Express Letters

A Novel Four-Step Search Algorithm for Fast Block Motion Estimation

Lai-Man Po and Wing-Chung Ma

Abstract—**Based on the real world image sequence's characteristic of center-biased motion vector distribution, a new four-step search (4SS) algorithm with center-biased checking point pattern for fast block motion estimation is proposed in this paper. Halfway-stop technique is employed in the new algorithm with searching steps of 2 to 4 and the total number of checking points is varied from 17 to 27. Simulation results show that the proposed 4SS performs better than the well-known three-step search and has similar performance to the new three-step search (N3SS) in terms of motion compensation errors. In addition, the 4SS also reduces the worst-case computational requirement from 33 to 27 search points and the average computational requirement from 21 to 19 search points as compared with N3SS.**

I. Introduction

In recent years, several video compression standards had been proposed for different applications such as CCITT H.261 [2], MPEG-1 [3], and MPEG-2 [4]. One common feature of these standards is that they use DCT transform coding to reduce spatial redundancy and block motion estimation/compensation to reduce the temporal redundancy. In addition, the encoders complexity of these video standards are dominated by the motion estimation, if full search (FS) is used as the block matching algorithm (BMA). FS matches all possible displaced candidate blocks within the search area in the reference frame, in order to find the block with the minimum distortion. Massive computation is, therefore, required in the implementation of FS. On the other hand, the H.261 and MPEG standards have not specified the BMA for motion estimation at the encoder. As a result, many fast BMA's [5]–[12] had been developed to alleviate the heavy computations of FS. For example, the three-step search (3SS) [5], the 2-D-logarithm search (LOGS) [6], the conjugate directional search [7], [8], the cross-search algorithm [9], and the dynamic search-window adjustment algorithm [10], etc. Among the proposed BMA's, the 3SS became the most popular one and it is also recommended by RM8 of H.261 and SM3 of MPEG owing to its simplicity and effectiveness. However, the 3SS uses a uniformly allocated checking point pattern in its first step, which becomes inefficient for the estimation of small motions. In addition, experimental results [1] show that the block motion field of real world image sequence is usually gentle, smooth, and varies slowly. It results in a center-biased global minimum motion vector distribution instead of a uniform distribution. This can be observed from the motion vector distribution based on the FS algorithm for the two well-known test sequences "Football" and "Tennis" in Fig. 1(a) and (b). For the "Football" sequence, nearly 80% of the blocks can be regarded as stationary or quasi-stationary blocks and most of the motion vectors are enclosed in the central 5 × 5 area. For the "Tennis" sequence, which consists of faster motion and camera zooming, the motion vector distribution is still highly center-biased. Although, it is more diverse as compared to "Football" sequence.

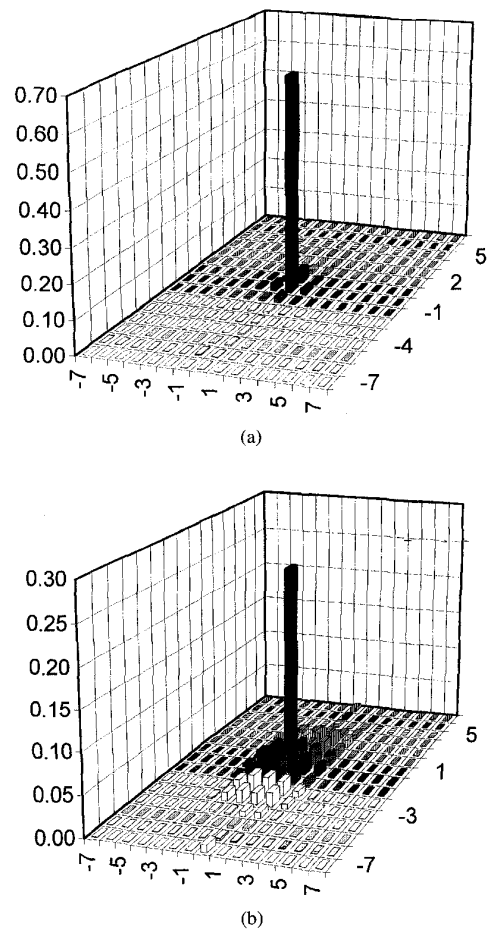


Fig. 1. Motion vector distribution for (a) "Football" and (b) "Tennis" sequences.

