Fast Motion Estimation using Circular Zonal Search

Alexis M. Tourapis*, Oscar C. Au**, Ming L. Liou***

Department of Electrical and Electronic Engineering
The Hong Kong University of Science and Technology,
Clear Water Bay, Kowloon, Hong Kong.

ABSTRACT

Block based motion estimation is widely used for exploiting temporal correlation within an image. Still the full search algorithm, which is considered to be the optimal, is computational intensive. In this paper a new fast motion estimation method for video coding, is presented. It will be shown that the new algorithm is not only much faster than traditional algorithms, but in some cases can achieve much better visual quality, even from the "optimal" full search algorithm.

Keywords: fast, motion, circular, zonal, search, block matching, video coding

1. INTRODUCTION

An important key in achieving high compression ratios in video coding is to efficiently exploit the temporal and spatial redundancy that exists within video data. Since variations between adjacent frames are mostly due to the movement of the camera or objects, motion estimation and compensation were used in the majority of the current video coding methods. Due to its simplicity, the technique of block matching motion estimation is the one mostly used. The current frame is divided into square blocks of pixels and then for each one of the blocks, a previous frame is searched in order to find a block that is the closest to it, according to a predetermined criterion. This block is then used as a predictor for the current one and the displacement between the two blocks is used to define the motion vector associated with the current block.

Due to its simplicity and since it gives similar performance as the mean square error (MSE), the distortion measure most commonly used is mean absolute error (MAE or MAD). The MAD for a block A of size $N \times N$ located at (x, y) inside the current block, compared to a block B located at a displacement of (v_x, v_y) relative to A in the previous one is given as:

$$MAD(v_x, v_y) = \frac{1}{N^2} \sum_{m,n=0}^{N-1} |I_t(x+m, y+n) - I_{t-i}(x+v_x+m, y+v_y+n)|, \quad (1)$$

where I_t is the current frame and I_{t-i} is a previous frame.

If a maximum displacement of w pixels/frame is allowed, then we will have $(2w+1)^2$ locations to search for the best match of the current block. The algorithm which examines all these locations is called the brute force exhaustive search (or full search (FS)) and it can be seen that it is very computational intensive. For a frame of size $P \times Q$ and a frame rate of T fps, the amount of computation is:

$$T \cdot \left(\frac{P}{N} \cdot \frac{Q}{N} \right) \cdot (2w+1)^2 (2N^2 - 1) \approx 8TPQW^2 \approx 1.09 \times 10^{10}, \quad (2)$$

for $T=30$, $P=288$, $Q=360$ and $W=21$. On the average it can use up to 80% of the computational power of the encoder. Even though currently there are some hardware devices and systems which use this approach for video coding, they are still mainly used for video production, and not for other real time applications, such as video conferencing, or even software implementations. In addition it has been shown [3] that full search is not exactly the optimal solution, especially for lower bit rates, since it does not take into consideration other factors that could affect the performance of the video coding, such as

* Correspondence: Email: alexis@ee.ust.hk;

** Correspondence: Email: eeau@ee.ust.hk; WWW: <http://www.ee.ust.hk/~eeau>; Telephone: +852 2358-7053

*** Correspondence: Email: eeliou@ee.ust.hk; WWW: <http://www.ee.ust.hk/~eeliou>

the rate requirements of the motion vectors. Many algorithms have been developed to overcome the computational cost of FS. But still, even though these algorithms make the search faster, they usually incur a significant loss in visual quality. Some examples include 3-step search [1], New 3-step search [4], 2-D Log search [2] etc. Most of these fast algorithms if not all, do not take into consideration the bits required to encode the motion vectors, and thus are not exactly the best alternatives. Other approaches [3] that take this factor in consideration, can be said that are even more complex than Full Search.

An additional problem of these algorithms, including FS, is that the estimated motion vectors are rather random since the motion vectors of adjacent blocks are not being considered when performing the search, and thus the motion vectors could have a rather chaotic pattern. This can create problems when transmitting in noisy environments, since it is rather hard to predict and make the necessary corrections.

