Expert Systems with Applications 39 (2012) 3249–3256

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Expert system based parallel multi-1D block matching algorithm with implementation for motion estimation ☆

Shin-Yeu Lin [a,*], Chong-Wei Su [b], Jung-Shou Huang [c]

[a] Department of Electrical Engineering & Green Technology Research Center at Chang Gung University, Taoyuan, Taiwan, ROC
[b] Institute of Electrical and Control Engineering at National Chiao Tung University, Hsinchu, Taiwan, ROC
[c] Elan Electronics Corporation, Hsinchu, Taiwan, ROC

ARTICLE INFO

Keywords:
Expert system
Knowledge base
Inference engine
Block matching algorithm
Motion estimation
Mixed signal

ABSTRACT

In this paper, we propose an expert-system based parallel multi-1-dimensional block matching algorithm (ESPM-1D-BMA) for motion estimation (ME). Instead of the conventional 2D block matching, we employ the parallel multi-1D blocks matching to improve the computing speed. To improve the ME accuracy, we design a knowledge base and inference engine to determine the true motion vector (MV) from the results of parallel multi-1D blocks matching. To speed up the computing speed further, we present a hardware architecture for implementing the ESPM-1D-BMA. We have demonstrated that the MV estimation accuracy achieved by the proposed ESPM-1D-BMA is much better than the comparing fast block matching algorithms and is close to the 2 dimensional full search block matching algorithm (2D-FSBMA). We also demonstrate that the computing speed of the proposed ESPM-1D-BMA is about two times as fast as the mixed-signal 2D-FSBMA (MS-2D-FSBMA).

1. Introduction

Two dimensional (2D)-block matching algorithm (BMA) is a commonly adopted method for searching the motion vector (MV) between two image frames namely the reference and the search frames, such that the MV is obtained when the best matched 2D-blocks, the reference and one search blocks are found. Among various BMAs, the 2D-full search BMA (2D-FSBMA) using the mean square error (MSE) criteria (Gharavi & Mills, 1990) is considered to be the most accurate algorithm for searching the MV. However, the 2D-FSBMA is computationally complex. Hence, some fast BMAs were proposed, such as the new three-step search (NTSS) (Li, Zeng, & Liou, 1994), diamond search (DS) (Zhu & Ma, 1997) and the hexagonal based search (HEXBS) (Zhu, Lin, & Chau, 2002). These algorithms use few search points to reduce computational complexity, however at the price of poor accuracy. Therefore, proposing a method to reduce the computational complexity of 2D-FSBMA while maintaining its accuracy in searching the MV is the *purpose* of this paper.

Instead of 2D-block matching, we will slice a 2D block into multiple, say K, 1D blocks and employ a parallel multi-1D blocks matching to reduce the computational complexity in searching the MV. Due to the noise appearing in the search and reference frames, the 1D-block matching should be less accurate than the 2D-block matching in searching the MV. Additionally, the MVs determined in each of the K 1D-blocks matching may be different due to various noise contaminations in various 1D blocks. Therefore, to remedy the possible inaccuracy in searching the MV using parallel multi-1D blocks matching, we propose an *expert system* based parallel multi-1D-BMA (ESPM-1D-BMA).

For the purpose of real-time motion estimation (ME), we need to improve the computing speed further by implementing the proposed algorithm in hardware. Therefore, we will present the hardware implementation architecture of the proposed algorithm.

We organize our paper in the following manner. In Section 2, we will present the proposed ESPM-1D-BMA. In Section 3, we will present the hardware implementation architecture of the proposed algorithm. In Section 4, we will test the performance of the proposed algorithm and compare with other existing methods in terms of ME accuracy and the computing speed using comprehensive simulations. Finally, we will draw a conclusion in Section 5.

2. Expert system based parallel multi-1D block matching algorithm (ESPM-1D-BMA)

2.1. Review of 2D-FSBMA

We let I_n and I_{n-1} in Fig. 1(a) denote the reference and search frames, respectively; we let block A inside I_n denote the reference

☆ This research work was supported in part by National Science Council in Taiwan under Grant NSC98-2221-E-182-065-MY2.

* Corresponding author. Address: Department of Electrical Engineering & Green Technology Research Center, Chang Gung University, 259 Wen-Hwa 1st Road, Kwei-Shan, Tao-Yuan 333, Taiwan, ROC. Tel.: +886 3 2118800x3221; fax: +886 3 2118026.

E-mail addresses: shinylin@mail.cgu.edu.tw (S.-Y. Lin), cwsu.ece97g@nctu.edu.tw (C.-W. Su), rong@emc.com.tw (J.-S. Huang).

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.09.012

block (RB) and let block B, which can be any block in I_{n-1}, denote the search block (SB). The idea of 2D-FSBMA is to search all possible SBs and find the one that is most similar to A. This searching task is performed by computing the MSE between blocks A and B as described below. We assume that the sizes of the RB A (or SB B) and frame I_n (or I_{n-1}) are $X \times Y$ and $M \times N$ pixels, respectively, as shown in Fig. 1(b). The MSE between two blocks, RB A and SB B, induced in 2D-FSBMA, denoted by MSE_{2D}, is defined as

$$\mathrm{MSE}_{2D} = \frac{1}{X \cdot Y} \sum_{i=1}^{X} \sum_{j=1}^{Y} (r(i,j) - s(i,j))^2 \quad (1)$$

where $r(i,j)$ and $s(i,j)$ denote the (i,j)th pixel values of the RB A and SB B, respectively. Therefore, the 2D-FSBMA will search through all possible SBs to find the one with smallest MSE, say B*. Then the MV is defined as the difference of position indices between RB A and SB B^* as shown in Fig. 1(a).

