Fast Full-Search Block-Matching Algorithm for Motion-Compensated Video Compression

Yih-Chuan Lin and Shen-Chuan Tai

***Abstract*—This paper proposes a fast block-matching algorithm that uses three fast matching error measures, besides the conventional mean-absolute error (MAE) or mean-square error (MSE). An incoming reference block in the current frame is compared to candidate blocks within the search window using multiple matching criteria. These three fast matching error measures are established on the integral projections, having the advantages of being good block features and having simple complexity in measuring matching errors. Most of the candidate blocks can be rejected only by calculating one or more of the three fast matching error measures. The time-consuming computations of MSE or MAE are performed on only a few candidate blocks that first pass all three fast matching criteria. Simulation results show that a reduction of over 86% in computations is achieved after integrating the three fast matching criteria into the full-search algorithm, while ensuring optimal accuracy.**

I. Introduction

Motion estimation using a block-matching algorithm (BMA) is widely used in many motion-compensated video coding systems, such as those recommended by the H.261 and MPEG standards [1], [2], to remove interframe redundancy and thus achieve high data compression. In a typical BMA, the current frame of a video sequence is divided into nonoverlapping square blocks of pixels, say, of size $N \times N$. For each reference block in the current frame, BMA searches for the best matched block within a search window of size $(2W+N) \times (2W+N)$ in the previous frame, where W stands for the maximum allowed displacement. Then the relative position between the reference and its best matched block is represented as the motion vector of the reference block. A nonnegative matching error function $D_p(i,j)$ is defined over all the positions to be searched, i.e.,

$$\begin{aligned} D_p(i,j) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} &|f_t(l+x,k+y) \\ &- f_{t-1}(l+i+x,k+j+y)|^p, \\ &\quad p = 1 \text{ or } 2 \quad \text{and} \quad -W \le i; j \le W \qquad (1)\end{aligned}$$

where $f_t(l,k)$ is the reference block of its upper left pixel at the coordinate (l,k) in the current frame, and $f_{t-1}(l+i,k+j)$ is a candidate block of its upper left pixel at the coordinate $(l+i,k+j)$ in the previous frame. The computations incurred in one complete measurement of $D_p(i,j)$ are N^2 absolute values (or squarings when $p=2$) and $2N^2-1$ additions. The best matched block corresponds to the candidate block of its upper left corner located at $(l+i_*,k+j_*)$ which has the minimum matching error $D_p(i_*,j_*)$. A straightforward method of BMA is the full-search BMA (FBMA) which requires to compute the $D_p(i,j)$'s for all $(2W+1)^2$ positions of candidate blocks in the search window; that is, the FBMA needs $(2W+1)^2N^2$ absolute values (or squarings), $(2W+1)^2(2N^2-1)$ additions, and $(2W+1)^2$ comparisons for each reference block; however, it is an intensive computation process, limiting its practical applications. Many well-known fast algorithms [3]–[13] have been developed to reduce such highly computational complexity of the full-search BMA by considering only a limited number of the motion vectors in the search window at the expense of estimate accuracy. That is, only suboptimal estimate accuracy is guaranteed by these algorithms. Concerning the VLSI implementation, most of these fast algorithms, e.g., the three-step search (TSS) [3], have the drawbacks of irregular data flow and high control overhead, while the full-search BMA has the advantages of regular data flow and low control overhead [14], [15].