In [5] we have introduced a zonal fast-search with modest speed up gain versus FS and good visual quality. In this paper, we propose a new algorithm called the Circular Zonal Search algorithm (CZS), which improves [5], and that is not only much faster than most traditional motion estimation algorithms, but its quality could also reach and even in some cases outperform even the "optimal" FS. Also due to the structure of the algorithm, the motion vectors that are created using CZS tend to be more regular than those of FS.

2. CIRCULAR ZONAL SEARCH (CZS)

In many cases, especially in videoconferencing sequences, the center of the search area is most likely to be optimal due to the center-biased property [4]. The remaining search points have decreasing likelihood to be optimal, as they move farther away from the center. In addition, a motion vector of a block will probably have a value close, if not equal to the motion vector of another adjacent block. Since motion vectors are differentially encoded in most of the common standards like MPEG1/2 and ITU-T H261/3, the closer the motion vectors are for two adjacent blocks, the fewer bits will be needed to code them. This allows us to allocate more bits to encode the residue signal and thus improving the overall quality.

Fig. 1. Definition of circular zones in a search window

In the proposed circular zonal search algorithm, we try to benefit from the center-biased property of the video by constructing several circular zones around a center (Fig. 1). The zones can be constructed using the following formula:

$$\text{round}(\sqrt{MV_x^2 + MV_y^2}) = r - 1, \quad (3)$$

where MV_x, MV_y , are the distance of the current block to be examined from the center, and r is the corresponding zone. We can consider all blocks corresponding to the same zone together, because as it can be seen from the formula they are at approximately the same distance from the center, and are thus equiprobable.

In addition, instead of selecting the optimal block according to the search criterion in Equation 1, we select one close enough to the optimal, by using a thresholding criterion. When the encoder finds a block that matches a predefined threshold inside a search zone, the search stops without having to examine any of the other zones. This method always benefits the center and all the locations that are closer to the center. In many cases the optimal block might be too far from the center of the search, which might lead into a large motion vector. By selecting a suboptimal one, the bits required from the motion vectors are very likely to be less and thus allowing us to use more bits for encoding the actual difference block. In this method we may also define multiple thresholds and/or use different thresholding criteria for the different zones or types of frames, which can lead to several tradeoffs between speed and quality.

Fig. 2. Circular zones after considering the predicted MV. First M zones will be examined around the predicted motion vector and we will examine whether a block satisfying our criterion exists. If it is found the search stops. Otherwise, if not, the search will move to the center and N zones will be examined in a similar way. Search stops if the thresholding criterion is met.

In addition, in order to benefit from the fact that motion vectors are always differentially encoded, it is essential to take in account the motion vectors of the previous blocks while doing the motion estimation. It is very likely that the motion vector of the current block is the same as, or very close to, the motion vector of the previous block, and it can be considered as an initial prediction. In CZS, before performing the search around the actual center of the search area, we might be able to increase the efficiency of the algorithm even further by defining smaller circular zones around the position corresponding to this predicted motion vector. We can again define different thresholds for these zones, and if a block is found that satisfies

these criteria, we may stop the search and use that block as our best match. Otherwise the search continues by examining all the other blocks starting again from the search window center (Fig. 2).

3. CZS ALGORITHM

Here is the algorithm of the proposed circular zone search (CZS) to estimate the motion vector MV of the current block:

Step 1: If the current block is the leftmost block, set $MV_{\text{predicted}}$ to be (0,0).
Otherwise, set $MV_{\text{predicted}}$ to be the motion vector of the previous block.
If $MV_{\text{predicted}} = (0,0)$, go to step 5.

(Circular search around predicted motion vector)

Step 2: Construct M circular zones around $MV_{\text{predicted}}$ in the search window.
Set $i=1$.

Step 3: Compute MAD for each search point in zone i.
Let MinMAD be the smallest MAD up to this point.

Step 4: If $\text{MinMAD} < T_1$, goto step 10. Else if $i < M$, set $i=i+1$, and goto step 3.