2.2. Motivation

To improve the computing speed of 2D-FSBMA, we will slice the 2D block into multi-1D blocks and apply a parallel multi-1D blocks matching algorithm. However, the MVs determined in each of the multi-1D blocks matching may be different due to various noise contaminations in various 1D blocks. Therefore, we will use the *expert system* concept to help determine the true MV. In the following, we will describe the proposed algorithm step by step.

2.3. The 1D block matching

We let B denote the number of pixels in 1D block, then a 1D block is formed by the B consecutive pixel values from a row of the frame. We let $r(i)$, $i = 1,\ldots,B$ and $s(i)$, $i = 1,\ldots,B$ denote the B pixel values of the 1D reference block and the 1D search block, respectively. Then, the MSE between $r(i)$, $i = 1,\ldots,B$ and $s(i)$, $i = 1,\ldots,B$ can be computed as follows:

$$\mathrm{MSE(SBC)} = \frac{1}{B} \sum_{i=1}^{B} (r(i) - s(i))^2 \quad (2)$$

where SBC represents the 1D search block coordinate (SBC), which is identified by the coordinate of the first pixel of the 1D search block, $s(1)$.

2.4. Parallel multi-1D blocks matching

To improve the ME accuracy of 1D block matching, while keeping its computational efficiency, we can use a parallel multi-1D blocks matching. The multi-1D blocks matching consists of K 1D reference blocks, and each 1D reference block represents one independent 1D reference block in the reference frame as shown in Fig. 2, in which we assume $K = 8$.

We let $r_k(i)$, $i = 1,\ldots,B$ denote the kth 1D reference block in I_n and let the $\mathrm{MSE}_k(\mathrm{SBC})$ denote the MSE between $r_k(i)$, $i = 1,\ldots,B$ in I_n and the search 1D block, $s(i)$, $i = 1,\ldots,B$ in I_{n-1}, then $\mathrm{MSE}_k(\mathrm{SBC})$ for $k = 1,\ldots,K$ can be computed by

$$\mathrm{MSE}_k(\mathrm{SBC}) = \frac{1}{B} \sum_{i=1}^{B} (r_k(i) - s(i))^2 \quad (3)$$

which can be performed independently and in parallel for each k.

2.5. Expert system based parallel multi-1D blocks matching algorithm (ESPM-1D-BMA)

As described earlier that the searched MV based on each of the K 1D-blocks matching may be different due to various noise

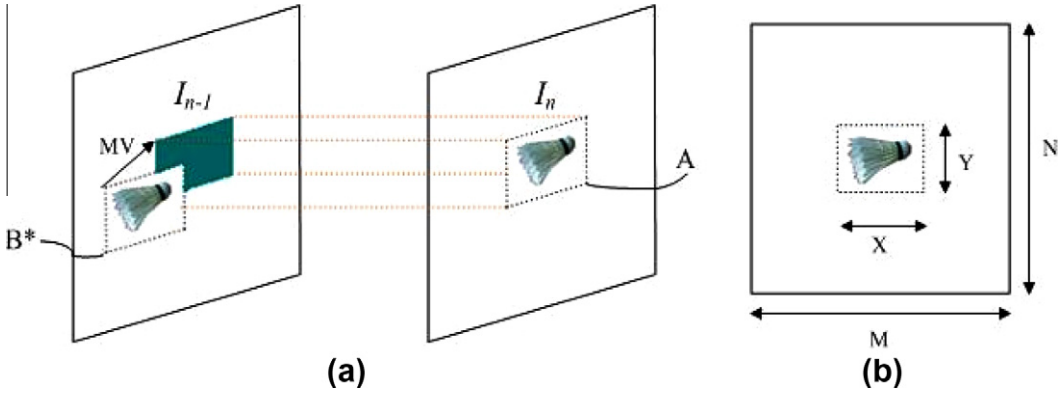


Fig. 1. Motion vector determination using 2D-FSBMA. (b) Example reference block and image frame.

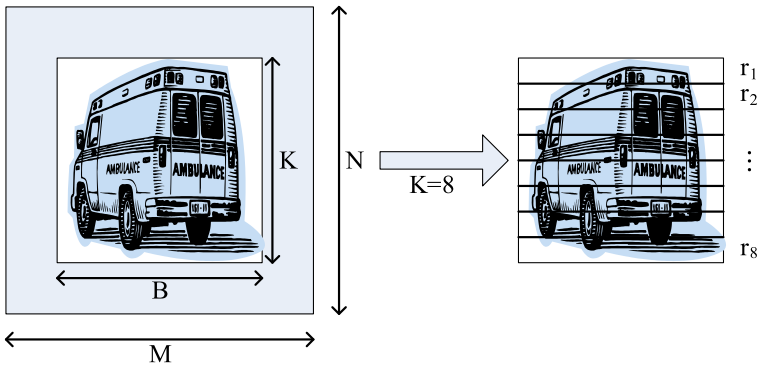


Fig. 2. A diagram of the K independent 1D blocks in a reference frame.

contamination in the search and reference 1D blocks. However, we can view the MSEs computed from the 1D block matching for a reference block in I_n as a result determined by an *expert*. Therefore, the MSEs resulted from K 1D blocks matching can be viewed as a result determined by K experts. Subsequently, to determine the true MV, we can employ the concept of expert system (Liao, 2005; Li & Sun, 2009; Sun & Li, 2008) to construct the *knowledge base* (KB) and the *inference engine* (IE) for the parallel multi-1D blocks matching as follows.

The input of the employed expert system is the overall resulted MSEs of the K 1D-blocks matching. For each of the K 1D-blocks matching, the true MV should be among the MVs with top smallest MSEs. Therefore, the employed KB of the proposed algorithm can be stated as follows. For each of the K 1D blocks matching, we select the P search blocks that correspond to the top P smallest MSEs and assign them with the marked numbers (MNs) $P, P-1, \ldots, 1$, such that the selected search block with smaller MSE is marked by a larger MN. For example, the MN assigned to the search block with smallest MSE is P. Since each selected search block may conclude an MV, there will be $K \times P$ MVs resulted from the parallel multi-1D blocks matching, and each MV is associated with the MN of the corresponding search block.

