

FAST EXHAUSTIVE BLOCK-BASED MOTION VECTOR ESTIMATION ALGORITHM USING FFT*

F. Essannouni †

*Laboratoire de Mathématiques Pures et Appliquées, Université du Littoral-Côte d'Opale
Calais, France, and GSCM_LRIT, Faculté des Sciences, Université Mohamed V-Agdal
B.P. 1014 Rabat, Maroc*

R. Oulad Haj Thami

*Laboratoire SI2M, Equipe WiM ENSIAS, Université Mohamed V-Souissi, B.P. 713
Rabat, Maroc*

D. Aboutajdine

*GSCM_LRIT, UFR IT, Faculté des Sciences, Université Mohamed V-Agdal, B.P. 1014
Rabat, Maroc*

and A. Salam

*Laboratoire de Mathématiques Pures et Appliquées, Université du Littoral-Côte d'Opale
C.U. de la Mi-Voix, 50 rue F. Buisson, B.P. 699, 62228 Calais, Cedex, France*

الخلاصة:

من المعلوم أن تقدير حركة الزمر في معالجة الفيديو (video) يمثل عبئاً كبيراً في وحدة المعالج المركزي (CPU). لذلك تم تطوير خوارزميات سريعة لتحسين طوري البحث والملاءمة. هذه الخوارزميات تعمل في الحقل المكاني. وفي هذا المقال نقترح خوارزمية جديدة تستفيد من السرعات المتوفرة في خوارزميات فورييه السريعة (FFT). والخوارزمية المقترحة تحدد الحركة بين الزمرة المشفرة والإطار المرجعي باستخدام العلاقات المتبادلة ذات البعدين، ومن ثم استخدام خوارزمية فورييه السريعة (FFT) للانتقال إلى حقل الذبذبات. وتوضح نتائج المحاكاة أن الخوارزمية المقترحة تعطي نتائج جيدة بينما تحتاج إلى تكلفة حسابية أقل.

† To whom correspondence should be addressed.

E-mails: efedwa@yahoo.fr; oulad@ensias.ma; aboutaj@fsr.ac.ma; Ahmed.Salam@lmpa.univ-littoral.fr

*This work has been supported by Maroc Telecom (EMOTION project).

ABSTRACT

In video processing, block motion estimation represents a CPU-intensive task. For this reason, many fast algorithms have been developed to improve searching and matching phases. These methods however work generally in the spatial domain. In this paper we propose to benefit from speed of the available FFT algorithms. The proposed algorithm computes the motion vector for two blocks using simultaneous two-dimensional cross correlations and use again the FFT to compute the sum square blocks in the frequency domain. Simulation results show that the proposed algorithm gives an optimal SSD (sum square differences) full search results while having a computational cost inferior to the classical fast block matching algorithms.

Key words: frequency domain, full search block matching, FFT algorithms, cross correlation algorithms

FAST EXHAUSTIVE BLOCK-BASED MOTION VECTOR ESTIMATION ALGORITHM USING FFT

1. INTRODUCTION

Motion compensation is used to improve the efficiency of the prediction from past or future frames. Motion estimation is the process of evaluating movements between adjacent frames. The so-called “block-matching algorithms” are the most important of these estimation methods. Pel-recursive and gradient-based motion estimation methods are less frequently used. For each test block in the current frame, the block-matching methods find the most similar candidate block in the frame used for prediction. The displacement between these two blocks is the motion vector for all pixels of the given test block.

The most accurate block-matching method is the full search (FS) that compares every possible candidate block in the search window with the test block; in this way, it produces accurate results, but it is computationally intensive. For this reason, several alternative and faster techniques have been developed. The existing block matching algorithms can be roughly grouped into two categories.

The first category consists of those algorithms that are not guaranteed to find the best matching block within a given search range, but instead use a heuristic approach to guide the search [1-8].

These methods examine only a subset of the possible locations within the search range, and hence can be computed very efficiently. Some of the most popular methods are Three Step Search (TSS) [1], New Three Step Search (NTSS) [2], Simple and Efficient TSS (SES) [3], Four Step Search (4SS) [4], Diamond Search (DS) [5], and Adaptive Rood Pattern Search (ARPS) [6] and so on. The second category contains several algorithms which are guaranteed to find the optimal matching block within a given search range. In recent years, many algorithms have been developed for this type of search for example: successive elimination algorithms (SEA) and its modified algorithms [9–12]. They achieve their speedup through early elimination of candidate search positions. Most of them compute the sum absolute difference (SAD) or the sum square difference (SSD) lower bound for the current vector, compare the best SAD or SSD found so far to the bound, and reject the candidate vector, if the lower bound is greater (worse). However they suffer from the fact that their performance depends largely on the content of the image sequence being encoded. In general the existing fast block matching methods are carried out in the spatial domain. To benefit from the huge amount of work that has been done in the frequency domain and the availability of large types of FFT algorithms and corresponding processors, we have been interested in optimal block matching algorithms using the only frequency domain.