(Circular search around (0,0))

Step 5: Construct N circular zones around (0,0) in the search window.
Set $i=1$. Set LAST = False.

Step 6: Compute MAD for each search point in zone i.
Let MinMAD be the smallest MAD up to this point.

Step 7: If $\text{MinMAD} < T_2$ or LAST = true, goto step 10.

Step 8: If $T_2 < \text{MinMAD} < T_3$, set LAST = true.

Step 9: If $i < N$, set $i = i+1$. Go to step 6.

(Final step. Use best MV found.)

Step 10: The motion vector of the block corresponding to MinMAD is chosen.
This can be optionally refined by performing a local half-pixel search.

Here T_1 , T_2 and T_3 are some predefined thresholds, and M is usually smaller than N.

4. RESULTS

For our simulation of the proposed CZS algorithm we used an MPEG1 simulator with bidirectional motion estimation. We used two B frames and GOP length of 15, i.e. IBBPBBPBBPBBPBBPBI. The search range is ± 7 , ± 14 and ± 21 for picture distance of 1, 2 and 3 respectively. The frame rate for all sequences is 30 fps and the same bit rate model in [6] is used for all the test sequences. Four ($M=4$) zones were defined around the predicted MV.

Simulation was performed using the Miss America sequence in CIF format (a video conferencing sequence), and tennis (SIF format) for several bitrates. The results are shown in tables 1, 2, 3 and 4. In tables 1 and 3 the speed up gain for the different search ranges is presented, for Miss America and tennis sequences respectively. Several thresholding values have been used for comparison. In tables 2 and 4 we can see the performance comparison in terms of MSE and PSNR, for the luminance (Y) and chrominance components (U,V) for the corresponding thresholds, and the difference from the MSE and PSNR obtained using the FS approach.

It can be seen that for Miss America, the new algorithm not only has a rather significant speed up compared to FS, but in several cases the visual quality is even higher than FS. At 768kbps, CZS can achieve a speedup factor of 29.41 while achieving a small gain (0.026 dB) in PSNR when compared with full search. Note that the gain in PSNR is very unusual for fast motion estimation algorithms, especially when CZS can achieve a decent speedup factor of 29. When the thresholds are increased to achieve a speed up factor of 74 it can be seen that the algorithm can still perform better than FS. At a speed up of 136 the CZS algorithm has a loss of 0.05dB which is almost negligible. This shows that CZS provides the flexibility of achieving different trade-off between speedup factor and quality, by allowing the thresholds to be changed. When the bit rate is lowered to 512 kbps, a larger gain in PSNR is achieved with the same speedup factors. This suggests that CZS can perform even better in lower bit rates, which is the likely case for video conferencing applications. In Fig. 4 we can see a comparison per frame between FS and CZS, and in Fig. 3 frame 52 is compared with the original for both FS (PSNR=39.94dB) and CZS (PSNR=40.42dB) using thresholding values $T_1=T_2=2$ and $T_3=4$.

For the tennis sequence, even though we can still achieve a small gain in PSNR (~ 0.01 dB) as shown in table 4, the speed up ratio compared to FS is very small (1.17). In Fig. 5 we present a per frame comparison between FS and CZS. By increasing the thresholds we can see that the speed up increases slightly, but the quality also reduces significantly. Tennis is a video sequence with large object motion, panning and zooming, unlike Miss America which is a video conferencing sequence, with relatively small object motion and no panning or zooming. This suggests that although CZS is very powerful for video conferencing situations, it may not be good for sequences with large or/and camera motion. Better tradeoff between the PSNR and the speed up ratio is possible by joining zones together and by using better-tuned thresholds.

In summary, the proposed circular zonal search (CZS) has the following characteristics:

1. it can achieve various speedup and quality tradeoffs by adjusting the thresholds,
2. a small gain in PSNR is possible at lower speedup factors,
3. the loss in PSNR at higher speedup factors is negligibly small,
4. it performs better in lower bit rates, which are the bit rates of interest for video conferencing,
5. it does not perform well in video sequences with large object or camera motion.

