Advances in Differential Equations and Control Processes 2025, 32(3), 3604.
doi: https://doi.org/10.59400/adecp3604

Article

Application of intelligent control to aircraft landing system

Teng-Chieh Yang[1], Jih-Gau Juang[2,*]

[1] Primax Electronics Ltd., Taipei 114, Taiwan

[2] Department of Communications, Navigation and Control Engineering, National Taiwan Ocean University, Keelung 20224, Taiwan

*** Corresponding author:** Jih-Gau Juang, jgjuang@mail.ntou.edu.tw

CITATION

Yang T-C, Juang J-G. Application of intelligent control to aircraft landing system. Advances in Differential Equations and Control Processes. 2025; 32(3): 3604. https://doi.org/10.59400/adecp3604

ARTICLE INFO

Received: 14 August 2025
Revised: 10 September 2025
Accepted: 20 September 2025
Available online: 30 September 2025

Abstract: Conventional automatic landing systems (ALS) primarily utilize proportional-integral-derivative (PID) controllers in combination with gain-scheduling techniques. Their designs are generally constrained to the specific flight envelopes established by Federal Aviation Administration regulations. However, when environmental disturbances exceed these operational boundaries, the ALS may be deactivated. Consequently, developing a more intelligent ALS is crucial for maintaining safety across a broader spectrum of turbulent conditions. This paper proposes an intelligent control method that integrates the Cerebellar Model Articulation Controller (CMAC) with a fuzzy logic system for the development of an advanced aircraft automatic landing system. Multiple fuzzy modules are embedded within the CMAC structure: Type-1 fuzzy CMAC, adaptive Type-1 fuzzy CMAC, Type-2 fuzzy CMAC, and adaptive Type-2 fuzzy CMAC. Flight control principles are incorporated into its design. The Lyapunov stability theory is applied to ensure system stability, and adaptive learning rules are established to maintain this stability. Simulation results verify that the proposed controller can accurately track the desired landing trajectory and effectively adapt to various environmental conditions. Therefore, even under turbulent conditions, the adaptive fuzzy CMAC achieves reliable aircraft guidance and landing performance.

Keywords: intelligent control; CMAC; fuzzy systems; landing system, turbulence; system stability

1. Introduction

On May 22, 2024, a Singapore Airlines flight encountered severe turbulence, causing the aircraft to plunge 6000 feet within five minutes. The incident resulted in one fatality and multiple injuries. Just days later, on May 27, 2024, twelve passengers and crew members were injured when turbulence struck a Qatar Airways flight from Doha to Dublin while flying over Turkey. Another turbulence-related incident occurred on July 2, 2024, when an Air Europa flight from Madrid to Uruguay experienced intense turbulence, injuring thirty passengers and forcing an emergency landing in Brazil. According to Aircraft Accident Statistics, the causes of aviation accidents are divided into several categories, as shown in **Table 1** [1], with weather being one of the primary contributing factors. In conventional automatic landing systems (ALS), most controllers rely on proportional–integral–derivative (PID) control combined with gain scheduling methods [2]. These systems are typically designed to operate within specific flight envelopes defined by Federal Aviation Administration (FAA) regulations [3]. However, during conditions such as turbulence or wind shear, these controllers may

fail to maintain safe aircraft guidance. When environmental disturbances exceed the operational limits, the ALS is often disabled. Therefore, to ensure safety under a wider range of turbulent conditions, a more intelligent ALS is essential.

Table 1. Causes of fatal accidents by decade (%), 1950s through 2000s.