In [13], we have shown that the sum square difference can be written on a new surface which can be computed using two cross correlation operations per one block. In this paper, we propose a new and faster frequency technique which can yield exactly the same optimal result as the direct full search (FS) under SSD metric using only one cross correlation per 2 blocks. The key of this algorithm is to compute in one operation for the whole frame, the sum square blocks using FFT algorithms and to compute the cross correlation for two blocks simultaneously using again the frequency domain.

In Section 2, the proposed frequency motion estimation technique is detailed. The principles of three state-of-the-art frequency motion estimation algorithms are outlined in Section 3. In Section 4, we present some experimental results to show how the suggested approach can work as a promising solution. Conclusions are drawn in Section 5.

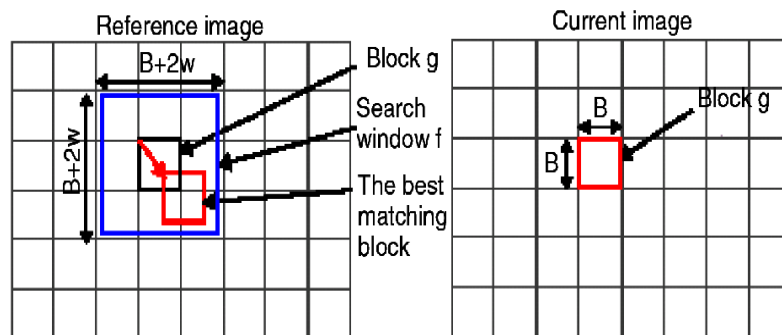


Figure 1. Search area in full search block matching algorithm

2. FAST FULL SEARCH BASED ON FFT ALGORITHMS

2.1. Mathematical Notations

The full search technique was originally described by Jain and Jain [14]. Each image frame is divided into a fixed number of usually square blocks $B \times B$. For each block g in the current frame I_c , a search is made in the reference frame I_r over a search area f within a fixed-sized of search window $\pm w$ (see Figure 1).

The search is for the best matching block, to give the least prediction error, usually minimizing either sum absolute difference (SAD), or sum square difference (SSD). This latter can be expressed as follows:

$$SSD(d_x, d_y) = \sum_{l=0}^{B-1} \sum_{k=0}^{B-1} (f(k + d_x, l + d_y) - g(k, l))^2, \quad (1)$$

where (d_x, d_y) is the motion vector candidate. For simplicity of notations, we assume that dx and dy are in $[0:2w]$. Note that dx between 0 and w indicates a negative displacement, meanwhile d_x between w and $2w$ indicates a positive one and the same for dy .

Let:

$$T_1(d_x, d_y) = \sum_{l=0}^{B-1} \sum_{k=0}^{B-1} f^2(k + d_x, l + d_y), \quad (2)$$

$$T_2(d_x, d_y) = \sum_{l=0}^{B-1} \sum_{k=0}^{B-1} -2f(k + d_x, l + d_y)g(k, l), \quad (3)$$

$$T_3 = \sum_{l=0}^{B-1} \sum_{k=0}^{B-1} g^2(k, l), \quad (4)$$

Then the SSD metric can be written as:

$$SSD(d_x, d_y) = T_1(d_x, d_y) + T_2(d_x, d_y) + T_3. \quad (5)$$

Let (m, n) be the position of the left corner of the search window f in the reference image I_r , therefore:

$$f(k, l) = I_r(m+k, n+l). \quad (6)$$

So if we define the function $S(x, y)$ as:

$$S(x, y) = \sum_{l=0}^{B-1} \sum_{k=0}^{B-1} I_r^2(k + x, l + y), \quad (7)$$

then:

$$T_1(d_x, d_y) = S(d_x + m, d_y + n). \quad (8)$$

Thus the SSD metric can be expressed as below:

$$SSD(d_x, d_y) = S(d_x + m, d_y + n) + T_2(d_x, d_y) + T_3. \quad (9)$$

The last term T_3 is independent of the motion vector (d_x, d_y) .

Therefore, minimizing SSD metric corresponds to minimize:

$$S(d_x + m, d_y + n) + T_2(d_x, d_y). \quad (10)$$

In the next section we show an optimal approach for minimizing this quantity using only FFT algorithms.

2.2. The Proposed Approach

The term T_2 can be viewed as a spatial correlation which can be efficiently computed using Fast Fourier transform (FFT). Given G the FFT of g and F the FFT of f , and the $IFFT()$ the Inverse fast Fourier transform. The term $T_2(dx, dy)$

can be computed as below:

$$T_2(d_x, d_y) = -2 \Re\left\{IFFT(F(u, v)G^*(u, v))\right\} \tag{11}$$

where \Re denotes the real part of a complex number and the asterisk denotes complex conjugation. Note that f and g are correlated with FFTs by zero padding the size of g to the size of f prior to taking the forward FFTs. And since the cross correlation that we have performed is cyclic, the last $B-1$ rows and $B-1$ columns of result will contain wraparound data that should be discarded. The computation of the term T_2 using FFT is not new and has been widely used in many works for speeding up the computation of the SSD metric. Take for instance [15]. In this latter, the method proposed combines the spatial and the frequency domain in its computation of the SSD metric. It uses the fast Fourier transform (FFT) for calculating the term $T_2(dx, dy)$ and a novel data structure called the Windowed-Sum-Squared-Table for computing the sum square blocks. The approach here is different. In this paper, we propose to use again the FFT algorithm for computing the sum square blocks which can be derived from the function $S(x, y)$ (see (8)), and to compute the cross correlation for two data blocks simultaneously.

