Theorem 5: EXTENSION(F_8^∞) and EXTENSION(F_7^∞) are polynomial-time solvable.

V. Conclusion

We have analyzed the complexity of the consistency problem for several important Post classes, which have shown to be relevant in the study of reliability of control systems. Additionally, we discussed a possible application of the considered classes to diagnosis of breast cancer. The positive result obtained here is that if μ is known *a priori* and fixed, which would reflect the case in practice, the consistency problem in classes F_8^μ, F_4^μ, F_7^μ and F_3^μ is solvable in polynomial time. In the context of breast cancer diagnosis, the parameter μ would reflect the user-settable degree of required reliability for the rules being inferred. The higher the value of μ, the more strict this requirement becomes.

Additionally, the consistency problem for classes F_8^∞, F_4^∞, F_7^∞ and F_3^∞ is also polynomial time solvable. As part of future work, it would be worthwhile to consider best-fit extensions in the above classes of Boolean functions, especially for high values of μ, when a fewer number of possible rules is expected.

References

[1] D. Angluin, "Learning regular sets from queries and counterexamples," *Inform. Comput.*, vol. 75, no. 2, pp. 87–106, 1987.

[2] M. Anthony and N. Biggs, *Computational Learning Theory*. Cambridge, U.K.: Cambridge Univ. Press, 1992.

[3] E. Boros, T. Ibaraki, and K. Makino, "Error-free and best-fit extensions of partially defined Boolean functions," *Inform. Comput.*, vol. 140, pp. 254–283, 1998.

[4] M. R. Garey and D. S. Johnson, *Computers and Intractability*. San Francisco, CA: Freeman, 1979.

[5] B. Kovalerchuk, E. Triantaphyllou, and J. F. Ruiz, "Monotonicity and logical analysis of data: A mechanism for evaluation of mammographic and clinical data," in *Proc. 13th Symp. Computer Applications in Radiology (SCAR)*, Denver, CO, June 6–9, 1996.

[6] B. Kovalerchuk, E. Vityaev, and E. Triantaphyllou, "How can AI procedures become more effective for manufacturing?," in *Proc. Artificial Intelligence and Manufacturing Research Planning Workshop*. Albuquerque, NM, 1996.

[7] B. Kovalerchuk, E. Triantaphyllou, and A. S. Deshpande, "Interactive learning of monotone Boolean functions," *Inform. Sci.*, vol. 94, no. 1–4, pp. 87–118, 1996.

[8] A. A. Muchnik and S. G. Gindikin, "On the completeness of systems of unreliable elements which realize functions of the algebra of logic" (in Russian), *Doclady AN SSSR*, vol. 144, no. 5, 1962.

[9] E. I. Nechiporuk, "On the complexity of circuits in some bases, containing nontrivial elements with zero weights" (in Russian), *Prob. Cybern.*, no. 8, 1962.

[10] E. Post, "Introduction to a general theory of elementary propositions," *Amer. J. Math.*, vol. 43, 1921.

[11] E. Triantaphyllou and J. Lu, "The knowledge acquisition problem in monotone Boolean systems," in *Encyclopedia of Computer Science and Technology*, A. Kent and J. G. Williams, Eds. New York: Marcel Dekker, 1998, vol. 39, pp. 89–106.

[12] L. G. Valiant, "A theory of the learnable," *Commun. Assoc. Comput. Mach.*, vol. 27, pp. 1134–1142, 1984.

[13] S. V. Yablonsky, G. P. Gavrilov, and V. B. Kudryavtsev, *Functions of the Algebra of Logic and Post Classes* (in Russian). Moscow, Russia: Nauka, 1966.

[14] R. Yang, L. Yin, M. Gabbouj, J. Astola, and Y. Neuvo, "Optimal weighted median filtering under structural constraints," *IEEE Trans. Signal Processing*, vol. 43, Mar. 1995.

[15] E. Triantaphyllou. "Discovery of decision rules for breast cancer diagnosis". [Online]. Available: http://www.imse.lsu.edu/vangelis/mining/cancer/cancer1.html-ssi.

An Efficient Blocking–Matching Algorithm Based on Fuzzy Reasoning

Pei-Yin Chen and Jer Min Jou

***Abstract*—Due to the temporal and spatial correlation of image sequence, the motion vector of a reference block is highly related to the motion vectors of its adjacent blocks in the same image frame. By using that idea, we propose a novel efficient fuzzy search (EFS) algorithm for block motion estimation. The experimental results show that the EFS performs better than other fast search algorithms, such as TSS, CS, NTSS, FSS, BBGDS, SES, and PSA in terms of picture quality, accuracy, computational complexity, and coding efficiency.**

***Index Terms*—Block motion estimation, fuzzy reasoning, motion vector.**

I. Introduction

Emerging information technologies such as video conferencing and high definition television (HDTV) require real time compression of video signals, which require a huge amount of bandwidth compared to audio or text information. The key to a successful video coding scheme is to exploit the spatial and temporal redundancies existing in video image sequences effectively. One common feature of most coding schemes is that they use motion estimation to reduce the temporal redundancy, and use block transform coding, such as the DCT or wavelet transform, to reduce spatial redundancy. The most popular technique for motion estimation is the block-matching algorithm (BMA) [1]–[7]. BMA finds the best match for a block in the current frame within a search area in the previous frame. Because of its simplicity and robustness, BMA has recently been adopted by various video coding standards, such as H263 for video conference and Moving Picture Experts Group (MPEG) for video communication. Besides, BMA is also used as a tool for computing the image flow [8], [9].

