

Application of Adaptive Self-Organizing CMAC_GBF to Aircraft Landing System

Chung-Ju Cheng

Department of Communications, Navigation and Control
Engineering, National Taiwan Ocean University
Keelung, Taiwan
M98670011@mail.ntou.edu.tw

Jih-Gau Juang

Department of Communications, Navigation and Control
Engineering, National Taiwan Ocean University
Keelung, Taiwan
jgjuang@mail.ntou.edu.tw

Abstract—This paper presents an intelligent control scheme that uses an adaptive self-organizing cerebellar model articulation controller with general basis function (CMAC_GBF) in aircraft automatic landing system (ALS). During landing phase if the flight condition is out of the desired or preset environments, such as encountering severe turbulence, the ALS is disabled. The proposed intelligent controller can act as an experienced pilot and guide the aircraft landed safely in the serious condition. Lyapunov theory is applied to obtain adaptive learning rule and stability analysis is also provided. The performance of the proposed controller is better than conventional CMAC_GBF.

Keywords—self-organizing CMAC; automatic landing system; general basis function; turbulence; PID control

I. INTRODUCTION

On March 1, 2008, at Hamburg airport, a Lufthansa Airbus A320 tried to land in crosswind conditions which exceeded the limit for the aircraft and made the left wing touch ground. The pilots then performed a go around and successfully saved the aircraft from crashing. The atmospheric disturbance causes the problem of flight safety and reduces the flying quality. According to International Air Transport Association (IATA) classification, aircraft accident contributing factor is divided into five categories as human factor, technical, environment, organization factor and insufficient. An aircraft accident survey of 19562 aircraft accidents from 1994 through 2003, weather was a contributing factor. The percentage of weather related to total accidents is 21.3% [1]. Percentage of weather related accidents by weather conditions is shown in Fig. 1. The first automatic landing system (ALS) was developed in England during 1965. Since then, most aircraft have had this system installed. The ALS relies on the instrument landing system (ILS) to guide the aircraft into the proper altitude, position, and approach angle during the landing phase. According to Federal Aviation Administration (FAA) regulations [2], environmental conditions considered in the determination of dispersion limits are: headwinds up to 25 knots, tailwinds up to 10 knots, crosswinds up to 15 knots, moderate turbulence, and wind shear of 8 knots per 100 feet from 200 feet to touchdown. Conventional automatic landing systems can provide a smooth landing, which is essential to the comfort of passengers. However, these

systems work only within a specified operational safety envelope. When the conditions, such as turbulence or wind shear, are beyond the envelope, they often cannot be used. Most conventional control laws generated by the ALS are based on the gain scheduling method [3]. Control parameters are preset for different flight conditions within a specified safety envelope, which is relative defined by FAA regulation. When the conditions, such as turbulence, are beyond the envelope, the ALS is disabled and manual operation is engaged. An inexperienced pilot may not be able to guide the aircraft safely. Therefore it is desirable to develop an intelligent ALS that expands the operational envelope to include more safe responses under a wider range of conditions. The goal of this study is to show that the proposed intelligent ALS can relieve human operation and guide the aircraft to a safe landing in wind-disturbance environment.

In recent years, intelligent control is more and more popular in control engineering applications. Many intelligent concepts have been applied into various scientific and engineering researches. There are also obvious achievements in flight control domain [4]-[9]. CMAC_GBF has attracted researchers' attentions for the following advantages: rapid learning speed, guarantee of learning convergence, good generalization, and low computational load. A simplified addressing technique procedure, S_CMAC_GBF, is presented by C.T. Chiang [10]. The simplified model not only provides fixed addressing procedure for CMAC_GBF but also reduces the usage of memory. The S_CMAC_GBF still possesses the same learning convergence characteristics of CMAC_GBF, but with more enhanced accurate system performance. The calculation speed of S_CMAC_GBF is faster than conventional CMAC_GBF. It is very useful in on-line control system. But it is hard to decide the number of layers and the value of learning rate of the S_CMAC_GBF. In this study, we utilize a self-organizing CMAC_GBF (SOCM) [11] by judging error rate to add or delete layer and introduce a SOCM with adaptive learning rate to automatic landing system. Because adaptive control theory has been successfully employed in various fields, more and more researchers integrated the adaptive concept into the intelligent control system. The performance of the intelligent ALS under severe environment can be improved by the advantages of the SOCM with adaptive learning rate, and

the advantages are local generalization, rapid learning convergence, and environment adaptive capability.

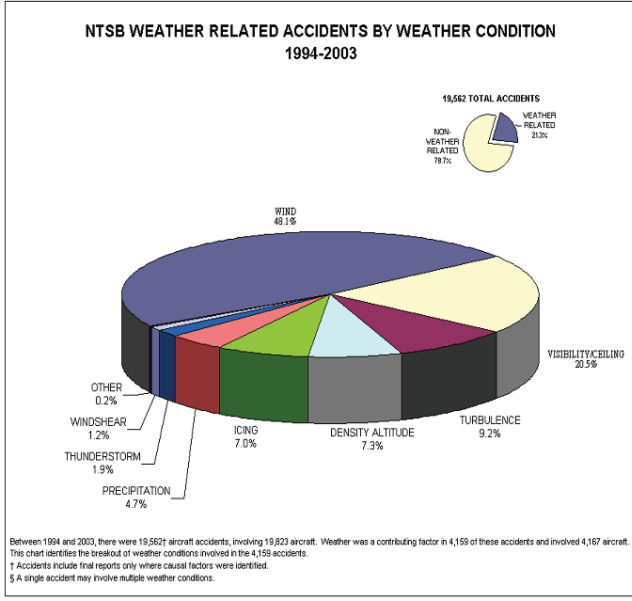


Figure 1. NTSB weather related accidents by weather condition.

