A Scheme of Adaptive Search Point Placement using DCT

Young-Min Park, Chu-Seok Chang, Changsoo Lee
Department of Software Engineering, Kyungwoon University

Abstract

In this paper, we propose the adaptive scheme to place more search points as long as the operation capability of the motion estimator in the video codec permits. And the proposed algorithm takes advantage of the intuitive fact that the quality of the decoded video is more degraded as the spatial frequency of the corresponding block is increased due to the augmentation of local minima per unit area. Therefore, we propose the scheme to enhance the quality by locating relatively more search points in the block with high frequency components by analyzing the spatial frequencies of the video sequence. As a result, the proposed scheme can adaptively place more search points possibly permitted by the motion estimator and guarantees better quality compared to the TSS and FS.

Keywords: adaptive motion search, motion estimation, fast search algorithm, three step search

1. Introduction

The data compression of digital video takes advantage of the temporal and/or spatial redundancy reduction and the statistical properties of the generation codes. Among them, the method using temporal redundancy is the most effective method that can reduce the amount of compressed data, which leads to high compression ratio by the inter-frame motion compensation coding(IMCC). The IMCC technique has been adopted as the moving picture coding standards in H.261 and H.263 by ITU-T and in MPEG-1, -2 and -4 by ISO/IEC. This method first codes motion vectors obtained from the subsequent frames. And then the prediction error between the original frames and the motion compensated frames are coded for transmission. Therefore, the prediction errors would be reduced as more exact motion vectors are obtained. So the motion estimation method can greatly affect the compression efficiency[1,2,5,6].

The motion estimation methods of video sequences are mainly classified into the following three categories: the block matching algorithm (BMA), the pel-recursive algorithm, and the optical flow-based algorithm[1,2]. Above all, the BMA is commonly used in IMCC due to its operation time and easy implementation capability in hardware. In BMA, the basic is the FSBMA(Full Search BMA) that searches all the blocks for every pixel in sequential within the pre-defined search range. It can produce the optimal motion vectors for every block, but requires very large computations. One of the fast block matching schemes introduced to overcome the disadvantage of the FSBMA is the three step search(TSS) algorithm, which has shown outstanding performance in searching speed. It has, however, high potentials to estimate erroneous motions by falling into the local minima. So, the prediction error becomes larger, which may cause low compression rate and quality degradation as well. As a consequence, placement of more search points enhances the quality of the decoded video sequences by reducing the possibility of falling into the local minima even though it may increase the amount of computations[1-3].

In this paper, we propose the adaptive scheme to place more search points as much as the operation capability of the motion estimator in the video codec permits. And the proposed algorithm takes advantage of the intuitive fact that the quality of the decoded video is more degraded as the spatial frequency of the corresponding block is increased due to the augmentation of local minima per unit area. Therefore, we propose the scheme to enhance the quality by locating relatively more search points in the block with high frequency components after analyzing the spatial frequencies of the video sequence.

2. Backgrounds

2.1 Block matching criteria

For the BMA, the current frame is divided into reference blocks with N× M pixels, and then each of the reference blocks is used to search its own matching block in the previous frame by comparing all the blocks within the search window with range *-w* to *+w* pixels to both horizontal and vertical directions. And the relative location of the matching block for the corresponding reference block is set as the motion vector which is transmitted together with the difference between them.

As criteria for finding the most matching block with the reference one, the sum of absolute differences(SAD) and the sum of squared differences(SSD) are commonly used as shown in the equations (1) and (2). We use the SAD like as almost BMA employs because it is implemented in simple hardware without using multiplier. Here, the size of the reference block is *N× M* pixels. R(i, j) and S(i, j) represent the luminances of the pixel(i, j) in the reference block and the search window, respectively. After all, the motion vector is determined as the coordinates(u, v) that minimizes the SAD(u, v).

$$SAD(u,v) = \sum_{i=1}^{N}\sum_{j=1}^{M}\left|R(i,j) - S(i+u, j+v)\right| \quad (1)$$

$$SSD(u,v) = \sum_{i=1}^{N}\sum_{j=1}^{M}\left|R(i,j) - S(i+u, j+v)\right|^2 \quad (2)$$