Recently, a number of algorithms with regard to the pattern-matching problems [16]–[19] make use of integral projections to simplify the computational complexity of the pattern-matching operation. However, all of the previous research work on motion estimation using integral projections has never provided any optimality-preserving ability like the FBMA. Integral projections are good features describing the block mean intensity and the edge location and orientation in a block of pixels, and are most likely to be different for different blocks. In this letter, a fast full-search BMA (FFBMA), which is also based on the uses of integral projections, is presented to provide much faster motion estimation than that using the traditional FBMA, while preserving the optimality of estimate accuracy. In fact, there still exist similar ideas being realized by other techniques for fast vector quantization (VQ), such as the partial distortion search (PDS) [20], the triangle-eliminating rule (TIE) [21], or VQ using mean pyramids of vectors [22]. These fast VQ algorithms converge to a common goal to reject most entries in the codebook that are not best matched to the target block using only the partial and simple information in the blocks. It is not straightforward or even difficult to extend directly these fast algorithms to the motion estimation task. For example, in [22], Lee and Chen defined a sequence of fast matching criteria, each associated with a different level of the mean pyramids of blocks, and employed these criteria in the "coarse-to-fine" manner to promote speed in searching for the nearest neighbor in a VQ

Paper approved by M. R. Civanlar, the Editor for Image Processing of the IEEE Communications Society. Manuscript received September 8, 1995; revised May 25, 1996. This work was supported in part by the National Science Council of Taiwan, R.O.C. under Contract NSC86-2221-E-006-057. This paper was presented in part at the 13th IEEE International Conference on Pattern Recognition, Vienna, Austria, August 1996.

The authors are with the Institute of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, Republic of China.

Publisher Item Identifier S 0090-6798(97)03723-9.

system; however, in trying to extend this fast hierarchical matching technique with optimality-perserving ability to the block-matching algorithm, it has to spend a large number of overhead computations to construct the mean pyramids of all the candidate blocks in the search window, and a significant amount of storage for these mean pyramids prior to the start of the block-matching process. On the other hand, using integral projections instead of the mean pyramids, the above two technical difficulties almost could be excluded completely by an efficient approach which can generate the integral projections in an on-line manner and only requiring six additions/substractions at each position of candidate block. In sum, this paper gives the optimality-preserving ability of motion estimation using integral projections, along with an efficient method for preparing the integral projections of all the candidate blocks in the search window, and shows the performance gain over that solely using all separate pixels in the blocks.

II. The Fast Full-Search Block-Matching Algorithm (FFBMA)

The basic idea behind the proposed fast full-search BMA relies on constructing three fast matching criteria and, during the period of block matching, employing these three fast matching criteria to discard the candidate blocks in the search window which are not matched to the reference block in the current frame, before using the time-consuming matching error defined in (1). These fast matching criteria are derived from the integral projections since the integral projections are simple and relevant features to a block of pixels.

Roughly speaking, the integral projections can be regarded as the intensity sums of spatial pixels along any fixed direction in a block of pixels. For any given block $f_t(l,k)$ in frame t, three kinds of integral projections are defined as follows:

1) vertical projections:

$$v_{t,(l,k)}(y) = \sum_{x=0}^{N-1} f_t(l+x, k+y), \qquad 0 \le y \le N-1 \quad (2)$$

2) horizontal projections:

$$h_{t,(l,k)}(x) = \sum_{y=0}^{N-1} f_t(l+x, k+y), \qquad 0 \le x \le N-1 \quad (3)$$

3) massive projection:

$$m_{t,(l,k)} = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f_t(l+x, k+y). \quad (4)$$

In the proposed FFBMA, the three fast matching error measures are defined as follows:

$$M_p(i,j) = |m_{t,(l,k)} - m_{t-1,(l+i,k+j)}|^p, \quad -W \le i,j \le W \quad (5)$$

$$V_p(i,j) = \sum_{y=0}^{N-1} |v_{t,(l,k)}(y) - v_{t-1,(l+i,k+j)}(y)|^p, \quad -W \le i,j \le W \quad (6)$$

$$H_p(i,j) = \sum_{x=0}^{N-1} |h_{t,(l,k)}(x) - h_{t-1,(l+i,k+j)}(x)|^p, \quad -W \le i,j \le W. \quad (7)$$

With these measures defined in (1) and (5)–(7), four different kinds of matching errors are available for each position within the search window. Obviously, (5) takes only one squaring (or absolute value) and one subtraction (or addition); as for (6) or (7), only $2N-1$ additions and N squarings (or absolute values) are sufficient. Each of these three computational complexities is relatively low in comparison with that of (1). In the following, Theorems 1 and 2 provide the relationships among the multiple matching errors on each position (i,j) within the search window .

