IEICE TRANS. FUNDAMENTALS, VOL.E93–A, NO.10 OCTOBER 2010

PAPER

An FFT-Based Full-Search Block Matching Algorithm with Sum of Squared Differences Criterion

Zhen LI[†a)], *Student Member*, **Atushi UEMURA**[†], *Member*, *and* **Hitoshi KIYA**[†], *Fellow*

SUMMARY An FFT-based full-search block matching algorithm (BMA) is described that uses the sum of squared differences (SSD) criterion. The proposed method does not have to extend a real signal into complex one. This reduces the computational load of FFT approaches. In addition, if two macroblocks share the same search window, they can be matched at the same time. In a simulation of motion estimation, the proposed method achieved the same performance as a direct SSD full search and its processing speed is faster than other FFT-based BMAs.

key words: *BMA, cross-correlation, SSD, pattern recognition, motion estimation, FFT*

1. Introduction

Block matching is widely used in many fields, including pattern recognition [1], [2], object tracking [3], image denoising [4], computer vision [5] and motion estimation. Because of its efficiency and simplicity, it has been widely adopted in many video coding standards, such as H.263, H.264, MPEG-2, and MPEG-4. However, the direct full-search block matching algorithm (which exhaustively searches for every possible candidate in the search window to find the most similar block) imposes a heavy computational load, which makes it almost impossible to use in any application. To solve this problem, many fast block matching algorithms (BMAs) have been developed. Their basic approaches can be generally divided into three types.

The first type uses an approximative search window instead of a direct full search window. For example three-step search [6], four-step search [7], and diamond search [8] are based on this approach. While the computational load is decreased, the accuracy is less than that of a full-search BMA, and the initial value affects the results.

The second type has the same performance as a direct full-search BMA in terms of accuracy, but imposes a lighter computational load so processing speed is higher. The successive elimination algorithm (SEA) [9] and the fast full-search block-matching algorithm [10] are representatives of this type. However, the degree to which the computational load can be reduced depends on the input signal.

These first two types operate in the spatial domain. The third type shifts the spatial domain problem into the frequency domain by using cross-correlation [11]–[13]. The third type of BMAs focuses on the relationship between the circular cross-correlation and the SSD criterion. It ensures the same accuracy as a direct full search does. At the same time, because of using FFT approaches, its computational load is low and does not depend on the input signal. Furthermore, the third type is available for the double-search-window BMA [14] which offers better block matching results than those in the full search BMA.

The method in reference [11] and SSDcorr [13] are two typical algorithms in all these FFT-based full-search BMAs. Particularly, the experimental result in Ref. [13] showed that SSDcorr was more effective than other BMAs. Here we describe a new full-search BMA that is of the third type. The calculation approach of the proposed method is based on FFT and a recursive summation method.

Compare with SSDcorr, the proposed method does not have to extend a real signal into a complex one before the FFT approach. As a result, it reduces the computational load of FFT approaches. Furthermore, if two macroblocks share the same search window, the proposed method can match them simultaneously by combine two real signals into a complex signal. When the number of macroblocks increases, the difference in processing speed between the two methods became larger.

Compare with the method in reference [11] which also does not have to extend a real signal into a complex one, the proposed method is advanced in the recursive summation part. In addition, if reference [11] uses the method we proposed, it is able to match two macroblocks simultaneously. However, it did not mention a word about how to match two macroblocks at the same time in their paper.

All the advantages above make the proposed method the most efficient one in the third type.

2. Preparation

We begin by defining block matching and discussing the relationship between the cross-correlation and SSD of two circular signals of the same size. Let $\mathbf{Z}$ denote the set of integer numbers.

2.1 Block Matching

First, a definition for block matching is needed. As shown in Fig. 1, let 2-D signal $b(x, y)$ be a macroblock, and let 2-D signal $f(x, y)$ be the search window. Suppose that the search window is bigger than the macroblock. That is,

Manuscript received December 28, 2009.
Manuscript revised May 16, 2010.