In the block-matching schemes, the coding performance depends heavily on accuracy, speed, and effectiveness of motion estimation. However, the characteristics of various image sequences bear a lot of uncertainty and are hard to extract. Therefore, it is not easy to devise a good estimation method which always provides accurate and fast motion estimation for different types of image sequences. Consequently fuzzy reasoning is proposed to develop an efficient algorithm for block motion estimation.

During the nearly thirty years since Zadeh first proposed the idea [10], a variety of applications of fuzzy logic [11], [12] have been implemented in various fields. By using the linguistic rules that capture the approximate and qualitative aspects of human knowledge, the fuzzy logic controller can infer desired output actions properly. In this paper, an efficient fuzzy search (EFS) algorithm for block motion estimation is proposed. With the help of fuzzy reasoning, the motion vectors of the adjacent blocks in the current frame are used to find the predicted motion vector of the current block. Using the predicted vector, the EFS can determine the initial center of the search area for the current block. Then, the EFS searches the whole search area with a 3×3 movable

Manuscript received April 1, 2000; revised April 16, 2000 and November 22, 2000. This work was supported in part by the National Science Council, R.O.C., under Grant NSC-89-2213-E-218-025. This paper was recommended by Associate Editor P. Willett.

P.-Y. Chen is with the Department of Electronic Engineering, Southern Taiwan University of Technology, Tainan, Taiwan, R.O.C.

J. M. Jou is with the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, R.O.C. (e-mail: jou@92a21.ee.ncku.edu.tw).

Publisher Item Identifier S 1083-4419(01)02509-2.

search window until the local minimum point lies in the center of the present search window, or the search iteration times reach a predefined limit. Experimental results show that EFS performs better than other search algorithms, such as TSS [1], CS [2], NTSS [3], FSS [4], BBGDS [5], SES [6], and PSA [7] in terms of picture quality, accuracy, computational complexity, and coding efficiency.

II. Block-Matching Algorithm

In typical BMA, an image frame is divided into nonoverlapped square blocks of $N \times N$ pels. Then for a maximum motion displacement of w pels per frame, the block of pels in the current frame (called a reference block B_{ref}) is compared with the corresponding blocks (called the candidate blocks) within a search area of size $(N + 2w) \times (N + 2w)$ pels in the previous frame. When the best-matched (or lowest-distortion) candidate block is found, the motion vector, representing the coordinate difference between B_{ref} and the best-matched candidate block, is recorded. Consequently, the motion vector and the prediction error between the B_{ref} and the best-matched candidate block can be coded and transmitted instead of the B_{ref}; therefore, temporal redundancy is removed and data compression is achieved.

There are two main factors which determine the performance of BMA: the block distortion measure (BDM) and the search method. Assume $B_{ref}\ (m, n)$ is a reference block whose upper most left pel is at the location (m, n), and $B_c\ (m + u, n + v)$ is a candidate block within the search area of the previous frame with (u, v) displacement from the left top. Let w be the maximum motion displacement. The two most popular block distortion measures can be briefly described as follows.

1) *Sum of Absolute Difference (SAD)* between the pixel values of any two blocks:

$$SAD(u,v) = \sum_{m=1}^{N}\sum_{n=1}^{N} |B_{ref}(m,n) - B_c(m+u, n+v)| \quad \text{where} \quad -w \le u, v \le w.$$

2) *Mean Squared Error (MSE)* between the pixel values of any two blocks:

$$MSE(u,v) = \frac{1}{N^2}\sum_{m=1}^{N}\sum_{n=1}^{N} [B_{ref}(m,n) - B_c(m+u, n+v)]^2 \quad \text{where} \quad -w \le u, v \le w.$$

The SAD/MSE is computed for each candidate block within the search area. A block with the minimum SAD/MSE is considered the best-matched block, and the value (u, v) for the best-matched block is called motion vector. That is, motion vector MV is

$$\text{MV} = (u, v)|_{\min \text{SAD}(u,v)/\text{MSE}(u,v)}.$$

In this paper, both SAD and MSE are used for testing the different search algorithms.

A straightforward BMA is the full search (FS) algorithm, which matches all the possible candidate blocks within the search area to find the optimal MV. Since the number of search points (or the number of block matches) needed to find the MV of each reference block is $(2w + 1)^2$ for FS, its speed is not appropriate to real-time applications. Thus, many suboptimal but faster search algorithms [1]–[7] have been developed to alleviate the heavy computations of FS.

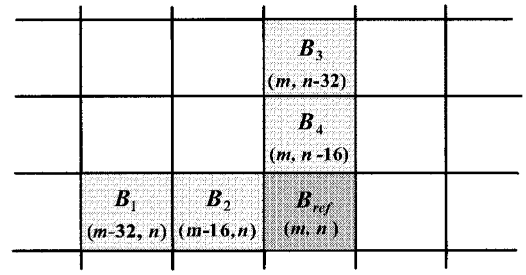


Fig. 1. Position-relation of the reference block and its adjacent blocks.