II. LANDING SYSTEM MODEL

At the aircraft landing phase, the pilot descends from the cruise altitude to an altitude of approximately 1200 ft above the ground. The pilot then positions the aircraft so that the aircraft is on a heading towards the runway centerline. When the aircraft approaches the outer airport marker, which is about 4 nautical miles from the runway, the glide path signal is intercepted, as shown in Fig. 2 [12]. As the airplane descends along the glide path, its pitch, attitude, and speed must be controlled. The descent rate is about 10 ft/sec and the pitch angle is between -5 to +5 degrees. Finally, as the airplane descends 20 to 70 feet above the ground, the glide path control system is disengaged and a flare maneuver is executed. The vertical descent rate is decreased to 2 ft/sec so that the landing gear may be able to dissipate the energy of the impact at landing. The pitch angle of the airplane is then adjusted, between 0 to 5 degrees for most aircraft, which allows a soft touchdown on the runway surface.

A simplified model of a commercial aircraft that moves only in the longitudinal and vertical plane is used in the simulations for implementation ease [7]. The motion equations of the aircraft are given as follows:

$$\Delta \dot{u} = X_u(\Delta u - u_g) + X_w(\Delta w - w_g) + X_q \Delta q - g \left(\frac{\pi}{180} \right) \cos(\gamma_0) \Delta \theta + Z_E \delta_E + Z_T \delta_T \quad (1)$$

$$\Delta \dot{w} = Z_u(\Delta u - u_g) + Z_w(\Delta w - w_g) + (Z_q - \frac{\pi}{180} U_0) \Delta q - g \left(\frac{\pi}{180} \right) \sin(\gamma_0) \Delta \theta + Z_E \delta_E + Z_T \delta_T \quad (2)$$

$$\Delta \dot{q} = M_u(\Delta u - u_g) + M_w(\Delta w - w_g) + M_q \Delta q + M_E \delta_E + M_T \delta_T \quad (3)$$

$$\Delta \dot{\theta} = \Delta q \quad (4)$$

$$\Delta \dot{h} = -\Delta w + \frac{\pi}{180} U_0 \Delta \theta \quad (5)$$

where u is the aircraft longitudinal velocity (ft/sec), w is the aircraft vertical velocity (ft/sec), q is the pitch rate (rate/sec), θ is the pitch angle (deg), h is the aircraft altitude (ft), δ_E is the incremental elevator angle (deg), δ_T is the throttle setting (ft/sec), γ_0 is the flight path angle (-3deg), and g is the gravity (32.2 ft/sec²). The parameters X_i , Z_i and M_i are the stability and control derivatives.

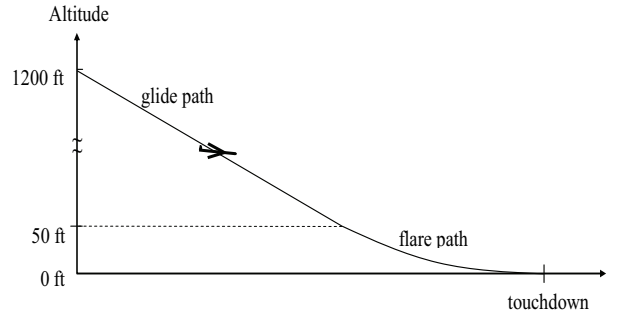


Figure 2. Glide path and flare path.

To make the ALS more intelligent, reliable wind profiles are necessary. Two spectral turbulence models by von Karman and Dryden are mostly used for aircraft response studies. In this study the Dryden form [7] was used for its demonstration ease. The model is given by:

$$u_g = u_{gc} + N(0,1) \sqrt{\frac{1}{\Delta t} \left(\frac{\sigma_u \sqrt{2a_u}}{s + a_u} \right)} \quad (6)$$

$$w_g = N(0,1) \sqrt{\frac{1}{\Delta t} \left(\frac{\sigma_w \sqrt{3a_w}(s + b_w)}{(s + a_u)^2} \right)} \quad (7)$$

where $u_{gc} = -u_{wind510} \left[1 + \frac{\ln(h/510)}{\ln(51)} \right]$, $a_u = \frac{U_o}{L_u}$, $L_w = h$, $a_u = \frac{U_o}{L_u}$, $a_w = \frac{U_o}{L_w}$, $b_w = \frac{U_o}{L_w \sqrt{3}}$, $L_u = 100h^{1/3}$ for $h > 230$, $L_u = 600$ for $h \leq 230$, $\sigma_u = 0.2|u_{gc}|$ for $h > 500$, $\sigma_w = 0.2|u_{gc}|(0.5 + 0.00098 * h)$ for $0 \leq h \leq 500$. The parameters are: u_g is the horizontal wind velocity (ft/sec), w_g is the vertical wind velocity (ft/sec), U_o is the nominal aircraft speed (ft/sec), $u_{wind510}$ is the wind speed at 510 ft altitude, L_u and L_w are scale lengths (ft), σ_u and σ_w are RMS values of turbulence velocity (ft/sec), Δt is the simulation time step (sec), $N(0,1)$ is the Gaussian white noise with zero mean and unity standards deviation, u_{gc} is the constant component of u_g , and h is the aircraft altitude (ft). Fig. 3 shows a turbulence profile with a wind speed of 30 ft/sec at 510 ft altitude.

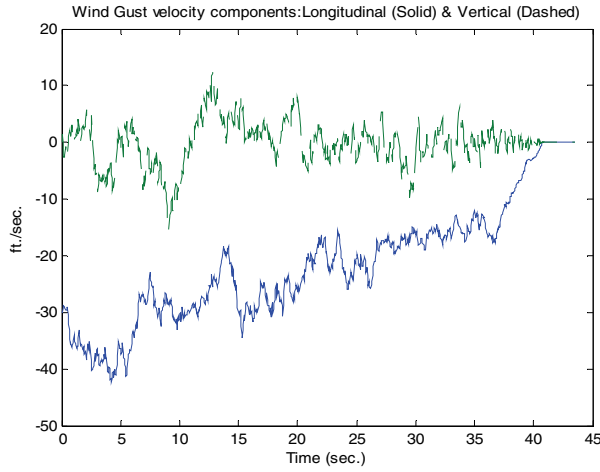


Figure 3. Turbulence profile.