[†]The authors are with the Graduate School of System Design, Tokyo Metropolitan University, Hino-shi, 191-0065 Japan.
a) E-mail: li-zhen@sd.tmu.ac.jp

DOI: 10.1587/transfun.E93.A.1748

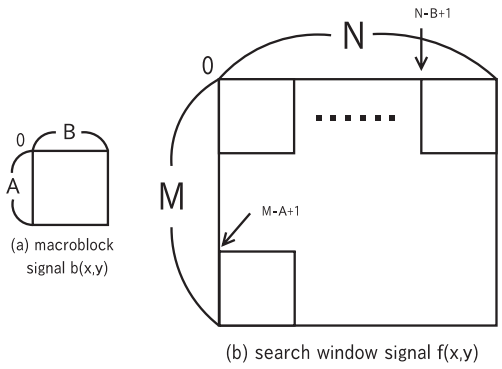

Fig. 1 Macroblock and search window.

$$b(x,y), \quad x = 0,1,\ldots,A-1, \quad y = 0,1,\ldots,B-1, \tag{1}$$

$$f(x,y), \quad x = 0,1,\ldots,M-1, \quad y = 0,1,\ldots,N-1, \tag{2}$$

$$A < M, B < N.$$

Inside the search window there are $(N-B+1)\times(M-A+1)$ different blocks, which have the same size as the macroblock in terms of integral pixels. Here we only focus on the integer precision of the shift amounts. All these different blocks are compared with the macroblock to find the most similar one (the one with the minimum matching error). This procedure can be defined as "full-search block matching." Generally, there are two different matching criteria: the sum of absolute differences (SAD) and the sum of squared differences (SSD):

$$\mathrm{SAD}_{b,f}(u,v) = \sum_{x=0}^{A-1}\sum_{y=0}^{B-1}|f(x+u,y+v)-b(x,y)| \tag{3}$$

$$\mathrm{SSD}_{b,f}(u,v) = \sum_{x=0}^{A-1}\sum_{y=0}^{B-1}\{f(x+u,y+v)-b(x,y)\}^2 \tag{4}$$

$$u \in [0, M-A+1], \quad v \in [0, N-B+1], \quad u,v \in \mathbf{Z}.$$

The u and v are shift amounts. The purpose of block matching is to find the particular (u,v) that yields the minimum SAD or SSD:

$$\mathrm{SAD}(u_0,v_0) = \min_{u,v}\{\mathrm{SAD}_{b,f}(u,v)\}, \tag{5}$$

$$\mathrm{SSD}(u_0,v_0) = \min_{u,v}\{\mathrm{SSD}_{b,f}(u,v)\}. \tag{6}$$

2.2 Circular Cross-Correlation and SSD

Next, let us take a look at the relationship between the circular cross-correlation and SSD of two circular signals, $g(x,y)$ and $f(x,y)$, that have the same period $(M\times N)$. Given shift amounts u and v, the circular cross-correlation of $g(x,y)$ and $f(x,y)$ can be defined as

$$\widehat{cor}_{g,f}(u,v) = \sum_{y=0}^{N-1}\sum_{x=0}^{M-1} g(x,y)f(x+u,y+v), \tag{7}$$

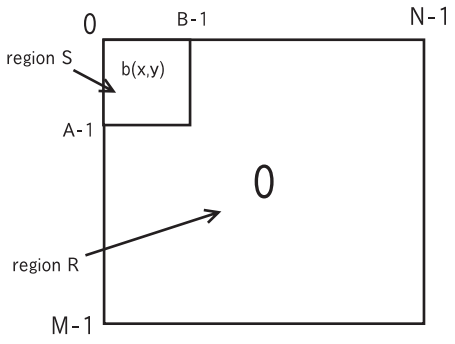

Fig. 2 Extended signal $g_b(x,y)$.

$$(u \in [0, M-1], v \in [0, N-1]).$$

The SSD of two signals can be written as

$$\begin{aligned}\mathrm{SSD}_{g,f}(u,v) &= \sum_{y=0}^{N-1}\sum_{x=0}^{M-1}\{g(x,y)-f(x+u,y+v)\}^2 \\ &= C_g - 2\widehat{cor}_{g,f}(u,v) + C_f,\end{aligned} \tag{8}$$

where

$$C_g = \sum_{y=0}^{N-1}\sum_{x=0}^{M-1}\{g(x,y)\}^2, \tag{9}$$

$$C_f = \sum_{y=0}^{N-1}\sum_{x=0}^{M-1}\{f(x+u,y+v)\}^2 = \sum_{y=0}^{N-1}\sum_{x=0}^{M-1}\{f(x,y)\}^2. \tag{10}$$

Since C_g and C_f are independent of shift amounts u and v,

$$\mathrm{SSD}(u_0,v_0) = \min_{u,v}\{SSD_{g,f}(\mathrm{u,v})\} = \max_{u,v}\{\widehat{cor}_{g,f}(u,v)\}. \tag{11}$$

The DFT of the cross-correlation, the "cross-spectrum" can be calculated using the FFT approach if both signals are circular and have the same size. If these conditions hold the $(u_0,v_0)_{SSD}$ in Eq. (11) can be easily found.