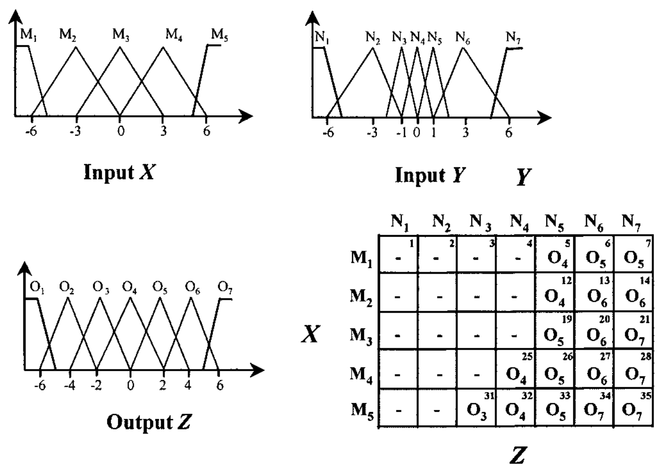


Fig. 2. Corresponding fuzzy sets and 35 fuzzy control rules.

III. Proposed EFS Algorithm

An image frame is processed from a top-left block to a bottom-right block. For each block, two processes are used to obtain its motion vector. They are finding the predicted MV and determining the final MV. In the first process, the MV's of the adjacent blocks of B_{ref} are used to determine the predicted MV of B_{ref} by using fuzzy reasoning. The fuzzy reasoning process, performed in the design, is based on the concepts of fuzzy implication and the compositional rules of inference for approximate reasoning [12]. In the second process, the EFS algorithm determines the initial center of the search area by using the predicted MV. Then, EFS searches the whole search area with a 3×3 movable search window and determines the final MV of B_{ref}. The detail of the EFS algorithm is described in the following subsections.

A. Finding the Predicted Motion Vector

The predicted MV's for blocks in the top two rows and the left two columns of a frame are all set to 0 in EFS. With the fuzzy reasoning process, the predicted MV of each block of other rows and columns of the frame is determined by using the MV's of its adjacent blocks, which are determined in the preceding search processes. Fig. 1 shows the position-relation of B_{ref} and its adjacent blocks, in which each block is of size 16×16. The upper most left pel of B_{ref} is located at (m, n) where $m(n)$ is the column (row) number, and its four adjacent blocks, at location $(m - 32, n)$, $(m - 16, n)$, $(m, n - 32)$, and $(m, n - 16)$, are denoted as B_i for $i \in \{1, 2, 3, 4\}$. Let MV of B_i be represented by $V_i = (u_i, v_i)|_{\min \text{SAD}(u,v)/MSE(u,v)} = (\Delta m_i, \Delta n_i)$ where Δm and Δn represent the displacement in horizontal and vertical directions respectively, and $\hat{V}_{ref} = (\Delta\hat{m}_{ref}, \Delta\hat{n}_{ref})$ mean the predicted MV of B_{ref}. First, we use the V_1 and V_2 as the inputs to infer the output

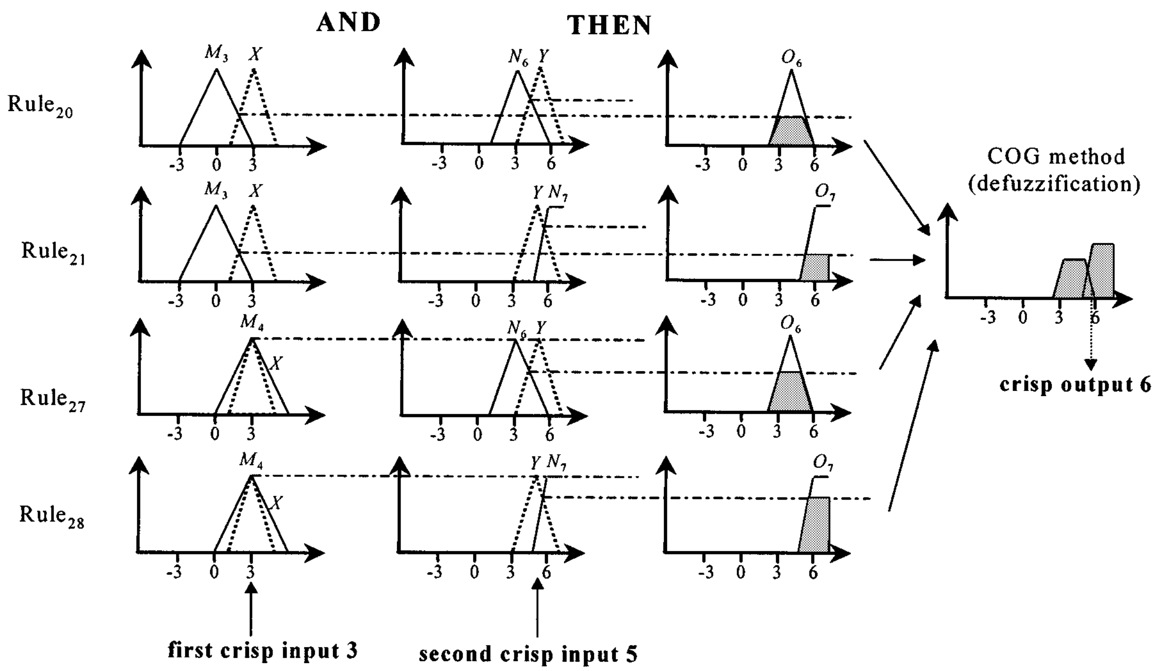


Fig. 3. Fuzzy inference process when $\Delta m_1 = 3$ and $\Delta m_2 = 5$.

