International Journal of Innovative Computing, Information and Control
Volume **9**, Number **1**, January **2013**

ICIC International ©2013 ISSN 1349-4198
pp. 99–109

DIGITAL IMAGE STABILIZATION USING A FUNCTIONAL NEURAL FUZZY NETWORK

Chi-Feng Wu[1] and Cheng-Jian Lin[2,*]

[1]Department of Digital Content Application and Management
Wenzao Ursuline College of Languages
No. 900, Mintsu 1st Rd., Kaohsiung 80793, Taiwan
cfwu@mail.wtuc.edu.tw

[2]Department of Computer Science and Information Engineering
National Chin-Yi University of Technology
No. 57, Sec. 2, Zhongshan Rd., Taiping Dist., Taichung 41170, Taiwan
*Corresponding author: cjlin@ncut.edu.tw

Received October 2011; revised March 2012

Abstract. *This study proposes a real-time video stabilization method to eliminate unwanted vibration, preserve the intended movement of camera, and improve the stability of the captured video sequence. The proposed method uses a functional neural fuzzy network to learn the characteristics of different vibrations and then choose the adequate compensation weight for two different methods to calculate the correction vector. Experimental results show that the proposed method has superior performance over other motion compensation methods.*

Keywords: Digital image stabilization, Neural fuzzy network, Functional link neural network

1. **Introduction.** Video-capturing devices have been used extensively in electronic devices, such as video camera recorder, video surveillance and monitoring system, and driver assistance system (DAS). The visual quality of the captured image sequences is degraded by vibration caused by hand or unstable mounted platform. In order to improve the system performance, motion compensation has been used to eliminate the impact of jitter and improving the stability of captured video. Video stabilization systems can be classified into electronic image stabilization (EIS), optical image stabilization (OIS), and digital image stabilization (DIS). EIS uses built-in sensors to detect vibrations and takes the image at the position of reverse displacement to eliminate the interference from vibration. EIS is normally equipped with digital zooming or CCD image sensors of increased sizes for motion compensation. OIS uses gyro type of sensors to detect vibration, and sends signals to a high-precision control circuit with fast computing capability. The circuit calculates the compensated position for a micro-OIS lens driven by a linear motor to allow the light through the lens to be accurately projected onto the CCD light sensor. DIS uses digital image processing techniques to remove the vibrating effects on images. The system does not require additional motion-sensing devices. In addition, DIS has been practically used not only in defensive weapons [1], but also equipped on the retail products such as consumer video cameras, handheld wireless visual communication devices [2] and vehicle-mounted camera [3,4]. The advantages of DIS possess are easy to operate, higher flexibility, higher stabilizing accuracy, and real-time processing. Simultaneously, it can distinguish jitter and intentional camera movement from the global camera motion.

Stabilization systems dealing with global motion containing translation, rotation and scaling effects are referred as three-dimensional stabilization. In real-time system, most proposed stabilization methods only deal with translational jitter for computing simplicity, and these models are referred as two-dimensional stabilization. In this paper, we only deal with 2D stabilization. DIS is generally composed of three major units: local motion estimation, global motion estimation and motion compensation. As shown in Figure 1, global motion estimation estimates the global motion vector (GMV) between two continuous frames of the acquired video sequence. Based on the GMV, the motion compensation unit produces a correction vector for the current frame. This vector is used to display the image frame inversely to jitter, to absorb the affection of annoying camera vibration while preserving the intended camera movement.

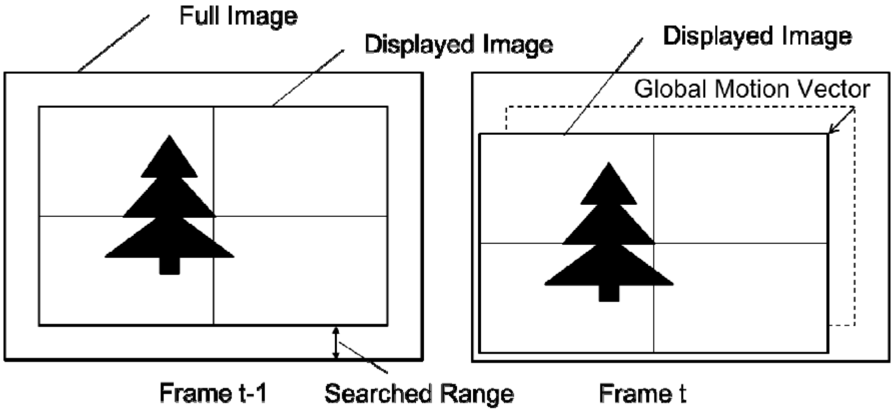


Figure 1. The illustration of motion compensation

There has been extensive research regarding the local motion estimation [3-9]. For example, Chen et al. [5] selected feature blocks from nine fixed observation blocks which distributed over one frame and used the full search method to get the local motion vector (LMV). When avoiding the impact of objects moving in and increasing the reliability of LMV, feature blocks with large content changed from the previous frame are excluded.