However, some of the $K \times P$ MVs may be the same. In general, the frequently appearing MVs and the MV with smaller MSE have higher probability to be the true MV. Consequently, if an MV appears q times in the $K \times P$ MVs, it will associate with q MNs. Therefore, the IE of our expert system can be stated as follows. For each distinct MV with q copies in the resulted $K \times P$ MVs, we will sum the q MNs and define the resulting value as the accumulated MN (AMN) of the MV. Consequently, the MV with largest AMN is determined to be the true MV.

For the sake of illustration, we use the following example to explain the proposed ESPM-1D-BMA. We assume $K = 8$ and $P = 3$. In Table 1, the first column shows the K 1D reference blocks, and the second, the third and the fourth columns show the P (=3) MVs with P smallest MSEs resulted from the 1D block matching for each reference block. Then the MVs in columns 2, 3, and 4 are assigned with MNs 3, 2, and 1, respectively. Calculating the AMNs of the seven distinct MVs presented in Table 1, we find that (3,2) has the largest AMN, 17, and is considered to be the true MV.

3. Implementation for motion estimation

For the purpose of real-time ME, we can speed up the computing speed of the proposed algorithm further by hardware implementation. However, a pure *digital circuit* implementation (Hsieh & Lin, 1992; Yang, Wolf, & Vijaykrishan, 2005) may suffer from some implementation problems, such as *high power consumption* and *large chip size*. To overcome these implementation problems, a *mixed-signal* approach that uses simple *current-summation circuit* to circumvent computationally complex digital MSE computation should be a good choice (Panovic & Demosthenous, 2006). In the following, we will present the implementation of the proposed algorithm using mixed-signal approach step by step.

3.1. Implementing 1D-block matching using mixed-signal approach

First of all, we transformed the sensored pixel values into voltages within the range [0 V, 2.5 V] by dividing the range of pixel values between black and white into 255 grey levels, which are represented by 255 least significant bits (LSBs), such that black and white correspond to 0 and 255 LSB, respectively. We let V_x denote the transformed voltage of a pixel value of x LSB, then $V_x = \frac{2.5V-0V}{255} \times x$. We let voltages $V_r(i)$ and $V_s(i)$ denote the transformed voltages of $r(i)$ and $s(i)$, respectively. Assuming $B = 8$, the mixed signal approach for the 1D block matching hardware implementation architecture is presented in Fig. 3, in which we preload the transformed voltage $V_r(i)$ of the pixel value of the 1D reference block $r(i)$, $i = 1, \ldots, B$ in I_n into the reference block memory (RBM). Then the transformed voltage $V_s(i)$ of the pixel value of the search frame I_{n-1} is serially fed into the search block memory (SBM) from left to right, from top to bottom, then the transformed voltage of the pixel value of the 1D search block will be fetched from the SBM as shown in Fig. 3. The two clock signals t_c and t_r are used

Table 1
The top 3 MV for each 1D reference block.

Index of reference block	MV		
	MV with smallest MSE	MV with 2nd smallest MSE	MV with 3rd smallest MSE
1	(3,2)	(5,8)	(1,5)
2	(3,2)	(5,5)	(8,9)
3	(5,8)	(3,2)	(8,8)
4	(8,8)	(3,2)	(5,8)
5	(8,9)	(6,3)	(5,5)
6	(5,8)	(6,3)	(3,2)
7	(3,2)	(5,5)	(1,5)
8	(3,2)	(6,3)	(1,5)

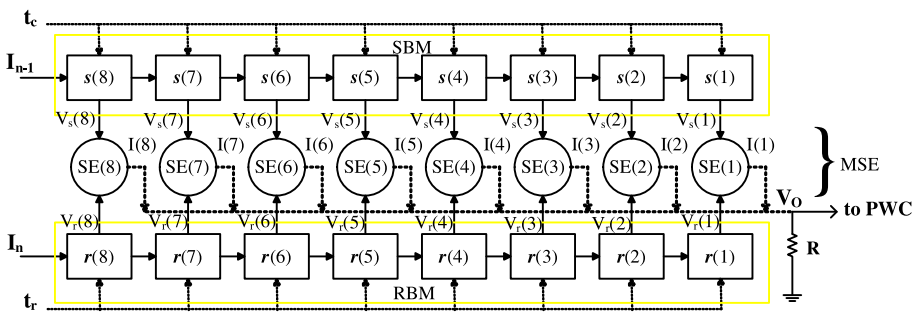


Fig. 3. The hardware implementation architecture of the 1D block matching using mixed-signal approach.

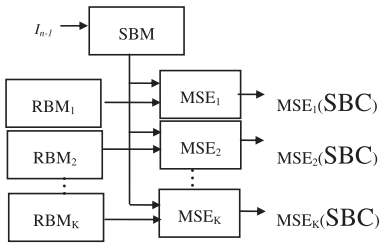


Fig. 4. The computing architecture of parallel multi-1D block matching.

to control the timing of the propagation of the pixel value fed into SBM and RBM. The SE(i) denoted by a circle in Fig. 3 represents the ith *square error computing circuit* to result in an output current $k_t(V_r(i) - V_s(i))^2$, where k_t is the *transconductance* parameter. The summation of the output currents of SE(i), denoted by $I(i)$ in Fig. 3, $i = 1,\ldots,8$, will flow into a single resistive load R as shown in Fig. 3. Then, the resulted voltage across R denoted by V_O is proportional to the MSE of 1D block matching and will be input to a P *winner comparator* (PWC), which will be presented later. Clearly, the computation of the MSE for all possible 1D search blocks can be completed when the last pixel value in the last row of the frame I_{n-1} is read out.