Based on the characteristic of center-biased motion vector distribution, an N3SS algorithm for improving the performance of the 3SS on the estimation of small motions was proposed in [1]. The N3SS employed a center-biased checking point pattern in the first step that combined the original checking point used in 3SS and eight extra neighboring points of the search window center. The N3SS use a halfway-stop technique to speed up the stationary or quasi-stationary blocks matching. Simulation results in [1] show that the N3SS is much more robust, and produces smaller motion compensation errors as compared with the 3SS. For the maximum motion displacements of ±7, however, the N3SS algorithm in the worst case requires 33 block matches while 3SS needs only 25 block matches. For some image sequences with a lot of large motions, the computational requirement of N3SS may be higher than

Manuscript received July 17, 1995; revised October 10, 1995. This paper was recommended by Associate Editor J. Brailean.

The authors are with the Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong.

Publisher Item Identifier S 1051-8215(96)03919-5.

1051–8215/96$05.00 © 1996 IEEE

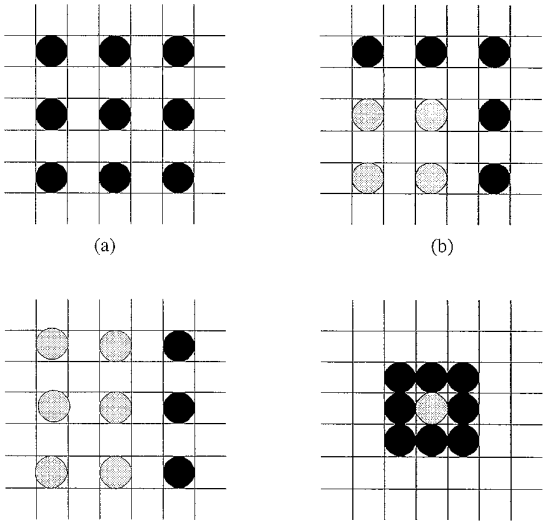


Fig. 2. Search patterns of the 4SS. (a) First step, (b) second/third step, (c) second/third step, and (d) fourth step.

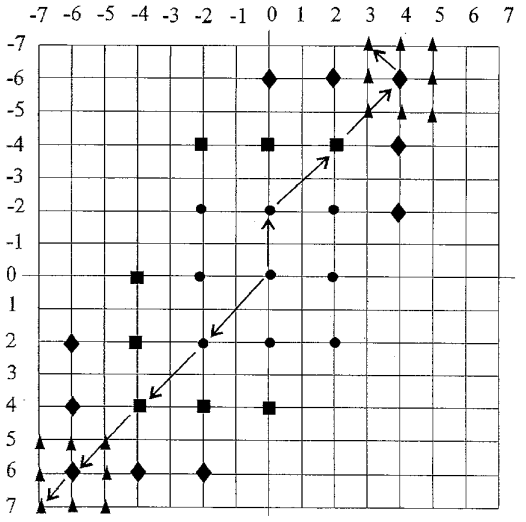


Fig. 3. Two different search paths of 4SS.

the 3SS. In addition, for the real-time or VLSI implementation of motion estimation, the worst case computational requirement should be considered instead of the average computation. In this paper, a new four-step search (4SS) algorithm based on the center-biased motion vector distribution characteristic is proposed. The proposed 4SS produces better performance than the 3SS and has similar performance as compared with the N3SS. While the worst case computational requirement of the 4SS is 27 block matches, which is only just two more block matches as compared with the 3SS.