By reducing the computation complexity of motion estimation, representative point matching (RPM) [3] and fast searching algorithm were utilized [6], but the LMV which is derived from it may not be reliable. In method of [3], an inverse triangle method was used to examine the reliability of the obtained LMV.

There are many methods used to obtain the GMV. Some of the methods compare the characteristic values, such as edges and feature points, instead of block matching [4,7-9]. Lane lines and their vanishing points are used as a feature comparison for in-car video capture [4].

For motion compensation of translational fluctuation, frame position smoothing (FPS) and motion vector integration (MVI) have been proposed to estimate the appropriate correction vector [10]. In FPS methods, the vibration is divided into intended movement of low frequency and jitters of high frequency. The high-frequency vibration of the accumulated global motion is then filtrated by some low-pass filter. The Kalman filter is frequently used as the low-pass filter for motion compensation [7,11-13]. However, the filter error will propagate while using Kalman filter. On the other hand, MVI method used adjustable damping factor to decay the quantity of shifting vector. This method is applicable in real-time system but due to the restricted area of compensation, this method does not work well for constant lens movement. An improved method [3] includes an integrator to remove the steady-state lag caused by constant movement. In this study, we focus on the issue of motion compensation and propose a solution based on functional neural fuzzy network.

The rest of this paper is organized as follows. The proposed video stabilization algorithm is described in Section 2. The structure of the functional neural fuzzy network is presented in Section 3. Lastly, the experimental results are described in Section 4 and conclusion is drawn in Section 5.

2. The Proposed Video Stabilization Algorithm.

The proposed method mainly involves motion estimation between two continues frames of captured video, and the motion compensation algorithm is based on functional neural fuzzy network. Details are given in the following subsections.

2.1. Local motion vector estimation.

First, frame $t-1$ is divided into four blocks of equal sizes, for an estimation of each individual *LMV*. Full-search block matching and mean absolute difference (MAD) is commonly used method for calculating LMV. The formula of MAD_f is as follows:

$$MAD_f(p,q) = \frac{1}{M \times N} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} |I(t-1, x+k, y+l) - I(t, x+p+k, y+q+l)| \quad (1)$$

where I is the intensity of the pixel (x, y), M and N are the sizes of block, (k, l) is the coordinates of the pixel at the top left corner and t is the frame number. The vector (p, q) with the minimum MAD_f is the desired motion vector. Although the full-search method can find the motion vector close to the movement of block, the method's computational complexity is too high to fit the requirement of real-time systems. Therefore, some fast search methods have been proposed to reduce the computational complexity. Examples of such methods are two-dimensional logarithmic search, three-step search, and diamond search. In this study, we use representative point matching (RPM) as the search method. The RPM method divides a block further into sub-blocks of equal sizes, and takes the center point of each sub-block as its representative point. In this way the amount of the calculation of MAD_r drops to S, where S is the number of sub-blocks. The formula is

$$MAD_r(p,q) = \sum_{k=1}^{S} |I(t-1, x_k, y_k) - I(t, x_k+p, y_k+q)| \quad (2)$$

where (x_k, y_k) are the coordinates of the representative points. Although RPM can greatly decrease the computing complexity; however, the reliability of LMV retrieved from RPM is decreased when the background image contains repeated pattern. This scenario will not be deal in this paper.

2.2. Global motion vector estimation.

The four LMVs, the mean vector of LMVs, the GMV of the previous frame, and the zero motion vector are selected as candidates to get the GMV of the current frame. Then, nine fixed observation blocks with a size of 16×16 pixels, as shown in Figure 2, are preselected. These blocks are distributed equally over the frame. MAD_p between preselected block in previous frame and block shifted by GMV candidate in current frame for each observation block and each GMV candidate are calculated. The formula is

$$MAD_p(u,v) = \sum_{k=1}^{16} \sum_{l=1}^{16} |I(t-1, x_i+k, y_i+l) - I(t, x_i+u+k, y_i+v+l))| \quad (3)$$

where (x_i, y_i) is the coordinates of the pixel at the top left corner of the i-th observation block, t is the frame number, and vector (u, v) is a candidate of GMV.

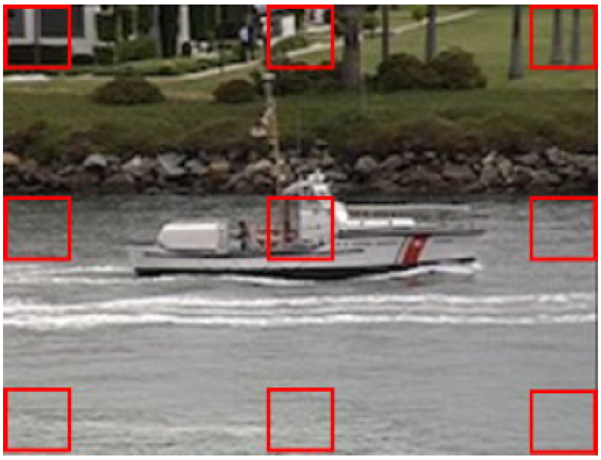


Figure 2. Nine pre-selected observation blocks

In addition to camera movements, there may be other objects moving into or out of the screen. These moving objects may move to the locations of observation blocks and cause abnormal MAD_p. Thus, the derived motion vector may reflect the movement of objects in the picture rather than the global movement of background. We use a ballot approach to avoid choosing the wrong GMV. Each observation block ranks the GMV candidates according to their MAD_p values. The candidate with the minimum sum of ranks will be the GMV of the current frame. If two or more candidates have the same sum of ranks, a comparison of the average MAD_p value is made.