To evaluate the objective quality of the video after motion estimation, we employ the PSNR(peak to peak signal to noise ratio) and the MSE(mean square error) as criteria together, shown in equations (3) and (4). Here, *w* and *h* denote the horizontal and vertical resolutions of the original frame, respectively. And R(i,j) and S(i, j) are the values of the pixel (i, j) in the original and the decoded images, respectively.

$$PSNR = 10\log_{10}\frac{255^2}{SSD^3} \quad (3)$$

$$MSE = (1/w \times h)\sum_{i=0}^{h}\sum_{j=0}^{w}\left|R(i,j) - S(i,j)\right|^2 \quad (4)$$

2.2 Discrete cosine transform

In the video compression, the discrete cosine transform(DCT) is the most commonly used to reduce spatial redundancies. The DCT gathers or concentrates video signals to the specific locations of the coordinates using the correlations between the neighboring pixels contained in the frame of the video sequences. And the data compression is realized by reducing or eliminating the redundancies contained in the video signals.

When the DCT is performed for a single frame, the video image is first divided into blocks with N × N pixels. Then the 2-dimensional DCT is applied to each of the partitioned blocks. The equation (5) shows the 2-D DCT coefficient for the input pixel f(x,y) of a given block. Here, the values of u, v, x and y range from *0* to *N-1*.

$$F(u,v) = \frac{C(u)}{2}\frac{C(v)}{2} *$$
$$\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right]$$
$$\text{where, } C(u),C(v) = \frac{1}{\sqrt{2}}, \text{ if } u = 0 \qquad (5)$$
$$C(u),C(v) = 0, \quad \text{if } u \neq 0$$

3. The proposed adaptive search point placement algorithm

The adaptive search point placement algorithm proposed in this paper for motion estimation have following basic concepts or properties. For the details of each are described in the subsections.

- To reduce the possibilities of falling into the local minima, or the disadvantages of the TSS, the spatial frequencies of the frames in the video sequences are examined. And then more search points are assigned to the portions with higher frequencies to decrease the possibility of estimating erroneous motions.
- The indices that imply the spatial frequencies of every block are calculated using the DCT coefficients and the weight table. And the three thresholds are determined according to the amount of computations that is permissible by the motion estimator of the video codec. Using the threshold, the search points are located.
- One threshold that adjusts the distance between search points is selected among three thresholds α, β, or γ. And then it is applied to the index *r*.

3.1 Properties of the spatial frequency

The spatial frequency is low when each frame of video sequences is plain and changes slowly. While the image components are fine and change rapidly, its spatial frequency is said to be high. In general, it is allowed to place search points at long distances in the image with low spatial frequency because of small SAD among them. To the contrary for the image with high spatial frequency it is easy to fall into local minima if the distances among the search points become far. We confirm this intuitive knowledge by simulation.

The test sequence used is the Stefan frame #2 with 4:2:0 YUV format and the size of 352× 288 pixels as shown in Fig. 1. The audience portion with higher spatial frequency and the ground portion with lower one are used for comparison. The results on both portions using the FSBMA with search window 32× 32(i.e., -16～+16) are shown in Fig. 2. It is obvious that the images with high spatial frequency holds high possibility than that in the lower one because of higher potential of falling into the local minima if the number of search points are placed equally.

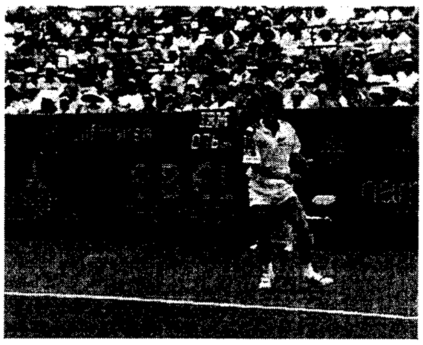
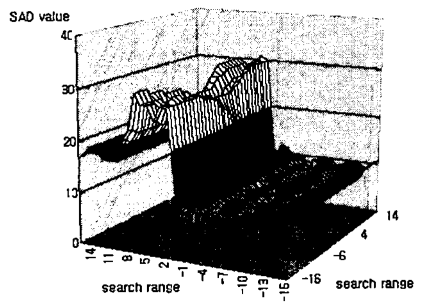
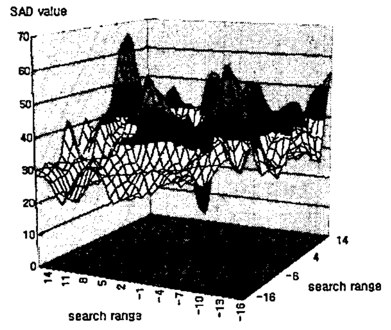