II. Four-Step Search Algorithm

For the maximum motion displacements of ± 7, the proposed 4SS algorithm utilizes a center-biased search pattern with nine checking points on a 5×5 window in the first step instead of a 9×9 window in the 3SS. The center of the search window is then shifted to the point with minimum block distortion measure (BDM). The search window size of the next two steps depended on the location of the minimum BDM points. If the minimum BDM point is found at the center of the search window, the search will go to the final step (Step 4) with 3×3 search window. Otherwise, the search window size is maintained in 5×5 for step 2 or step 3. In the final step, the search window is reduced to 3×3 and the search stops at this small search window. The 4SS algorithm is summarized as follows:

Step 1: A minimum BDM point is found from a nine-checking-points pattern on a 5×5 window located at the center of the 15×15 searching area as shown in Fig. 2(a). If the minimum BDM point is found at the center of the search window, go to Step 4; otherwise go to Step 2.

Step 2: The search window size is maintained in 5×5. However, the search pattern will depend on the position of the previous minimum BDM point.

a) If the previous minimum BDM point is located at the corner of the previous search window, five additional checking points as shown in Fig. 2(b) are used.

b) If the previous minimum BDM point is located at the middle of horizontal or vertical axis of the previous search window, three additional checking points as shown in Fig. 2(c) are used.

If the minimum BDM point is found at the center of the search window, go to Step 4; otherwise go to Step 3.

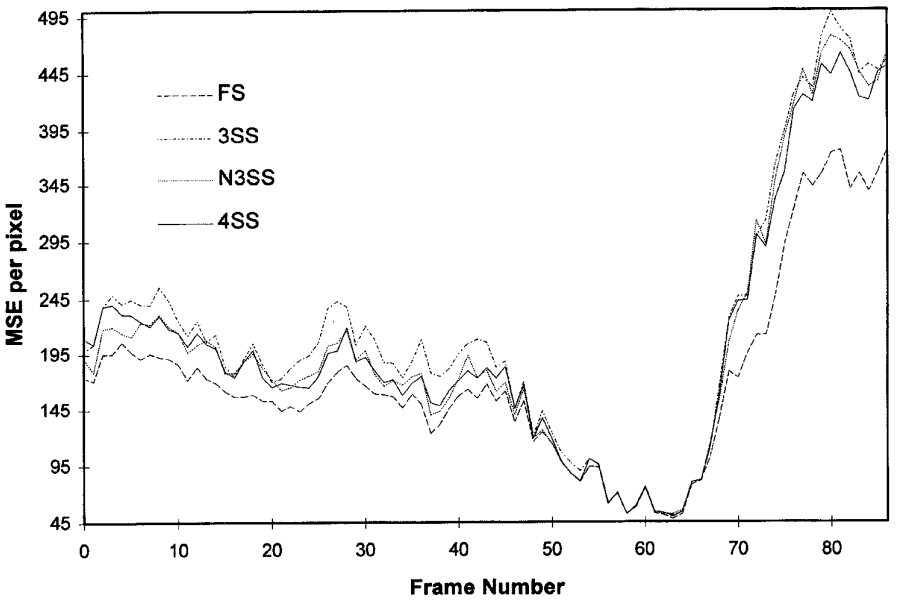
Step 3: The searching pattern strategy is the same as Step 2, but finally it will go to Step 4.

Step 4: The search window is reduced to 3×3 as shown in Fig. 2(d) and the direction of the overall motion vector is considered as the minimum BDM point among these nine searching points.