Theorem 1: a) $M_1(i,j) \le D_1(i,j)$; b) $V_1(i,j) \le D_1(i,j)$; c) $H_1(i,j) \le D_1(i,j)$.

Theorem 2: a) $M_2(i,j) \le N^2 D_2(i,j)$; b) $V_2(i,j) \le N D_2(i,j)$; c) $H_2(i,j) \le N D_2(i,j)$.

Notice that $D_1(i,j)$ corresponds to the mean-absolute error (MAE), and $D_2(i,j)$ is the mean-square error (MSE). The validity of these two theorems can be shown according to two mathematical inequalities. They are

$$\left|\sum_{i=1}^{k} a_i\right| \le \sum_{i=1}^{k} |a_i| \quad (8)$$

$$\frac{1}{k}\left|\sum_{i=1}^{k} a_i\right|^2 \le \sum_{i=1}^{k} |a_i|^2 \quad (9)$$

where $a_i, i = 1, 2, \cdots, k$ are k arbitrary real numbers. Inequality (8) follows the well-known triangle inequality. As for inequality (9), a brief explanation is given as follows. When considering any pair of two real numbers (a_i, a_j), we have $|a_i - a_j|^2 \ge 0$ or, equivalently,

$$a_i^2 + a_j^2 \ge 2a_i a_j. \quad (10)$$

Taking summation over all the k^2 pairs on both sides of (10) yields inequality (9). Theorems 1 and 2 can be derived easily by processing (1) according to (8) and (9), respectively, where the integral projections of the error terms in (1) are formulated accordingly to form the $M_p(i,j), V_p(i,j)$, or $H_p(i,j)$. For example,

$$\begin{aligned}
D_2(i,j) &= \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} |f_t(l+x, k+y) \\
&\quad - f_{t-1}(l+i+x, k+j+y)|^2 \\
&\ge \sum_{x=0}^{N-1} \frac{1}{N} \left| \sum_{y=0}^{N-1} [f_t(l+x, k+y) \right. \\
&\quad \left. - f_{t-1}(l+i+x, k+j+y)] \right|^2 \\
&\ge \frac{1}{N} \sum_{x=0}^{N-1} |h_{t,(l,k)}(x) - h_{t-1,(l+i,k+j)}(x)|^2 \\
&= \frac{1}{N} H_2(i,j).
\end{aligned}$$

This proves c) of Theorem 2. A similar process can be employed for the remaining parts of the two theorems.

Returning to the block-matching problem, let us assume that the known current most matched motion vector (i_*, j_*) is initially set to (0,0), and the associated MSE is $D_2(i_*, j_*)$ (if the MSE criterion is used). For any other candidate block, $f_t(l+i, k+j)$ within the search window if one or more of the following conditions holds:

1) $N^2 D_2(i_*, j_*) < M_2(i,j)$
2) $N D_2(i_*, j_*) < V_2(i,j)$
3) $N D_2(i_*, j_*) < H_2(i,j)$
4) $D_2(i_*, j_*) < D_2(i,j)$.

Then, applying Theorem 2, the candidate block $f_{t-1}(l+i, k+j)$ can be rejected without calculating the time-consuming MSE measurement. In the FFBMA, for each candidate block, test conditions 1)–4) are checked sequentially; if any of the first three test conditions fails, then the candidate block should be checked further by its successive test condition; otherwise, it is rejected, and the next other candidate block is then compared. Once test condition 4) is tested and unsatisfied, both the best matched motion vector and the associated matching error should be updated. In this algorithm, all of the candidate blocks in the search window need be examined by test condition 1); this requires $(2W+1)^2$ additions, $(2W+1)^2$ squarings, and $(2W+1)^2$ comparisons. Each of the candidate blocks that fails on the check of test condition 1) should go through the test of condition 2) for further rejection, and this test using condition 2) for one block matching needs $2N-1$ additions, N squarings, and one comparison. Similarly, test condition 3) also takes $2N-1$ additions, N squarings, and one comparison for one candidate block that violates the preceding test. Finally, when all of the first three test conditions are unsatisfied with a certain candidate block, the FFBMA requires $2N^2-1$ additions, N^2 squarings, and one comparison to decide whether the best matched motion vector should be updated to the current candidate position.