Cause	50s	60s	70s	80s	90s	00s
Pilot Error	40	32	24	25	27	25
Pilot Error (weather-related)	11	18	14	17	21	17
Pilot Error (mechanical-related)	7	5	4	2	4	3
Total Pilot Error	58	57	42	44	53	45
Other Human Error	0	8	9	6	8	9
Weather	16	10	13	15	9	8
Mechanical Failure	21	20	23	21	21	28
Sabotage	5	5	11	13	10	9
Other Cause	0	2	2	1	0	1

In recent years, modern control strategies and intelligent methodologies have been widely implemented across numerous engineering applications [4–7], including various flight control systems [8–17]. The cerebellar model articulation controller (CMAC), originally proposed in Albus [18], is a neural network model inspired by the structure and functioning of the human cerebellum. Acting as an associative memory neural network, CMAC emulates the learning and coordination mechanisms of the cerebellum. Unlike conventional backpropagation neural networks, CMAC employs a constant local weight update rule rather than a global one, resulting in faster convergence and reduced computational complexity. Owing to these advantages, CMAC has been successfully applied to a wide range of nonlinear systems, automatic control applications, as well as pattern recognition, signal processing, and image processing tasks [19–22].

Numerous studies have focused on enhancing the performance of the CMAC. One notable development is the introduction of the fuzzy CMAC, which integrates the learning ability of neural networks with the reasoning advantages of fuzzy systems [23]. Fuzzy logic, first proposed by Zadeh [24], is founded on the concept of fuzzy sets and seeks to emulate human reasoning through fuzzy rules embedded in an inference system. Type-2 fuzzy logic [25], an extension of this concept based on Zadeh's extension principle, introduces an additional layer of uncertainty by allowing variability in both membership functions and inference processes. Systems employing type-2 fuzzy sets and logic are known as type-2 fuzzy systems, while those using conventional fuzzy sets and inference mechanisms are referred to as type-1 fuzzy systems. In a type-1 fuzzy system, each element is assigned a precise membership value between 0 and 1, whereas type-2 fuzzy systems are better equipped to manage linguistic and numerical uncertainties that type-1 systems cannot effectively address.

Another major advancement in CMAC research is the development of adaptive learning rules. An adaptive CMAC approach was proposed for controlling linear piezoelectric ceramic motors [26]. Four years later, the same authors presented a robust adaptive CMAC system for brushless DC (BLDC) motors [27], featuring an improved adaptive mechanism that further enhanced control performance. In this study,

type-1 and type-2 fuzzy CMACs with adaptive learning rules are employed to design an intelligent automatic landing control system (ALS) for aircraft. The goal is to enhance the performance of conventional ALS and ensure safe aircraft landings under various conditions. Although intelligent control strategies for aircraft landing have been explored in previous studies [10–17], which utilized neural networks, model predictive control, and linear quadratic regulator. Most of the controllers lack adaptability in the presence of severe disturbances. For example, traditional PID controllers can only handle turbulence up to 30 ft/sec, while backpropagation networks, multilayer functional-link networks, counter-propagation networks, improved backpropagation networks, and radial basis function networks can manage turbulence between 30 ft/sec and 65 ft/sec. To overcome these limitations, this study introduces adaptive learning rules into the control framework, enabling the system to handle a wider range of wind disturbances better. The proposed controller utilizes a gradient descent algorithm with variable learning rates, allowing it to adapt effectively to higher levels of turbulence compared to previous methods. Furthermore, the stability of the control system is rigorously ensured and proven through Lyapunov stability theory.

2. System description

During the landing phase, the pilot guides the aircraft from cruising altitude down to approximately 1200 feet above ground level, aligning its heading with the runway centerline. When the aircraft is about four nautical miles from the runway, it reaches the outer marker and intercepts the glide path signal, as shown in **Figure 1** [28]. While descending along the glide path, the aircraft must maintain stable pitch, attitude, and airspeed. The typical descent rate is around 10 ft/sec, with pitch angles ranging from −5° to +5°. As the aircraft approaches an altitude of 20–70 feet above ground level, the glide path control system disengages, initiating the flare maneuver. During this phase, the descent rate is reduced to about 2 ft/sec, allowing the landing gear to absorb the impact energy. Meanwhile, the pitch angle is adjusted between 0° and 5° to achieve a smooth and safe touchdown on the runway surface [29].