2.3 Non-circular Cross-Correlation

Let us consider the simplest case of block matching with only one macroblock and one search window. Because the macroblock $b(x,y)$ and search window $f(x,y)$ are non-circular signals of different sizes, the property discussed in Sect. 2.2 can not be used directly.

To solve this problem, we can extend macroblock signal $b(x,y)$ into a new signal, $g_b(x,y)$, by padding it with zeros, as shown in Fig. 2. Namely,

$$\begin{cases}\text{region R} = \\ \quad \{(x,y)|A-1 < x \le M-1 \text{ or } B-1 < y \le N-1\}, \\ \text{region S} = \{(x,y)|0 \le x \le A-1 \text{ and } 0 \le y \le B-1\}, \\ \text{entire region Q} = R \cup S.\end{cases} \tag{12}$$

$$\begin{cases} g_b(x,y) = 0, & (x,y) \in R, \\ g_b(x,y) = b(x,y), & (x,y) \in S. \end{cases} \tag{13}$$

Then it is assumed that the extended signal, $g_b(x,y)$, and search window signal, $f(x,y)$, are both circular. Therefore, the circular cross-correlation of the two signals can be written as

$$\widehat{cor}_{g_b,f}(u,v) = \sum_{y=0}^{N-1}\sum_{x=0}^{M-1} g_b(x,y) f(x+u, y+v), \tag{14}$$
$$(u \in [0, M-1], v \in [0, N-1]).$$

Let the range of shift amounts u and v be appropriate to the case of block matching in Eq. (4) and define the cross-correlation as

$$cor_{g_b,f}(u,v) = \sum_{y=0}^{N-1}\sum_{x=0}^{M-1} g_b(x,y) f(x+u, y+v), \tag{15}$$
$$(u \in [0, M-A+1], v \in [0, N-B+1]).$$

Then we can get the relation

$$cor_{g_b,f}(u,v) = \widehat{cor}_{g_b,f}(u,v), \tag{16}$$
$$(u \in [0, M-A+1], v \in [0, N-B+1]).$$

As mentioned in Sect. 2.2, $\widehat{cor}_{g_b,f}(u,v)$ can be calculated by using the FFT. Therefore, Eq. (16) means that the non-circular cross-correlation, $cor_{g_b,f}(u,v)$, can also be calculated by using the FFT. In references [11] and [12], this property is used for calculating $cor_{g_b,f}(u,v)$.

3. Proposed Method

3.1 Non-circular Cross-Correlation and SSD

The SSD criterion can be expressed by three different terms, therefore according to Eq. (8), $\mathrm{SSD}_{g_b,f}(u,v)$ can be written as

$$\mathrm{SSD}_{g_b,f}(u,v) = C_{g_b} - 2cor_{g_b,f}(u,v) + \sum_{(x,y)\in Q} \{f(x+u, y+v)\}^2, \tag{17}$$

where

$$C_{g_b} = \sum_{y=0}^{B-1}\sum_{x=0}^{A-1} \{b(x,y)\}^2. \tag{18}$$

Since Eq. (4) and $g_b(x,y)$ are made up of two regions, $\mathrm{SSD}_{g_b,f}(u,v)$ can also be written as

$$\begin{aligned} &\mathrm{SSD}_{g_b,f}(u,v) \\ &= \sum_{(x,y)\in R} \{g_b(x,y) - f(x+u, y+v)\}^2 + \\ &\quad \sum_{(x,y)\in S} \{g_b(x,y) - f(x+u, y+v)\}^2, \\ &= \sum_{(x,y)\in R} \{f(x+u, y+v)\}^2 + \mathrm{SSD}_{b,f}(u,v). \end{aligned} \tag{19}$$

