a very compact X-shaped search point and diamond search points configuration. This allows a minimum of only 9 candidate search points per block, and a speed improvement of up to 34% over the UCBDS and 22.2 times faster than FS.

• Robustness : As CBHS is unrestricted and dose not have a predetermined number of search steps, it is flexible enough to work well for any search range/window size.

Reference

[1] ITU Telecommunication Standardization Sector LBC-95, Study Group 15, Working Party 15/1, Expert's Group on Very Low Bitrate Visual Telephony, from Digital VideoCoding Group, Telenor Research and Development.

[2] ISO/IEC CD 11172-2 (MPEG-1 Video), "Information technology - Coding of moving pictures and associated audio for digital storage media at up to about 1.5Mbits/s: Video," 1993.

[3] ISO/IEC CD 13818-2 - ITU-T H.262(MPEG-2 Video), "Information technology - Generic coding of moving pictures and associated audio information: Video", 1995.

[4] CCITT SGXV, "Description of reference model 8(RM8)," Document 525, Working Party XV/4, Specialists Group on Coding for Visual Telephony, June 1989.

[5] T.Koga, K. llinuma, A. Hirano, Y.Iijima, and T.Ishiguro, "Motion compensated inter frame coding for video conferencing," in Proc. NTC 81, New Orleans, LA, Nov./Dec. 1981, pp. C9.6.1 - C9.6.5

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

[7] 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. 6, pp. 313-317, June 1996.

[8] Jong-Nam Kim and Tae-Sun Choi, "A fast motion estimation for software based real-time video coding", IEEE Transaction on Consumer Electronics, vol. 45, No. 2, pp. 417-425, May 1999.

[9] Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath, and Ashraf Ali Kassim, "A novel unrestricted center-biased diamond search algorithm for block motion estimation", IEEE Trans. Circuits Syst. Video Technol., Vol. 8, No. 4, pp. 369-377, August 1998.