Indeed, we can remark that the function $S(x, y)$ can be written as:

$$S(x, y) = \sum_{l=0}^{H-1} \sum_{k=0}^{W-1} I_r^2(k+x, l+y)m(k, l), \tag{12}$$

where:

$$m(k, l) = \begin{cases} 1 & \text{for } 0 \leq k < B \text{ and } 0 \leq l < B \\ 0 & \text{for } B \leq k < W \text{ or } B \leq l < H' \end{cases} \tag{13}$$

H and W are the numbers of pixels in vertical and horizontal directions of the reference image I_r , respectively. This remark is crucial in our paper. Indeed from (12), the function $S(x, y)$ can be also considered as a spatial correlation. Therefore, using the correlation Fourier theorem, the function $S(x, y)$ can be computed using again the FFT algorithms.

$$S(x, y) = \Re\left\{IFFT\left((R(u, v)M^*(u, v))\right)\right\} \tag{14}$$

where $R(u, v)$, $M(u, v)$ denote respectively the FFT of $I_r^2(x, y)$, and the FFT of $m(x, y)$.

Finally, the motion vector (d_x, d_y) can be easily determined by minimizing (10) which can be computed using (11) and (14).

Moreover, to decrease the computation time further, we propose to compute the cross correlation given by (11) for two data blocks simultaneously. The steps of our proposed algorithm are given in the following section.

2.3. The Proposed Algorithm Steps

- (1) Computing the sum square blocks for the whole image using Equation (14).
- (2) Selecting first and second input data blocks g_1 and g_2 in said current image, and selecting first and second input search areas f_1 and f_2 in said reference image.

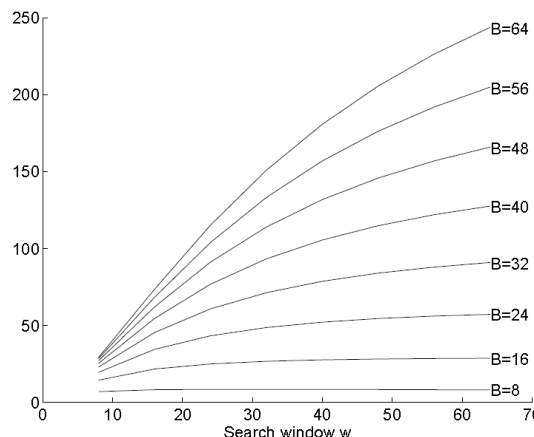


Figure 2. Complexity improvement when using the proposed frequency method instead of the full search, versus search range w

- (3) Converting the input data blocks to a complex data block as below:

$$g_c(x, y) = g_1(x, y) + jg_2(x, y), \quad (15)$$

where j is the square root of -1 .

Note here that g_1 and g_2 have been padded by zeros to the size N^2 . For simplicity of notation, N^2 denotes the size of the search areas.

- (4) Let G_c be the FFT of g_c , determining a first frequency domain data block G_1 and a second frequency domain data block G_2 from G_c as follows:

$$G_1(u, v) = \frac{G_c(u, v) + G_c^*(N-u, N-v)}{2}, \quad (16)$$

and

$$G_2(u, v) = \frac{G_c(u, v) - G_c^*(N-u, N-v)}{2j}. \quad (17)$$

These formulas can be obtained easily by utilizing the symmetrical properties of the FFT process by which real inputs produce even real and odd imaginary outputs and imaginary inputs produce odd real and even imaginary outputs; (See the appendix for more details).

- (5) In the same way as we have computed G_1 and G_2 , determining the FFT of the search areas F_1 and F_2 .
 (6) Computing the following surface:

$$F_1(u, v)G_1^*(u, v) + jF_2(u, v)G_2^*(u, v). \quad (18)$$

- (7) Inverse transforming the complex resultant surface to a resultant spatial blocks having real and imaginary pads;
 (8) Determining first and second cross-correlations between said input data blocks and said input search areas by separating the real and imaginary pads of said resultant spatial block wherein said real part is the first cross correlation between the first input data and search area and said imaginary part is the second cross correlation between the second input data and the corresponding search area. These two cross correlations are exactly the ones given by Equation (11) between the data blocks and their corresponding search areas.
 (9) Repeating steps (2) through (8) for other of said plurality of data blocks so as to generate a plurality of motion vectors.

2.4. Running Time Analysis

The frequency block matching algorithm that we have presented consists of two principal steps. In the first step we compute the sum square blocks through the computation of $S(x, y)$ using FFT algorithms. This step takes only $O(WH \log_2(WH))$, per frame.