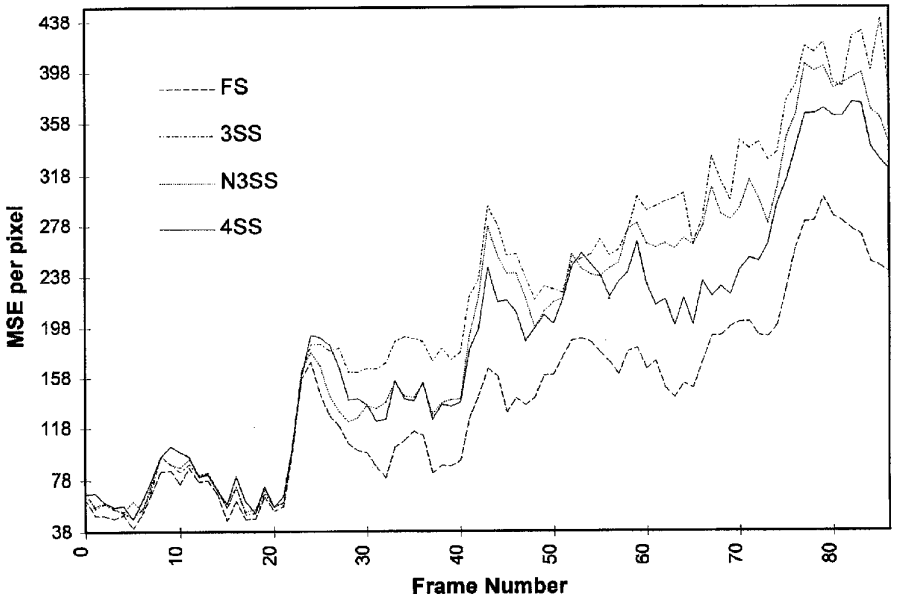
From the algorithm, we can find that the intermediate steps of the 4SS may be skipped and then jumped to the final step with a 3×3 window if at any time the minimum BDM point is located at the center of the search window. Based on this four-step search pattern, we can cover the whole 15×15 displacement window even only small search windows, 5×5 and 3×3, are used. There are overlapped checking points on the 5×5 search windows in the step 2 and step 3, thus the total number of checking points is varied from $(9 + 8) = 17$ to $(9 + 5 + 5 + 8) = 27$. The worst case computational requirement of the 4SS is 27 block matches, it will happen for the estimation of large movement. Two examples of 4SS are shown in Fig. 3 with different search paths. For the upper search path, a total of 25 checking points are used with the estimated motion vector equal to $(3, -7)$. For the worst case example of the lower search path, the number of checking points required is 27 and the estimated motion vector is equal to $(-7, 7)$. We can find that the computational complexity of 4SS is only two block matches more than the 3SS while six block matches less than the N3SS in the worst case.

III. Simulation Results

The 4SS algorithm is simulated using the luminance component of the first 90 frames of the "Football" and "Tennis" sequences. These two sequences consist of large displacement and fast motion. In the "Tennis" sequence, camera zooming and panning are also involved. The size of each individual frame is 352×240 pixels quantized uniformly to 8 bits. The mean absolute error (MAE) distortion function is used as the BDM. The maximum displacement in the search area is ± 7 pixels in both the horizontal and the vertical directions for 16×16 block size. The performance comparison of 4SS, N3SS, 3SS, and FS in terms of mean-square error (MSE) between the estimated frames and the original frames are shown in Fig. 4(a) and (b). The MSE comparisons show that the 4SS almost



(a)



(b)

Fig. 4. MSE comparisons of 4SS, N3SS, 3SS, and FS for (a) "Football" and (b) "Tennis" sequences.

TABLE I
AVERAGE MSE OF THE FIRST 90 FRAMES

Searching Algorithm	Football	Tennis
FS	175.74	143.99
3SS	219.26	221.97
N3SS	206.72	203.05
4SS	205.99	189.39

TABLE II
AVERAGE SEARCH POINTS PER MOTION VECTOR ESTIMATION FOR THE FIRST 90 FRAMES

Searching Algorithm	Football	Tennis
FS	225	225
3SS	25	25
N3SS	19.76	22.66
4SS	18.27	19.87

always provides better performance than the 3SS and similar to the N3SS, especially in the area where fast motion is involved.