2.3. **Motion compensation.** Accumulated Motion Vector (AMV) [14] is a MVI based method. AMV uses damping factor k to achieve the purpose of vibrating absorption, which is defined as follows:

$$\begin{aligned} AMV(t) &= kAMV(t-1) + (\alpha GMV(t) + (1-\alpha)GMV(t-1)) \\ AMV(t) &= clipper(AMV(t)) \\ &= \tfrac{1}{2}(|AMV(t)+l| - |AMV(t)-l|) \end{aligned} \tag{4}$$

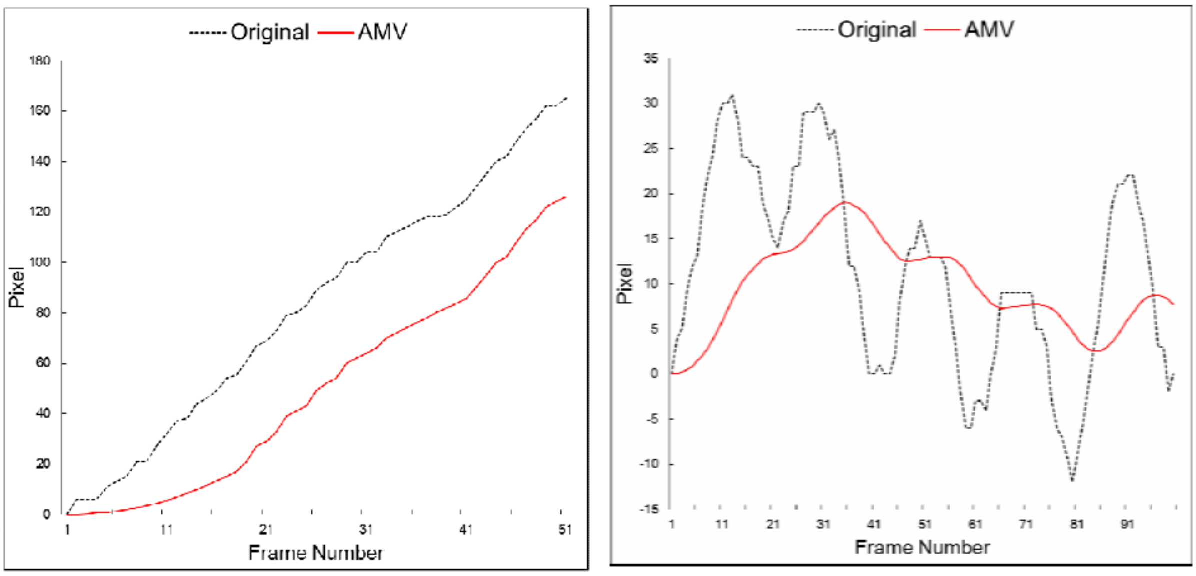
where l is the search boundary, k is the damping factor, $0 < k < 1$. The greater the value of k is, the greater the ability of vibration absorption. However, when the camera moves at constant speed, a larger k will make the AMV value larger. Due to fixed search range of motion estimation, the ability of vibrating absorption will be decreased and the compensated trajectory of AMV will lag behind the actual trajectory, the simulation results are displayed in Figure 3. The trajectory formula is as follows:

$$\begin{aligned} Tr_{original}(t) &= \sum_{i=1}^{t} GMV(i) \\ Tr_{AMV}(t) &= \sum_{i=1}^{t} GMV(i) - AMV(t) \end{aligned} \tag{5}$$

The vector of constant movement of camera should be subtracted in the calculation when calculating motion compensation during the constant movement of camera. This will reduce the size of the correction vector as well as increase the ability of compensation. Therefore, Hsu has added an integrator in the formula to improve this lag problem while the camera is in constant speed [3]. The formula is as follows:

$$\begin{aligned} &CMV(t) = k \times CMV(t-1) + GMV(t) - \beta \times CMV\_I(t-1) \\ &CMV\_I(t) = CMV\_I(t-1) + CMV(t) \\ &CMV(t) = clipper(CMV(t)) \end{aligned} \tag{6}$$

