Matching Quality Investigation of Motion Estimation Algorithms

Alexander Setkov
St-Petersburg University of Aerospace Instrumentation
190000, St-Petersburg, Bolshaya Morskaya 67, Russia
alexsetkov@gmail.com

Abstract

The scope of this paper is block-matching algorithms study. There is a wide range of algorithms performing image block matching to estimate the offset of one block comparing to some block in another image. Roughly they can be divided into two classes referred as full-search and fast algorithms. In this paper, we present a study of a frequency domain based motion estimation method. We show that this fast FFT-based algorithm being applied for motion video compression can be simplified without loss of compression efficiency.

Index Terms: Motion Estimation (ME), Template matching, Sum of Absolute Differences (SAD), Sum of Square Differences (SSD), FFT-based ME algorithm.

I. INTRODUCTION

Every year in the world internet-channel bandwidths becomes more rapid, new high-performance computing devices appear and programs parallelization becomes more effective. As of result of these facts the issues of quality and information processing speed remain still relevant. Video signal transmission and processing are subclasses of signal processing. For the time being there is a wide range of video codecs which perform encoding and video compressing with a different speed and quality of output video sequences. To transmit a streaming video through Internet video codecs with adjacent speed of processing, quality and output size are of significant importance. This can be caused due to problems with channel speed in different parts of network, errors in transmission, intensity and overload of network elements and performance limitations of terminal devices.

H.264 SVC (Scalable Video Codec) is an example of such kind of codecs. It has been standardized to within distinct blocks. Recently in the world hard work is being performed on speed increasing and decreasing of processing costs without quality losses. This can be reached in different ways: partial or full transfer of video coding algorithms to multi-core computing platforms or choosing the faster algorithms which however don’t guarantee the proper quality of output video.

Object of this paper is motion estimation block. Motion estimation (ME) is a process of determining motion vectors that describe the transformation from one image to another; usually from adjacent frames in a video sequence. It is important in many areas of image processing. For example, video coding schemes often exploit the high temporal redundancy between successive frames in a sequence by predicting the current frame from the previous frame based on an estimated motion field. The prediction error image,

which is the difference between the previous frame after motion compensation and the current frame, is then transmitted. Typically, the prediction error has a much lower information content than the original frame if the motion estimates are accurate, allowing high data compression ratios to be achieved [1].

II. Motion estimation algorithms

ME algorithms are divided into two main groups: spatial-domain and frequency-domain algorithms. First algorithms search a matching in spatial domain while the second transform input frames to frequency domain; calculate its cross correlation and then do inverse transform to determine coordinates of a best-matched domain in search area. The second types of algorithms are called fast algorithms. This paper gives complexity estimations of these algorithms and results of experiments on quality of matching. Description of these two types will be also provided in this paper.

A. Description of matching metrics

There are different metrics being used in algorithms to estimate quality of matching. The most widespread metrics are Sum of Absolute Differences (SAD) and Sum of Squared Differences (SSD). These are represented below.

$$SAD(x,y) = \sum_{l=0}^{n-1}\sum_{k=0}^{m-1}\left| f(x+k, y+l) - g(k,l)\right|$$

$$SSD(x,y) = \sum_{l=0}^{n-1}\sum_{k=0}^{m-1}\left(f(x+k, y+l) - g(k,l)\right)^2$$

Where *f* and *g* are previous (based) frame and searched domain respectively.

SAD is a widely used, extremely simple algorithm for finding the correlation between image blocks. It works by taking the absolute difference between each pixel in the original block and the corresponding pixel in the block being used for comparison. These differences are summed to create a simple metric of block similarity, the L1 norm of the difference image [3].

SSD takes sum of square differences between each pixel in original block and in the block being matched. SSD metrics is much accurate than SAD but it is applied rarely due to its complexity (multiplications are involved) [3].

Sometimes MAD (Mean Absolute Difference) is used. It can be calculated as a mean value of sum of absolute differences.

B. Motion Estimation algorithms overview

Motion Estimation algorithms subdivide into block matching algorithms (object is a square or a rectangular), object matching (objects have a sophisticated random shape) and global matching (global motion vector is derived from a local one based on the theory of probability). First group is widely spread (about 90%) due to its simplicity of implementation, low computational cost. These are significant characteristics for real-

time processing. Also they can be easily deployed in hardware. Second group is more sophisticated and is also presented in MPEG standard. Third group is used for elimination of a temporal difference caused by camera shaking.