For the sake of clarity, the computations required in the FFBMA for one reference block should include:

1) $(p_1+p_2+p_3+p_4)$ comparisons;
2) $[p_1+(p_2+p_3)(2N-1)+p_4(2N^2-1)]$ additions (or subtraction);
3) $[p_1+(p_2+p_3)N+p_4N^2]$ squarings (or absolute values if MAE is concerned);

where $p_1=(2W+1)^2$, and p_2, p_3, and p_4 stand for the occurrence frequencies of evaluating test conditions 2)–4), respectively, required for finding the best matched motion vector within the search window.

To evaluate the first three conditions 1)–3), the integral projections of each candidate block have to be known prior to matching. We do not have to calculate all of the integral projections for each candidate block in the previous frame. If the integral projections for the two candidate blocks $f_{t-1}(l, k-1)$ and $f_{t-1}(l-1, k)$ are known, only a few terms are updated for obtaining all of the integral projections of the block $f_{t-1}(l,k)$, i.e.,

$$\begin{aligned}
v_{t-1,(l,k)}(i) &= v_{t-1,(l,k-1)}(i+1) \quad \text{and} \\
h_{t-1,(l,k)}(i) &= h_{t-1,(l-1,k)}(i+1), \\
&\qquad \text{for} \quad i = 0,1,\cdots,N-2 \\
v_{t-1,(l,k)}(N-1) &= v_{t-1,(l-1,k)}(N-1) \\
&\quad - f_{t-1}(l-1, k+N-1) \\
&\quad + f_{t-1}(l+N-1, k+N-1) \\
h_{t-1,(l,k)}(N-1) &= h_{t-1,(l,k-1)}(N-1) \\
&\quad - f_{t-1}(l+N-1, k-1) \\
&\quad + f_{t-1}(l+N-1, k+N-1) \\
m_{t-1},(l,k) &= m_{t-1,(l,k-1)} \\
&\quad - v_{t-1,(l,k-1)}(0) \\
&\quad + v_{t-1,(l,k)}(N-1).
\end{aligned}$$

Therefore, the computations required for the integral projections of block $f_{t-1}(l,k)$ are six arithmetic operations of addition/subtraction. Suppose the frame size is $W \times H$ pixels. The integral projections of all of the blocks $f_{t-1}(0,j)$ for $j=0,1,\cdots,W-N$ and $f_{t-1}(i,0)$ for $i=1,2,\cdots,H-N$ in the considered frame $t-1$ are first calculated. This precomputation needs $[(3N+1)(H+W)-(4N^2+3N+1)]$ additions/subtractions. To calculate the integral projections of the remaining blocks in the frame, we need $6(H-N)(W-N)$ additions/subtractions since each of the remaining blocks requires six additions/subtractions and there are $(H-N)(W-N)$ blocks remaining. Since the integral projections can convey the most information in a block of pixels and the arithmetic operations required for calculating the three fast matching errors are both much fewer and simpler than those for MSE, a great deal of computations or number of MSE (or MAE) measurements are thus saved. The next section shows several experiments to demonstrate the effectiveness of using integral projections to speed up the FBMA.