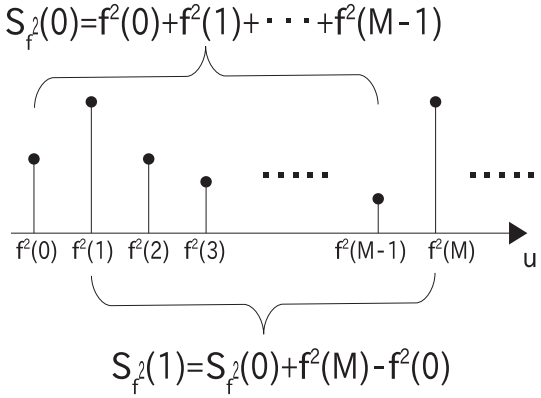

Fig. 3 The conceptual diagram for the recursive summation method in 1D.

Therefore, combining Eqs. (17) and (19) we obtain

$$\mathrm{SSD}_{b,f}(u,v) = C_{g_b} - 2cor_{g_b,f}(u,v) + S_{f^2}(u,v), \tag{20}$$

where

$$S_{f^2}(u,v) = \sum_{(x,y)\in S} \{f(x+u, y+v)\}^2, \tag{21}$$
$$= \sum_{(x,y)\in S} f^2(x+u, y+v). \tag{22}$$

Thus,

$$\begin{aligned} (u_0, v_0)_{SSD} &= \min_{u,v}\{\mathrm{SSD}_{b,f}(u,v)\}, \\ &= \max_{u,v}\{2cor_{g_b,f}(u,v) - S_{f^2}(u,v)\}. \end{aligned} \tag{23}$$

The $cor_{g_b,f}(u,v)$ can be easily calculated by using the FFT. And from Eq. (22), $S_{f^2}(u,v)$ can be considered as a recursive summation. We propose to calculate it using an FIR filter with feed back [15], [16].

For a simple explanation, let us consider the 1D situation shown in Fig. 3. We need to calculate the summation of every M point. If we have the $S_{f^2}(0)$ as the initial value,

$$S_{f^2}(0) = f^2(0) + f^2(1) + f^2(2) + ... + f^2(M-1). \tag{24}$$

$S_{f^2}(1)$ and $S_{f^2}(2)$ can be written as

$$S_{f^2}(1) = S_{f^2}(0) + f^2(M) - f^2(0), \tag{25}$$
$$S_{f^2}(2) = S_{f^2}(1) + f^2(M+1) - f^2(1). \tag{26}$$

And in general,

$$S_{f^2}(u) = S_{f^2}(u-1) + f^2(u+M-1) - f^2(u-1). \tag{27}$$

Only two additions are needed for every $S_{f^2}(u)$. Therefore, in the case of calculating $S_{f^2}(u,v)$, besides the calculation of the initial value, only 4 additions are needed for one coordinate point.

Figure 4 shows the flow chart for the proposed method when there is only one macroblock needs to be matched.

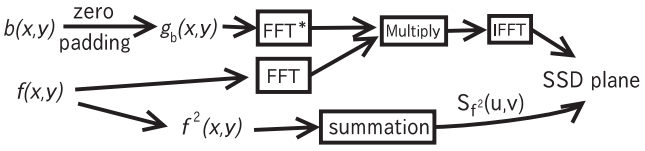

Fig. 4 Flow chart for the proposed method.

Table 1 Comparison of computational load for calculating $S_{f^2}(u,v)$.

Search window size: $M \times N$, Macroblock size: $A \times B$

Method	Addition
Ref. [11]'s	$3MN + 8((M-A+1)(N-B+1) - MN/AB)$
Proposed	$N(A-1) + (M-A-1)(4N-B-3)$

	Addition		
	A=B=16		A=B=32
Method	M=N=32	M=400,N=512	M=400,N=512
Ref. [11]'s	5352	2138760	2032712
Proposed	2205	787085	756845
ratio	2.4:1	2.7:1	2.7:1

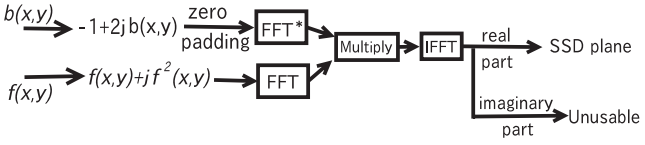

Fig. 5 Flow chart for SSDcorr [13].