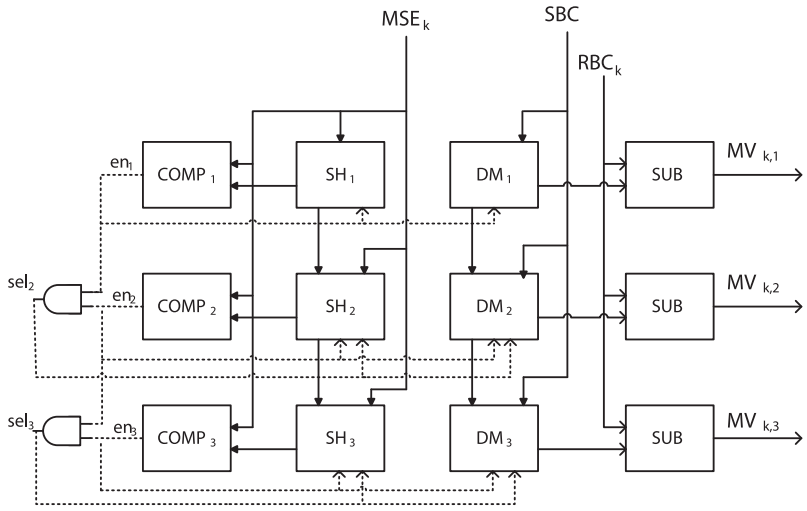
3.2. Implementing parallel multi-1D blocks matching

The parallel computing architecture for the K 1D blocks matching is presented in Fig. 4. The K 1D reference blocks are preloaded into RBM_k, $k = 1,\ldots,K$, and the search block in the frame I_{n-1} are serially fed into SBM. The MSE_k, $k = 1,\ldots,K$ are computed in parallel to obtain MSE_k(SBC), $k = 1,\ldots,K$. The detailed structure of SBM, RBM_k, and MSE_k in Fig. 4 are the same as that presented in Fig. 3.

3.3. Implementing ESPM-1D-BMA

To implement the ESPM-1D-BMA, we need a PWC to select the P search blocks that correspond to the top P smallest MSEs resulted from the 1D block matching for each 1D reference block. The circuit of PWC for the kth 1D reference block, denoted by PWC_k, is presented in Fig. 5, which is designed based on a sorting logic; the solid lines and dotted lines in this figure represent the transmission of data and signals, respectively.

At the very beginning, the P (=3) sample and hold circuits (SHs) are reset to a default value, which is the largest value that SH can take; similarly, the corresponding P digital memories (DMs) are reset to null values. Then, for each incoming MSE_k(SBC), we will compare it with the MSE values stored in the P SHs as indicated by the P comparators, denoted by $COMP_i$, $i = 1,\ldots,P$, shown in Fig. 5. $COMP_i$ will generate an *enable signal* en_i, whose value depends on the comparison result such that if $MSE_k(SBC) < MSE_{SH_i}$ then $en_i = 1$; otherwise $en_i = 0$. The combination of $en_1,\ldots,en_P$ will indicate which of the following ranges that MSE_k(SBC) lies: $[0, MSE_{SH_1}]$, $(MSE_{SH_1}, MSE_{SH_2}]$, $\ldots$, $(MSE_{SH_{P-1}}, MSE_{SH_P}]$, or $(MSE_{SH_P}, \infty]$ as presented in the illustrative table in Fig. 5, in which we set $P = 3$. From the value range of the incoming MSE_k(SBC), we can easily update the P winners as follows. If $en_i = 1$, MSE_{SH_i}, the content of SH_i, should be replaced by either $MSE_{SH_{i-1}}$ for the case $en_{i-1} = 1$ or MSE_k(SBC) for the case that $en_{i-1} = 0$ or $i = 1$, and the content of the corresponding DM_i will be replaced by the proper SBC accordingly. To implement the above P winners updating logic, we will design a *selection signal*, sel_i, to determine what to replace the contents of SH_i and DM_i when the enable signal $en_i = 1$ in the following manner. If $en_i = 1$ and $sel_i = 1$, the contents of SH_i and DM_i will be replaced by SH_{i-1} and DM_{i-1}, respectively. If $en_i = 1$ and $sel_i = 0$, the contents of SH_i and DM_i will be replaced by MSE_k (SBC) and the corresponding SBC, respectively. If $en_i = 0$, SH_i and DM_iremain unchanged. The values of sel_i, $i = 2,\ldots,P$, which are determined based on the values of en_i, $i = 1,\ldots,P$, are presented in the illustrative table of Fig. 5, and they can be generated using AND



	en_1	en_2	en_3	sel_2	sel_3
MSE_k SBC) < MSE_{SH_1}	1	1	1	1	1
$MSE_{SH_1} < MSE_k$ SBC) < MSE_{SH_2}	0	1	1	0	1
$MSE_{SH_2} < MSE_k$ SBC) < MSE_{SH_3}	0	0	1	0	0
$MSE_{SH_3} < MSE_k$ (SBC)	0	0	0	0	0

Fig. 5. The circuit of PWC_k and illustrations.

gates as shown in Fig. 5. Notably, sel_1 is not needed because if $en_1 = 1$, MSE_{SH_1} and DM_1 must be replaced by MSE_k(SBC) and the corresponding SBC. For the kth 1D reference block with reference block coordinate RBC_k, the above comparison and replacement process will continue until the last pixel value in the last row of the search frame is read out. The final contents in SH_i and DM_i, $i = 1,\ldots,P$ are the MSEs and the SBCs of the P search blocks with top P smallest MSEs, respectively. Consequently, the P SBCs stored in DM_i, $i = 1,\ldots,P$ and the kth reference block coordinate (RBC_k) will be input to the subtracter, SUB, as shown in Fig. 5 to calculate the top P MV_ks corresponding to the P smallest MSEs. This constitutes the operations of PWC_k.