III. INTELLIGENT CONTROL

A simplified PID control structure is applied to aircraft landing control as shown in Fig. 4. Its inputs consist of altitude and altitude rate commands along with aircraft altitude and altitude rate. Via aircraft landing controller we can obtain the pitch command θ_c . Then, the pitch autopilot is controlled by pitch command. The pitch autopilot is shown in Fig. 5. Detail descriptions can be found in [7]. In order to enable aircraft to land more steady when an aircraft approaches to the flare path, a constant pitch angle is added to the controller. In general, the PID controller is simple and effective but there are some drawbacks such as apparent overshoot and sensitive to external noise and disturbance. When severe turbulence is encountered the PID controller may not be able to guide the aircraft to land safely. With SOCM compensator the proposed controller can overcome these disadvantages. The control scheme utilizes a conventional PID controller to stabilize the controlled system and train the SOCM to provide precise control. The gains of PID controller are adjusted based on experiences, what it provides are tolerable solutions, not desired solutions. The SOCM can effectively meliorate these disadvantages.

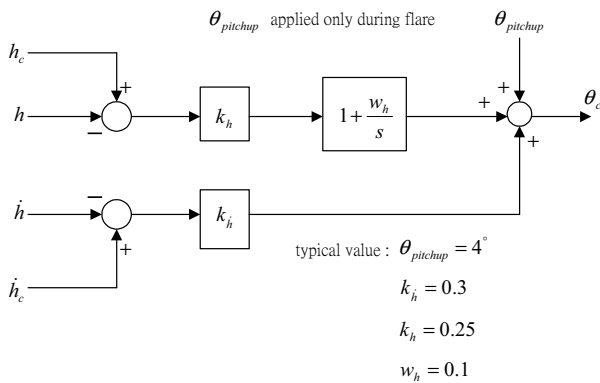


Figure 4. PID-controller.

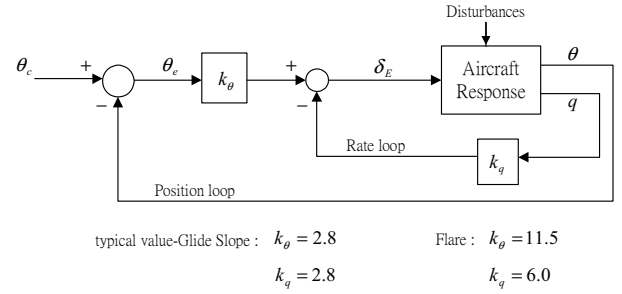


Figure 5. Pitch autopilot.

The output of CMAC is the sum of several values of hypercube and the input vector is the values of the aircraft status. The overall control scheme is described in Fig. 6, in which the control signal U is the sum of the PID controller output and the SOCM output. The inputs of the SOCM and PID controller are: altitude, altitude command, altitude rate, and altitude rate command. The PID controller provides tolerable solutions. In each time step k , the SOCM involves a recall process and a learning process. In the recall process, it uses the desired system output of the next time step and the actual system output as the address to generate the control signal U_{SOCM} . In the learning process, the control signal of the pitch autopilot, U , is treated as a desired output. It is used to modify the weights of SOCM stored at desired location which is addressed by the actual system output and the system output of the next time step. The output of the SOCM is the compensation for pitch command. When the wind turbulence is too strong, the ALS can not control the aircraft to land safely. In here we use a SOCM control scheme to improve the ability of turbulence resistance of the ALS.

A. CMAC_GBF

The CMAC represents one kind of associative memory technique. In the addressing technique, each input space (state variable) is quantized and the output space is divided into discrete states. A quantized input vector specifies a discrete state and is used to generate addresses for retrieving information from memory for this state. There is no big difference on the structure and quantization method and the mapping process between CMAC-GBF and original CMAC. The difference is that while the input vector maps the hypercube by way of the association in Fig. 7, the stored data can be the differential basis function value. Fig. 8 illustrates the block division of the CMAC-GBF for a two-variable case. This simple example has two state variables (x_1 and x_2) with each quantized into four discrete regions, called blocks. For instance, x_1 can be divided into A, B, C , and D and x_2 can be divided into a, b, c , and d . Areas $Aa, Ab, \dots, Dd$ formed by quantized regions are called hypercubes. By shifting each block a small interval (called an element) different hypercubes can be obtained. For instance, E, F, G , and H in the second row for x_1 and e, f, g , and h in the second row for x_2 are possible shifted regions. $Ee, Ef, \dots, Hh$ are new hypercubes from the shifted regions. With this kind of decomposition, if there are N_e elements in a complete block, we will have N_e layers of hypercubes. In the given example there are four

layers as shown in Fig. 8. The state is covered by N_e different hypercubes, one from each layer. In the original CMAC, a constant value is assigned to each hypercube. By inputting these values into hypercubes, no matter what the positions are, they are treated as memory addresses. Therefore, the output is irrelevant to input and the differential of output to input is also not obtainable. Thus, in the CMAC-GBF, a general basis function (GBF) is used to replace constants in every hypercube. In this way, the positions of input vectors are related to output value, and the learning precision is also increased.

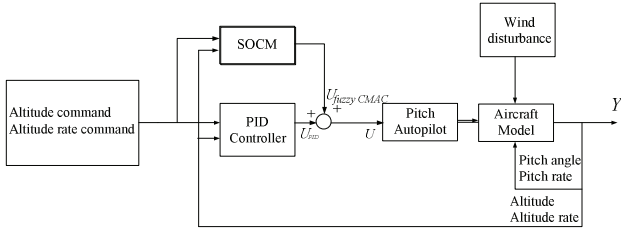


Figure 6. The SOCM control scheme.