3.2 Compare with Other FFT-Based Methods

Here we compare the proposed method with two other FFT-based BMAs, one is the method in Ref. [11], another one is SSDcorr [13]. One important difference among the three methods is how to calculate $S_{f^2}(u,v)$.

Reference [11], proposed a windowed sum squared table for calculating $S_{f^2}(u,v)$. This method costs almost 11 additions for every coordinate (u,v), which makes it less efficient than the proposed method. Table 1 shows the computational load difference between reference [11]'s method and the proposed method in calculating $S_{f^2}(u,v)$. We can see that the proposed method is 2.4 to 2.7 times better.

Figure 5 shows the flow chart for SSDcorr [13], where j is the square root of -1. SSDcorr combines the calculation of $S_{f^2}(u,v)$ with the correlation approach. As a result, the input signal of FFT must be complex. However, if the input signal is complex, the computational load of FFT is much heavier than the case that the input signal is real.

Because the proposed method does not extend real input signals into complex ones, imaginary part of the output is not used. This advantage enables the proposed method to match two macroblocks at the same time which will be discussed in the next section.

3.3 BMA for Two and Multiple Macroblocks

Here we propose a method which can match two macroblocks at the same time if they are sharing the same search window. In this case, the inverse FFT approach is only needed once for matching two macroblocks.

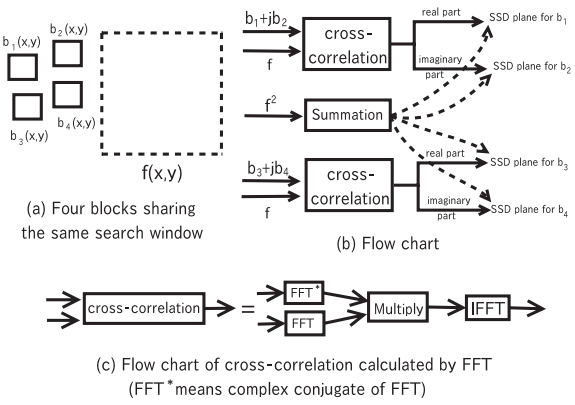

Fig. 6 Four macroblocks sharing the same search window.

Table 2 Times of using FFT and IFFT approaches and calculating $S_{f^2}(u,v)$ in block matching. (*: The input signal is complex.)

	Number of macroblock					
	One			Four		
Method	FFT	IFFT	S_{f^2}	FFT	IFFT	S_{f^2}
SSDcorr	2*	1*	-	5*	4*	-
Ref. [11]'s	2	1*	1	5	4*	1
Proposed	2	1*	1	3	2*	1

For a simple explanation, let us assume three 1-D real signals of the same length, N: $g_1(n)$, $g_2(n)$, and $f(n)$. $G_1(k)$, $G_2(k)$, and $F(k)$ are the DFTs of the signals, respectively. We combine $g_1(n)$ and $g_2(n)$ to create a new complex signal, $h(n) = g_1(n) + jg_2(n)$. The DFT of $h(n)$ can be written as $H(k) = G_1(k) + jG_2(k)$. Therefore, the cross-correlation between $h(n)$ and $f(n)$ can be written as

$$\begin{aligned}\widehat{cor}_{h,f}(n) &= \frac{1}{N}\sum_{k=0}^{N-1}\overline{H(k)}F(k)W_N^{-nk} \\ &= \frac{1}{N}\sum_{k=0}^{N-1}\overline{G_1(k)}F(k)W_N^{-nk} - j\frac{1}{N}\sum_{k=0}^{N-1}\overline{G_2(k)}F(k)W_N^{-nk},\end{aligned} \tag{28}$$

where $\overline{H(k)}$ means the complex conjugate of $H(k)$, and $W_N = e^{-j2\pi/N}$.

Note that the real part of $\widehat{cor}_{h,f}(n)$ is the cross-correlation between $g_1(n)$ and $f(n)$ and that the imaginary part of $\widehat{cor}_{h,f}(n)$ is the cross-correlation between $g_2(n)$ and $f(n)$. This means the inverse FFT approach is only needed once for calculating the two FFT-based cross-correlations.

Figure 6 shows the flow chart of the proposed method when there are four macroblocks sharing the same search window. Please note that $S_{f^2}(u,v)$ is separated from the macroblocks and the search window. Therefore we calculate this part only one time.