In the second step, we perform 2 complex FFT and one IFFT for two blocks. This step has a complexity close to $O((2w+B)^2 \log_2(2w+B))$ per 2 blocks.

Then the complexity of the proposed algorithm per frame is:



(a) Proposed method



(b) Fast robust correlation



(c) Orientation correlation



(d) Phase correlation

Figure 3. Sample of reconstructed frame from bus sequence for block size 16×16 and search range ± 8

$$O \left(WH \left(\frac{((2w+B)^2 \log_2(2w+B))}{2B^2} + \log_2(WH) \right) \right) \quad (19)$$

Figure 2 shows the improvement in computational complexity by using the proposed frequency method over the full search. The transform method becomes relatively more efficient with larger search range w . Note also, that this computation time can be greatly reduced by using machine specific optimized FFT implementations which are widely available.

3. PREVIOUS FREQUENCY MOTION ESTIMATION

The existing frequency motion estimation techniques are mainly used for the global motion estimation. They can also be applied to local motion estimation as block based approaches. The best known and popular frequency technique is the

phase correlation method. In this section we review the principle of this method and two other correlation techniques that have recently appeared in the literature. The methods considered are based on the correlation methodology in the frequency domain and as such have similar computational complexity to our scheme.

3.1. Phase Correlation

The phase correlation method [16] capitalizes on the well-known Fourier shift theorem which states that shifts in the spatial domain correspond to linear phase changes in the Fourier domain. Its formula between two images f_1 and f_2 is given in [14] by:

$$P(x, y) = IFFT \left\{ \frac{F_2(u, v) F_1^*(u, v)}{|F_2(u, v) F_1^*(u, v)|} \right\}, \quad (20)$$

- $IFFT()$ is the inverse Fourier Transform
- F_1 and F_2 denote the Fourier Transforms of the two images f_1 and f_2 .

For block based motion estimation the phase correlation technique works by computing the phase correlation function (20) between two regions of image data, usually in the form of co-sited rectangular blocks in the current and the next frame of a video sequence [17–20]. However, in local motion estimation, the phase correlation method fails in several cases.

Indeed the mathematical analysis of this technique assumes ideal translation which is not true between two compared blocks from real frames. (Sample of image predicted using the phase correlation method is shown in Figure 3(d)).

3.2. Robust Correlations

3.2.1. Orientation Correlation

Orientation correlation estimates the motion between two images by correlating their orientation images [21]. Each pixel in an orientation image is a complex number that represents the orientation of intensity gradient. This method is based on Andrews's wave M estimator [22] which makes it statistically robust.

3.2.2. Fast Robust Correlation

A generalization of robust correlation using Andrews's wave M estimator has been proposed in [23]. This approach works by expressing the matching surface in terms of many cross correlations which can be computed using the fast Fourier transform. However, even if the robust correlation methods outperform the standard cross and phase correlation methods, these approaches give a suboptimal result when compared with FS. Figure 3(b–c) shows the results of the prediction of the second frame from bus sequence using first the orientation correlation and then the fast robust correlation.

4. SIMULATION RESULTS

The proposed algorithm is simulated using the luminance of the popular video sequences: Football, Bus, and Foreman CIF formats (352×288) and three QCIF (176 ×144) video sequences called Mobile, Silent, and Stefan.

The frames are subdivided into equal blocks of size 16x 16 and the current blocks are searched into the reference image using a search range of ±8. In this paper we choose the FFTW3 for FFT computation [24]. FFTW is a widely used free-software library written in C that computes the discrete Fourier transform (DFT) and its various special cases. Its performance is competitive even with vendor optimized programs, but unlike these programs, FFTW is not tuned to a fixed machine. Instead, FFTW uses a planner to adapt its algorithms to the hardware in order to maximize performance. The planner applies a set of rules to recursively decompose a problem into simpler sub-problems of the same type. "Sufficiently simple" problems are solved directly by optimized, straight-line code that is automatically generated by a special-purpose compiler. Our test platform is a Pentium 4, 3.06 GHz processor with 512 MB of RAM.

Table 1. MSE Results for the First 80 Frames from Test Video Sequences

Algorithm	Tempete	Football	Bus	Mobile	Silent	Table
OC	150.90	512.14	274.93	156.11	51.14	140.13
PC	163.06	675.26	506.73	159.31	62.02	172.75
CORR	149.99	478.28	386.57	157.21	50.58	128.99
FRcorr	153.75	539.56	361.55	162.46	48.60	130.75
Proposed method	118.81	219.12	222.46	153.45	32.97	72.54

Figure 4 and Table I compare our proposed frequency method to the classical cross correlation (CORR), the phase correlation method (PC) [16, 20], the orientation correlation method (OC) [21], and the fast robust correlation (FRcorr) (we announce that, for this method, we have chosen the same coefficients used in video coding experiments [23]). The comparison is done through the minimization of the mean square error (MSE) between the predicted frame and the current one.