In the typical block-matching algorithms current frame breaks up into blocks and the goal is to determine the motion vectors which are best suited to each small block according to real shift in the previous frame. There is a wide range of block-matching algorithms. They can be roughly grouped into two categories. The first category consists of those algorithms that are not guaranteed to find the best matching template. These algorithms achieve speed-up because of early elimination of some candidate vectors. Many of them calculate a lower bound for the current vector, compare the best SAD found so far to the bound, and reject the candidate vector, if the lower bound is greater (worse). However, the problem with the fast full search algorithms using early elimination of candidates is the unpredictable amount of computation. If the video sequence is noisy, or there is a large amount of motion, these algorithms reject only small part of the candidate motion vectors and require more computation. Even if, many approaches exist for speeding up the process of SAD matching. These methods can only give the position of the SAD minimum. When the sum of absolute difference matching criteria should be calculated for every location in the image, direct computation requires much more time [2].

Some of the most popular methods of the second group are Three Step Search (TSS), New Three Step Search (NTSS), Simple and Efficient TSS (SES), Four Step Search (4SS), Diamond Search (DS), and Adaptive Rood Pattern Search (ARPS) and so on. These types of algorithms however, are prone to getting trapped in local minima [2].

All the algorithms described above fulfill calculations in spatial domain. As an alternative such calculations can be performed in frequency domain. In this case phase correlation will be estimated rather than block luma matching. Using of the Fourier Transform can dramatically speed up the calculations because of fast implementations of this transform which have computational complexity equal to O(Nlog(N)). A new approach for the computation of any kernel function using FFT has been recently proposed by Fitch. Their approach works by expressing a given matching surface as a series of cosine terms. The more the number of the cosine terms is large, the slower is the algorithm. Fourier's theorem states that any continuous function can be described with a series of sinusoids. The main complication is that we must deal with infinite series rather than finite sums. Therefore convergence issues that do not appear in the finite dimensional situation become of at most importance. Some kernel functions can be easily expressed in the frequency domain using finite cosine terms whereas the absolute kernel cannot [2].

In this paper, we will show how the number of expansion terms will influence on quality and we will give an answer to the question of an optimal number of expansion terms.

C. Description of the fast ME algorithm in frequency domain

In this section we will give a short review of the fast FFT-based ME algorithm. In this method SAD function is approximated by Fourier expansion terms. The formula of the Fourier sequence:

$$f(x) = a_0 + \sum_{p=1}^{\infty} a_p \cos(px) + b_p \sin(px)$$

Fourier transform terms:

$$\begin{cases} a_0 = \dfrac{1}{2\pi}\int_{-\pi}^{\pi} f(x)dx \\ a_p = \dfrac{1}{\pi}\int_{-\pi}^{\pi} f(x)\cos(px)dx,\ p \geq 1 \\ b_p = \dfrac{1}{\pi}\int_{-\pi}^{\pi} f(x)\sin(px)dx,\ p \geq 1 \end{cases}$$

Let f(x) be an even function in interval [-π,π]. The expansion of this function:

$$f(x) = \frac{\pi}{2} - \frac{4}{\pi}\left(\cos(x) + \frac{1}{9}\cos(3x) + \frac{1}{25}\cos(5x) + \ldots\right)$$

Work in threshold value L which limits length of sequence.

$$f_L(x) = \frac{\pi}{2} - \frac{4}{\pi}\sum_{p=1}^{L}\frac{1}{(2p-1)^2}\cos\left((2p-1)x\right)$$

In this case the function is approximated by the term sequence of a finite length. The number of approximation terms of expansion affects accuracy and number of calculations [2].

After summation on whole search area we obtain:

$$SAD_L(x,y) = \frac{\pi}{2}mn - \frac{4}{\pi}\left(\sum_{p=1}^{L}\frac{1}{(2p-1)^2}\sum_{l=0}^{n-1}\sum_{k=0}^{m-1}\cos\left((2p-1)d_{x,y}(k,l)\right)\right)$$

The last equation can be overwritten as:

$$SAD_L(x,y) = \frac{\pi}{2}mn - \frac{4}{\pi}SCD_L(x,y)$$

From this formula it is obviously that to get minimum SAD we have to maximize SCD value.