5. ACKNOWLEDGMENT

This work was funded by an HKTIT grand.

6. REFERENCES

1. T. Koga et al, "Motion compensated interframe coding for video conferencing," *Proc. Nat. Telecommun. Conf.*, New Orleans, LA, Dec. 1981, pp. G5.3.1-G5.3.5.
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. D. T. Hoang, P. M. Long, and J. S. Vitter, "Efficient Cost Measures for Motion Estimation at Low Bit Rates," *IEEE Trans. On Circuits & System for Video Technology*, Vol. 8, Pp. 488-500, Aug. 1998.
4. R. Li, B. Zeng, and M.L. Liou, "A New Three-step Search Algorithm for Fast Motion Estimation," *IEEE Trans. On Circuits & System for Video Technology*, Vol. 4, Pp. 438-442, Aug. 1994.
5. Z. He and M.L. Liou, "A High Performance Fast Search Algorithm for Block Matching Motion Estimation," *IEEE Trans. On Circuits & System for Video Technology*, Vol. 7, Pp. 826-828, Oct. 1997.
6. Bellcore, "MPEG-1 simulator", 1991

Bit Rate (kbps)	Algorithm	Thresholds	± 7 window Speed up	± 14 window Speed up	± 21 window Speed up	Total Speed up
768	FS	None	1.00	1.00	1.00	1.00
	CZS	$T_1, T_2, T_3 = 2$	9.06	7.84	6.63	7.41
	CZS	$T_1, T_2 = 2, T_3 = 4$	35.44	32.94	25.05	29.41
	CZS	$T_1, T_2 = 2, T_3 = 6$	63.94	85.08	67.49	74.10
	CZS	$T_1, T_2 = 4, T_3 = 8$	121.26	163.60	117.58	135.88
	CZS	$T_1, T_2, T_3 = 6$	94.55	102.14	71.50	86.15
512	FS	None	1.00	1.00	1.00	1.00
	CZS	$T_1, T_2, T_3 = 2$	8.50	7.54	6.46	7.15
	CZS	$T_1, T_2 = 2, T_3 = 4$	33.90	31.68	24.16	28.30
	CZS	$T_1, T_2 = 2, T_3 = 6$	64.29	78.83	64.50	70.43
	CZS	$T_1, T_2, T_3 = 4$	40.59	33.28	24.73	29.73
	CZS	$T_1, T_2 = 4, T_3 = 8$	123.21	161.51	114.69	133.90
	CZS	$T_1, T_2, T_3 = 6$	93.51	95.80	68.93	82.39
	CZS	$T_1, T_2, T_3 = 8$	148.29	182.65	122.11	148.24

Table 1. Speed up comparison between FS and CZS for the Miss America Sequence

Bit Rate (kbps)	Alg.	Threshold	MSE	Diff	Y-PSNR (dB)	Diff (dB)	U-PSNR (dB)	Diff (dB)	V-PSNR (dB)	Diff (dB)
768	FS	None	2.298	0	40.909	0	40.017	0	43.236	0
	CZS	$T_1, T_2, T_3 = 2$	2.280	-0.018	40.982	0.073	40.170	0.153	43.417	0.181
	CZS	$T_1, T_2 = 2, T_3 = 4$	2.292	-0.006	40.935	0.026	40.102	0.085	43.349	0.113
	CZS	$T_1, T_2 = 2, T_3 = 6$	2.298	0	40.911	0.002	40.098	0.081	43.322	0.086
	CZS	$T_1, T_2 = 4, T_3 = 8$	2.314	0.016	40.853	-0.049	40.022	0.005	43.225	-0.011
	CZS	$T_1, T_2, T_3 = 6$	2.316	0.018	40.850	-0.051	40.068	0.051	43.228	-0.008
512	FS	None	2.479	0	40.250	0	39.302	0	42.341	0
	CZS	$T_1, T_2, T_3 = 2$	2.421	-0.058	40.453	0.203	39.652	0.350	42.709	0.358
	CZS	$T_1, T_2 = 2, T_3 = 4$	2.434	-0.045	40.409	0.158	39.603	0.301	42.636	0.295
	CZS	$T_1, T_2 = 2, T_3 = 6$	2.444	-0.035	40.372	0.122	39.563	0.261	42.574	0.232
	CZS	$T_1, T_2, T_3 = 4$	2.436	-0.043	40.403	0.153	39.609	0.307	42.581	0.240
	CZS	$T_1, T_2 = 4, T_3 = 8$	2.453	-0.026	40.344	0.094	39.534	0.232	42.474	0.133
	CZS	$T_1, T_2, T_3 = 6$	2.461	-0.018	40.314	0.064	39.525	0.223	42.450	0.109
	CZS	$T_1, T_2, T_3 = 8$	2.491	0.012	40.212	-0.038	39.474	0.172	42.322	-0.019