Tables I to IV give some statistical performance comparisons of the four BMA's. Table I shows the average MSE of the first 90 frames using different algorithms. From this table, we can find that 4SS gives the smallest MSE among the three fast algorithms. The speed-up ratio of the BMA's is compared by the average search points

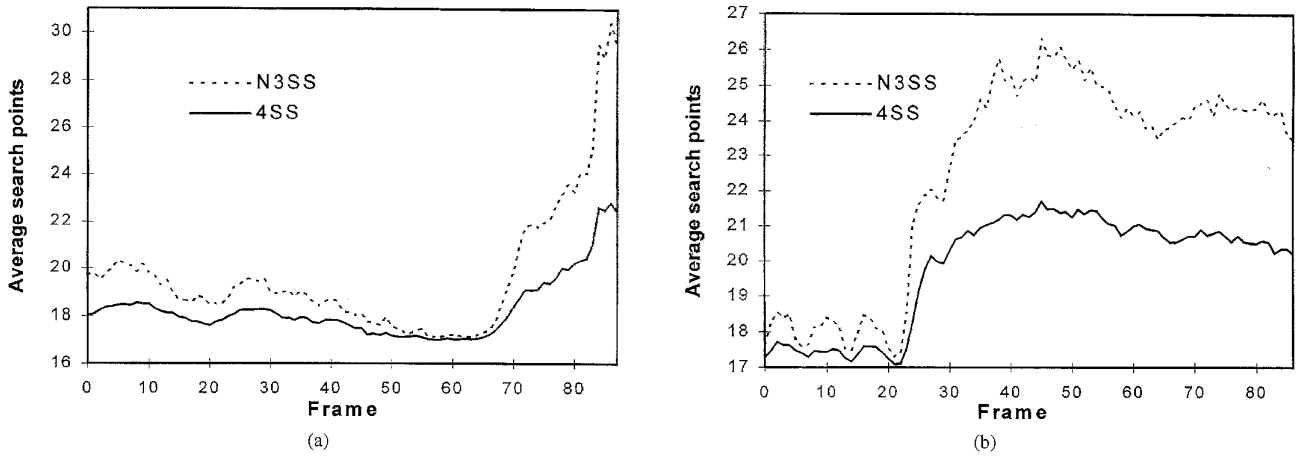


Fig. 5. Comparison of N3SS and 4SS on average search points per motion vector estimation versus frame number for (a) "Football" and (b) "Tennis" sequences.