Table 2 shows the time of using FFT and IFFT approaches and calculating $S_{f^2}(u,v)$ for matching one or four macroblocks sharing the same search window. We can

see that the proposed method is the most efficient one. Please note that despite SSDcorr does not have to calculate $S_{f^2}(u,v)$ separately, its input signals of FFT are all complex. The calculation of $S_{f^2}(u,v)$ in the proposed method is more efficient than that of Ref. [11]'s method. In addition, Reference [11]'s method is able to match two macroblocks simultaneously if it uses the method we proposed.

4. Simulation

4.1 Block Matching

We first simulated the simplest case of block matching. There is only one macroblock and one search window. Table 3 shows the execution time in different sizes. Every block matching algorithm was repeated 1000 times. We compared four different methods: direct SSD full search, reference [11]'s method, SSDcorr [13] and the proposed method. We can see that the proposed method is the fastest one in every size and is 5 to 15 times faster compared with direct SSD full search.

Tables 4 and 5 show the execution time against number of macroblock sharing the same search window. The search window size were 512×512 and 1024×1024, the macroblock size were 32×32 and 128×128.

In the proposed method, the inverse FFT approach and the multiplication are only needed once for every two macroblocks. In contrast, SSDcorr and Ref. [11]'s method need to do them twice to match two macroblocks. Therefore, the simulation results showed that the proposed method was the fastest one and as the number of macroblock increased the difference became larger. In addition, inside the proposed method the time of calculating S_{f^2} takes up 6 to 28 percent of the total execution time and only needs to calculate once, which proved that the heaviest part for block matching in Eq. (23) is the calculation of $cor_{g_b,f}(u,v)$.

All the simulations above were based on C language with a GCC compiler version 4.2.4. We used FFTW3[17] library for the FFT approach.

4.2 Motion Estimation

We then simulated motion estimation by using video sequences listed in Tables 6. We used one frame before the current one to generate a predicted frame. The macroblock size was fixed at 16×16. There are two search window sizes: 32×32 (± 8 pixels around the macroblock) and the entire frame.

We compared six different methods in terms of both the PSNR (between the predicted frame and the current one) and the processing speed. The methods were direct SSD full search, direct SAD full search, SSD diamond search [8], SSDcorr [13], and the proposed method.

Figure 7 plots the PSNR between the predicted frame and the current frame for different methods when using the video sequence "Caltrain." The proposed method and SSDcorr are exactly the same as the direct SSD full search from the view point of the PSNR, thus, they share the same curve in Fig. 7. We can see that the PSNR under the SSD criterion is better than that under the SAD criterion. The PSNR of the SSD diamond search depends on the start point. If we do not select the start point appropriately, both the PSNR and the processing speed become worse.

Tables 7 show the processing speed of generating one predicted frame using different methods based on Matlab 6.0. We can see that the proposed method is faster than the others.

All our experiments were performed on a computer with an Intel Core2 2.4-GHz CPU, 2-GB memory running

Table 3 Execution time for block matching in different sizes (based on C language, repeated 1000 times).

Size(pixels)		Execution time(ms)			
SW	MB	Direct FS	Ref. [11]'s	SSDcorr	Proposed
32×32	16×16	160	60	40	30
48×48	16×16	500	180	150	120
64×64	16×16	1050	280	200	160
64×64	32×32	1940	290	210	180
128×128	32×32	15800	1280	1130	1020
256×256	32×32	85410	6740	6440	5620

Table 4 Execution time for block matching against number of macroblock sharing the same search window. The search window size was 512×512, the MB1 size was 32×32, the MB2 size was 128×128 (based on C language).

	Execution time(ms)						
Number of	Ref. [11]'s		SSDcorr		Proposed		
MB	MB1	MB2	MB1	MB2	S_{f^2}	MB1	MB2
1	110	110	80	80	20	70	70
2	140	150	140	140	20	100	110
3	200	200	190	200	20	140	140
4	270	270	260	270	20	160	170
5	310	320	320	320	20	200	210
8	460	460	480	480	20	270	270
10	570	580	600	600	20	320	330

Table 5 Execution time for block matching against number of macroblock sharing the same search window. The search window size was 1024×1024, the MB1 size was 32×32, the MB2 size was 128×128 (based on C language).