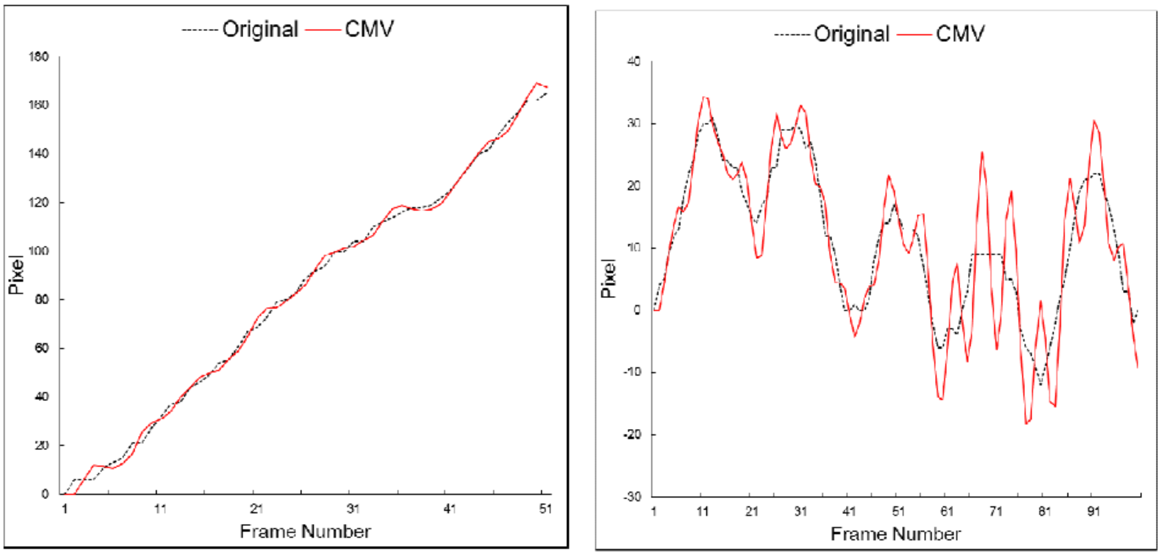
where the constant k is used to smooth the playback of camera panning, and constant β is used to filter out abnormal vibrations. However, compensated trajectory will exceed the original path and oscillate around it, referred as the overshooting phenomena, which will occur often in this method, as shown in Figure 4. These two methods have different advantages and disadvantages. AMV has a better stability in reciprocating motion for motion compensation, and on the other hand, CMV produced a better performance in constant movement of camera.



(a) Constant movement

(b) Reciprocating motion

FIGURE 3. Simulation results of accumulated GMV trajectory for AMV method where k is set to 0.95, and α is set to 1



(a) Constant movement

(b) Reciprocating motion

FIGURE 4. Simulation results of accumulated GMV trajectory for CMV method where k is set to 0.95, and β is set to 1

Hsu integrates these two compensation methods with fuzzy reasoning [15]. Instead, we use a functional neural fuzzy network (FNFN) to learn the characteristics of AMV and CMV under different vibrating conditions. The combination of these two distinct compensation method forms a novel means in which the incorporation of compensation weight $\gamma(t)$ significantly increases the stability of compensated video. The inputs of the FNFN are the vector difference and the difference of short-term smoothness index between AMV and CMV defined in [15]. The formula of short-term smoothness index is as follows:

$$SSI(t) = \frac{1}{p} \sum_{i=t-p-1}^{t} |GMV(i) - GMV(i-1)| \tag{7}$$

where p is window size. The output of the FNFN is adequate compensation weight $\gamma(t)$ for the two different methods. The formula of compensated motion vector is modified as follows:

$$\begin{aligned} FMV(t) &= k \times FMV(t-1) + \gamma(t) \times ((\alpha GMV(t) + (1-\alpha)GMV(t-1))) \\ &\quad +(1-\gamma(t)) \times (GMV(t) - \beta \times CMV\_I(t-1)) \\ FMV(t) &= clipper(FMV(t)) \end{aligned} \tag{8}$$

3. **A Functional Neural Fuzzy Network (FNFN).** This section briefly introduces the structure of our published FNFN model and its learning mechanism [16], which uses a nonlinear combination of input variables (FLNN). Each fuzzy rule corresponds to a sub-FLNN, comprising a functional link. Figure 5 presents the structure of the FNFN model.