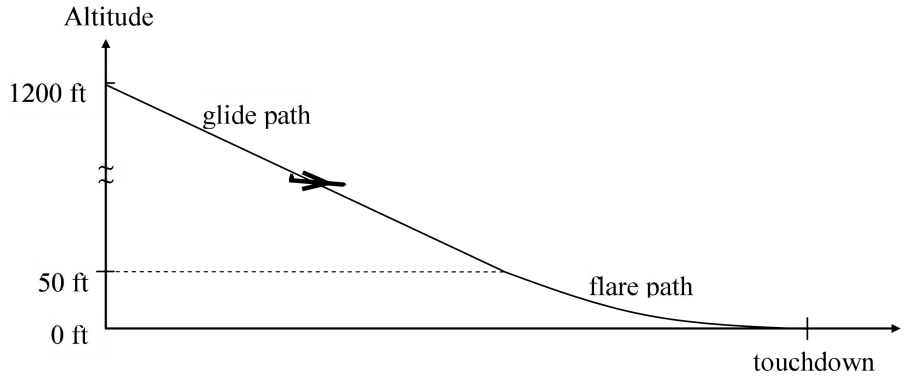

Figure 1. Glide path and flare path.

A commercial aircraft model is employed in the simulations, with motion constrained to the longitudinal and vertical planes [12]. The equations governing the

aircraft's motion are expressed as follows:

$$\begin{aligned}\dot{u} = X_u\,(u-u_g) + X_w(w-w_g) + X_q q \\ -g(\tfrac{\pi}{180})\cos(\gamma_0)\theta + X_E\delta_E + X_T\delta_T\end{aligned} \tag{1}$$

$$\dot{q} = M_u(u-u_g) + M_w(w-w_g) + M_q q + M_E\delta_E + M_T\delta_T \tag{2}$$

$$\begin{aligned}\dot{w} = Z_u\,(u-u_g) + Z_w(w-w_g) + (Z_q - \tfrac{\pi}{180}U_0)q \\ -g(\tfrac{\pi}{180})\sin(\gamma_0)\theta + Z_E\delta_E + Z_T\delta_T\end{aligned} \tag{3}$$

$$\dot{\theta} = q \tag{4}$$

$$\dot{h} = -w + \frac{\pi}{180}U_0\theta \tag{5}$$

Here, u denotes the aircraft's longitudinal velocity (ft/sec), w represents the vertical velocity (ft/sec), and q is the pitch rate (rad/sec). The pitch angle is denoted by θ (degrees), h indicates the altitude (f), and δ_E represents the incremental elevator angle (degrees). The variable δ_T refers to the throttle setting (f/t/sec), γ_o denotes the flight path angle (typically $-3°$), U_0 is the nominal aircraft speed (ft/sec), and g represents gravitational acceleration (32.2 ft/sec^2). The parameters X_i, Z_i and M_i are the aircraft's stability and control derivatives.

To improve the intelligence of the ALS, precise wind profile modeling is essential. Among the commonly used spectral turbulence models—the von Kármán and Dryden models—the Dryden model [12] is adopted in this study for its simplicity and ease of implementation. A turbulence profile with a speed of 30 ft/sec is shown in **Figure 2**.