	Execution time(ms)						
Number of	Ref. [11]'s		SSDcorr		Proposed		
MB	MB1	MB2	MB1	MB2	S_{f^2}	MB1	MB2
1	580	580	500	500	100	400	410
2	830	830	830	820	100	620	620
3	1170	1180	1150	1150	100	860	870
4	1360	1360	1480	1490	100	950	950
5	1600	1610	1800	1800	100	1230	1230
8	2320	2320	2700	2690	100	1630	1640
10	2890	2900	3450	3450	100	1980	1980

Table 6 Video sequences using in the simulation of motion estimation.

Name	Frame size	Frame number	Total frame length
"Caltrain"	400×512	No.0 to No.9	10
"Hall monitor"	288×352	No.30 to No.39	10
"Garden"	240×352	No.1 to No.12	12
"Football"	240×352	No.30 to No.45	16

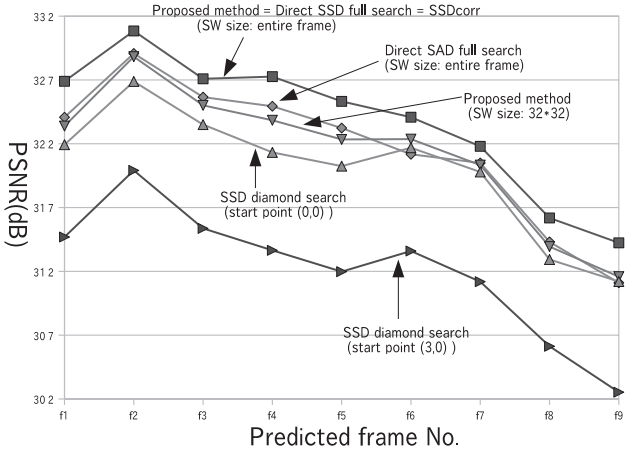

Fig. 7 PSNR between predicted frame and current frame ("Caltrain").

Table 7 Execution time for generating one predicted frame. The macroblock size was fixed at 16×16 (based on Matlab).

	Execution time(s)	
Method	"Caltrain"	"Hall monitor"
Proposed (SW size: 32×32)	0.38	0.19
SSD diamond search (start point (0,0))	0.77	0.33
SSD diamond search (start point (3,0))	0.99	0.51
Direct SSD FS (SW size: 32×32)	5.58	2.79
Proposed (SW size: entire frame)	26.6	6.87
SSDcorr (SW size: entire frame)	44.1	11.48
Direct SAD FS (SW size: entire frame)	1950	460
Direct SSD FS (SW size: entire frame)	1970	464

	Execution time(s)	
Method	"Garden"	"Football"
Proposed	0.17	0.17
SSD diamond search (start point (0,0))	0.36	0.33
SSD diamond search (start point (3,0))	0.46	0.45
Direct SSD FS (SW size: 32×32)	2.35	2.35
Proposed (SW size: entire frame)	4.81	4.81
SSDcorr (SW size: entire frame)	7.95	7.96
Direct SAD FS (SW size: entire frame)	318	317
Direct SSD FS (SW size: entire frame)	320	320

Linux.

5. Conclusions

In this paper, we proposed a full-search block-matching algorithm that uses the sum of squared differences (SSD) criterion. The calculation approach of the proposed method is based on FFT and a recursive summation method. The proposed method does not have to extend a real signal into a complex one. This reduces the computational load of FFT approaches. If two macroblocks share the same search window, the proposed method can match them at the same time. The simulation results showed that the proposed method was faster than other FFT-based full search BMAs.

In future work, we plan to decrease the memory consumption and the computational load of FFT approaches in the proposed method by dividing the search window. And we also plan to extend this method into the sub-pixel level block matching.