By the aid of PWC_k, $k = 1,\ldots,K$, we can design the IE using the following components. Except for the indicators for indicating the index of and the number of distinct MVs in the $K \times P$ MVs, we employ a counter to cyclically generate the MN for each MV, an identifier to recognize the distinct MVs, an adder to calculate the AMN for each distinct MV and a comparator to identify the MV with largest AMN. Since the circuit to interconnect the above mentioned components for implementing the IE and generating the true MV is complicated and tedious, we will present it in Appendix A.

3.4. Hardware Implementation Architecture of ESPM-1D-BMA for ME

Now, combining the parallel multi-1D blocks matching computing architecture (Figs. 3 and 4), PWC_k (Fig. 5), $k = 1,\ldots,K$, and IE (Fig. 8 in Appendix A), the hardware implementation architecture of the proposed ESPM-1D-BMA for ME can be described in Fig. 6. The solid lines and dotted lines in Fig. 6 represent the transmission of data and signals, respectively.

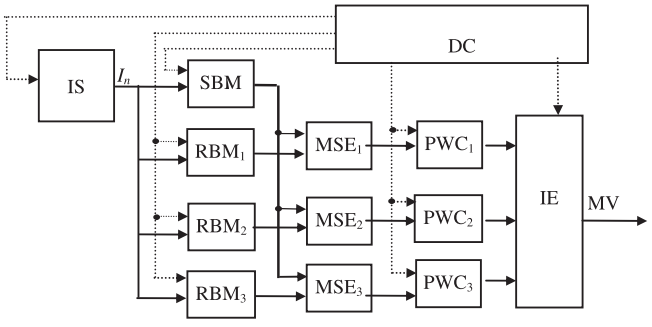


Fig. 6. The block diagram of ESPM-1D-BMA for ME.

For the sake of simplicity in illustration, we assume $K = 3$ in Fig. 6, where SBM, RBM_k and MSE_k, $k = 1,\ldots,K$, are the same as those in Fig. 4. The Image Sensor, IS, shown in Fig. 6 is used to obtain the image data of the 1D search block and the K reference 1D blocks. The purpose of Digital Controller, DC, is to control the activation timing of each unit and the synchronization of the parallel processing architecture. Therefore, the operations of the hardware implementation architecture presented in Fig. 6 can be described as follows. First of all, the image data of the K 1D reference blocks will be preloaded into RBM_k, $k = 1,\ldots,K$. The DC will send a control signal to the IS to obtain the image data of I_{n-1}, which will be input to SBM. Since MSE_k, $k = 1,\ldots,K$ are analog circuits, they will compute MSE_k(SBC), $k = 1,\ldots,K$, directly and in parallel once the data are ready at the output of both SBM and RBM_k, $k = 1,\ldots,K$. The computed MSE_k(SBC), $k = 1,\ldots,K$ will be input to PWC_k, $k = 1,\ldots,K$ controlled by the signal output from DC to determine whether it is among the P smallest MSE_ks for the kth 1D reference block. The above process will repeat until the last pixel value in the last row of the search frame is read out. Then DC will send a signal to PWC_k to output the top P MV_ks for $k = 1,\ldots,K$ in parallel. These $K \times P$ MVs will be input to IE to generate the true MV.

3.5. Time complexity

We let $T(\cdot)$ denote the computation time or propagation time of unit $(\cdot)$. In addition to $T(\cdot)$, there are other time delays need be considered such as (i) the time delay incurred from DC to guarantee a safety margin of controlling the action of the K units of same type of components in parallel and (ii) the wire delay between units. However, comparing with $T(\cdot)$, these time delays are negligible. Therefore, the estimated time needed to generate an MV of the ESPM-1D-BMA can be stated in the following:

$$(M \times N)(T_{SH} + T_{MSE} + T_{PWC}) + T_{IE} \quad (4)$$

where T_{SH} denotes the time for loading a pixel value into RBM or SBM, T_{MSE} denotes the time required in computing the MSE of 1D block matching, which is the propagation time of the circuit

Table 2
The average ME accuracies of the 5000 estimated MVs.

K	P	Football	Greens	Concord	Fabric	Hestain	Pears	Tissue	Board
2	1	91.70	98.84	89.98	96.62	98.60	81.16	98.04	98.34
	2	95.11	98.96	92.12	98.33	99.14	86.20	97.82	98.76
	3	96.04	99.00	93.18	98.89	99.30	90.16	97.92	99.00
	4	96.51	99.36	94.18	99.08	99.20	91.84	97.82	98.70
	5	97.36	99.04	94.12	99.17	98.92	91.30	98.04	98.72
4	1	98.27	99.60	94.32	99.59	99.54	93.74	98.66	98.72
	2	98.83	99.36	95.18	99.53	99.16	95.96	98.30	99.98
	3	99.08	99.44	95.40	99.59	99.04	97.24	98.28	99.70
	4	99.39	99.42	96.20	99.79	99.20	97.60	98.32	99.66
	5	99.56	99.48	96.10	99.70	99.12	97.80	98.12	99.66
6	1	99.30	99.68	96.12	99.92	99.44	96.96	98.60	100
	2	99.55	99.52	96.80	99.65	99.18	97.54	98.54	99.44
	3	99.58	99.32	97.10	99.68	99.22	98.24	98.44	99.62
	4	99.50	99.60	97.52	99.63	99.32	98.30	98.54	99.60
	5	99.57	99.68	97.96	99.62	99.24	98.32	98.52	99.72
8	1	99.63	99.82	96.98	99.94	99.48	97.70	98.88	99.98
	2	99.65	99.58	97.12	99.72	98.98	98.70	98.78	99.64
	3	99.58	99.54	97.20	99.57	98.90	99.28	98.52	99.70
	4	99.60	99.60	97.92	99.63	99.36	99.14	98.78	99.74
	5	99.69	99.46	98.28	99.54	99.24	98.98	98.58	99.70
2D-FSBMA		100	99.998	99.947	100	99.625	99.923	99.285	100

presented in Fig. 3, T_{PWC} denotes the propagation time of the PWC_k presented in Fig. 5 and T_{IE} denotes the processing time of the IE presented in Fig. 8.