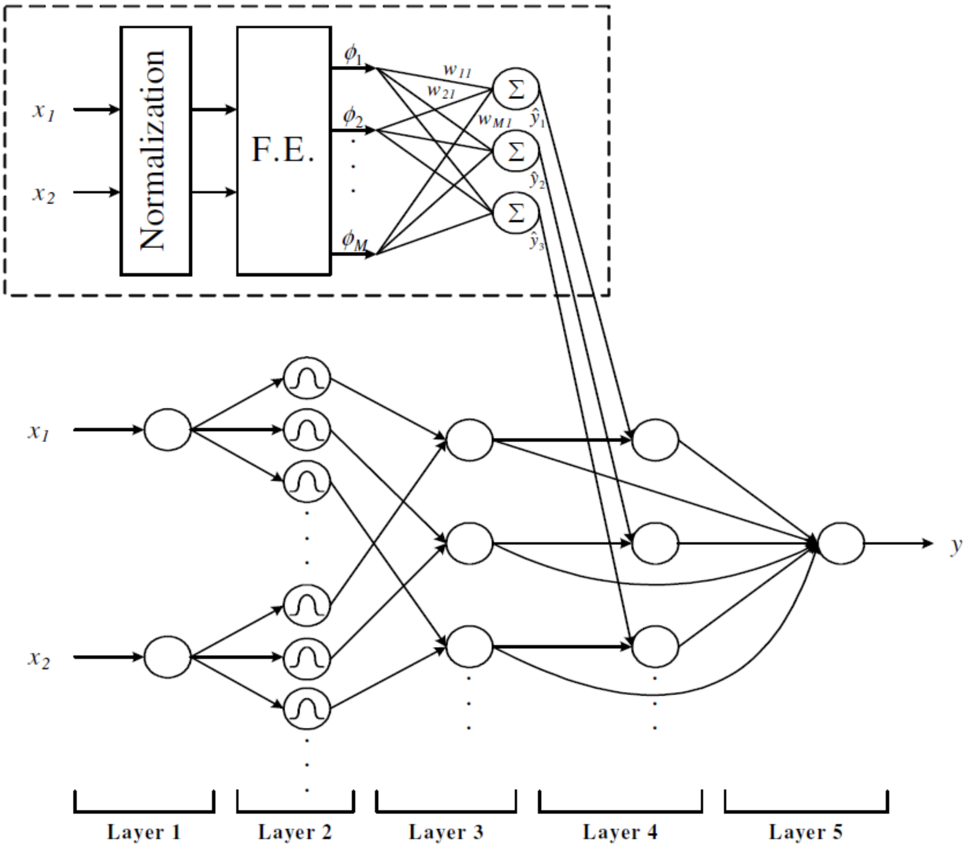


Figure 5. Structure of FNFN model

The FNFN model realizes a fuzzy if-then rule in the following form.
Rule-j

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{1j} \text{ and } x_2 \text{ is } A_{2j} \ldots \text{ and } x_i \text{ is } A_{ij} \ldots \text{ and } x_N \text{ is } A_{Nj} \\ &\text{THEN} \quad \hat{y}_j = \sum_{k=1}^{M} w_{kj}\phi_k \\ &\qquad\qquad = w_{1j}\phi_1 + w_{2j}\phi_2 + \ldots + w_{Mj}\phi_M \end{aligned} \tag{9}$$

where x_i and $\hat{y}_j$ are the input and local output variables, respectively; A_{ij} is the linguistic term of the precondition part with Gaussian membership function; N is the number of input variables; w_{kj} is the link weight of the local output; ϕ_k is the basis trigonometric function of input variables; M is the number of basis function, and Rule-j is the jth fuzzy rule.

The learning algorithm comprises a structure learning phase and a parameter learning phase. Figure 6 presents flow diagram of the learning scheme for the FNFN model. The structure learning is used to determine whether a new rule should be extracted from the training data and to determine the number of fuzzy sets in the universal of discourse of each input variable. One cluster in the input space corresponds to one potential fuzzy logic rule, in which m_{ij} and σ_{ij} represent the mean and variance of that cluster, respectively.