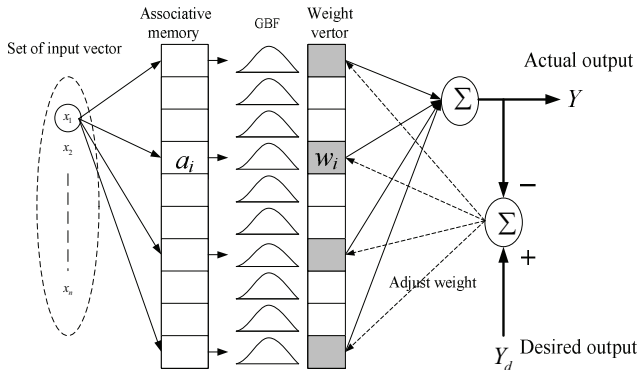


Figure 7. The structure of CMAC-GBF.

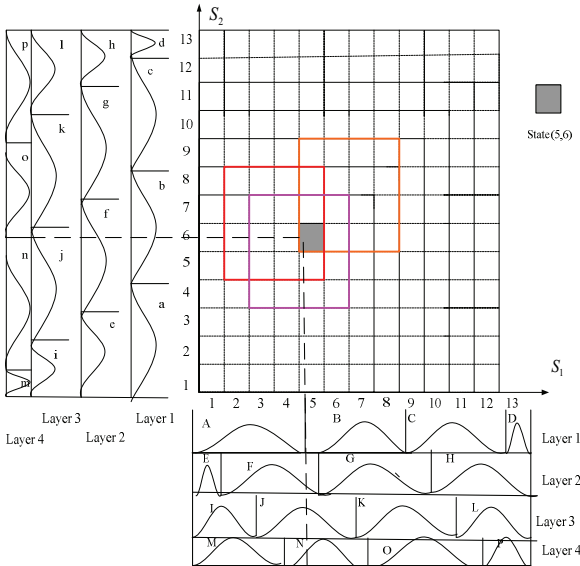


Figure 8. Block division of CMAC-GBF for a two-variable case.

In the CMAC-GBF, the content of hypercube can be expressed as $w_i(x_s) = v_i b_i(x_s)$, where $b_i(x_s)$ is a general basis function and v_i is a weight to be obtained through learning. The output of the CMAC-GBF can be written as below.

$$y(x_s) = a_s^T w(x_s) = \sum_i^N [a_{s,i} v_i b_i(x_s)] \quad (8)$$

where $a_s^T = [a_{s,1} \ a_{s,2} \dots a_{s,N}]$ is a basis function selection vector (or so-called association cells) and $W^T = [w_1 \ w_2 \dots w_N]$ is the vector of memory contents of CMAC-GBF. The Gaussian functions are employed as the basis functions

$$b_i(x_s) \equiv \prod_{j=1}^N \phi_{ij}(x_{s,j}) \quad (9)$$

with

$$\phi_{ij}(x_{s,j}) = \exp\left[-\frac{(x_{s,j} - m_{ij})^2}{\sigma_{ij}^2}\right] \quad (10)$$

m_{ij} is the mean, σ_{ij} is the variance and N is the number of variables in the target function. Consequently, the weight function is

$$w_i(x_s) = v_i \prod_{j=1}^N \phi_{ij}(x_{s,j}) \quad (11)$$

The output from the CMAC with Gaussian basis functions can be mathematically expressed as

$$E = \frac{1}{2} (\hat{y}_s - y_s)^2 \quad (12)$$

The updated amount for v_i can be set to

$$\begin{aligned} \Delta v_i &= -\frac{\alpha_v}{N_e} \frac{\partial E}{\partial v_i} \\ &= -\frac{\alpha_v}{N_e} \frac{\partial E}{\partial e_s} \frac{\partial e_s}{\partial y(x_s)} \frac{\partial y(x_s)}{\partial w_i(x_s)} \frac{\partial w_i(x_s)}{\partial v_i} \\ &= \frac{\alpha_v}{N_e} (\hat{y}_s - a_s^T w(x_s)) a_{s,i} b_i(x_s) \end{aligned} \quad (13)$$

where α_v is the learning rate for v . The mean and variables of the Gaussian function can also be adjusted to increase the approximation capability. The updating rules for these parameters can be derived as

$$\begin{aligned} \Delta m_{ij} &= -\frac{\alpha_m}{N_e} \frac{\partial E}{\partial m_{ij}} \\ &= -\frac{\alpha_m}{N_e} \frac{\partial E}{\partial e_s} \frac{\partial e_s}{\partial y(x_s)} \frac{\partial y(x_s)}{\partial w_i(x_s)} \frac{\partial w_i(x_s)}{\partial \phi_{ij}} \frac{\partial \phi_{ij}}{\partial m_{ij}} \\ &= \frac{\alpha_m}{N_e} (\hat{y}_s - a_s^T w(x_s)) a_{s,i} v_i b_i(x_s) \frac{2(x_{s,i} - m_{ij})}{\sigma_{ij}^2} \end{aligned} \quad (14)$$

$$\begin{aligned}
\Delta\sigma_{ij} &= -\frac{\alpha_\sigma}{N_e} \frac{\partial E}{\partial \sigma_{ij}} \\
&= -\frac{\alpha_\sigma}{N_e} \frac{\partial E}{\partial e_s} \frac{\partial e_s}{\partial y(x_s)} \frac{\partial y(x_s)}{\partial w_i(x_s)} \frac{\partial w_i(x_s)}{\partial \phi_{ij}} \frac{\partial \phi_{ij}}{\partial \sigma_{ij}} \\
&= \frac{\alpha_\sigma}{N_e} (\hat{y}_s - \mathbf{a}_s^T \mathbf{w}(x_s)) \alpha_{s,i} v_i b_i(x_s) \left(\frac{2(x_{s,i} - m_{ij})^2}{\sigma_{ij}^3} \right)
\end{aligned} \quad (15)$$

where α_m and α_σ are the learning rates for mean and variance.