4. Test results and comparisons

In this section, we will demonstrate the ME accuracy achieved by the proposed ESPM-1D-BMA in comparison with the 2D-FSBMA and some fast block matching algorithms (Li et al., 1994; Zhu et al., 2002; Zhu & Ma, 1997) using extensive simulations. We will also compare the computational efficiency of the proposed ESPM-1D-BMA with the 2D-FSBMA. We use *Matlab* and their eight pictures, *football, greens, concord, fabric, hestain, pears, tissue* and *board*, as our simulation tool and test bed, respectively. The eight tested pictures were all formatted as grey-level image in eight-bit. We set $M = 24$ and $N = 24$ for all tests, $X = 8$ and $Y = 8$ for 2D-FSBMA, and $B = 8$ for ESPM-1D-BMA. However, we will use various combinations of K and P to test the performance of ESPM-1D-BMA. For each tested picture, we arbitrarily pick an image frame of $M \times N$ pixels to serve as the reference frame I_n. For each reference frame I_n, we prepare the search frame I_{n-1} by the following procedures. We randomly generate an MV ranging from $-\frac{M-X}{2}$ to $\frac{M-X}{2}$ and from $-\frac{N-Y}{2}$ to $\frac{N-Y}{2}$ in x and y directions, respectively, and apply it to I_n to form a noise free I_{n-1}. For each pixel in the noise free I_{n-1}, we randomly generate a noisy signal based on a normal distribution with mean 0 LSB and variance 3 LSBs and add it to the pixel. The resulted frame will serve as the tested search frame I_{n-1}. For each one of the eight tested pictures, we prepare 5000 (I_{n-1}, I_n)s based on the above process.

4.1. Comparisons of ME accuracy

Now for each of the eight tested pictures, we use the prepared 5000 (I_{n-1}, I_n) s to estimate the corresponding randomly generated MVs using the proposed ESPM-1D-BMA and the 2D-FSBMA, the average accuracy of the 5000 estimated MVs for each picture and for various combinations of K and P are presented in Table 2. From Table 2, we can observe that the larger the values of K and P in the proposed ESPM-1D-BMA, the more accurate the MV estimation will be. We can also observe that when $K \geqslant 8$ and $P \geqslant 4$, the proposed algorithm is at most 1% less accurate than the 2D-FSBMA on the average.

Putting the test results of ESPM-1D-BMA for $K = 8$ and $P = 3$ presented in Table 2 in the second row of Table 3 for reference, we also use the same 5000 prepared (I_{n-1}, I_n) s to test the three fast block matching algorithms, the DS, the NTSS, the HEXBS. The average ME accuracy resulted by these three methods are presented in the last three rows of Table 3, which show that the proposed ESPM-1D-BMA is far better than the three fast block matching algorithms in ME accuracy.

4.2. Comparison of computational efficiency

Now, to evaluate the computing time of ESPM-1D-BMA and the mixed-signal approach 2D-FSBMA (MS-2D-FSBMA), we need to review the computational complexity of the latter first. Similar to the

```
Inference Engine( )
{
  % - - initialization - -
  Dmv_d = 0, d = 1,..., KP; AMN_d = 0, d = 1,..., KP;
  wMV = 0;  wAMN = 0;  D = 1;
  for (k = 1 to K) {
    for ( p = 1 to P) {
     % - - step 1 - -
      d' = D;
      smv = mv_{k,p};
      for (d = 1 to KP)
        if ( Dmv_d == smv )  d' = d;
      if ( d' == D )  new = 1;
      else  new = 0;
      % - - step 2 - -
      AMN_old = AMN_{d'};
      AMN_new = AMN_old + MN_{k,p};
      % - - step 3 - -
      if ( AMN_new > wAMN)  win = 1;
      else  win = 0;
      % - - step 4 - -
      Dmv_{d'} = smv;
      AMN_{d'} = AMN_new;
      if ( win == 1) {
        wMV = smv;
        wAMN = AMN_new;
      }
      if ( new == 1)  D = D + 1;
    }
  }
}
```

Fig. 7. The pseudo-code of IE.

$1 \times B$ current summation circuit employed in 1D block matching presented in Fig. 3, the MS-2D-FSBMA employed a $X \times Y$ current-summation circuit to obtain the 2D-MSE (Panovic & Demosthenous, 2006). Instead of PWC_k, $k = 1, \ldots, K$, the MS-2D-FSBMA need only one *single winner comparator* (SWC), which consists of a comparator, COMP, a SH, and a DM to identify the coordinate of the SB with minimum MSE. In the case of a frame with size $M \times N$, a block with size $X \times Y$, the computing time of the MS-2D-FSBMA for computing an MV can be calculated by:

$$2 \times X \times Y \times T_{SH} + (M - X)(N - Y + 1)(Y \times T_{SH} + T_{MSE_{2D}} + T_{COMP}) + (N - Y)(X \times T_{SH} + T_{MSE_{2D}} + T_{COMP}) \quad (5)$$

where T_{SH} is the same as that in (4); $T_{MSE_{2D}}$ denotes the time for performing a 2D-MSE computation; since the most time consuming

Table 3
Comparisons of ESPM-1D-BMA with the three fast block matching algorithms.