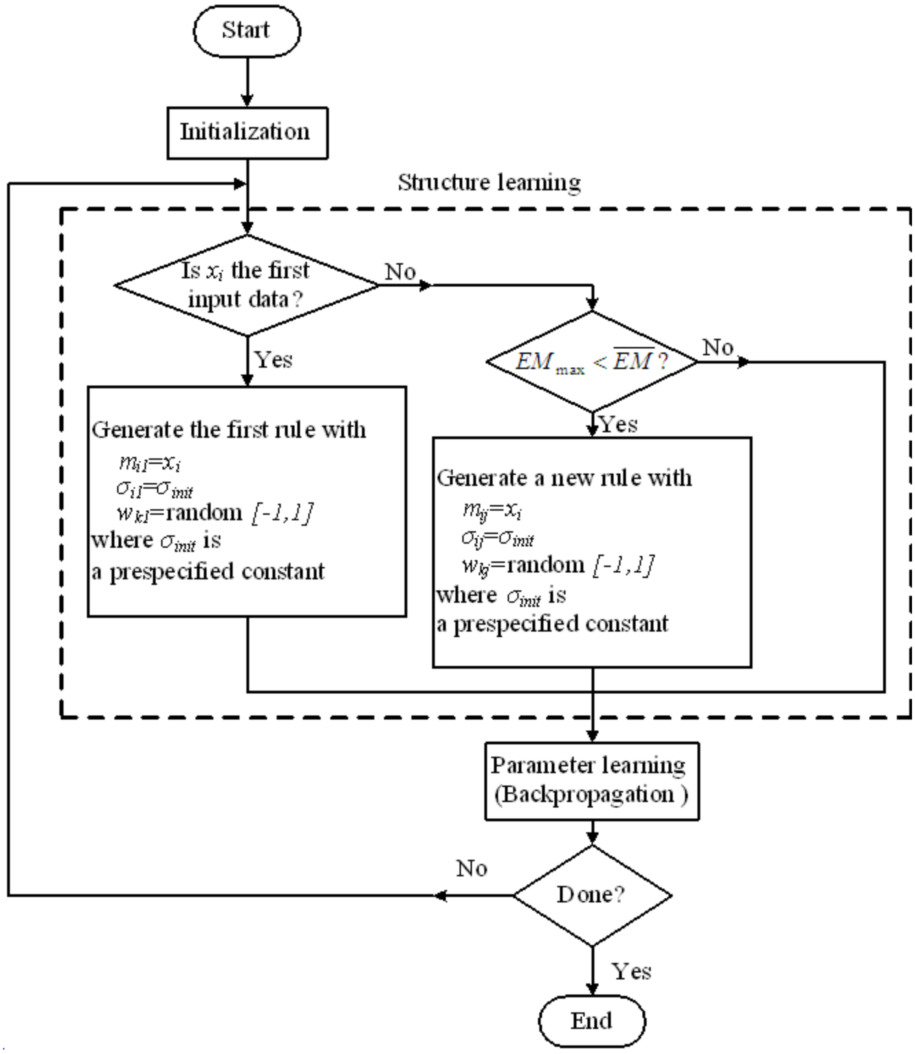

Start
Initialization
Structure learning
Is x_i the first input data?
No
Yes
$EM_{\max} < \overline{EM}$?
No
Yes
Generate the first rule with $m_{i1}=x_i$ $\sigma_{i1}=\sigma_{init}$ w_{k1}=random [-1,1] where σ_{init} is a prespecified constant
Generate a new rule with $m_{ij}=x_i$ $\sigma_{ij}=\sigma_{init}$ w_{kj}=random [-1,1] where σ_{init} is a prespecified constant
Parameter learning (Backpropagation)
Done?
No
Yes
End

Figure 6. Flow diagram of the structure/parameter learning for the FNFN model

Entropy values between data points and current membership functions are calculated to determine whether or not to add a new rule. After the structure learning phase, the network enters the parameter learning phase to adjust the parameters of the network optimally based on the same training data. The goal to minimize the cost function E is defined as

$$E(t) = \frac{1}{2}[y(t) - y^d(t)]^2 = \frac{1}{2}e^2(t) \tag{10}$$

where $y^d(t)$ is the desired output and $y(t)$ is the model output for each discrete time t.

4. **Experimental Results.** To get the adequate compensation weight $\gamma(t)$, defined in Equation (8), a functional neural fuzzy network (FNFN) is adopted to learn the characteristics of AMV and CMV under different vibrating conditions. The sets of training data used here are obtained from multiple real experiments. After the learning phase, only 11 rules are generated in the FNFN model. The obtained fuzzy rule is as follows, and all the rules are shown in Table 1.

$$\begin{aligned} &\text{If } x_1 \text{ is } \mu(-0.803121, 0.788202) \text{ and } x_2 \text{ is } \mu(1.40939, 8.02976) \\ &\text{Then} \quad \hat{y} = -0.8023x_1 - 0.614932\sin(\pi x_1) - 0.783226\cos(\pi x_1) \\ &\qquad\qquad +0.720673x_2 - 0.156326\sin(\pi x_2)0.741401\cos(\pi x_2) \end{aligned} \tag{11}$$