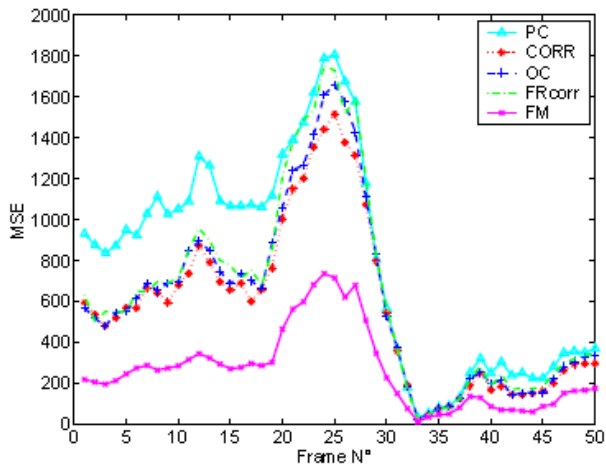
We can clearly see from this experiment, that our proposed method gives more accurate results than the other frequency method in terms of minimizing the mean square error. It should be noted that the other methods are slightly faster since they are implemented in a cyclic match.

In Table 2, the proposed algorithm is compared against five traditional BMAs: DS [5], TSS [1], SS4 [4], NTSS [2], and ARPS [6]. The comparison between the motion estimation algorithms is done through four aspects which are: (1) MSE performance; (2) speedup ratio with respect to the FS; (3) average distance from the true motion vector per block; and (4) probability of finding the true motion vector. Note that the “true” motion vectors are regarded as those found in FS under SSD. We can see that the first two aspects provide the prediction quality and searching speed improvement. The last two aspects show how the found vectors are far from the true ones and the percentage of finding these latter, but these two aspects are independent of the first two ones. That means that a motion vector far away from the optimal one could even give better quality within the search area. In all BMA simulations, the SSD is chosen as the block distortion measure.

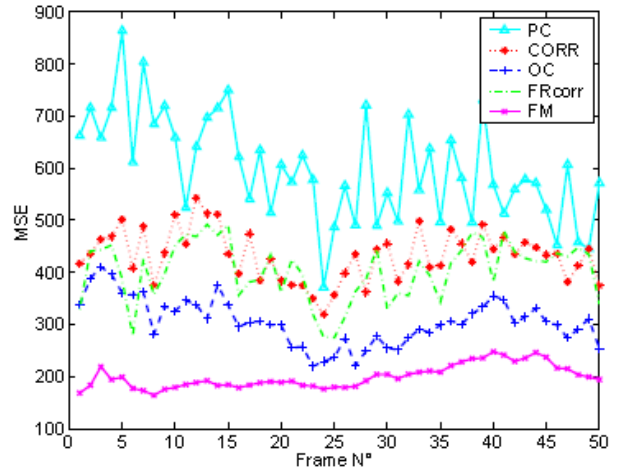
Table 2 shows that, the proposed algorithm always gives the same result as direct (FS) while decreasing dramatically the computation time. Furthermore, it outperforms the other algorithms in terms of MSE, probability of finding the true motion vectors and distance between the found vectors and the true ones while giving the smallest execution time. To know if this performance will change in the presence of an additive noise, we add a Gaussian noise to the test video sequences before motion estimation process (see Tables 3 and 4).

From a maximum likelihood perspective, it is well known that the SSD is justified when the additive noise distribution is Gaussian [25]. For that reason, we have chosen this type of noise in this experiment.

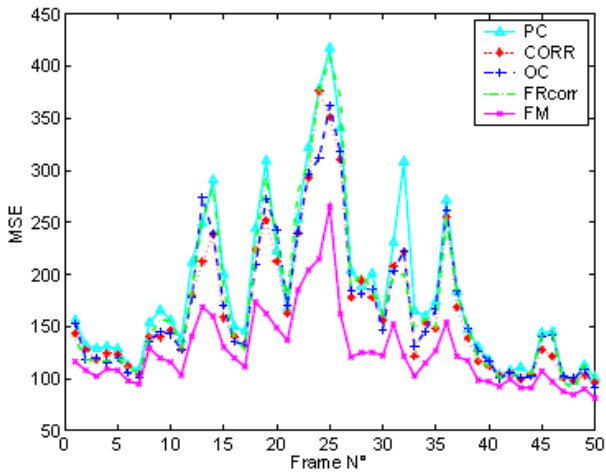
The advantages of the proposed method against the other algorithms are clearly highlighted in Tables 3 and 4. These advantages can be seen in rapidity and the ability to give the same result as the FS algorithm, unlike the other fast block matching algorithms which require more computation time and are prone to fall in local minima.