Fig. 1 Stefan frame #2



(a) Ground portion with low spatial frequency.



(b) Audience portion with high spatial frequency.
Fig. 2 The SAD distribution of the Stefan frame #2.

3.2 Index and weight tables

To produce the index representing the spatial frequency for every block, the image of the video frame is partitioned into blocks with 8× 8 pixels and the DCT coefficients are calculated. Here, the DCT coefficients are calculated from the I-frame of the current GOP(group of pictures). Then, the weights proportional the spatial frequencies by the equation (6) are imposed to each of the corresponding coefficient.

The index r is the value normalized with the sum of the weighted DCT coefficients for the sum of the DCT coefficients in a single block. Here, M and N denotes the number of pixels in a block. And (i,j), K(i,j), and W(i,j) are the coordinate for each pixel in a block, the DCT coefficients, and the weights, respectively. Therefore, the index gets higher proportional to the spatial frequency of the corresponding block because the weights are proportionally assigned. The Fig. 3 shows the weight table employed in our experiments. The same results showed for both tables regardless of linear or nonlinear.

$$r = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} W(i,j)K(i,j)}{\sum_{i=1}^{M}\sum_{j=1}^{N} K(i,j)} \qquad (6)$$

i: LOW → HIGH

j ↓	LOW							HIGH
LOW	0	1	2	3	4	5	6	7
	1	2	3	4	5	6	7	8
	2	3	4	5	6	7	8	9
	3	4	5	6	7	8	9	10
	4	5	6	7	8	9	10	11
	5	6	7	8	9	10	11	12
	6	7	8	9	10	11	12	13
HIGH	7	8	9	10	11	12	13	14

(a) Linear

i: LOW → HIGH

j ↓	LOW							HIGH
LOW	0	1	3	4	7	8	9	10
	1	4	7	9	11	12	14	15
	3	5	8	10	12	14	15	16
	5	7	9	13	14	15	17	18
	7	8	12	14	15	18	19	20
	9	11	13	15	19	20	22	23
	10	12	15	20	21	23	25	27
HIGH	13	15	19	21	22	24	28	29

(b) Nonlinear

Fig. 3 Weight Tables $W(i, j)$.

3.3 Determining threshold

The goal of the threshold determination process is to enhance the PSNR by assigning the maximum number of search points within the processing capability of the motion estimator. In video compression, the fast and effective motion estimation is required because the motion estimation is the most time consuming part in the overall coding. Also, the video codec has processing capability of 30 frames per second in order that the human recognizes just like continuous, natural, and smooth video sequences. Moreover, there are various kinds of video sequences in size. So, we would like to estimate motions effectively by adjusting the amount of calculations regularly to have the motion estimator of the codec process more than 30 frames in a second regardless of the size of the video sequences.

Once the processing capability of the motion estimator - the number of blocks possibly compared in a second - is given as an input parameter, it is possible to get the required amount of calculations per block by dividing it with the number of blocks in a input image. The amount of calculations becomes smaller as the size of input image becomes larger, and it can be denoted as shown in the equation (7). Here, C_b, C and B_N are the amount of calculation required per block, the processing capability of the motion estimator, and the number of blocks in input images, respectively.