Table 1. Fuzzy rules obtained from FNFN

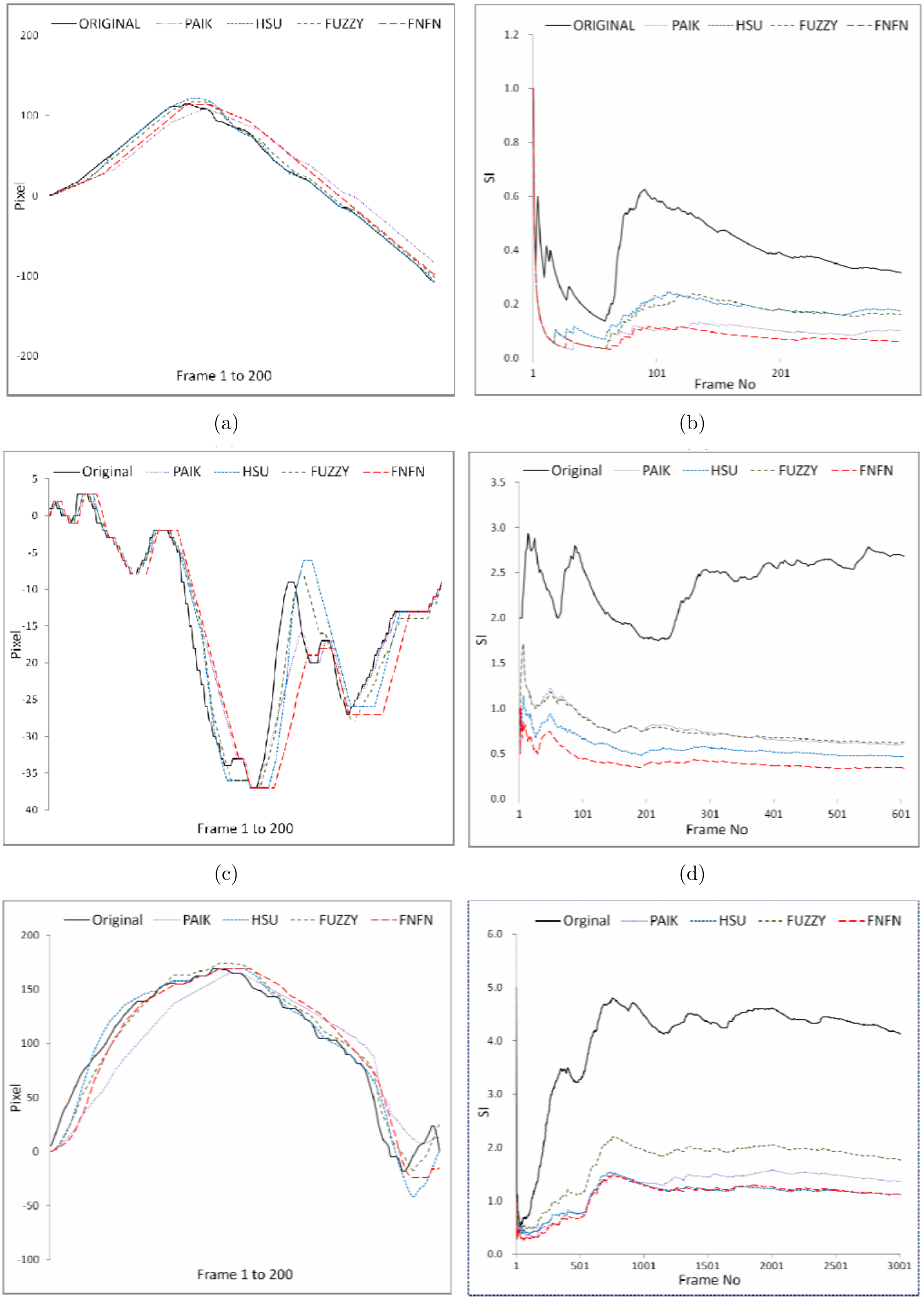
Rule	x_1		x_2		$\hat{y}$					
1	–0.803121	0.788202	1.40939	8.02976	–0.8023	–0.614932	–0.783226	0.720673	–0.156326	0.741401
2	0.634284	0.0920552	0.428304	–0.433525	0.51115	0.0251177	–1.01894	–0.176738	–1.20562	0.246145
3	1.12275	0.130162	0.241527	0.360755	0.239879	–0.808921	0.387948	–0.483494	–0.325137	–0.337143
4	1.23541	–0.596997	2.13897	–0.266948	0.352526	–0.438174	–0.177513	–0.490436	–0.215599	0.489558
5	0.495829	0.0827615	–0.826947	–1.70216	1.01667	0.686472	1.31271	–1.30856	–0.286245	1.20784
6	0.734625	–1.47996	–0.141216	0.798323	0.664787	–0.903905	1.51441	–0.0580628	0.597347	0.876388
7	0.524395	–2.28738	1.10038	0.770059	0.484585	0.858025	1.50912	0.672275	0.312475	0.00197925
8	–0.180453	1.69076	1.11272	927.13	0.395382	–0.676834	0.0132132	0.433716	0.245315	–0.130672
9	0.855605	14.8815	–0.0430686	0.177996	0.391229	0.735451	0.784755	0.02544	–0.158432	0.484081
10	0.404052	0.0628271	0.140628	0.441529	0.64156	0.995318	1.96048	1.07425	–0.59744	–0.778386
11	0.475612	0.246999	0.993545	–2.1012	0.535832	0.407712	1.55535	–0.69541	0.52416	0.971248

In order to prove that our methods can maintain robust performance, we select three video sequences of different motion patterns for testing. The content of video 1 is a boat of coastguard moving to the right at constant speed. Video 2 captures a baby crawling around a room. Video 3 shows a kart driven within a fixed zone. The resolution of the video sequences is 320 × 240, searching region l is set to 40, resolution of displayed sequence is 240×160, RPM sub-block size is 8×8, window size p of short-term smoothness index is set to 6, α is set to 0.95, and β is set to 0.015. The performance of smoothing is evaluated by the smoothness index (SI) [3] and is the average value of motion vector variations among frames. The SI is given by

$$SI = \frac{1}{N-1}\sum_{t=2}^{N} |GMV(t) - GMV(t-1)| \tag{12}$$

where t is the frame number, N is the number of all the frames.

Table 2 shows the SI values of our FNFN method and other motion compensation methods. The results show that the Paik's method [14] and the Hsu's method [3] work well in some cases. However, FNFN generates either the best results or the same result as the best method. Figure 7(a), Figure 7(c) and Figure 7(e) depict the motion trajectories of the first 200 frames of the original image and the compensated motion trajectories.