Table 2. Performance comparison between FS and CZS in terms of MSE and PSNR for the Miss America sequence

(a)

(b)

(c)

(d)

(e)

Fig. 3. Frame 52 of the Miss America sequence as encoded using Full search and Circular zonal search at 512kbps. Only the luminance component is shown. (a) Original. (b) FS with PSNR 39.94dB. (c) Error image between FS and the original. (d) CZS with PSNR 40.42dB (e) Error image between CZS and the original.

Fig.4. PSNR of full search and CZS for Miss America at 512 kbps using $T_1=T_2=2$, $T_3=4$

Fig.5. PSNR of full search and CZS for Tennis at 768 kbps using $T_1=T_2=2$, $T_3=4$

Bit Rate (kbps)	Algorithm	Thresholds	± 7 window Speed up	± 14 window Speed up	± 21 window Speed up	Total Speed up
768	FS	None	1.00	1.00	1.00	1.00
	CZS	$T_1, T_2, T_3 = 2$	1.17	1.15	1.14	1.15
	CZS	$T_1, T_2 = 2, T_3 = 4$	2.00	1.92	1.86	1.90
	CZS	$T_1, T_2 = 2, T_3 = 6$	2.77	2.60	2.46	2.56
	CZS	$T_1, T_2, T_3 = 4$	2.06	1.94	1.86	1.92
	CZS	$T_1, T_2 = 4, T_3 = 8$	4.01	3.61	3.36	3.55
	CZS	$T_1, T_2, T_3 = 6$	2.55	2.46	2.36	2.43
	CZS	$T_1, T_2, T_3 = 8$	4.21	3.67	3.38	3.60

Table 3. Speed up comparison between FS, and CZS for the tennis Sequence.

Bit Rate (kbps)	Alg	Threshold	MSE	Diff	Y-PSNR (dB)	Diff (dB)	U-PSNR (dB)	Diff (dB)	V-PSNR (dB)	Diff (dB)
768	FS	None	8.053	0	30.276	0	37.893	0	36.857	0
	CZS	$T_1, T_2, T_3 = 2$	8.047	-0.006	30.285	0.009	37.890	-0.003	36.862	0.005
	CZS	$T_1, T_2 = 2, T_3 = 4$	8.066	0.013	30.262	-0.014	37.872	-0.020	36.848	-0.010
	CZS	$T_1, T_2 = 2, T_3 = 6$	8.241	0.188	30.092	-0.184	37.791	-0.102	36.735	-0.122
	CZS	$T_1, T_2, T_3 = 4$	8.083	0.030	30.245	-0.031	37.876	-0.017	36.837	-0.020
	CZS	$T_1, T_2 = 4, T_3 = 8$	8.379	0.326	29.942	-0.334	37.706	-0.187	36.623	-0.234
	CZS	$T_1, T_2, T_3 = 6$	8.333	0.280	29.998	-0.278	37.742	-0.151	36.657	-0.201
	CZS	$T_1, T_2, T_3 = 8$	8.506	0.453	29.819	-0.457	37.644	-0.249	36.521	-0.336

Table 4. Performance comparison between FS and CZS in terms of MSE and PSNR for the tennis sequence.