B. Self Organizing CMAC_GBF (SOCM)

The learning method for S_CMAL_GBF [10] and CMAC_GBF are the same, but they are different in the structures. The quantified method of input variables is different from the quantified conventional CMAC_GBF. In S_CMAL_GBF, each layer has only one block, and each block has different basis function. The weighting of each layer can be obtained by multiplying basis function of each layer with its height. Fig. 9 shows a two variables case, from recall process of S_CMAL_GBF, the output is the sum of the weights of 20 layers. The learning method of S_CMAL_GBF is just like the one for CMAC_GBF. In S_CMAL_GBF, the output equation is the same as CMAC_GBF. $\alpha_{s,i}$ (element of selection vector) is either 1 (selected) or 0 (not selected) for CMAC_GBF, but in S_CMAL_GBF each element in the hypercube selection vector is equal to 1. It is noted that the learning rules of S_CMAL_GBF are also the same as CMAC_GBF. Adjust the required parameters in the recall process, and after repeated learning processes, the output of S_CMAL_GBF can reach the same accurate level as of CMAC_GBF.

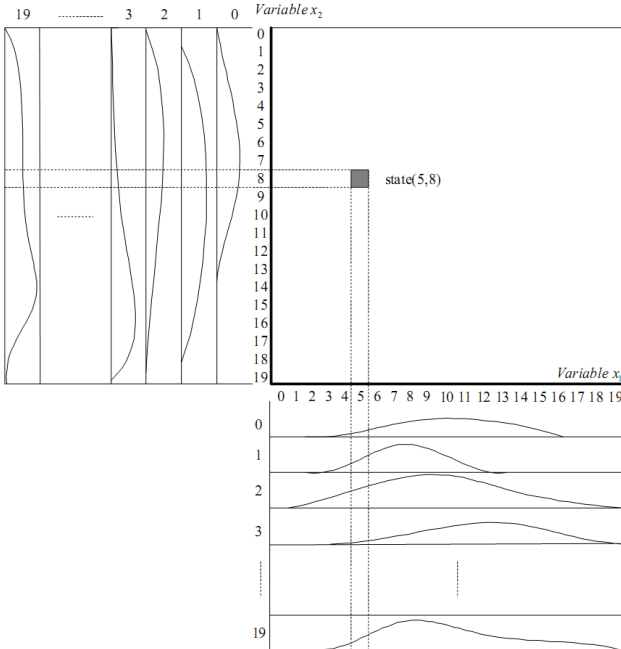


Figure 9. Block division of S_CMAL_GBF for two-variable case [10].

The difference of S_CMAL_GBF and SOCM [11] is that SOCM can create a new layer and delete a layer. The first process of the structure learning is to determine whether to add a new layer in association memory and to create its hypercube and weight memory, simultaneously. In the layer generating process, the mathematical description of the existing layers can be expressed as clusters. The firing strength of a rule for each incoming data can be represented as the degree to which the incoming data belong to the cluster. If a new input data falls within the boundary of clusters, SOCM will not generate a new layer but update parameters of the existing rules. Before SOCM decides which layer should be added or deleted, we have to choose the upper threshold (UT) (layer should be added) and lower threshold (LT) (layer should be deleted). The error rate can be expressed as

$$E_{rate} = \left| \frac{E(t)}{E(t-1)} \right| \quad (16)$$

$$\text{If } E_{rate} > UT \text{ then } n_k(t+1) = n_k(t) + 1 \quad (17)$$

where $n_k(t)$ is the number of the existing layers at time t . In the meanwhile, for the new layer, the initial mean and variance of Gaussian basis function in association memory space are defined as $m_{ink} = x_s$ and $\sigma_{ink}(t) = \sigma_{ink}(t-1)$, respectively.

$$\text{If } E_{rate} < LT \text{ then } n_k(t+1) = n_k(t) - 1 \quad (18)$$

We choose the least contribution layer of the SOCM output to be deleted. Using the ratio of the output from every layer

$$LO_i = \frac{v_i b_i(x_s)}{v_{SOCM}}, i = 1, \dots, n_k \quad (19)$$

The layer which has minimum output ($\min_{i=1 \sim n_k} LO_i$) should be deleted.

IV. STABILITY ANALYSIS OF SOCM

In order to improve the shortcoming of conventional CMAC-GBF on the crisp relation, adaptive learning rule is introduced. Here, we use the discrete-time Lyapunov function to define the adaptive learning rate α . Let the tracking error e_s be

$$e_s(t) = \hat{y}_s - y_s \quad (20)$$

where t is the time index. A discrete-type Lyapunov function can be expressed as

$$V = \frac{1}{2} e_s^2(t) \quad (21)$$

Thus, the change in the Lyapunov function is obtained by

$$\Delta V = V(t+1) - V(t) = \frac{1}{2} [e_s^2(t+1) - e_s^2(t)] \quad (22)$$

The error difference can be represented by [13]

$$\Delta e_s(t) = e_s(t+1) - e_s(t) \approx \begin{bmatrix} \frac{\partial e_s(t)}{\partial v_i(t)} & \frac{\partial e_s(t)}{\partial m_{ij}(t)} & \frac{\partial e_s(t)}{\partial \sigma_{ij}(t)} \end{bmatrix} \begin{bmatrix} \Delta v_i(t) \\ \Delta m_{ij}(t) \\ \Delta \sigma_{ij}(t) \end{bmatrix} \quad (23)$$

From (13) to (15) we have

$$\frac{\partial e_s(t)}{\partial v_i(t)} = \frac{\partial e_s}{\partial y} \frac{\partial y}{\partial w_i} \frac{\partial w_i}{\partial v_i} = -a_{s,i} b_i(x_s) \quad (24)$$

$$\frac{\partial e_s(t)}{\partial m_{ij}(t)} = \frac{\partial e_s}{\partial y} \frac{\partial y}{\partial w_i} \frac{\partial w_i}{\partial \phi_{ij}} \frac{\partial \phi_{ij}}{\partial m_{ij}} = -a_{s,i} v_i b_i(x_s) \frac{2(x_{s,i} - m_{ij})}{\sigma_{ij}^2} \quad (25)$$

$$\frac{\partial e_s(t)}{\partial \sigma_{ij}(t)} = \frac{\partial e_s}{\partial y} \frac{\partial y}{\partial w_i} \frac{\partial w_i}{\partial \phi_{ij}} \frac{\partial \phi_{ij}}{\partial \sigma_{ij}} = -a_{s,i} v_i b_i(x_s) \frac{2(x_{s,i} - m_{ij})^2}{\sigma_{ij}^3} \quad (26)$$