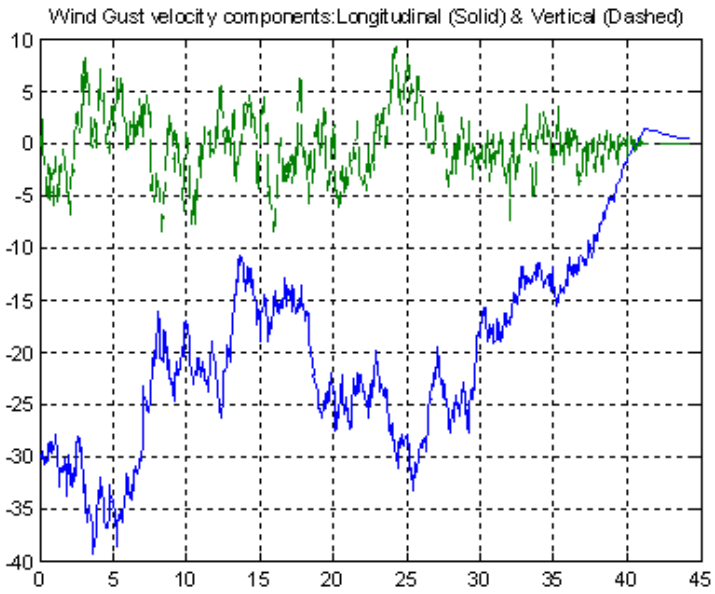

Figure 2. Turbulence profile.

3. Control scheme

The PID controller serves as a basic structure for the aircraft landing system, as illustrated in **Figure 3**. Its inputs consist of the commanded altitude and altitude rate, as well as the actual altitude and altitude rate of the aircraft. Using these inputs, the landing controller produces a pitch command to regulate the pitch autopilot. Further

details are provided in Jorgensen and Schley [12], which has all the limitations of inputs. To achieve a smooth touchdown, a constant pitch angle is added to the controller while the aircraft reaches the flare phase. Although the PID controller is simple and generally effective, it has inherent limitations, including overshoot and high sensitivity to noise and external disturbances. In conditions of severe turbulence, it may fail to guide the aircraft safely to the runway. To address these issues, a CMAC compensator is integrated into the proposed controller [17]. In this hybrid configuration, the PID controller ensures system stability, while the CMAC component learns to refine and enhance control precision. Since the PID gains are usually tuned empirically—yielding satisfactory but suboptimal results—the inclusion of the CMAC effectively compensates for these shortcomings and improves overall performance.

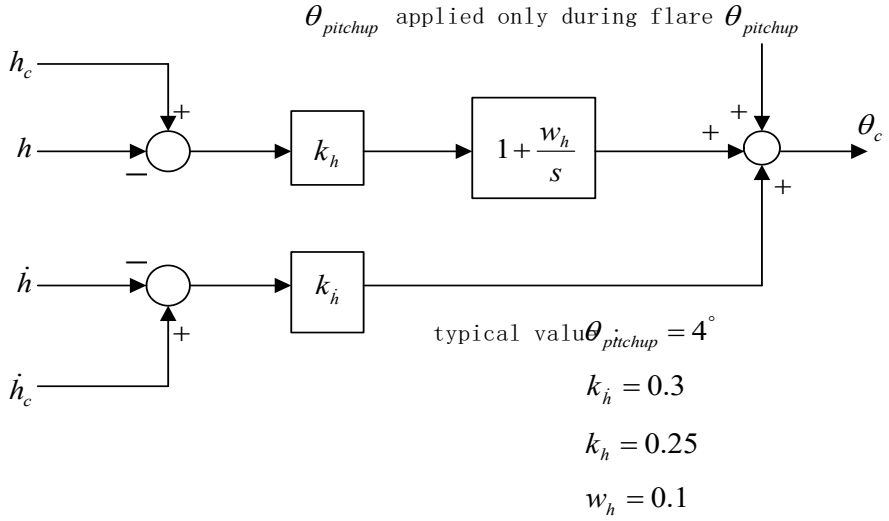

Figure 3. PID controller.

Figure 4 shows the overall control architecture. The pitch autopilot control signal U is generated by summing the outputs of the PID controller and the fuzzy CMAC. Both controllers share identical input signals: aircraft altitude, altitude command, altitude rate, and altitude rate command. The PID controller provides a reliable baseline response, while the fuzzy CMAC operates through recall and learning processes at each sampling instant k. During the recall phase, the fuzzy CMAC utilizes the desired and actual system outputs of the next time step as addressing inputs to produce its control output $U_{fuzzy\ CMAC}$. In the learning phase, the pitch autopilot input serves as the desired output, allowing the fuzzy CMAC to update its stored weights by comparing the actual system output with that of the subsequent time step. The resulting fuzzy CMAC output functions as a compensatory term for the pitch command. This integrated control strategy significantly enhances the system's turbulence rejection capability, effectively addressing the performance limitations of conventional automatic landing systems under severe turbulence conditions.