References

[1] A. Koschan, V. Rodehorst, and K. Spiller, "Color stereo vision using hierarchical block matching and active color illumination," Proc. 13th Int. Conf. on Pattern Recognition, vol.1, pp.835–839, Vienna, Austria, Aug. 1996.
[2] Y. Kawai, H. Sumiyoshi, and N. Yagi, "A cut detection based on a block matching with variable block size," Proc. IEICE Gen. Conf. 2007, D-12-96, March 2007.
[3] L. Di Stefano and E. Viarani, "Vehicle detection and tracking using the block matching algorithm," in Recent Advances in Signal Processing and Communications, N. Mastorakis ed., World Scientific and Engineering Society Press, ISBN 960-8052-03-3, 1999.
[4] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising with block-matching and 3D filtering," Proc. SPIE-IS&T Electronic Imaging, SPIE vol.6064, pp.606414(1)–606414(12), 2006.
[5] T. Kaneko and O. Hori, "Tracking block selection based on upper bound of block matching error," IEICE Trans. Inf. & Syst. (Japanese Edition), vol.J88-D-II, no.3, pp.562–572, March 2005.
[6] T. Koga, K. Iinuma, A. Hirano, and Y. Iijima, "Motion compensated interframe coding for video conferencing," Proc. IEEE NTC Conf., pp.G5.3.1–G5.3.5, New Orleans, Dec. 1981.
[7] L. Po and W. Ma, "A novel four-step search algorithm for fast block motion estimation," IEEE Trans. Circuits Syst. Video Technol., vol.6, no.3, pp.313–317, June 1996.
[8] S. Zhu and K. Ma, "A new diamond search algorithm for fast block motion estimation," IEEE Trans. Image Process., vol.9, no.2, pp.287–290, Feb. 2000.
[9] W. Li and E. Salari, "Successive elimination algorithm for motion estimation," IEEE Trans. Image Process., vol.4, no.1, pp.105–107, Jan. 1995.
[10] Y. Noguchi, J. Furukawa, and H. Kiya, "A fast full search block matching algorithm for MPEG-4 video," Proc. IEEE Int. Conf. Image Process., vol.1, pp.61–65, Kobe, Japan, Oct. 1999.
[11] S.L. Kilthau, M.S. Drew, and T. Moller, "Full search content independent block matching based on the fast fourier transform," Proc. IEEE Int. Conf. Image Process., vol.1, pp.669–672, New York, Sept. 2002.
[12] A. Fitch, A. Kadyrov, W. christmas, and J. Kittler, "Fast robust correlation," IEEE Trans. Image Process., vol.14, no.8, pp.1063–1073, Aug. 2005.
[13] F. Essannouni, R. Oulad Haj Thami, D. Aboutajdine, and A. Salam, "Simple noncircular correlation method for exhaustive sum square difference matching," Optical Engineering, vol.46, no.10, 107004, Oct. 2007.
[14] Z. Li and H. Kiya, "Double-search-window block matching using the fast fourier transform," Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., no.IVMSP-P11.PG8, Dallas, TX, the U.S., March 2010.
[15] J.W. Adams and A.N. Willson, "A new approach to FIR digital filter design with fewer multipliers and reduced sensitivity," IEEE Trans. Circuits Syst., vol.30, no.5, pp.277–283, May 1983.
[16] J.W. Adams and A.N. Willson, "Some efficient digital prefilter structure," IEEE Trans. Circuits Syst., vol.31, no.3, pp.260–265, May 1984.
[17] M. Frigo and S.G. Johnson, "The design and implementation of FFTW3," Proc. IEEE, vol.93, no.2, pp.216–231, 2005.

Zhen Li received his B.Eng. degree from Tokyo Metropolitan University, Japan in 2009. From 2009, he has been a Master course student at Tokyo Metropolitan University, Japan. His research interests include image processing.

Atsushi Uemura received his B.Eng. and M.Eng. degrees from Tokyo Metropolitan University, Japan in 2008 and 2010, respectively. He joined Techno Mathematical Co., Ltd. in 2010. His research interests include image processing.

Hitoshi Kiya received his B.Eng. and M.Eng. degrees from Nagaoka University of Technology, Japan in 1980 and 1982, respectively, and his D.Eng. degree from Tokyo Metropolitan University, Japan in 1987. In 1982, he joined Tokyo metropolitan University, where he is currently a Professor of Information and Communication Systems Engineering, Faculty of System Design. He was a Visiting Fellow at the University of Sydney, Australia from Oct. 1995 to Mar. 1996. His research interests are in the areas of multirate systems and image processing, including filter bank design and theory, image and video coding, image watermarking and data hiding. He received an Excellent Paper Award in 2008 from IEICE. He served as an associate editor for the IEICE Trans. Fundamentals and the IEEE Trans. Signal Processing from 1998 to 2002 and from 1998 to 2000, respectively. He was also the Guest Editor-in-Chief for the Special Issues of IEICE Transactions published in 1999, 2006, 2007, and 2009 respectively. He currently serves as the President-elect of the IEICE ESS and a Vice President of the APSIPA. He is a Senior Member of the IEEE.