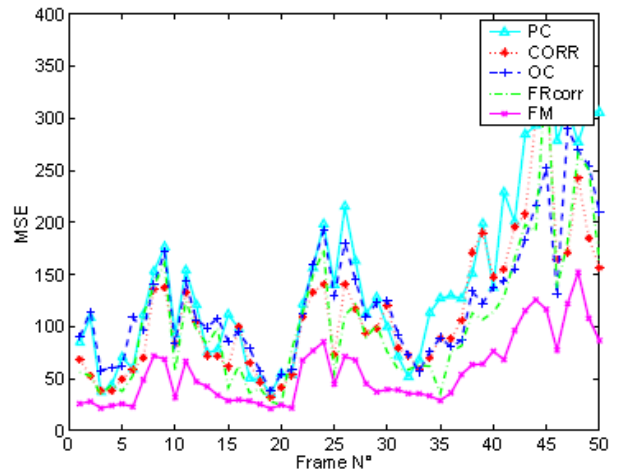
(a) Football



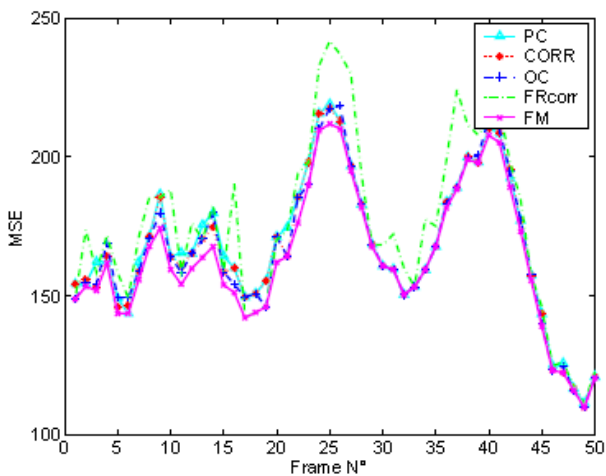
(b) Bus



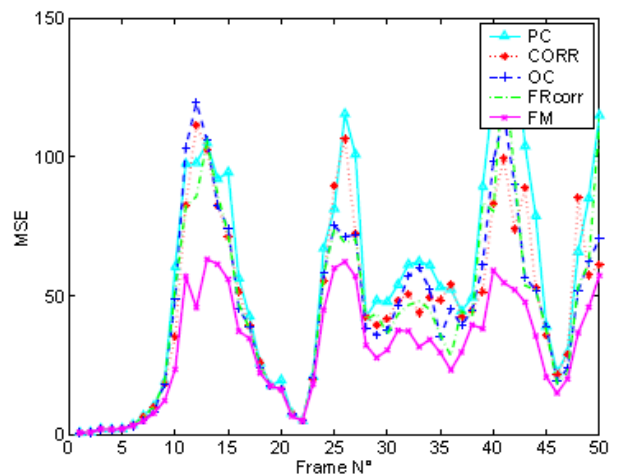
(c) Tempete



(d) Table



(e) Mobile



(f) Silent

Figure 4. MSE results using different frequency method; in this figure, FM denotes the proposed frequency method

Table 2. Performance of the Proposed Algorithm Against Spatial Block Matching Algorithms for the First 80 Frames from CIF Sequences

Football sequence						
Algorithm	Proposed method	DS	TSS	SS4	NTSS	ARPS
MSE	219.12	259.09	263.41	277.38	265.26	246.75
Time (%)	3.74	9.82	9.15	8.79	9.84	5.96
Probability (%)	100.00	67.80	44.44	38.64	40.78	89.14
Distance	0.00	1.98	1.64	2.35	1.86	0.63
Bus sequence						
Algorithm	Proposed method	DS	TSS	SS4	NTSS	ARPS
MSE	222.46	458.03	341.18	463.04	327.38	339.00
Time (%)	3.74	8.79	9.12	8.43	10.18	4.90
Probability (%)	100.00	76.89	85.98	78.41	84.47	88.76
Distance	0.00	1.12	0.85	1.07	0.90	0.54
Tempete sequence						
Algorithm	Proposed method	DS	TSS	SS4	NTSS	ARPS
MSE	118.81	121.77	123.14	123.97	122.39	122.35
Time (%)	3.74	6.31	9.15	6.65	6.67	3.51
Probability (%)	100.00	99.62	98.99	98.99	99.49	98.48
Distance	0.00	0.02	0.03	0.03	0.02	0.04

Table 3. Performance of the Proposed Algorithm Against Spatial Block Matching Algorithms for the First 80 Frames from CIF Sequences Under an Additive Gaussian Noise with Mean 0 and Variance 0.01

Football sequence						
Algorithm	Proposed method	DS	TSS	SS4	NTSS	ARPS
MSE	1334.23	1467.97	1431.78	1462.12	1438.75	1503.70
Time (%)	3.28	8.43	9.12	8.76	9.84	5.26
Probability (%)	100.00	32.58	31.82	27.90	30.93	32.58
Distance	0.00	3.19	3.08	3.12	3.05	3.12
Bus sequence						
Algorithm	Proposed method	DS	TSS	SS4	NTSS	ARPS
MSE	1371.20	1648.82	1511.14	1639.57	1503.57	1569.36
Time (%)	3.28	8.43	9.46	8.43	9.84	5.26
Probability (%)	100.00	52.15	63.26	53.91	60.48	54.04
Distance	0.00	2.09	1.91	2.10	1.88	2.01
Tempete sequence						
Algorithm	Proposed method	DS	TSS	SS4	NTSS	ARPS
MSE	1269.98	1288.18	1287.89	1289.60	1286.29	1296.34
Time (%)	3.28	6.65	9.12	7.01	7.37	3.84
Probability (%)	100.00	80.56	81.94	81.44	82.95	78.54
Distance	0.00	0.78	0.76	0.77	0.70	0.79