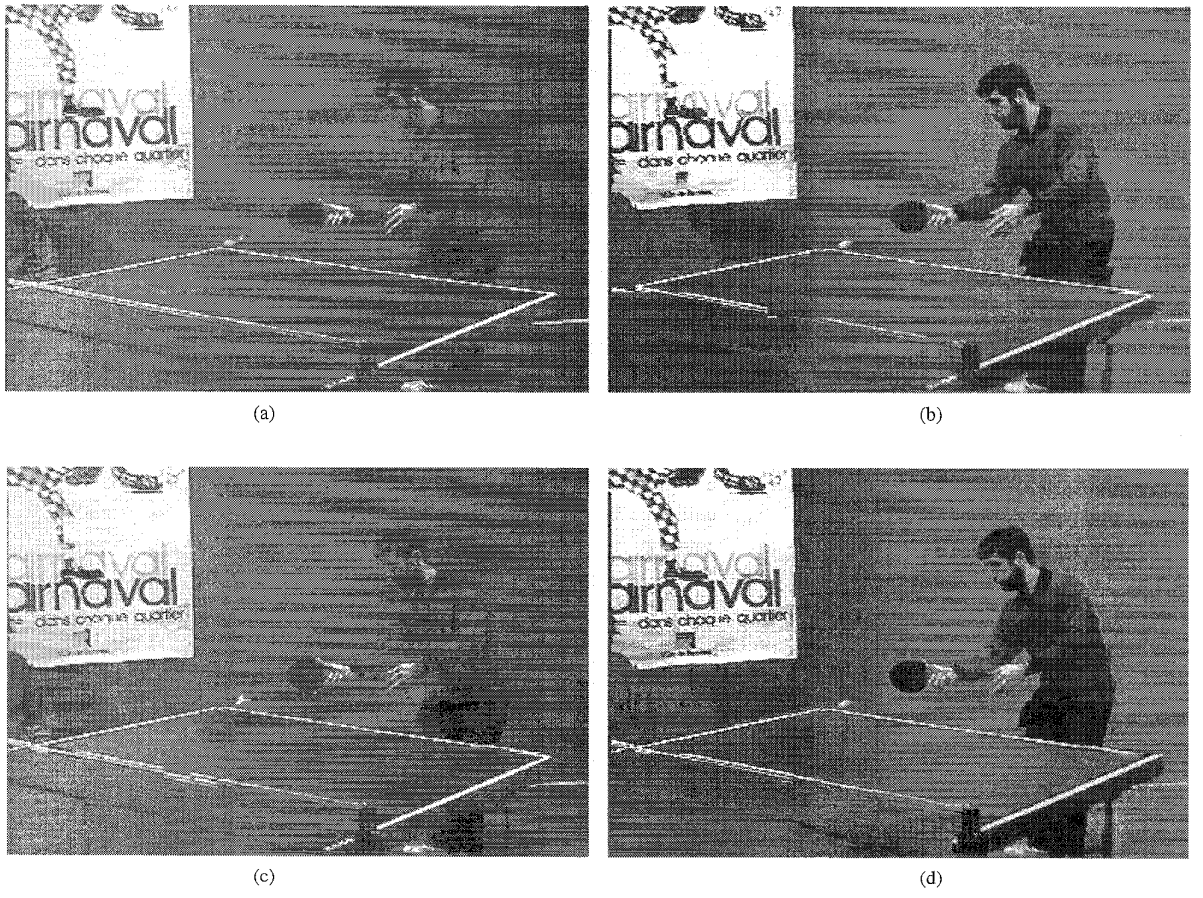


Fig. 6. The 80th estimated frames for the "Tennis" sequence using difference searching algorithms. Estimated frames using (a) FS, (b) 3SS, (c) N3SS, and (d) 4SS.

required for a motion vector estimation. The average search points per motion vector estimation for the first 90 frames using FS, 3SS, N3SS, and 4SS algorithms are shown in Table II. In addition, the average search points required by N3SS and 4SS versus the frame number for the "Football" and "Tennis" sequences are also shown in Fig. 5(a) and (b), respectively. These figures show the average search points required by 4SS for each frame is always less than 25 and much less than N3SS for the frames with fast or large motions. From Table II and Fig. 5, we can find that the search points needed by 4SS is very near to its minimum 17 but the search points needed by N3SS is highly dependant on the motion content of the image sequences. Table II also shows that 4SS out-performed other algorithms whether the image sequence contained fast or slow motion.

Table III gives us a measure of the accuracy of different algorithms using the motion vectors found by FS as reference. The N3SS and 4SS give similar accuracy and they are much better than the conventional

TABLE III
AVERAGE DISTANCE TO THE MOTION VECTOR FOUND BY THE FS ALGORITHM

Searching Algorithm	Football	Tennis
3SS	0.3971	1.6833
N3SS	0.3299	0.9657
4SS	0.3329	1.0022

TABLE IV
THE PROBABILITY TO FIND THE TRUE MOTION VECTOR W.R.T. THE FS ALGORITHM

Searching Algorithm	Football	Tennis
3SS	0.8710	0.6214
N3SS	0.8884	0.7095
4SS	0.8900	0.7203

3SS. Using the motion vector found by FS as the true motion vector, Table IV shows the probability of finding the true motion vector using different fast BMA's. Again, the performance of N3SS and 4SS is similar and they are much better than 3SS. Fig. 6 shows the estimated 80th frames of FS, 3SS, N3SS, and 4SS of the "Tennis" sequence. In terms of subjective image quality, the performance of 4SS is very close to FS and better than 3SS and similar to N3SS.

IV. CONCLUSION

Based on the center-biased global minimum motion vector distribution characteristic of real world image sequence, a new 4SS algorithm for fast block-based motion estimation is proposed in this paper. Experimental results show that the proposed 4SS algorithm performs better than the well-known 3SS and have similar performance to the N3SS in terms of mean-square error measure with smaller computational requirement. In addition, 4SS is more robust as compared with 3SS and N3SS. It is because the performance of 4SS is maintained for image sequence that contains complex movement such as camera zooming and fast motion. On the other hand, the 4SS also possesses the regularity and simplicity of hardware-oriented features.