If we note that:

$$g_p(x,y) = \exp\left(j(2p-1)g(x,y)\right)$$

$$f_p(x,y) = \exp\left(j(2p-1)f(x,y)\right)$$

Using Euler's identity $\cos(x) = \Re\left(\exp(jx)\right)$ the SCD_L can be written as:

$$SCD_L(x,y) = \Re\left\{\sum_{p=1}^{L}\frac{1}{(2p-1)^2}\sum_{l=0}^{n-1}\sum_{k=0}^{m-1} g_p^*(k,l)\, f_p(k+x,l+y)\right\}$$

Finally using Fourier transform we get:

$$SCD_L(x,y) = \Re\left\{ IFFT\left(\sum_{p=1}^{L} \frac{1}{(2p-1)^2} F_p(u,v) G_p^*(u,v) \right) \right\}$$

Where G_p is *FFT* of g_p and F_p is *FFT* of f_p, $\Re$ denotes the real part of a complex number and asterisk denotes the complex conjugation [2].

III. Calculation of complexity and experiments on quality

Evaluation of the algorithms was carried out by means of several criteria such as computational complexity, probability of finding correct motion vectors comparing with ones found by SAD and SSD metrics as well as calculation of residual energy between based and restored frames by SNR metrics.

A. Computational complexity of the algorithms

Arithmetic complexity was calculated as number of operations. These algorithms have determined behavior so it is possible to calculate exact number rather than asymptotic complexity [2]. Computation costs are calculated for different domain and search area sizes. Used sizes for domains are 4x4, 8x8, 16x16 pels. Used sizes for search area are 32x32, 64x64 pels. Table 1 shows total number of operations: assignment, addition, multiplication, cos/sin calculation and FFT/IFFT operations. Level of expansion is equal to 1.

Total number of operations used in full search (exhaustive search) SAD algorithm:

$$O = (M - m + 1)^2 \cdot m^2$$

where M is a search area size, m is a domain size.

Total number of operations used in FFT-based algorithm:

$$O = M^2 \cdot (2 + 36 \cdot L + \log M \cdot (4L + 2))$$

where L is a level of expansion.

TABLE I
TOTAL NUMBER OF OPERATIONS

Search area/domain size	FS SAD	FFT	Speed-up of FFT-based, %
64/16	1844017	327680	82.23
64/8	623865	327680	47.48
64/4	178669	327680	-83.4
32/16	221969	75776	65.86
32/8	120025	75776	36.87
32/4	40397	75776	-87.58

As shown in Table 1 FS SAD matching algorithm works faster than FFT-based algorithm for domain sizes 4x4. In other cases FFT-based algorithm is better. This fact can be explained due to non-dependency of the FFT-based algorithm from domain sizes.

We can resume that small domains (4x4) are better to be searched by FS SAD method and others domains more suitable for FFT-based method.

With a search area increasing improvement of the fast frequency algorithm becomes more evident. For the areas of 32x32 and 64x64 pels reached speed-up is up to 82%. This is a good result for the frequently used area sizes in video coding.

B. Experiments on quality

Quality was measured by different ways: we compared numbers of error matched domains against Full-search SAD and SSD methods. Also we calculated energy of difference between restored and original frames calculated by PSNR metrics.

Experiments were carried out on Carphone video sequence on first 150 frames with resolution 352x288. Used size of search area are 32x32, 48x48, 64x64. Used domain sizes are 4x4, 8x8, 16x16 pels. Used levels of expansion are 1, 5, 10.

Table 2 represents results of experiments. The first criteria which we used to estimate a quality is an error probability of domain matching compared with SAD metrics. As we have shown before we approximated SAD function by Fourier terms. With expansion level increasing the probability goes down. But the error probability remains quite considerable.

Comparing with SSD metrics we got very low probabilities. It is a good result because in this case only one level of expansion is enough to reach the accuracy comparable with SSD metrics.