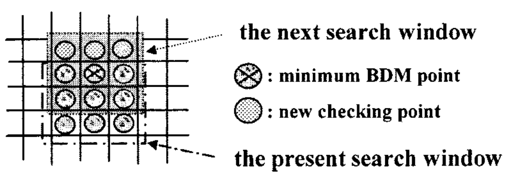


Fig. 4. Next search window when the minimum BDM point is located at the middle of the boundary column (row) of the present search window.

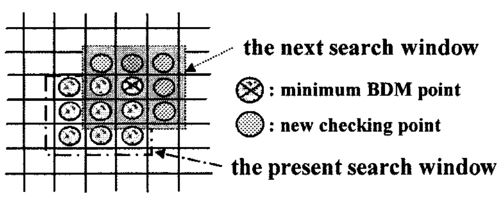


Fig. 5. Next search window when the minimum BDM point is located at the corner of the present search window.

$\hat{V}_{ref\_H} = (\Delta\hat{m}_{ref\_H}, \Delta\hat{n}_{ref\_H})$, which represents the horizontally predicted MV of B_{ref}. Similarly, the V_3 and V_4 are used as the inputs to infer the output $\hat{V}_{ref\_V} = (\Delta\hat{m}_{ref\_V}, \Delta\hat{n}_{ref\_V})$, which represents the vertically predicted MV of B_{ref}. Finally, $\hat{V}_{ref}$ is determined by $\hat{V}_{ref} = (\hat{V}_{ref\_H} + \hat{V}_{ref\_V}/2)$.

The maximum displacement w is set to 7, that is $-7 \le \Delta m, \Delta n \le 7$. We also assume that a fuzzy set is represented by a triple (L, C, R) in which L, C, and R are the left boundary, the center value, and the right boundary along the horizontal axis, respectively. Fig. 2 shows the corresponding fuzzy sets used for the fuzzy inputs and output. The fuzzy sets M_i used for the first input X are described as follows:

$$M_1(-\infty, -6, -5), M_2(-6, -3, 0), M_3(-3, 0, 3)$$
$$M_4(0, 3, 6), M_5(5, 6, +\infty).$$

For the second input Y, the following seven fuzzy sets are used:

$$N_1(-\infty, -6, -5), N_2(-6, -3, -1), N_3(-2, -1, 0)$$
$$N_4(-1, 0, 1), N_5(0, 1, 2), N_6(1, 3, 6), N_7(5, 6, +\infty).$$

The output Z is characterized by the fuzzy sets as follows:

$$O_1(-\infty, -6, -5), O_2(-6, -4, -2), O_3(-4, -2, 0)$$
$$O_4(-2, 0, 2), O_5(0, 2, 4), O_6(2, 4, 6), O_7(5, 6, +\infty).$$

Since B_2 (B_4) is closer to B_{ref} than B_1 (B_3) is, V_2 (V_4) would do more help for the determination of the possible value of $\hat{V}_{ref}$ than V_1 (V_3). Thus, the coarse fuzzy control rules are constructed. To determine the final fuzzy rules, a backpropagation learning algorithm is adopted [13]. Using the first image frame of each of the three sequences, Football, Tennis, and Mobile, as the training data, we can obtain the 35 tuned fuzzy rules. Some of those rules are shown in Fig. 2. The corresponding output values can be inferred for different input values by using the fuzzy sets and the fuzzy rules.

The detail to determine $\hat{V}_{ref}$ is described as follows.

Step 1) Use Δm_1 and Δm_2 as the fuzzy input X and input Y, respectively, and then fuzzy infer the output $\Delta\hat{m}_{ref\_H}$ by using the 35 fuzzy control rules shown in Fig. 2.

Step 2) Use Δn_1 and Δn_2 as the inputs, and then infer the output $\Delta\hat{n}_{ref\_H}$ with the 35 fuzzy rules.

Step 3) Use Δm_3 and Δm_4 as the inputs, and then infer the output $\Delta\hat{m}_{ref\_V}$ with the 35 fuzzy rules.

Step 4) Use Δn_3 and Δn_4 as the inputs, and then infer the output $\Delta\hat{n}_{ref\_V}$ with the 35 fuzzy rules.

Step 5) Determine $\hat{V}_{ref}$ as follows:

$$\hat{V}_{ref} = (\Delta\hat{m}_{ref}, \Delta\hat{n}_{ref}) = \frac{\hat{V}_{ref\_H} + \hat{V}_{ref\_V}}{2}$$
$$= \left(\frac{\Delta\hat{m}_{ref\_H} + \Delta\hat{m}_{ref\_V}}{2}, \frac{\Delta\hat{n}_{ref\_H} + \Delta\hat{n}_{ref\_V}}{2}\right).$$

TABLE I
COMPARISON OF THE SEARCH ALGORITHMS (MSE)