Table 4. Performance of the Proposed Algorithm Against Spatial Block Matching Algorithms for the First 80 Frames from CIF Sequences Under an Additive Gaussian Noise with Mean 0 and Variance 0.001

Football sequence						
Algorithm	Proposed method	DS	TSS	SS4	NTSS	ARPS
MSE	1332.21	1464.91	1428.93	1459.92	1435.65	1500.51
Time (%)	3.50	8.76	9.46	8.79	10.18	4.92
Probability (%)	100.00	31.19	27.90	25.25	28.03	31.57
Distance	0.00	3.33	3.16	3.31	3.09	3.17
Bus sequence						
Algorithm	Proposed method	DS	TSS	SS4	NTSS	ARPS
MSE	1361.56	1635.80	1501.85	1627.45	1493.20	1558.66
Time (%)	3.50	8.43	9.12	8.43	10.18	4.92
Probability (%)	100.00	53.79	62.37	53.54	59.09	60.23
Distance	0.00	2.10	1.99	2.11	2.01	1.79
Tempete sequence						
Algorithm	Proposed method	DS	TSS	SS4	NTSS	ARPS
MSE	1257.53	1276.10	1275.12	1277.40	1273.67	1284.20
Time (%)	3.50	6.65	9.15	7.01	7.37	3.84
Probability (%)	100.00	83.08	83.71	82.07	83.46	79.29
Distance	0.00	0.65	0.63	0.73	0.61	0.76

Table 5. Performance of the Proposed Method Against DS, and ARPS for the First 80 Frames from Football Sequence for a Search Range ±16 and ±24

w=±16			
Algorithm	Proposed method	DS	ARPS
MSE	128.37	206.06	180.92
Probability (%)	100.00	61.11	82.83
Distance	0.00	3.13	1.60
w=±24			
Algorithm	Proposed method	DS	ARPS
MSE	102.06	197.55	164.94
Probability (%)	100.00	58.96	80.43
Distance	0.00	3.64	2.13

The proposed method benefits principally from the fast and deterministic time of the FFT algorithms. These features are required especially for sequences with large motion (e.g. Football sequence). For this type of sequences, the search range is generally chosen as more than ±8 (see Table 5).

5. CONCLUSION

The FFT block matching algorithm discussed in this paper exploits only the FFT algorithms in its computation of the SSD metric. This method is not heuristic-based and thus can consistently identify the best matching blocks minimizing MSE.

The algorithm is well suited for software implementations. It is independent of image content, and can be even faster by using machine specific optimized FFT implementations which are widely available. On the other hand, the proposed frequency method greatly outperforms the existing frequency motion estimation techniques in terms of accuracy while still keeping the same order of complexity.

APPENDIX

Here we explain how we can compute two simultaneous real FFTs using one complex FFT. For simplicity, this explanation concerns a 1D dimensional signal. The 2D dimensional case can be obviously derived from the 1D one.

In order to save cost when performing FFTs on the data, the two-for-one FFT can be used. The two-for-one FFT utilizes the properties of a complex FFT to perform two real FFTs.

To begin with, the two real sequences $x(t)$ and $y(t)$ are used to create a complex sequence $z(t)$ such that:

$$z(t) = x(t) + jy(t). \tag{21}$$

The buffer for z is twice the length of the FFT and alternates real data and imaginary data. The Fourier Transform of $z(t)$ is:

$$Z(f) = \sum_{n=0}^{N-1} [x(t) + jy(t)] \times W_N^{-fn}. \tag{22}$$

It is not obvious from Equation (22) how to separate the Fourier Transforms of our two sequences, $X(f)$ and $Y(f)$. The data output by the FFT also alternates real data and imaginary data. It is not as simple as $X(f)$ being the real part of $Z(f)$ and $Y(f)$ being the imaginary part of $Z(f)$. To find how to separate the two sequences, we can write:

$$Z(N-f) = Z(-f) = \sum_{n=0}^{N-1} [x(t) + jy(t)] \times W_N^{-fn}, \tag{23}$$

and

$$Z^*(N-f) = Z^*(-f) = \sum_{n=0}^{N-1} [x(t) - jy(t)] \times W_N^{fn}. \tag{24}$$

Adding Equations (22) and (24), we find that:

$$Z(f) + Z^*(N-f) = 2 \sum_{n=0}^{N-1} [x(t)] \times W_N^{-fn} = 2X(f). \quad (25)$$

Subtracting Equation (24) from Equation (22), we find that

$$\begin{aligned} Z(f) - Z^*(N-f) &= 2 \sum_{n=0}^{N-1} [jy(t)] \times W_N^{-fn}, \\ &= 2jY(f). \end{aligned} \quad (26)$$