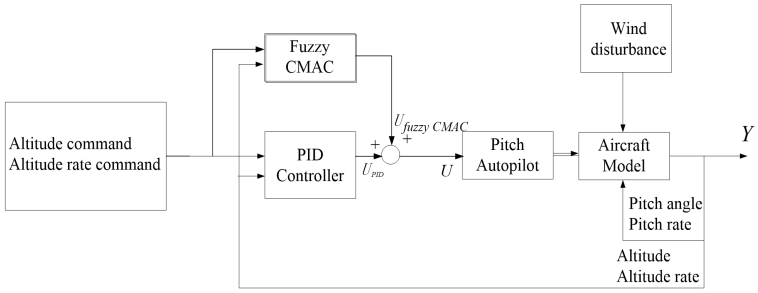

Figure 4. The fuzzy CMAC control scheme.

3.1. Type-1 fuzzy CMAC

Figure 5 depicts the structure of the fuzzy CMAC, which functions similarly to an associative memory network. Compared with conventional neural networks, it achieves a faster self-learning rate, as it can adjust memory weights with minimal modifications. Moreover, it exhibits strong local generalization capability. The fuzzy CMAC operates much like a look-up table, generating its output through a linear combination of memory-stored weights. Knowledge is stored associatively within overlapping storage hypercubes, known as the remembering regions, which facilitate effective data retrieval. The fuzzy CMAC executes two primary operations: one for output computation and another for learning and weight adjustment. The output is derived through the mapping process X→S→C→W→Y [30].

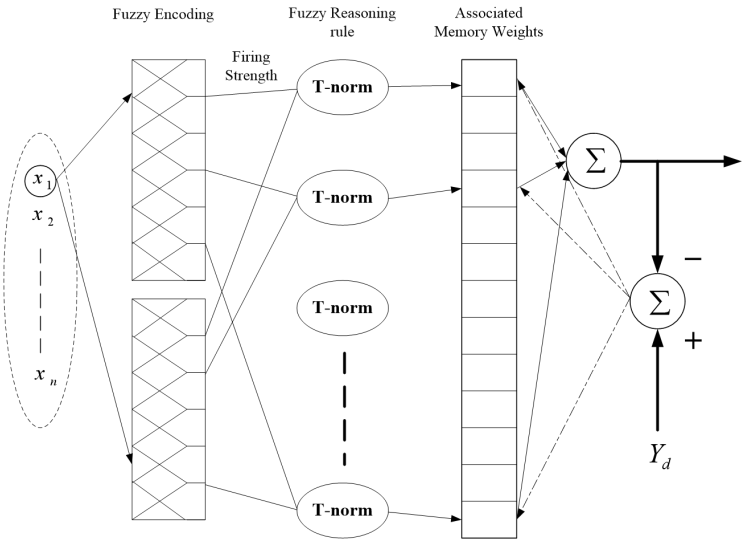

Figure 5. Structure of fuzzy CMAC.

Phase I. Quantization ($X \rightarrow S$):

The input vector $X = [x_1, x_2, \ldots, x_n]^T$ is n-dimensional. Its quantization vector, denoted as $S = [s_1, s_2, \ldots, s_n]^T$, defines the discrete state of each input variable before the fuzzification process.

Phase II. Associative Mapping segment ($S \rightarrow C$):

The next step involves fuzzifying the quantization vector obtained from X into C. In the fuzzy CMAC, this fuzzification process serves as the addressing mechanism. After fuzzification, the input state values are transformed into corresponding *firing strengths* according to their membership functions.