	Prediction Error	MSE	PSNR	Unpredictable Pixels	Distances	Probabilities	Search Points	Coding Bits
FS	6.78	198.74	27.97	41.36%	0	100.00%	202.05	1061.94
TSS	7.31	228.17	27.39	43.96%	0.90	79.70%	23.14	1179.31
CS	8.15	279.06	26.45	46.19%	1.25	64.18%	15.48	1244.90
NTSS	6.92	208.75	27.77	42.52%	0.70	85.29%	19.66	999.66
FSS	7.12	217.49	27.59	43.28%	0.76	81.68%	17.54	1022.93
BBGDS	6.95	212.27	27.73	42.45%	0.73	85.26%	12.16	919.25
SES	7.46	239.65	27.21	44.24%	1.34	72.34%	16.21	1166.89
PSA	6.91	209.63	27.74	42.51%	0.72	85.21%	10.71	916.59
EFS	6.87	207.95	27.79	42.37%	0.69	85.58%	10.22	907.05

For example, $V_1 = (3, -1), V_2 = (5, -6), V_3 = (2, -3)$ and $V_4 = (4, -2)$. At the first step, "3" and "5" are used as the inputs to fuzzy infer the output value "6" ($\Delta\hat{m}_{ref\_H}$) as shown in Fig. 3. Then, "-1" and "-6" are used to infer output "-6" ($\Delta\hat{n}_{ref\_H}$) at the second step. Similarly, "2" and "4" are used to infer "5," and "-3" and "-2" are used to obtain "-2." Finally, the $\hat{V}_{ref}$ is equal to $((6 + 5/2), (-6\ -2/2)) = (5, -4)$.

B. Determining the Final Motion Vector

In this process, the EFS first determines the initial center of the search area by using the $\hat{V}_{ref}$. Then, the EFS searches the whole search area with a 3 × 3 movable search window until the local minimum point lies in the center of the present search window, or until when the search iteration times reach the given maximum of iteration times. Let Count represent the maximum iteration times of the search loop set by users and BDM mean the block distortion measure used for motion estimation. Then, the final motion vector is calculated by the following steps.

Step 1) Set the loop counter (S) to 1 and set the initial center of the search area at the coordinate $((m+\Delta\hat{m}_{ref}), (n+\Delta\hat{n}_{ref}))$.

Step 2) Find the minimum BDM point among the nine checking points on a 3 × 3 movable search window.

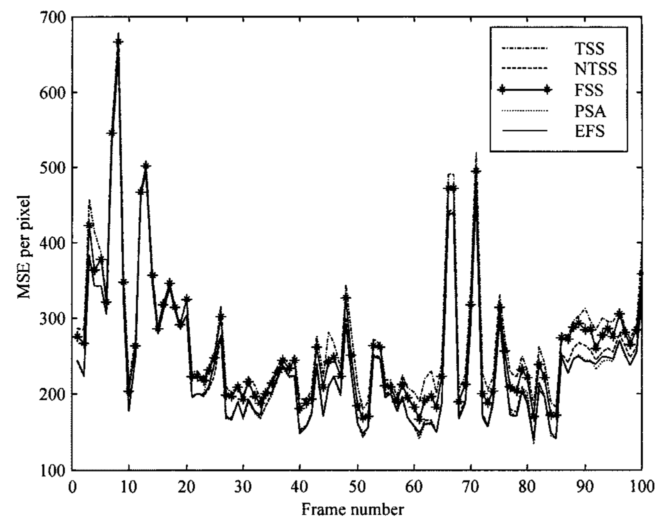
Step 3) If $S > Count$ or the minimum BDM point is located at the center of the present search window, go to Step 4; otherwise increase S by 1, perform the search process with the following two search patterns, and then repeat this step.

  a) If the minimum BDM point is located at the middle of the boundary column (row) of the present search window, the center of the next search window is shifted to the minimum BDM point, and three additional checking points along the vertical (horizontal) direction as shown in Fig. 4 are used.

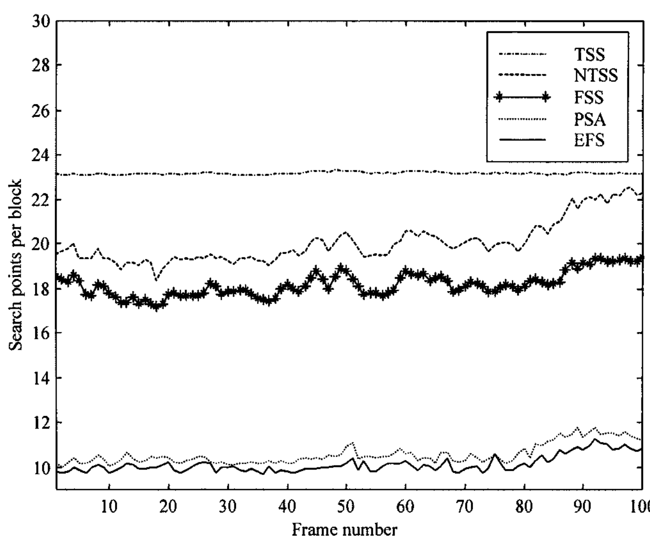
  b) If the minimum BDM point is located at the corner of the present search window, the center of the next search window is shifted to the minimum BDM point, and five sadditional checking points as shown in Fig. 5 are used.

Step 4) Stop the search and calculate the final motion vector.

Even if we use a 3 × 3 movable search window, only part of the window but not all the nine checking points is searched. Thus, some unnecessary search points can be omitted. With the help of the halfway-stop technique, the EFS algorithm can stop the search process as soon as the minimum BDM point is found at the center of the present search window. If we can make a better prediction, the minimum BDM point can be found at the center of the present search window either after the first search iteration (nine search points) or the second search iteration (12 or 14 search points).