$$C_b = C / B_N \qquad (7)$$

To compare the computational complexities among the various motion searching algorithms including the proposed one, we summarize the expression s to calculate the number of search points in the Table 1. Using the C_b from the equation (7), we can determine the rate thresholds α, β, and γ to adjust the distances between the search points adaptively according to the spatial frequencies from the index *r*. From the number of the search points in the Table 1, we can derive the expressions on α, β, and γ, as shown in the equation (8). Here, $1 - \alpha - \beta - \gamma \geq 0$, $\gamma \geq 0$ and *T* is the amount of calculations of the motion estimator required on a single block with the maximum calculations as 1089. The total amount of calculations applied TSS, TSS-2, TSS-3 and FS to the input image should be less than or equal to those of the motion estimator on a single block.

$$\begin{aligned} &33\alpha + 97\beta + 385\gamma + 1089(1-\alpha-\beta-\gamma) \leq T \\ &1056\alpha + 992\beta + 704\gamma \geq 1089 - T \\ &\gamma \geq 1/704(1089 - T - 1056\alpha - 992\beta) \end{aligned} \qquad (8)$$

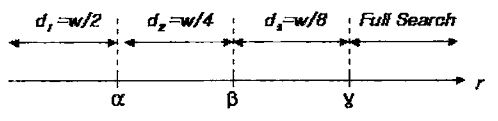
Table 1. Computational Complexities
(The number of SAD calculations per block)

Schemes	The number of search points (A : the amount of calculations)	
	Expression	Values(w=16)
FSBMA	$(2w+1)^2$	1,089
TSS(d=w/2)	$1 + 8 \log_2 w$	33
TSS-2(d=w/4)	$1 + 48 \log_4 w$	97
TSS-3(d=w/8)	$1 + 288 \log_8 w$	385
Proposed (Combined)	$(1 + 8 \log_2 w) * \alpha +$ $(1 + 48 \log_4 w) * \beta +$ $(1 + 288 \log_8 w) * \gamma +$ $\{(2w+1)^2(1-\alpha-\beta-\gamma)\}$	$33 \leq A \leq 1089$

w : Search window size
d : Distance between search points

3.4 Search point placement

The proposed algorithm classifies four levels based on the thresholds and accordingly adjusts to the distance *d* among the search points: the traditional TSS, TSS-2 and TSS-3 with more increased number of search points for TSS, and FS . In other words, the distance in TSS is fixed as an half of the search window size(*d*=*w/2*). But in the proposed scheme, the distances are adjusted using the index by comparing the thresholds α, β, and γ. The concept of the proposed scheme is depicted in Fig. 4.



w: Search window size
d : distance between search points
Fig. 4. Adjusting distance by thresholds.

Table 2. Simulation results

T	Sequences	α	β	γ	α +β	T	Sequences	α	β	γ	α +β
10% of FS	Children	20	70	10	90	60% of FS	Children	0	40	6	40
	Coast	40	45	15	85		Coast	40	0	2	40
	Foreman	40	45	15	85		Foreman	0	40	6	40
	Stefan	65	15	20	80		Stefan	20	20	4	40
	Weather	40	45	15	85		Weather	40	0	2	40
20% of FS	Children	25	25	50	50	70% of FS	Children	0	30	5	30
	Coast	25	25	50	50		Coast	30	0	2	30
	Foreman	5	45	38	50		Foreman	0	30	5	30
	Stefan	25	25	50	50		Stefan	15	15	3	30
	Weather	25	25	50	50		Weather	25	5	2	30
30% of FS	Children	0	20	80	20	80% of FS	Children	0	20	3	20
	Coast	15	5	78	20		Coast	20	0	1	20
	Foreman	0	20	80	20		Foreman	0	20	3	20
	Stefan	15	5	78	20		Stefan	20	0	1	20
	Weather	15	5	78	20		Weather	20	0	1	20
40% of FS	Children	0	65	2	65	90% of FS	Children	10	0	1	10
	Coast	5	60	1	65		Coast	5	5	1	10
	Foreman	5	60	1	65		Foreman	0	10	2	10
	Stefan	10	55	1	60		Stefan	0	10	2	10
	Weather	10	55	1	65		Weather	10	0	1	10
50% of FS	Children	0	50	7	50						
	Coast	45	5	3	50						
	Foreman	0	50	7	50						
	Stefan	35	15	4	50						
	Weather	50	0	3	50						

* T: processing capabilities of the motion estimator

As an example, the distance is adjusted and set to d_3 in the block with index r if the r has a value between α and β. This approach enhances the quality of the decoded video sequences and regulates the amount of calculations by changing to assign the locations of the unnecessary search points in the region with low frequency to the area with higher frequency.