From (22) to (24), with respect to v_i , we have

$$\begin{aligned} \Delta V &= \frac{1}{2} [e_s^2(t+1) - e_s^2(t)] \\ &= \frac{1}{2} [e_s(t+1) - e_s(t)] [e_s(t+1) + e_s(t)] \\ &= \frac{1}{2} [\Delta e_s(t)] [2 \cdot e_s(t) + \Delta e_s(t)] \\ &= \Delta e_s(t) \left[e_s(t) + \frac{1}{2} \Delta e_s(t) \right] \\ &= \left[-a_{s,i} b_i(x_s) \frac{\alpha_v}{N_e} e_s(t) a_{s,i} b_i(x_s) \right] \\ &\quad \left\{ e_s(t) + \frac{1}{2} \left[-a_{s,i} b_i(x_s) \frac{\alpha_v}{N_e} e_s(t) a_{s,i} b_i(x_s) \right] \right\} \\ &= \left\{ -\frac{1}{2} \frac{\alpha_v}{N_e} e_s(t) [a_{s,i} b_i(x_s)]^2 \right\} \\ &\quad \left[2e_s(t) - \frac{\alpha_v}{N_e} e_s(t) [a_{s,i} b_i(x_s)]^2 \right] \\ &= \left[-\frac{1}{2} \frac{\alpha_v}{N_e} e_s^2(t) [a_{s,i} b_i(x_s)]^2 \right] \left[2 - \frac{\alpha_v}{N_e} [a_{s,i} b_i(x_s)]^2 \right] \quad (27) \end{aligned}$$

Let $\left[2 - \frac{\alpha_v}{N_e} [a_{s,i} b_i(x_s)]^2 \right] > 0$ then $\Delta V < 0$, i.e., we can select the learning rate α_v in the following range

$$\frac{2N_e}{[a_{s,i} b_i(x_s)]^2} > \alpha_v > 0 \quad (28)$$

From (22), (23), and (25), with respect to m_{ij} , we have

$$\begin{aligned} \Delta V &= \frac{1}{2} [e_s^2(t+1) - e_s^2(t)] = \Delta e_s(t) \left[e_s(t) + \frac{1}{2} \Delta e_s(t) \right] \\ &= \left\{ -\frac{1}{2} \frac{\alpha_m}{N_e} e_s(t) \left[a_{s,i} v_i b_i(x_s) \frac{2(x_{s,i} - m_{ij})}{\sigma_{ij}^2} \right]^2 \right\} \\ &\quad \left[2e_s(t) - \frac{\alpha_m}{N_e} e_s(t) \left[a_{s,i} v_i b_i(x_s) \frac{2(x_{s,i} - m_{ij})}{\sigma_{ij}^2} \right]^2 \right] \\ &= \left\{ -\frac{1}{2} \frac{\alpha_m}{N_e} e_s^2(t) \left[a_{s,i} v_i b_i(x_s) \frac{2(x_{s,i} - m_{ij})}{\sigma_{ij}^2} \right]^2 \right\} \\ &\quad \left[2 - \frac{\alpha_m}{N_e} \left[a_{s,i} v_i b_i(x_s) \frac{2(x_{s,i} - m_{ij})}{\sigma_{ij}^2} \right]^2 \right] \end{aligned} \quad (29)$$

Let $\left[2 - \frac{\alpha_m}{N_e} \left[a_{s,i} v_i b_i(x_s) \frac{2(x_{s,i} - m_{ij})}{\sigma_{ij}^2} \right]^2 \right] > 0$ then $\Delta V < 0$, i.e., we can select the learning rate α_m in the following range

$$\frac{2N_e}{\left[a_{s,i} v_i b_i(x_s) \frac{2(x_{s,i} - m_{ij})}{\sigma_{ij}^2} \right]^2} > \alpha_m > 0 \quad (30)$$

From (22), (24), and (26), with respect to σ_{ij} , we have

$$\begin{aligned} \Delta V &= \frac{1}{2} [e_s^2(t+1) - e_s^2(t)] = \Delta e_s(t) \left[e_s(t) + \frac{1}{2} \Delta e_s(t) \right] \\ &= \left\{ -\frac{1}{2} \frac{\alpha_\sigma}{N_e} e_s(t) \left[a_{s,i} v_i b_i(x_s) \left(\frac{2(x_{s,i} - m_{ij})^2}{\sigma_{ij}^3} \right) \right]^2 \right\} \\ &\quad \left[2e_s(t) - \frac{\alpha_\sigma}{N_e} e_s(t) \left[a_{s,i} v_i b_i(x_s) \left(\frac{2(x_{s,i} - m_{ij})^2}{\sigma_{ij}^3} \right) \right]^2 \right] \\ &= \left\{ -\frac{1}{2} \frac{\alpha_\sigma}{N_e} e_s^2(t) \left[a_{s,i} v_i b_i(x_s) \left(\frac{2(x_{s,i} - m_{ij})^2}{\sigma_{ij}^3} \right) \right]^2 \right\} \\ &\quad \left[2 - \frac{\alpha_\sigma}{N_e} \left[a_{s,i} v_i b_i(x_s) \left(\frac{2(x_{s,i} - m_{ij})^2}{\sigma_{ij}^3} \right) \right]^2 \right] \end{aligned} \quad (31)$$

Let $\left[2 - \frac{\alpha_\sigma}{N_e} \left[a_{s,i} v_i b_i(x_s) \left(\frac{2(x_{s,i} - m_{ij})^2}{\sigma_{ij}^3} \right) \right]^2 \right] > 0$ then $\Delta V < 0$, i.e., we can select the learning rate α_σ in the following range

$$\frac{2N_e}{\left[a_{s,i} v_i b_i(x_s) \left(\frac{2(x_{s,i} - m_{ij})^2}{\sigma_{ij}^3} \right) \right]^2} > \alpha_\sigma > 0 \quad (32)$$

ΔV becomes negative definite. This implies that $E(t) \rightarrow 0$ for $t \rightarrow \infty$. The convergence of the adaptive SOCM learning process is then guaranteed. The aircraft landing control system is locally asymptotically stable.