III. Simulation Results

The efficiency of the proposed algorithm was tested by using two benchmark video sequences, *Salesman* and *Flower Garden*. We first used 60 consecutive frames of size 360 × 288 pixels in *Salesman* and 60 consecutive frames of size 360 × 240 pixels in *Flower Garden*. The block and search window sizes were fixed at 16 × 16 and 33 × 33, respectively. Thus, the traditional FBMA requires computing 1089 MSE or MAE measurements for each reference block in the current frame. In addition to the FBMA and FFBMA, we also implemented a partial distortion search block-matching algorithm (PDSBMA) that involves the partial distortion search (PDS) technique to speed up the block-matching process. In the PDSBMA, the matching process with a certain candidate block accomplishes the distortion measurement by accumulating the individual error terms of block elements one at a time, and can be quit partially without completing the accumulation of the full absolute difference, that is, to check if the accumulation thus far had already exceeded the distortion to the best match; if so, there is no need to continue the accumulation. Table I shows

TABLE I
COMPARISON OF THE COMPUTATIONAL COMPLEXITY FOR VARIOUS BLOCK-MATCHING ALGORITHMS WITH THE MSE CRITERION ACCORDING TO THE NUMBERS OF THE ARITHMETIC OPERATIONS REQUIRED FOR EACH 16 × 16 REFERENCE BLOCK

Test Sequences	Methods	Squarings	+/-	Compares	Total
Salesman	FFBMA	8,817	22,731	1,437	32,985
	PDSBMA	59,964	119,928	59,964	239,856
Flower Garden	FFBMA	32,865	70,434	1,830	105,129
	PDSBMA	69,559	139,118	69,559	278,236
FBMA		278,784	556,479	1,089	836,352

TABLE II
COMPARISON OF THE COMPUTATIONAL COMPLEXITY FOR VARIOUS BLOCK-MATCHING ALGORITHMS WITH THE MAE CRITERION ACCORDING TO THE NUMBERS OF THE ARITHMETIC OPERATIONS REQUIRED FOR EACH 16 × 16 REFERENCE BLOCK

Test Sequences	Methods	$\vert . \vert$	+/-	Compares	Total
Salesman	FFBMA	7,393	19,957	1,363	28,713
	PDSBMA	62,518	125,036	62,518	250,072
Flower Garden	FFBMA	22,449	49,803	1,629	73,881
	PDSBMA	72,242	144,484	72,242	288,968
FBMA		278,784	556,479	1,089	836,352

TABLE III
COMPARISON OF THE COMPUTATION COMPLEXITY FOR VARIOUS ALGORITHMS WITH MSE WORKING ON 60 *FLOWER GARDEN* FRAMES OF 720 × 480 SIZE ACCORDING TO THE NUMBERS OF ARITHMETIC OPERATIONS REQUIRED FOR EACH 16 × 16 REFERENCE BLOCK

Methods	Squarings	+/-	Compares	Total
FFBMA	43,457	·91,541	1,907	136,905
PDSBMA	110,570	221,140	110,570	442,280
FBMA	278,784	556,479	1,089	836,352

the averaged numbers of the various arithmetic operations, including squarings, additions/subtractions, and comparisons, required by the three considered algorithms, respectively, for the two test sequences. As can be seen from this table, the FFBMA can achieve over 96% reduction of computation complexity compared to the FBMA and 86% compared to the PDSBMA in terms of the total number of arithmetic operations. Table II shows similar results when the MAE measure is considered. As shown in this table, over 96 and 88% of the total arithmetic operations, respectively, in the FBMA and the PDSBMA are also saved by the FFBMA. In these two tables, it is clearly indicated that in comparing between the *Salesman* and *Flower Garden* sequences, more computation complexity is needed in both the FFBMA and the PDSBMA for the *Flower Garden* sequence. This increase of computation complexity is mainly due to the abrupt scene changes in *Flower Garden*, which could make the current known minimum matching error $D_p(i_*, j_*)$ be larger during the period of block matching. Obviously, this larger $D_p(i_*, j_*)$ does reduce the rejection rate of candidate blocks in the FFBMA and PDSBMA. Therefore, the reduction in computations of the FFBMA is dependent on the sequence envisaged. The more significant the motion of objects in the sequence, the less reduction of complexity the FFBMA exhibits.