	Football	Greens	Concord	Fabric	Hestain	Pears	Tissue	Board
ESPM-1D-BMA ($K = 8$, $P = 3$)	99.58	99.54	97.20	99.57	98.90	99.28	98.52	99.70
DS	45.93	69.04	81.79	82.88	89.04	75.54	79.24	38.73
NTSS	48.69	81.13	86.87	86.22	87.89	75.01	87.69	43.99
HEXBS	38.15	63.02	73.89	67.10	74.53	62.13	74.69	28.19

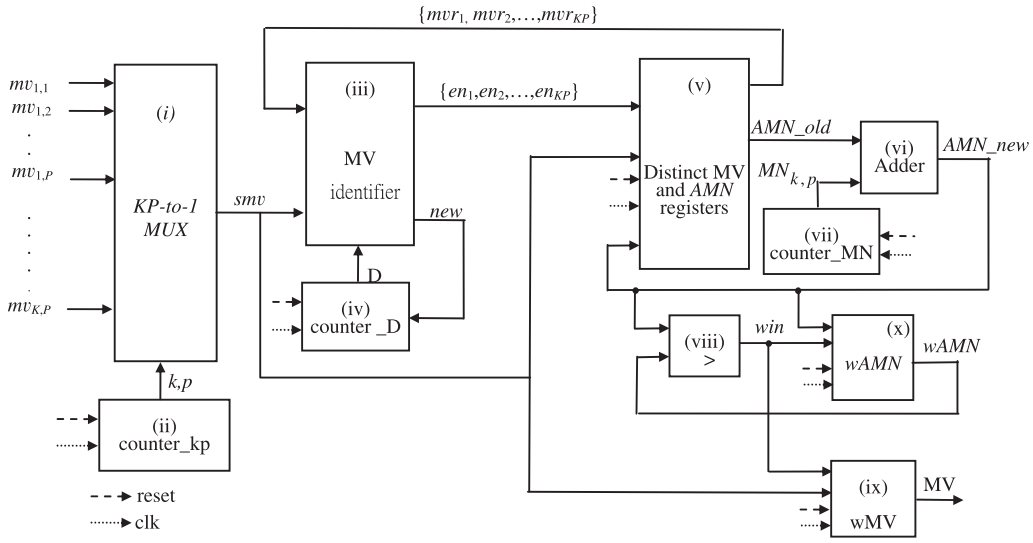


Fig. 8. Hardware architecture for implementing the IE.

component in SWC is the COMP, the time needed for comparison in SWC is T_{COMP}.

Based on the existing circuit for SH (Chen, Gu, Shen, Wu, & Hsu, 1998), MSE_{2D} (Panovic & Demosthenous, 2004) and COMP (Razavi & Wooley, 1992), we obtain the following computing time using HSPICE simulation: $T_{SH} = 50$ ns, $T_{MSE_{2D}} = 10$ ns, and $T_{COMP} = 100$ ns. Although the analog circuit for computing MSE presented in Fig. 3 is simpler than computing MSE_{2D} in MS-2D-FSBMA, the operations of all SEs in both circuits are carried out in parallel. Therefore, $T_{MSE} \cong T_{MSE_{2D}} = 10$ ns. Similarly, the operations of PWC_k (Fig. 5), $k = 1,\ldots,K$ are also carried out in parallel, and the most time consuming component in PWC_k is $COMP_k$, therefore, $T_{PWC} \cong T_{SWC} = T_{COMP} = 100$ ns. According to the hardware implementation architecture of the IE presented in Appendix A, the processing time for IE is $T_{IE} = K \times P \times T_{clock}$, where $T_{clock} = 13.1$ ns is the critical path delay of IE. Therefore, for $K = 8$, $P = 3$, we have $T_{IE} = 314.4$ ns. Consequently, based on the parameters employed in our tests, the computing time needed for an MV estimation of ESPM-1D-BMA with $K = 8$, $P = 3$ and MS-2D-FSBMA are 92474.4 ns and 153,280 ns, respectively. This demonstrates that the proposed ESPM1D-BMA uses only 60% of computing time of the MS-2D-FSBMA and achieves almost the same ME accuracy. Notably, the critical point of the computing efficiency achieved by the ESPM-1D-BMA is the small amount of time spending on loading the pixel values of the frame I_{n-1} into the SBM. The part of loading time in (4) and (5) for ESPM-1D-BMA and MS-2D-FSBMA are $M \times N \times T_{SH}$ and $2 \times X \times Y \times T_{SH} + (M - X)(N - Y + 1)(Y \times T_{SH}) + (N - Y)(X \times T_{SH})$, respectively, which constitutes 31.14% and 79.33% of the corresponding total computing time, respectively.

5. Conclusion

In this paper, we have proposed a hardware implementable ESPM-1D-BMA and demonstrated that its ME accuracy is close to the 2D-FSBMA and better than the three comparing fast block matching algorithms. Above all, the computing speed of ESPM-1D-BMA is about two times as fast as the MS-2D-FSBMA.

Appendix A

To implement the IE, we will first describe the *pseudo code* for carrying out the IE then present the hardware architecture for implementing the pseudo code. We let $mv_{k,p}$ and $MN_{k,p}$, $p = 1,\ldots,P$ denote the P MVs that correspond to the top P smallest MSEs for the kth 1D block matching and the corresponding MN, respectively; we let D denote the number of distinct MVs among the $K \times P$ $mv_{k,p}$'s and let Dmv_d, $d = 1,\ldots,D$, denote the D distinct MVs; we let AMN_ddenote the AMN of Dmv_d and let *wMV* and *wAMN* denote the resulting best-so-far MV and the corresponding AMN during the process, respectively. Based on the above notations, the pseudo code for carrying out the IE is presented in Fig. 7.