(a)



(b)

Fig. 6. Results of each frame of Windmill sequence for (a) MSE per pixel and (b) search points per block.

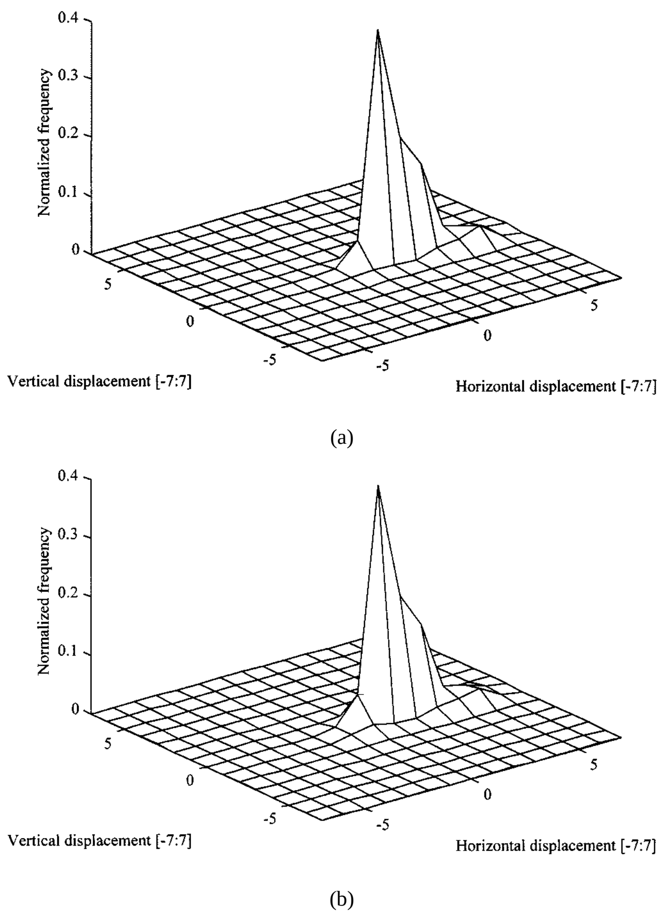


Fig. 7. Histograms of MV locations of Flower sequence for (a) FS and (b) EFS.

IV. Experimental Results and Performance Comparisons

To reduce the implementation complexity and to achieve high-speed fuzzy reasoning, we implement the whole fuzzy inference process with a table-look-up manner. In the EFS, a two-input and one-output fuzzy system is employed. Each input (or output) represents a vector displacement value between -7 and 7 (4 bits), so a table with $2^{4\times 2}$ entries must be constructed. For each possible input combination, we used the 35 fuzzy rules shown in Fig. 2 to obtain its output z (4 bits) in advance, and then saved z in the corresponding entry. With the table, which needs only 128 bytes of memory, fuzzy reasoning can be implemented easily.

In our experiments, the block size is fixed at 16×16 for MV estimation as specified by the MPEG. The first 90 frames of the "Football," "Tennis," "Flower," "Mobile," "Windmill," "Miss America," and "Salesman" sequences are used to test various search algorithms. These sequences have been selected to highlight different kind of motions, such as low-to-high amount of movement, camera zooming and panning motion, etc. The size of each individual frame is 352×240 pixels, quantized uniformly to 8 bits. Using MSE as the block distortion measure, we compare the nine search algorithms:

1) FS;
2) TSS [1];
3) CS [2];
4) NTSS [3];
5) FSS [4];
6) BBGDS [5];
7) SES [6];
8) PSA [7];
9) EFS in terms of picture quality, accuracy, computational complexity, and coding efficiency.

Table I shows the experimental results of nine search algorithms for eight different measures. To evaluate the picture quality of various algorithms, four measures are considered. The first is the average absolute prediction error per pixel between the original frame and the motion-compensated prediction frame. The second is the average mean square error (MSE) per pixel between the original frame and the motion-compensated prediction frame. The next is the average peak signal-to-noise ratio (PSNR), and the last one is the average percentage of unpredictable pixels per frame (pixels with absolute prediction errors larger than three, over a range of 255, are classified as unpredictable pixels). The accuracy of the algorithms is evaluated by using two measures. One is the average Euclidean error distances between the estimated motion vectors and the true motion vectors (motion vectors obtained by full search algorithm are called the true motion vectors), and the other is the average probabilities of finding the true motion vectors. In addition, the average number of search points per block is used as the measure of computational complexity. According to Table I, the EFS performs better than other search algorithms.

In MPEG, the MV differentials, obtained from spatially adjacent blocks, are encoded and transmitted instead of the MVs. By spatial ordering, we use the block at the top left corner as the reference block and order the blocks from top left to top right, and then go to the next row of blocks and so on. Thus, the bits per frame for encoding the MV differentials, as the measure of coding efficiency, can be calculated and are also shown in Table I. According to the result, we can see that EFS needs fewer bits (only about 85% as compared with FS) than other search algorithms. This implies that EFS generates a more spatially correlated MV sequence, and saves more bits than other algorithms. When video coding is used, the saved bits can improve the visual quality of the reconstructed images.

By showing the different results of each frame of Windmill sequence in terms of MSE/pixel and search points/block for TSS, NTSS, FSS, PSA, and EFS, a clearer comparison is presented in Fig. 6. The histogram of MV locations of the entire Flower sequence obtained by using FS and EFS is plotted in Fig. 7. One can clearly see that the histograms of the MV locations of the two figures are similar in shape and their peaks are concentrated around the central region.