TABLE IV
COMPARISON OF THE COMPUTATION COMPLEXITY FOR VARIOUS ALGORITHMS WITH MAE WORKING ON 60 *FLOWER GARDEN* FRAMES OF 720 × 480 SIZE ACCORDING TO THE NUMBERS OF ARITHMETIC OPERATIONS REQUIRED FOR EACH 16 × 16 REFERENCE BLOCK

Methods	$\vert . \vert$	+/-	Compares	Total
FFBMA	34,721	74,240	1,736	110,697
PDSBMA	144,522	289,044	144,522	578,088
FBMA	278,784	556,479	1,089	836,352

As for the higher resolution sequences, we also have done an experiment on the *Flower Garden* sequence with a size of 720 × 480 pixels. This *Flower Garden* of higher resolution consists of 60 frames which are all the finer sampling versions of those in the preceding experiments. For this higher resolution sequence, Tables III and IV compare the computation complexities of the three algorithms with the MSE and MAE, respectively. Notice that the block and search window sizes were also set to 16 × 16 and 33 × 33, respectively. Referring to Tables III and IV, we can observe that the computational performance gain of the FFBMA for the higher resolution of *Flower Garden* is less than that for the lower resolution one. This is because, for the same scene, the finer sampling version could result in the fact that most candidate blocks' matching errors within the search window are close to each other, especially for those smooth parts.

From these tables, we can find that the FFBMA exhibits a larger reduction of arithmetic operations for MAE as compared to the MSE. This is because the MAE criterion inherently can offer more rejection ratios of candidate blocks than the MSE criterion. To explain this fact, we show a 2-D example as follows. Assume that the 2-D vector (1,0) is the best matched vector found thus far to the origin (0,0). The minimum MSE and MAE values are both set to 1. Considering another candidate vector (0.5,0.6) which is not closer to the origin vector than the vector (1,0) according to either the MSE or MAE criteria, the MAE criterion can certainly reject this candidate vector by using the massive projection, that is, $|(0.5 + 0.6) - 0| > 1$; however, in the MSE case, the vector (0.5,0.6) cannot be rejected by means of the massive projection because $|(0.5 + 0.6) - 0|^2 < 2^*1$. This example shows that in the FFBMA, the MAE is superior to the MSE in the reduction of arithmetic operations.

When comparing to the suboptimal algorithms, e.g., Liu and Zaccarin's subsampled motion-field estimation algorithm [10] that reduces the complexity of the FBMA by a fixed factor of 8 at the expense of estimation accuracy, the FFBMA with MAE can provide a greater computation reduction up to a factor of 29. Although the computation complexity of the FFBMA is dependent on the input sequence, for the worst case in Table IV, a comparable computation reduction factor of about 7.5 can be achieved by the FFBMA. In [17], Fok and Au proposed a feature domain BMA that can offer a computation reduction factor of about $0.9N$ for the search block size of $N \times N$. This algorithm is also suboptimal due to the employment of the integral projection features. For the search block size of 16 × 16, the FFBMA with MAE produces a computation reduction over twofold better than Fok and Au's algorithm for the kind of sequences like *Salesman*. As for the sequence of *Flower Garden*, a smaller computation reduction is achieved by the FFBMA as compared to Fok and Au's method.

These results verify the efficiency of the proposed FFBMA with either MSE and MAE. With the experiments, it is concluded that the FFBMA, which uses the three fast matching criteria, can perform much faster than the FBMA and the PDSBMA at the same estimate accuracy.

IV. Conclusions

A new fast full-search block-matching algorithm is presented in this paper. It runs much faster than the traditional full-search BMA, while the optimal accuracy of motion estimation is guaranteed. This improvement of speed is based on the fact that multiple matching errors which have different levels of computation complexity are available on each position to be searched. The relationships among the multiple matching errors of a candidate position are utilized to construct three test conditions which can be employed during block matching to avoid the time-consuming computations of MSE or MAE measurements. With the experiments, the proposed method can give a great amount of savings of computations, and thus can be well suited for a wide range of applications, such as videotelephony, videoconferencing, and HDTV.