The operations of the pseudo code can be summarized in the following. In the initialization step, we reset all the variables. In step 1, we check whether the incoming MV, $mv_{k,p}$, matches any one of the stored distinct MVs, Dmv_d. If it matches, we identify the matched distinct MV and output a flag *new* = 0; otherwise, we denote the incoming MV as a new distinct MV and set *new* = 1. In step 2, we compute the AMN of the identified distinct MV. In step 3, we check whether *AMN_new* of the distinct MV identified previously greater than the *wAMN* of *wMV* and output a flag *win* = 1 if the result is positive; otherwise we set *win* = 0. In step 4, we update the distinct MV and the associated AMN and update *wMV* and *wAMN* if *win* = 1; furthermore, if the flag *new* resulted in step 1 is 1, we set $D = D + 1$ that is to increase the number of distinct MVs by 1.

The hardware implementation architecture of the pseudo code is presented in Fig. 8. In the leftest part of this architecture, we use a counter, counter_*kp* marked by (ii) in Fig. 8, to generate the index $p = 1,\ldots,P$ for each $k = 1,\ldots,K$ sequentially and use a *KP*-to-1 Multiplexer, denoted by MUX and marked by (i) in Fig. 8, to select $mv_{k,p}$ based on the generated index; this corresponds to the statement $smv = mv_{k,p}$ in step 1 of Fig. 7. The *smv* will be input to the MV identifier marked by (iii) in Fig. 8, which will compare the *smv* with the distinct MVs stored in the registers denoted by Distinct MV and AMN. The counter, counter_*D* marked by (iv) in Fig. 8, will generate the index *D*, such that if *smv* does not match any existing distinct MVs, this *smv* will be the *D*th distinct MV, and the MV identifier will set the enable signal $en_D = 1$ to activate the Distinct MV and AMN registers, marked by (v) in Fig. 8, to store the current *smv* and the corresponding AMN. In the meantime, the MV identifier also sets *new* = 1 to increase counter_*D* by 1. We will then use an adder marked by (vi) in Fig. 8 to update the AMN register for the current *smv*, which may be one of the existing distinct MVs or a new distinct MV, as follows. Add the $MN_{k,p}$ corresponding to the current *smv* to the *AMN_old* then output *AMN_new*, which will be fed back to the

Distinct MV and AMN registers. Notably, the $MN_{k,p}$ is generated by the counter_*MN* marked by (vii) in Fig. 8. Furthermore, the *AMN_new* will compare with *wAMN* through the comparator, >, marked by (viii) in Fig. 8. If *AMN_new* > *wAMN*, we set *win* = 1, which will update *wAMN* by *AMN_new* and *wMV* by *smv* as shown by the blocks marked by (x) and (ix), respectively. This completes the implementation of the pseudo code presented in Fig. 7.

We design the clock period to be long enough such that the signal $mv_{k,p}$ can travel through the critical path, which includes the propagation of counter_*kp*, *KP*-to-1 MUX, and MV identifier, the access of the Distinct MV and AMN registers, the processing time of adder and comparator, and the update of *wMV* and *wAMN*. Based on Taiwan Semiconductor Manufacturing Corporation (TSMC) 0.18 μm CMOS technology, the critical path's delay of the IE presented in Fig. 8 can be within 13.1 ns. That means we can design the clock period for IE as T_{clock} = 13.1 ns. Then, the time required for processing the IE is $T_{IE} = K \times P \times T_{clock}$. Therefore, for $K = 8$, $P = 3$, the IE can obtain the true MV within 314.4 ns.

References

Chen, M. J., Gu, Y. B., Shen, W. C., Wu, T., & Hsu, P. C. (1998). A compact high-speed Miller-capacitance based sample-and-hold circuit. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, 45*, 198–201.

Gharavi, H., & Mills, M. (1990). Block matching motion estimation algorithms new results. *IEEE Transactions on Circuits and Systems, 37*, 649–651.

Hsieh, C. H., & Lin, T. P. (1992). VLSI architecture for block-matching motion estimation algorithm. *IEEE Transactions on Circuits Systems Video Technology, 2*(2).

Liao, S. H. (2005). Expert systems methodologies and applications – a decade review from 1995 to 2004. *Expert Systems with Applications, 28*(1), 93–103.

Li, H., & Sun, J. (2009). Majority voting combination of multiple case-based reasoning for financial distress prediction. *Expert Systems with Applications, 36*(3), 4363–4373.

Li, R., Zeng, B., & Liou, M. L. (1994). A new three-step search algorithm for block motion estimation. *IEEE Transactions on Circuits Systems for Video Technology, 4*, 438–442.

Panovic, M., & Demosthenous, A. (2004). A compact block matching cell for analogue motion estimation processors. *Proceedings of 2004 IEEE International Symposium on Circuits and Systems (ISCAS'04)* (Vol. 2, pp. 229–232). Canada: Vancouver.

Panovic, M., & Demosthenous, A. (2006). Motion estimation processor using mixed-signal approach. *IEEE Transactions on Circuits and Systems II, 53*(6), 492–496.

Razavi, B., & Wooley, B. A. (1992). Design techniques for high-speed, high-resolution comparators. *IEEE Journal of Solid-State Circuits, 27*(12), 1916–1926.

Sun, J., & Li, H. (2008). Listed companies' financial distress prediction based on weighted majority voting combination of multiple classifiers. *Expert Systems with Applications, 35*(3), 818–827.

Yang, S., Wolf, W., & Vijaykrishan, N. (2005). Power and performance analysis of motion estimation based on hardware and software realizations. *IEEE Transactions on Computers, 54*, 714–726.

Zhu, S. & Ma, K.-K. (1997). A new diamond search algorithm for fast block matching motion estimation. In *IEEE international conference on communications and signal processing* (pp. 292–296).

Zhu, C., Lin, X., & Chau, L. P. (2002). Hexagon-based search pattern for fast block motion estimation. *IEEE Transactions on Circuits Systems for Video Technology, 12*, 349–355.