Phase III. Memory Weight Mapping ($C \rightarrow W$):

After the block regions are fuzzified, the firing strength of the i^{th} rule in the fuzzy CMAC can be calculated as:

$$C_j(x) = c_{j1}(x_1)^* c_{j2}(x_2)^* \ldots c_{jn}(x_n) = \prod_{i=1}^{n} c_{j1}(x_i) \tag{6}$$

In this expression, $c_{j_i}(x_i)$ denotes the j^{th} membership function of the i^{th} input variable, and n represents the total number of input states. The asterisk (*) signifies a fuzzy t-norm operator, which can be defined using operations such as maximum, minimum, or product. In this study, the product operator is adopted as the t-norm because of its simplicity and ease of implementation.

Phase IV. Memory weight learning and output generation ($W \rightarrow Y$):

Because of the overlapping membership functions and the structure of fuzzy rule definitions, multiple fuzzy rules may be activated at the same time. During the defuzzification stage, the outputs of these activated rules are aggregated by summing their respective weights (w_j), each scaled by its corresponding firing strength ($C_j(x)$). The overall output of the network is then determined as follows:

$$y = \sum_{j=1}^{N} (w_j C_j(x) / \sum_{i=1}^{N} C_i(x)) \tag{7}$$

In the fuzzy CMAC, learning is achieved by adjusting the memory weights according to the error between the desired output and the actual network output. The weight update rule for the fuzzy CMAC is expressed as:

$$w_j^{(i)} = w_j^{(i-1)} + \frac{\alpha}{m}(y_d - y) C_j(x) / \sum_{i=1}^{N} C_i(x) \tag{8}$$

where α is the learning rate, m is the size of the floor (called generalization), y_d is the desired output.

In conventional fuzzy CMAC, the weight update rule employs a fixed learning rate, which can constrain learning efficiency by rendering the adaptation process either excessively slow or potentially unstable. To mitigate this limitation, we introduce an adaptive learning rate that is dynamically adjusted throughout the weight update procedure, thereby improving overall learning performance. This adaptive learning rate is systematically derived using a Lyapunov function. Let $e(t)$ denote the tracking error, defined as:

$$e(t) = y_d(t) - y(t) \tag{9}$$

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

$$V(t) = \frac{1}{2} e^2(t) \tag{10}$$

Thus, the change in the Lyapunov function is obtained by

$$\Delta V = V(t+1) - V(t) = \frac{1}{2}\left[e^2(t+1) - e^2(t)\right] \tag{11}$$

The error difference can be represented by Juang et al. [31] as follows

$$\Delta e(t) \approx \Big[\frac{\partial e(t)}{\partial W}\Big]\Delta W(t) \tag{12}$$

Using the gradient descent method, the weight change can be obtained as

$$\Delta w_j(t) = -\frac{\alpha}{m}\frac{\partial V(t)}{\partial w_j(t)} \tag{13}$$

Since the derivative of the Lyapunov function with respect to weight is

$$\frac{\partial V(t)}{\partial w_j(t)} = e(t)\frac{\partial e(t)}{\partial w_j(t)} = (y_d - y(t))(-C(x)^T/\Sigma C(x)) \tag{14}$$

Thus, the weight change can be obtained as

$$\Delta w_j(t) = \frac{\alpha}{m}(C_j(x)/\Sigma C(x)(y_d - y(t)), j = 1, 2, ..., N \tag{15}$$

$$\Delta W(t) = \begin{bmatrix} \Delta w_i(t) \\ \Delta w_2(t) \\ \vdots \\ \Delta w_N(t) \end{bmatrix} = \frac{\alpha}{m}\begin{bmatrix} (C_1(x)/\Sigma C(x) \\ (C_2(x)/\Sigma C(x) \\ \vdots \\ (C_N(x)/\Sigma C(x) \end{bmatrix}(y_d - y(t)) = \frac{\alpha}{m}(C(x)/\Sigma C(x)(y_d - y(t)) \tag{16}$$

From (7) to (9), we have

$$\frac{\partial e(t)}{\partial w_j(t)} = -C_j(x)/\Sigma C(x),\ \frac{\partial e(t)}{\partial W} = -C(x)^T/\Sigma C(x) \tag{17}$$