V. SIMULATIONS

The aircraft starts the initial states of the ALS as follows: the flight height is 500 ft, the horizontal position before touching the ground is 9240 ft, the flight angle is -3 degrees, the speed of the aircraft is 234.7 ft/sec. Successful touchdown landing conditions are defined as follows:

$$\begin{aligned} -3 \leq \dot{h}_{TD} \text{ ft/sec} \leq 0, \quad 200 \leq V_{TD}(t) \text{ ft/sec} \leq 270, \\ -300 \leq x_{TD} \text{ ft} \leq 1000, \quad -1 \leq \theta_{TD} \text{ degree} \leq 5, \end{aligned}$$

where T is the time at touch down, $\dot{h}_{TD}$ is the vertical speed of the aircraft at touch down, x_{TD} is the position at touch down, $V_{TD}(t)$ is horizontal speed, θ_{TD} is the pitch angle at touch down.

Table I shows the result of conventional PID controller with different wind turbulence speeds. The conventional controller can only successfully guide an aircraft flying through wind speeds of 0 ft/sec to 30 ft/sec. The situations at wind turbulence 30 ft/sec are that the pitch angle is -0.1115 degrees, vertical speed is -2.1704/sec, horizontal velocity is 234.6779 ft/sec, and horizontal position at touchdown is 902.6000 ft, as shown in Fig. 10 to Fig. 12. If the wind speed is higher than 30 ft/sec, the ALS will be unable to guide the aircraft to land safely.

TABLE I. RESULTS FROM USING CONVENTIONAL PID CONTROLLER ($k_1=2.8$; $k_2=2.8$; $k_3=11.5$; $k_4=6.0$)

Wind Speed (ft/sec)	Landing Point (ft)	Vertical Speed (ft/sec)	Pitch angle (degree)
0	785.2610	-2.8449	-1.3511
10	853.9561	-2.5786	-0.9523
20	910.0164	-2.3443	-0.4800
30	902.6000	-2.1704	-0.1115
40	1090.3424	-1.7348	0.3873

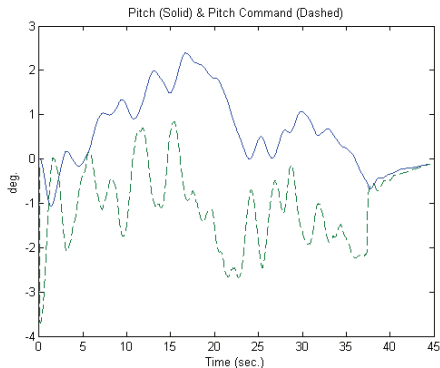


Figure 10. Aircraft pitch and pitch command.

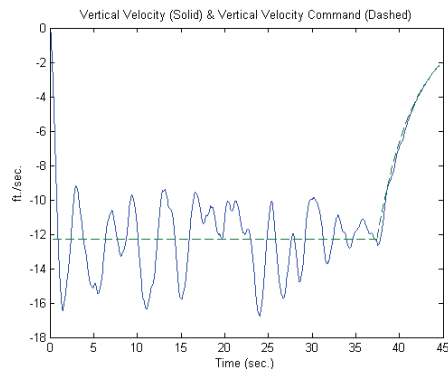


Figure 11. Aircraft vertical velocity and command.

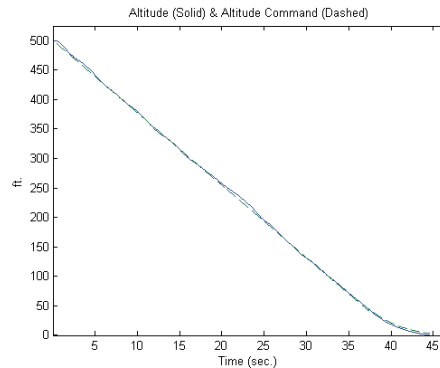


Figure 12. Aircraft altitude and command.

For the safe landing of an aircraft using the adaptive SOCM control scheme under different wind turbulence speeds are tested. The simulations show that the adaptive SOCM controller has well adaptive capability against severe wind disturbance. It is more robust than conventional PID type controller, as shown in Table II. The adaptive SOCM control scheme can successfully guide the aircraft flying through wind speeds of 0 ft/sec to 96 ft/sec. The situations at wind turbulence 96 ft/sec are that the pitch angle is 1.5011 degrees, vertical speed is -1.4125 ft/sec, horizontal velocity is 234.6779 ft/sec, and horizontal position at touchdown is 937.8017 ft, as shown in Fig.13 to Fig. 15. The layer variance is shown in Fig. 16.

TABLE II. RESULTS FROM USING ADAPTIVE SOCM CONTROL SCHEME ($k_1=2.8$; $k_2=2.8$; $k_3=11.5$; $k_4=6.0$)

Wind Speed (ft/sec)	Landing Point (ft)	Vertical Speed (ft/sec)	Pitch angle (degree)
20	937.8017	-1.9125	-0.3762
40	867.3983	-1.8542	-0.1497
60	949.5356	-1.4679	1.0374
80	738.3255	-1.4031	2.4873
90	926.0678	-1.5081	1.8440
96	937.8017	-1.4125	1.5011
97	1278.085	-0.7979	2.0038

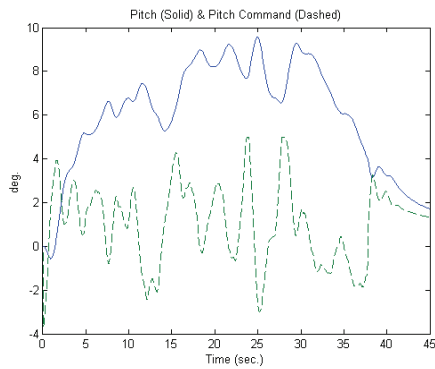


Figure 13. Aircraft pitch and pitch command.