References

[1] CCITT SG XV, "Recommendation H.261—Video codec for audiovisual services at p*64 kbits/s," Tech. Rep. COM XV-R37-E, Aug. 1990.
[2] MPEG, "ISO CD 11172-2: Coding of moving pictures and associated audio for digital storage media at up to about 1.5 M bits/s," Nov. 1991.
[3] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," in *Proc. Nat. Telecommun. Conf.* New Orleans, LA, Nov. 1981, pp. G5.3.1–G5.3.5.
[4] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Trans. Commun.*, vol. COM-29, pp. 1799–1808, Dec. 1981.
[5] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. Commun.*, vol. COM-33, pp. 888–896, Aug. 1985.
[6] S. Kapagantula and K. R. Rao, "Motion predictive interframe coding," *IEEE Trans. Commun.*, vol. COM-33, pp. 1011–1015, Sept. 1985.
[7] A. Puri, H. M. Hang, and D. L. Schilling, "An efficient block-matching algorithm for motion-compensated coding," in *Proc. Int. Conf. Acoust., Speech, Signal Processing*, 1987, pp. 25.4.1–25.4.4.
[8] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Trans. Commun.*, vol. 38, pp. 950–953, July 1990.
[9] C. H. Hsieh, P. C. Lu, J. S. Shyn, and E. H. Lu, "Motion estimation algorithm using interblock correlation," *Electron. Lett.*, vol. 26, pp. 276–277, Mar. 1990.
[10] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, pp. 148–157, Apr. 1993.
[11] L. W. Lee, J. F. Wang, J. Y. Lee, and J. D. Shie, "Dynamic search-window adjustment and interlaced search for block-matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, pp. 85–87, Feb. 1993.
[12] M. J. Chen, L. G. Chen, and T. D. Chiueh, "One-dimensional full search motion estimation algorithm for video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, pp. 504–509, Oct. 1994.
[13] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: A review and a new contribution," *Proc. IEEE*, vol. 83, pp. 858–876, June 1995.
[14] K. M. Yang, M. T. Sun, and L. Wu, "A family of VLSI designs for the motion compensation block-matching algorithm," *IEEE Trans. Circuits Syst.*, vol. 36, pp. 1317–1325, Oct. 1989.
[15] H. M. Jong, L. G. Chen, and T. D. Chiueh, "Parallel architectures for 3-step hierachical search block-matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*. vol. 4, pp. 407–416, Aug. 1994.
[16] J. S. Kim and R. H. Park, "A fast feature-based block matching algorithm using integral projections," *IEEE J. Select. Areas Commun.*, vol. 10, pp. 968–971, June 1992.
[17] Y. H. Fok and O. C. Au, "An improved fast feature-based block motion estimation," in *Proc. IEEE 1994 Int. Conf. Image Processing*, 1994, pp. 741–745.
[18] H. B. Park and C. Wang, "Image compression by vector quantization of projection data," *IEICE Trans. Inform. Syst.*, vol. E75-D, pp. 148–155, Jan. 1992.
[19] K. H. Jung and C. Wang, "Projective image representation and its application to image compression," *IEICE Trans. Inform. Syst.*, vol. E79-D, pp. 136–142, Feb. 1996.
[20] C. Bei and R. M. Gray, "An improvement of the minimum distortion encoding algorithm for vector quantization," *IEEE Trans. Commun.*, vol. COM-33, pp. 1132–1133, Oct. 1985.
[21] S. H. Huang and S. H. Chen, "Fast encoding algorithm for VQ-based image coding," *Electron. Lett.*, vol. 26, pp. 1618–1619, Sept. 1990.
[22] C. H. Lee and L. H. Chen, "A fast search algorithm for vector quantization using mean pyramids of codewords," *IEEE Trans. Commun.*, vol. 43, pp. 1697–1702, Feb./Mar./Apr. 1995.