From (10) to (17), we can rewrite the change of the Lyapunov function as

$$\begin{aligned} &\Delta V = \tfrac{1}{2}\Big[e^2(t+1) - e^2(t)\Big] = \tfrac{1}{2}\big[\,e(t+1) - e(t)\big]\big[\,e(t+1) + e(t)\big] \\ &= \tfrac{1}{2}\big[\Delta e(t)\big]\big[2e(t) + \Delta e(t)\big] = \Delta e(t)\Big[\,e(t) + \tfrac{1}{2}\,\Delta e(t)\Big] \\ &= \Big[\tfrac{\partial e(t)}{\partial W}\tfrac{\alpha}{m}(C(x)/\Sigma C(x))e(t)\Big] \\ &\Big\{e(t) + \tfrac{1}{2}\Big[\tfrac{\partial e(t)}{\partial W}\tfrac{\alpha}{m}(C(x)/\Sigma C(x))e(t)\Big]\Big\} \end{aligned} \tag{18}$$

Since $\frac{\partial e(t)}{\partial W} = -C(x)^T/\Sigma C(x)$, the final change of the Lyapunov function can be obtained as

$$\begin{aligned} &\Delta V = \Big[\,-C(x)^T/\Sigma C(x)\tfrac{\alpha}{m}C(x)/\Sigma C(x)e(t)\Big] \\ &\Big\{e(t) + \tfrac{1}{2}\Big[\,-C(x)^T/\Sigma C(x)\tfrac{\alpha}{m}C(x)/\Sigma C(x)e(t)\Big]\Big\} \\ &= \Big[\tfrac{1}{2}\tfrac{\alpha}{m}e(t)\tfrac{C(x)^TC(x)}{(\Sigma C(x))^2}\Big]\Big[2e(t) - \tfrac{\alpha}{m}e(t)\tfrac{\|C(x)^2\|}{(\Sigma C(x))^2}\Big] \\ &= \Big[\,-\tfrac{1}{2}\tfrac{\alpha}{m}e^2(t)\tfrac{\|C(x)^2\|}{(\Sigma C(x))^2}\Big]\Big[2-\tfrac{\alpha}{m}\tfrac{\|C(x)^2\|}{(\Sigma C(x))^2}\Big] \end{aligned} \tag{19}$$

Let $\left[2-\frac{\alpha}{m}\frac{\|C(x)^2\|}{(\Sigma C(x))^2}\right]>0$ then $\Delta V<0$, i.e., select the learning rate as

$$\frac{2m(\Sigma C(x))^2}{\|C(x)^2\|}>\alpha>0 \tag{20}$$

Since ΔV is negative definite, it follows that $\Delta V<0$ for all $t>0$, indicating that $e(t)\to 0$ for $t\to\infty$. This ensures the convergence of the adaptive type-1 fuzzy CMAC learning process. As a result, the aircraft landing control system is locally asymptotically stable.

3.2. Type-2 fuzzy CMAC

The CMAC structure is extended using a type-2 fuzzy framework to improve resolution and accuracy over the conventional fuzzy CMAC. Functionally, the type-2 fuzzy CMAC works similarly to its type-1 counterpart. Its structure is presented in **Figure 6**, with the mapping procedure for each phase outlined below. Let $\mathcal{X}$ denote an n-dimensional input space, as illustrated in **Figure 7**. The type-2 fuzzy CMAC employs an addressing scheme derived from interval type-2 fuzzification. Upon fuzzifying the input vector into an interval type-2 fuzzy set, the input state values are mapped to upper and lower firing strengths determined by their corresponding membership functions. The t-norm operation uses product inference. Accordingly, the upper and lower firing strengths for the j^{th} rule are obtained as follows:

$$\overline{c}^j(x)=\overline{c}_{j_1}(x_1)^*\overline{c}_{j_2}(x_2)^*.....\overline{c}_{j_n}(x_n)=\prod_{i=1}^{n}\overline{c}_{j_i}(x_i) \tag{21}$$

$$\underline{c}^j(x)=\underline{c}_{j_1}(x_1)^*\underline{c}_{j_2}(x_2)^*.....\underline{c}_{j_n}(x_n)=\prod_{i=1}^{n}\underline{c}_{j_i}(x_i) \tag{22}$$