(a) (b)

(c) (d)

(e) (f)

Figure 7. Original and compensated motion trajectories of the first 200 frames in horizontal direction and SI of each frame with respect to video 1, video 2 and video 3

TABLE 2. SI of compensation methods

Video #	Total no of frame	SI				
		Original	*Paik [14]*	*Hsu [3]*	*Fuzzy [15]*	*FNFN*
1	299	0.3154	0.1007	0.1745	0.1611	0.0604
2	607	2.6810	0.6083	0.4678	0.6314	0.3421
3	3007	4.1304	1.3598	1.1174	1.7679	1.1174

In video 1, a small boat moves to the left, where the background will move right. Then the coastguard boast will appeared and move in the opposite direction. In Figure 7(a) it is evident that computation of RPM is reduced, but the results remains accurate. Only the trajectory of the first 200 frames in video 2 and 3 is depicted due to the length of the video, otherwise the trails will be too condensed to distinguish. It can be apparently seen that the trajectories of our proposed method is smoother than the other methods, is closer to the original trajectory, and causes no overshooting problems. The results from the third video prove that FNFN and Hsu's method generate the same SI. Nevertheless, when comparing the SI of the first 200 frames, FNFN has a better result. The SI of FNFN is 0.4040 whereas Hsu's SI is 0.5228. The SI diagrams shown in Figure 7(b), Figure 7(d) and Figure 7(f) prove that the compensated videos from our proposed method contain less vibrations throughout the whole video sequences.

5. **Conclusions.** In this study, we propose a new digital image stabilization method. The proposed method is done through FNFN method by learning the characteristics of AMV and CMV under different vibrating conditions. Then, the combination of the two distinct compensations method forms a novel means in which the incorporation of compensation weight, the output of FNFN, significantly increases the stability of compensated video. The use of this method will retrieve a better correction vector. This is suitable in real-time system. Experimental results indicate that the proposed method can produce smoother motion trajectories of stabilized videos, and contain less vibration throughout the whole video sequences. In future, we plan to apply the proposed method to DAS systems in our future research.

REFERENCES

[1] Y. L. Bu, *Research on Digital Image Stabilization in Infrared Homing Seeker of Single Channel-Controlled Air Defense Missile*, Master Thesis, National University of Defense Technology, 2004.

[2] A. Engelsberg and G. Schmidt, A comparative review of digital image stabilizing algorithms for mobile video communications, *IEEE Trans. on Consum. Electron.*, vol.45, pp.591-597, 1999.

[3] S. C. Hsu, S. F. Liang, K. W. Fan and C. T. Lin, A robust in-car digital image stabilization technique, *IEEE Trans. on System, Man, and Cyber., Part C: Applications and Reviews*, vol.37, no.2, pp.234-247, 2007.

[4] Y. M. Liang, H. R. Tyan, S. L. Chang, H. Y. M. Liao and S. W. Chen, Video stabilization for a camcorder mounted on a moving vehicle, *IEEE Trans. on Vehicular Technology*, vol.53, no.6, pp.1636-1648, 2004.

[5] C.-H. Chen, Y.-L. Kuo, T.-Y. Chen and J.-R. Chen, Real-time video stabilization based on motion compensation, *Proc. of the 4th International Conference on Innovative Computing, Information and Control*, Kaohsiung, pp.1495-1498, 2009.

[6] C.-H. Yeh, M.-T. Wu and S.-J. Chern, Coarse-to-fine partial distortion search algorithm for motion estimation, *International Journal of Innovative Computing, Information and Control*, vol.5, no.9, pp.2523-2530, 2009.

[7] C. Wang, J. H. Kim, K. Y. Byun, J. Ni and S. J. Ko, Robust digital image stabilization using Kalman filter, *IEEE Trans. on Consumer Electronics*, vol.55, no.1, pp.6-13, 2009.

[8] J. Cai and R. Walker, Robust video stabilization algorithm using feature point selection and delta optical flow, *IET Comput. Vis.*, vol.3, no.4, pp.176-188, 2009.

[9] J. Yang, D. Schonfeld and M. Mohamed, Robust video stabilization based on particle filter tracking of projected camera motion, *IEEE Trans. on Circuits and Systems for Video Technology*, vol.19, no.7, 2009.

[10] S. Erturk, Image sequence stabilisation: Motion vector integration (MVI) versus frame position smoothing (FPS), *Proc. of the 2nd International Symposium on Image and Signal Processing and Analysis*, pp.266-271, 2001.

[11] Y. Zhou, Real-time electronic digital image stabilization system based on multi-core computation, *Proc. of the 9th International Conference on Electronic Measurement & Instruments*, pp.4-308-4-314, 2009.

[12] M. K. Gullu, E. Yaman and S. Erturk, Image sequence stabilization using fuzzy adaptive Kalman filtering, *Electronic Letters*, vol.39, no.5, pp.429-431, 2003.

[13] S. Erturk, Digital image stabilization with sub-image phrase correclation based global motion estimation, *IEEE Trans. on Consumer Electronics*, vol.49, no.4, pp.1320-1325, 2003.

[14] J. K. Paik, Y. C. Park and D. W. Kim, An adaptive motion decision system for digital image stabilizer based on edge pattern matching, *IEEE Trans. on Consum. Electron.*, vol.38, no.3, pp.607-616, 1992.

[15] S. C. Hsu and C. T. Lin, Fuzzy inference applied to digital image stabilization techniques, *Image and Recognition*, vol.13, no.3, pp.55-66, 2007.

[16] C. Lin, Y. Liu and C. Lee, An efficient neural fuzzy network based on immune particle swarm optimization for prediction and control applications, *International Journal of Innovative Computing, Information and Control*, vol.4, no.7, pp.1711-1722, 2008.