4. Experiments and result analysis

We performed experiments to evaluate the quality measures and to verify the proposed scheme using the standard MPEG test sequences. They are the 4:2:0 YUV sequences with CIF format having 352× 288 pixels in each frame. In the experiments, 15 frames among the sequences such as Children, Stefan, Foreman, Coast, Weather are used.

And we set the block size and search window range as 8× 8 and -16~+16, respectively. The simulation is done assuming the processing capability corresponding to eh 10% of the calculation based on the FS to the 90% for every 10%. The Table 2 represents the investigated rates showing the optimal PSNRs for the given video sequences among α, β, and γ. As shown in Table 2, the sum of three rate thresholds is under 100% and the full search does not employed when T is less then 30%. But in 40%, the rate for γ is so much reduced to find motions using FS. These facts are shown in Fig. 5 separately.

For the blocks whose calculations belong to 10~30% of the FS, it is effective to assign the search points only using the TSS, TSS-2, and/or TSS-3 because they are lack of capabilities for supporting the FS. But from a certain point between 30 and 40%, the rates of TSS, TSS-2 and TSS-3 are reduced because the FS can be affordable. That is, the quality is enhanced as more search points are assigned.

The amounts of calculations for both α and β is very small, so they can hardly affect the efficiency compared to the TSS-3 and FS. As a consequence, the values for α, β, and γ on five test sequences are almost uniformly produced. This means that the proposed scheme places as many search points as the motion estimator can support and shows the enhanced quality with almost constant amount of calculations for any kinds of video sequences.

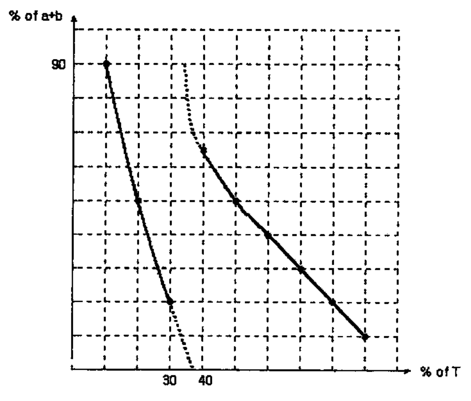


Fig. 5. The processing capabilities of the motion estimator

5. Conclusions

In this paper, we propose the adaptive scheme to place more search points as much as the operation capability of the motion estimator in the video codec permits. We intended to enhance the quality by locating relatively more search points in the block with high frequency components after analyzing the spatial frequencies of the video sequence.

After confirming the effects of the spatial frequencies by simulation, we experiment the our scheme using the several MPEG test sequences. As a result, the proposed scheme can adaptively place the maximum number of the search points possibly permitted by the motion estimator and guarantees the good quality if compared to the TSS and FS. Also, our method can fully utilize the hardware performance. The contribution in this paper presents good tradeoff between the processing time and the quality at the motion estimation essential in the video compression.

References

[1] H. G. Muamann, P. Pirch, and H. J. Grallert, "*Advances in picture coding*," Proceedings of IEEE, Vol. 73, No. 4, pp. 523-548, April 1985.

[2] Joan L. Mitchell, William B. Pennebaker, Chad E. Fogg, and Didier J. LeGall, *MPEG video compression standard*, pp. 283-312, Chapman & Hall, 1996.

[3] A. K. Jain, "*Image data compression: A review,*" Proceedings of IEEE, Vol. 69, No. 3, pp. 349-389, March 1981.

[4] Murat A. Tekalp, *Digital Video Processing*, Prentice Hall, pp. 72-94, 1995.

[5] Borko Furht, Joshua Greenberg, and Raymond Wetwater, *Motion Estimation Algorithms for Video Compression*, KAP, 1997.

[6] John Watkinson, *MPEG-2*, Focal Press, pp. 104-107, 1999.