To see the effect of block distortion measure, SAD is also employed to test the search algorithms. The results are given in Table II, and clearly EFS performs well. To explore robustness of EFS, a skipping of two frames is adopted. Frame skipping is a strategy used by current standards, such as H.263, for the purpose of controlling the bit rate and adjusting the coding to specific bandwidth limitations. Table III gives the average results of the seven image sequences when MSE is adopted as the BDM and a skipping of two frames is used. Similarly, Table IV gives the results as SAD is applied. The results indicate EFS consistently produces better performance and more reliable MV's. This demonstrates the capability of our algorithm in coping with larger movements. To investigate the practical effects of EFS, we also implement the various search algorithms for H.263 video coding scheme. Table V shows PSNR of the nine search algorithms for seven image sequences. Obviously, EFS still performs better than other search algorithms.

V. Conclusion

An efficient fuzzy search algorithm for block motion estimation is presented in this paper. By using fuzzy reasoning, the algorithm can

TABLE II
COMPARISON OF THE SEARCH ALGORITHMS (SAD)

	Prediction Error	MSE	PSNR	Unpredictable Pixels	Distances	Probabilities	Search Points	Coding Bits
FS	6.72	202.59	27.88	41.44%	0	100%	202.05	1021.37
TSS	7.17	226.88	27.40	43.06%	0.729	83.57%	23.13	1134.60
CS	7.89	263.08	26.62	44.65%	1.354	58.45%	15.52	1199.00
NTSS	6.85	212.17	27.69	41.81%	0.566	88.79%	19.39	960.02
FSS	7.03	219.74	27.53	42.52%	0.624	85.69%	17.47	1013.42
BBGDS	6.89	216.58	27.62	41.77%	0.584	88.21%	12.09	913.93
SES	7.31	237.77	27.21	43.40%	0.887	78.91%	16.33	1125.99
PSA	6.83	211.11	27.66	41.77%	0.593	88.34%	10.75	905.00
EFS	6.81	210.48	27.70	41.70%	0.561	**89.60%**	10.20	897.47

TABLE III
COMPARISON OF THE SEARCH ALGORITHMS FOR FRAME SKIPPING = 2 (MSE)

	Prediction Error	MSE	PSNR	Unpredictable Pixels	Distances	Probabilities	Search Points	Coding Bits
FS	8.56	325.32	26.47	43.42%	0	100%	202.05	1196.05
TSS	9.64	406.40	25.68	45.62%	1.248	74.08%	23.22	1403.16
CS	13.10	443.34	24.82	47.85%	2.080	48.89%	15.53	1617.69
NTSS	8.99	358.10	26.09	44.36%	1.152	80.24%	21.91	1218.56
FSS	9.28	390.19	25.82	44.49%	1.156	79.40%	18.57	1176.96
BBGDS	9.64	431.27	25.55	44.64%	1.290	75.62%	14.25	1131.05
SES	9.92	430.76	25.41	46.09%	1.732	67.59%	15.98	1407.71
PSA	8.84	351.87	26.10	44.04%	1.300	80.46%	11.99	1067.79
EFS	8.80	347.31	26.17	44.01%	1.094	80.74%	11.30	1057.15

TABLE IV
COMPARISON OF THE SEARCH ALGORITHMS FOR FRAME SKIPPING = 2 (SAD)

	Prediction Error	MSE	PSNR	Unpredictable Pixels	Distances	Probabilities	Search Points	Coding Bits
FS	8.48	331.19	26.37	42.73%	0	100.00%	202.05	1143.4
TSS	9.31	392.40	25.76	44.45%	0.984	80.43%	23.20	1324.6
CS	11.19	534.29	24.96	46.30%	2.063	46.44%	15.58	1521.6
NTSS	8.84	359.64	26.05	43.53%	0.945	83.48%	21.53	1157.6
FSS	9.10	388.52	25.80	43.60%	0.991	83.06%	18.48	1146.6
BBGDS	9.45	429.37	25.50	43.78%	1.100	78.75%	14.14	1105.9
SES	9.57	414.97	25.50	44.88%	1.200	75.48%	16.11	1328.9
PSA	8.70	351.73	26.05	43.17%	1.110	83.72%	11.76	1035.9
EFS	8.66	348.75	26.11	43.18%	0.921	84.13%	11.16	1025.6

TABLE V
COMPARISON OF PSNR OF THE SEARCH ALGORITHMS FOR H.263

	Football	Tennis	Flower	Mobile	Windmill	Miss-A	Salesman	Average
FS	22.48	29.10	23.83	22.51	24.59	37.35	35.25	27.87
TSS	21.92	27.92	22.90	22.35	23.82	36.98	35.19	27.30
CS	21.41	27.10	19.46	22.18	22.59	36.65	35.09	26.35
NTSS	22.12	28.52	23.63	22.49	24.48	37.28	35.19	27.67
FSS	21.98	28.41	23.36	22.44	23.99	37.02	35.19	27.48
BBGDS	21.96	28.50	23.52	22.49	24.46	37.28	35.22	27.63
SES	21.30	27.50	22.93	22.30	23.76	36.79	35.12	27.10
PSA	22.12	28.23	23.60	22.43	24.45	37.29	35.20	27.62
EFS	22.20	28.52	23.62	22.48	24.48	37.30	35.21	27.69

determine the motion vectors of image blocks both quickly and correctly. To reduce the implementation complexity and to increase prediction speed, a fuzzy reasoning procedure is implemented with the table-look-up approach. Experimental results show that the proposed algorithm works better than other search algorithms in terms of picture quality, accuracy, computational complexity and coding efficiency.