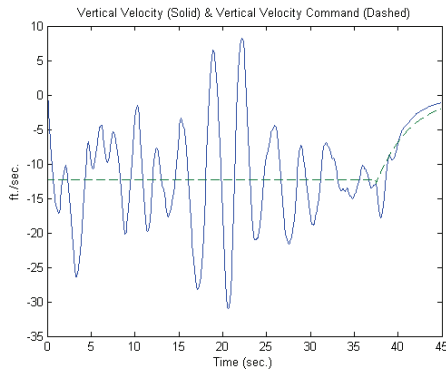


Figure 14. Aircraft vertical velocity and command.

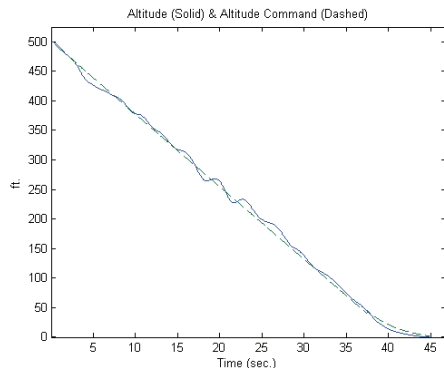


Figure 15. Aircraft altitude and command.

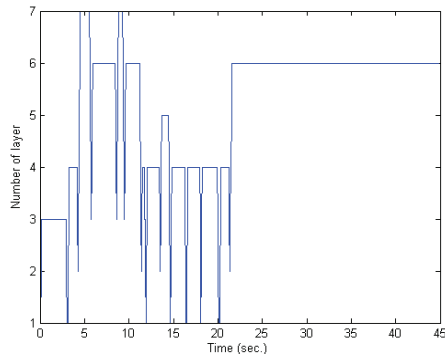


Figure 16. The layer variance of adaptive SOCM.

VI. CONCLUSION

The purpose of this paper is to investigate the use of adaptive SOCM in aircraft automatic landing system and to make the automatic landing system more intelligent. Current flight control law is adopted in the intelligent controller design. Tracking performance and adaptive capability are demonstrated through software simulations. For the safe landing of an aircraft using a conventional controller, CMAC or a conventional fuzzy CMAC, the wind speed of turbulence limits are 30, 58, and 90 ft/sec [14], respectively. In this study, a well-trained adaptive SOCM control scheme can reach 96 ft/sec. The proposed controllers have better performance than previous works, and it can act as an experienced pilot and guide the aircraft to a safe landing in severe wind turbulence environment.

REFERENCES

- [1] NASDAC Review of TSB Weather-Related accidents, http://www.asias.faa.gov/aviation_studies/weather_study/windcond.html
- [2] Federal Aviation Administration, "Automatic Landing Systems," AC 20-57A, Jan. 1971.
- [3] H. Buschek and A. J. Calise, "Uncertainty Modeling and Fixed-Order Controller Design for Hypersonic Vehicle Model," *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 1, pp. 42-48, 1997.
- [4] M.G. Cooper, "Genetic Design of Rule-Based Fuzzy Controllers," Ph.D. dissertation, University of California, Los Angeles, 1995.
- [5] Y. Iiguni, H. Akiyoshi, and N. Adachi, "An Intelligent Landing System Based on Human Skill Model," *IEEE Transactions on Aerospace and Electronic Systems*, 34(3), pp.877-882, 1998.
- [6] H. Izadi, M. Pakmehr, and N. Sadati, "Optimal Neural-Controller in Longitudinal Autolanding of a Commercial Jet Transport," *Proc. IEEE International Conference on Control Applications*, CD-000202, 1-6, Istanbul, Turkey, 2003.
- [7] C.C. Jorgensen, and C. Schley, "A Neural Network Baseline Problem for Control of Aircraft Flare and Touchdown," *Neural Networks for Control*, pp.403-425, 1991.
- [8] J.G. Juang, and J.Z. Chio, "Fuzzy Modeling Control for Aircraft Automatic Landing System," *International Journal of Systems Science*, 36(2), pp.77-87, 2005.
- [9] S.M.B. Malaek, N. Sadati, H. Izadi, and M. Pakmehr, "Intelligent autolanding controller design using neural networks and fuzzy logic," *Proc. IEEE 5th Control Conference*, 1, pp.365-373, 2004.
- [10] C.T. Chiang, T.S. Chiang, and C.K. Li, "A Simple and Converged Structure of Addressing Technique for CMAC_GBF," *Proc. IEEE International Conference on Systems, Man and Cybernetics*, pp. 6097-6101, 10-13 Oct. 2004.
- [11] C.M. Lin and T.Y. Chen, "Self-Organizing CMAC Control for a Class of MIMO Uncertain Nonlinear Systems," *IEEE Transaction on Neural Network*, pp. 1377-1384, Sep. 2009.
- [12] Flight Safety Foundation-Taiwan, http://www.flightsafety.org.tw/news1.php?Code=1&main_id=3&pages=5
- [13] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning Internal Representations by Error Propagation," in D.E. Rumelhart, J.L. McClelland, et al., *Parallel Distributed Processing*, vol. 1, Chap. 8, Cambridge, MA : MIT Press, 1986.
- [14] J.G. Juang and W.P. Lin, "Aircraft Landing Control Based on CMAC and GA Techniques," *Proc. of IFAC WC 2008*, P1173, 2008.