REFERENCES

[1] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, no. 4, pp. 438–442, Aug. 1994.
[2] CCITT SGXV, "Description of reference model 8 (RM8)," Document 525, Working Party XV/4, Specialists Group on Coding for Visual Telephony, June 1989.
[3] ISO/IEC 11172-2 (MPEG-1 Video), "Information technology—Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s: Video," 1993.
[4] ISO/IEC 13818-2 | ITU-T H.262 (MPEG-2 Video), "Information technology—Generic coding of moving pictures and associated audio information: Video," 1995.
[5] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in *Proc. NTC81*, New Orleans, LA, Nov. 1981, pp. C9.6.1–9.6.5.
[6] 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.
[7] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. Commun.*, vol. COM-33, pp. 888–896, Aug. 1985.
[8] S. Kappagantula and K. R. Rao, "Motion compensated interframe image prediction," *IEEE Trans. Commun.*, vol. COM-33, pp. 1011–1015, Sept. 1985.
[9] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Trans. Commun.*, vol. 38, no. 7, pp. 950–953, July 1990.
[10] 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.
[11] S. C. Kwatra, C.-M. Lin, and W. A. Whyte, "An adaptive algorithm for motion compensated color image coding," *IEEE Trans. Commun.*, vol. COM-35, pp. 747–754, July 1987.
[12] B. Liu and A. Zaccarin, "New fast algorithm for estimation of block motion vectors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, pp. 148–157, Apr. 1993.

Adaptive Interpolation Technique for Scanning Rate Conversion

Chung J. Kuo, Ching Liao, and Ching C. Lin

***Abstract*— An adaptive technique for scanning rate conversion and interpolation is proposed in this paper. This technique performs better than the edge-based line average algorithm, especially for an image with more horizontal edges. Moreover, it is easy to implement and a simple VLSI architecture is proposed in this paper. Computer simulation shows that a 37.0 dB image can be obtained via our proposed technique, while edge-based line average algorithm only achieves 35.2 dB.**

I. INTRODUCTION

It is well known that current TV systems suffer from the uncomfortable visual artifacts such as edge flicker, interline flicker, and line crawling due to the inherent nature of the interlaced scanning process. Although as the technology progresses, wide screen (16:9) TV with 1050 scanning lines is available and the high definition TV (HDTV) is just at the corner, the current National Television Systems Committee (NTSC) system (with 512 scanning lines) will still exist for a long time before being replaced. Therefore, how the NTSC signals can be displayed on a HDTV system becomes a problem. In order to solve this problem, scanning rate conversion or interpolation is thus important and has been widely studied [1]–[4]. To have a video sequence with better quality, the problem above with consideration in motion rendition artifacts is also solved in [5] and [6].

The most famous scanning rate conversion technique is the edge-based line average (ELA) algorithm [1] which uses the directional correlations between pixels to interpolate a missing line linearly between two adjacent lines in the interlaced signal before displaying. The ELA algorithm is used widely because it requires simple computation and can be easily implemented in hardware. Though the ELA algorithm performs very well for most images, it is inefficient for images with horizontal edges. Therefore, the ELA algorithm is usually combined with the line doubling method for performance improvement [4].

In this paper, based on the ELA algorithm, we propose an adaptive ELA technique for scanning rate conversion and interpolation. It is known that the ELA algorithm is inadequate for horizontal edges;

Manuscript received August 15, 1995; revised November 5, 1995. This paper was recommended by Associate Editor Y. Wang. This work was supported in part by National Science Council, Republic of China, under contract NSC-84-2213-E-194-016.

The authors are with the Department of Electrical Engineering, National Chung Cheng University Chiayi, Taiwan, 62107 R.O.C.

Publisher Item Identifier S 1051-8215(96)02252-5.

1051–8215/96$05.00 © 1996 IEEE