Table 2 also shows energy of difference of residual frame by PSNR metrics. In order to decrease the energy it is enough to use only one level of expansion.

As a result of these experiments on quality we can see that for the first level of expansion FFT-based method gives the best convergence with SSD metrics. With continuing of an expansion SAD curve will convergence to SAD metrics.

Figure 4 represents the initial (based) frame and the frames restored by FFT-based method. In this example there is a one region which is restored with errors. This disparity caused by region modification in the successive frame in comparison with the previous one. As we can see there is no visual difference between restored frames.

III. CONCLUSION

In this paper, we considered an application of FFT-based motion estimation algorithms for video compression. Computational complexity was estimated. It was shown that the FFT-based algorithm can significantly decrease computational cost for domain sizes larger then 4x4. Experiments on quality were performed. The table shows that the best quality is achieved using one level of expansion. In the future works the variant with implementation of motion estimation algorithm will be considered on different computational platforms.

TABLE II
COMPARISON OF TEMPLATE MATCHING ERROR BETWEEN DIFFERENT METHODS

search area / domain size		L=1	L=5	L=10	SAD	SSD
32/16	Mean matching error probability comparing with SAD, %	9,24	8,34	7,05	0	9,23
	Mean matching error probability comparing with SSD, %	0,15	1,78	3,36	9,23	0
	APSNR	34,72	34,70	34,67	34,65	34,72
32/8	Mean matching error probability comparing with SAD, %	13,95	14,17	13,27	0	11,19
	Mean matching error probability comparing with SSD, %	2,12	3,31	4,44	11,19	0
	APSNR	39,39	39,37	39,33	39,26	39,39
32/4	Mean matching error probability comparing with SAD, %	21,46	22,30	21,88	0	13,60
	Mean matching error probability comparing with SSD, %	8,16	9,39	10,05	13,60	0
	APSNR	41,93	41,92	41,89	41,77	41,93
48/16	Mean matching error probability comparing with SAD, %	11,13	10,32	8,97	0	10,88
	Mean matching error probability comparing with SSD, %	0,37	1,52	3,36	10,88	0
	APSNR	37,84	37,81	37,77	37,71	37,84
48/8	Mean matching error probability comparing with SAD, %	15,04	14,59	13,79	0	12,20
	Mean matching error probability comparing with SSD, %	2,94	3,44	4,67	12,20	0
	APSNR	39,57	39,55	39,51	39,44	39,57
48/4	Mean matching error probability comparing with SAD, %	23,43	22,99	22,56	0	14,33
	Mean matching error probability comparing with SSD, %	9,44	9,33	9,99	14,33	0
	APSNR	42,27	42,26	42,23	42,10	42,27
64/16	Mean matching error probability comparing with SAD, %	11,24	10,50	9,17	0	11,04
	Mean matching error probability comparing with SSD, %	0,30	1,56	3,43	11,04	0
	APSNR	37,94	37,91	37,86	37,80	37,94
64/8	Mean matching error probability comparing with SAD, %	15,16	15,00	14,20	0	12,40
	Mean matching error probability comparing with SSD, %	2,86	3,65	4,81	12,40	0
	APSNR	39,69	39,67	39,64	39,56	39,69
64/4	Mean matching error probability comparing with SAD, %	23,74	24,18	23,72	0	14,82
	Mean matching error probability comparing with SSD, %	9,26	10,07	10,64	14,82	0
	APSNR	42,58	42,57	42,54	42,40	42,58

Fig.4. Restored Frames by FFT-based method with different levels L. Top-left – initial frame; Top-right – L1; Bottom-left – L4; Bottom-right – L10.

REFERENCES

[1] Young Robert W., Kingsbury Nick G., “Frequency-domain Motion Estimation Using a Complex Lapped Transform”, IEEE TRANSACTIONS ON IMAGE PROCESSING, vol. 2, No 1, 1993

[2] Essannouni F., Oulad Haj Thami R., Aboutajdine D., Salam A., “Adjustable SAD matching algorithm using frequency domain”, J. Real-Time Image Process., vol. 1, no. 4, pp 105-107, Jan. 2007

[3] E. G. Richardson, Iain., “H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia”. Chichester: John Wiley & Sons Ltd., 2003.