From Equations (25) and (26), we can extract the formulas for separating the data for $X(f)$ and $Y(f)$ from $Z(f)$:

$$X(f) = \frac{Z(f) + Z^*(N-f)}{2}, \quad (27)$$

and

$$Y(f) = \frac{Z(f) - Z^*(N-f)}{2j}. \quad (28)$$

REFERENCES

- [1] T. Koga, K. Iinuma, A. Hirano, and Y. Iijima, "Motion Compensated Interframe Coding for Video Conferencing," in *NTC' 81 Conference Record*, 1981, pp. G5.3.1–G5.3.5.
- [2] R. Li, B. Zeng, and M. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation," *IEEE Trans. On Circuits and Systems for Video Technology*, **4(4)**, pp. 438–442.
- [3] J. Lu and M. Liou, "A Simple and Efficient Search Algorithm for Block-Matching Motion Estimation," *IEEE Trans. On Circuits and Systems for Video Technology*, **7(2)**(1997), pp. 429–433.
- [4] L. Po and W. Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation," *IEEE Trans. On Circuits and Systems for Video Technology*, **6(3)**(1996), pp. 313–317.
- [5] S. Zhu and K. Ma, "A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation," *IEEE Transactions on Image Processing*, **9(2)**(2000), pp. 287–290.
- [6] Y. Nie and K. Ma, "Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation," *IEEE Transactions on Image Processing*, **11(12)**(2002), pp. 1442–1449.
- [7] J.-Y. Nam, J.-S. Seo, J.-S. KwaK, M.-H Lee, and Y. H. Ha, "New Fast-Search Algorithm for Block Matching Motion Estimation Using Temporal and spatial Correlation of Motion Vector," *IEEE Trans. Consumer Electron.*, **46(4)**(2000), pp. 934–942.
- [8] X. Q. Banh and Y. P. Tan, "Adaptive Dual-Cross Search Algorithm for Block-Matching Motion Estimation," *IEEE Trans. on Consumer Electronics*, **50(2)**(2004), pp. 766–775.
- [9] W. Li and E. Salari, "Successive Elimination Algorithm for Motion Estimation," *IEEE Transactions on Image Processing*, **4(1)**(1995), pp. 105–107.
- [10] Y. Lin and S. Tai, "Fast Full-Search Block-Matching Algorithm for Motion-Compensated Video Compression," *IEEE Transactions on Communications*, **45(5)**(1997), pp. 527–531.
- [11] Y. Chen, Y. Hung, and C. Fuh, "Fast Block Matching Algorithm Based on the Winner-Update Strategy," *IEEE Transactions on Image Processing*, **10(8)**(2001), pp. 1212–1222.
- [12] T. Ahn, Y. Moon, and J. Kim, "An Improved Multilevel Successive Elimination Algorithm for Fast Full-Search Motion Estimation," in *ICIP03 2003*, pp. II: 351–354.
- [13] F. Essannouni, R. Oulad, Haj Thami, A. Salam, and D. Aboutajdine, "A New Fast Full Search Block Matching Algorithm Using Frequency Domain," in *IEEE ISSPA., Sydney, Australia*, 2005, vol. 2, pp. 559–562.
- [14] J. Jain and A. Jain, "Displacement Measurement and its Application in Interframe Image Coding," *IEEE Transactions on Communications*, **COM- 29(12)**, pp. 1799–1806.
- [15] S. Drew Kilthau and T. M. S. Moller, "Full Search Content Independent Block Matching Based on the Fast Fourier Transform," in *ICIP 2002*, pp. I-669– I-672.

- [16] C. Kuglin and D. Hines, "The Phase Correlation Image Alignment Method", in *IEEE 1975 International Conference on Systems, Man and Cybernetics*, September 1975, pp. 163–165.
- [17] G. Thomas, "Television Motion Measurement for DATV and Other Applications", *Research Report*, 1987.
- [18] Y. Chou and H. Hang, "A New Motion Estimation Method Using Frequency Components", *JVCIR*, **8(1)**(1997), pp. 83–96.
- [19] T. Vlachos and G. Thomas, "Motion Estimation for the Correction of Twin-Lens Telecine Flicker", in *ICIP96*, 1996, p. 16A4.
- [20] Y. Liang, "Phase-Correlation Motion Estimation", *EE392J Project Report*, 2000.
- [21] A. Fitch, A. Kadyrov, W. Christmas, and J. Kittler, "Orientation Correlation", in *BMVC02*, 2002, p. Matching/Recognition.
- [22] P. Huber, *Robust Statistics*. New York: John Wiley, 1981.
- [23] A. Fitch, A. Kadyrov, W. Christmas, and J. Kittler, "Fast Robust Correlation", *IEEE Transactions on Image Processing*, **14(8)**(2005), pp. 1063–1073.
- [24] M. Frigo and S. G. Johnson, "The Design and Implementation of FFTW3," *Proceedings of the IEEE*, **93(2)**(2005) pp. 216–231, (Special Issue on "Program Generation, Optimization, and Platform Adaptation".)
- [25] N. Sebe, M. S. Lew, and D. P. Huijsmans, "Toward Improved Ranking Metrics," *IEEE Trans. Pattern Anal. Mach. Intell.*, **22(10)**(2000), pp. xx-xx.