REFERENCES

[1] T. Koga, K. Iinuma, A. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in *Proc. of NTC81*, New Orleans, LA, 1981, pp. G5.3.1–G.5.3.5.

[2] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Trans. Commun.*, vol. 38, pp. 950–953, 1990.

[3] 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, pp. 438–442, 1994.

[4] L.-M. Po and W.-C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst., Video Technol.*, vol. 6, pp. 313–317, 1996.

[5] L.-K. Liu and E. Feig, "A block-based gradient descent search algorithm for block motion estimation in video coding," *IEEE Trans. Circuits Syst., Video Technol.*, vol. 6, pp. 419–422, 1996.

[6] J. Lu and M. L. Liou, "A simple and efficient search algorithm for block-matching motion estimation," *IEEE Trans. Circuits Syst., Video Technol.*, vol. 7, pp. 429–433, 1997.

[7] L. Luo, C. Zou, X. Gao, and Z. He, "A new prediction search algorithm for block motion estimation in video coding," *IEEE Trans. Consumer Electron.*, vol. 43, pp. 56–61, 1997.

[8] L. Li and J. H. Duncan, "3-D translational motion and structure from binocular image flow," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, pp. 657–667, 1993.

[9] R. Beb, D. Paulus, and H. Niemann, "3D recovery using calibrated active camera," in *IEEE Int. Conf. Image Processing*, Lausanne, Switzerland, 1996, pp. 855–858.

[10] L. A. Zadeh, "Fuzzy sets," *Inf. Control*, vol. 8, pp. 338–353, 1965.

[11] E. H. Mamdani, "Applications of fuzzy algorithms for simple dynamic plant," *Proc. Inst. Elect. Eng.*, vol. 121, pp. 1585–1588, 1974.

[12] C. C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller—Parts I & II," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, pp. 404–435, Mar. 1990.

[13] J. M. Jou and P.-Y. Chen, "A hybrid adaptive fuzzy system using error backpropagation learning," in *CFSA/IFIS/SOFT'95 on Fuzzy Theory and Applications*, Taipei, Taiwan, R.O.C., pp. 537–542.

Piecewise Lyapunov Stability Conditions of Fuzzy Systems

Ming Feng and C. J. Harris

***Abstract*—In this paper we address the stability of a class of nonlinear fuzzy systems that can be decomposed into a set of local models characterized as Takagi–Sugeno models. This new approach includes a consideration of the input membership functions. Via this approach, a reduction in the number of candidate Lyapunov functions and associated linear matrix inequalities (LMIs) is produced. This approach significantly reduces the computational load associated with determining closed loop stability as the input dimension increases.**

***Index Terms*—Fuzzy systems, stability.**

I. INTRODUCTION

For fuzzy and neurofuzzy systems, stability analysis has been difficult because these systems are both nonlinear and represent linguistic/symbolic knowledge in terms of rules with variables that encapsulate vague or imprecise notions. Recently, some stability results for fuzzy systems have been reported [1]–[5]. An investigation of input/output (I/O) data based stability of a given direct static multiple-input single-output neuro-fuzzy controller operating under feedback control has been developed [1]. For Lyapunov stability, one of the more recent approaches [4] to determining stability of fuzzy/neurofuzzy systems is to decompose the global process into a series of local models/subsystems represented as Takagi–Sugeno models. A sufficient condition for the asymptotic stability of a fuzzy system in the sense of Lyapunov through the existence of a common Lyapunov function for all the subsystems has been derived [2]; whereas a general method for the computation of piecewise quadratic Lyapunov functions for hybrid systems which include Takagi–Sugeno fuzzy systems as a special case has also been derived [3]. Compared with the global result [2], the 'local' result of [3] is significant, in that it searches for different quadratic Lyapunov function in different operating regions in state space and so significantly relaxes the stability conditions for a global fuzzy system. However, there are many situations where continuous Lyapunov functions are too restrictive; so a construction method [5] for generating stability conditions of hybrid systems using discontinuous Lyapunov functions has been derived. Both local and hybrid methods [3], [5] are attractive since the search for local Lyapunov functions can be reformulated as a set of linear matrix inequalities (LMIs).

A drawback that limits the practical use of the methods presented in [3] and [5] is that it may be required to solve a large number of LMIs in the interpolation regions between the system submodels. In addition to the high number of LMIs, the computation complexity and cost also increases dramatically as the input dimensionality increases. This means that the number of parameters involved in the optimization process becomes prohibitively large for large dimensional systems. Also for fuzzy membership functions with global support the stability conditions reduce to the case of global quadratic stability; hence, the preference for basis functions such as B-splines with compact support. The majority of current methods for stability analysis of fuzzy systems ignore the

Manuscript received March 1, 1999, February 29, 2000, and October 21, 2000. This paper was recommended by Associate Editor S. Lakshmivarahan.

The authors are with the Image, Speech and Intelligent Systems Research Group, Department of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K. (e-mail:cjh@ecs.soton.ac.uk).

Publisher Item Identifier S 1083-4419(01)02510-9.