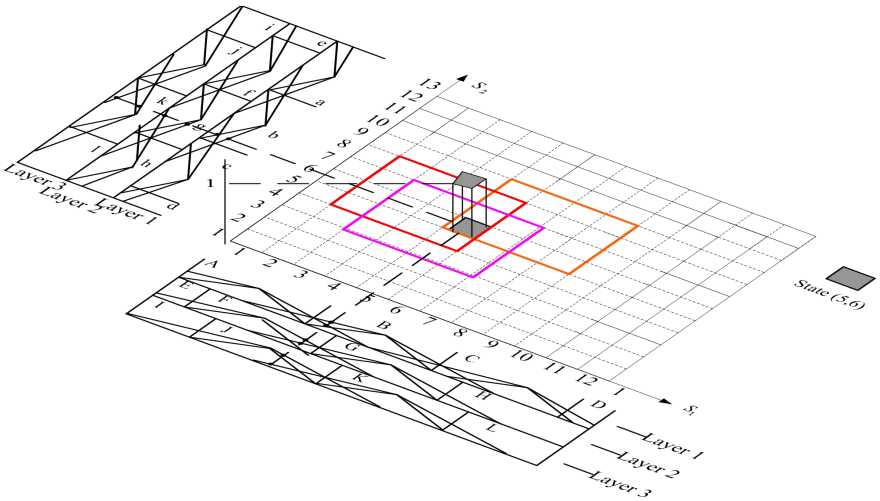

Figure 6. Diagram of a type-2 fuzzy CMAC in 3-D.

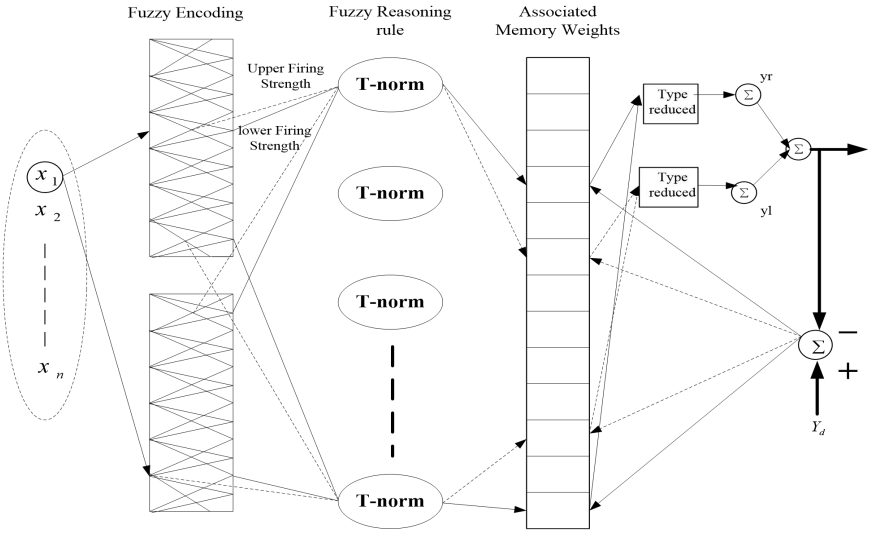

Figure 7. Architecture of a type-2 fuzzy CMAC network.

The center of set type reduction is applied to the type-2 fuzzy CMAC:

$$y_{\cos} = [y_l, y_r] = \int_{W^1 \in [\underline{w}^1, \overline{w}^1]} \cdots \int_{W^N \in [\underline{w}^M, \overline{w}^N]} \cdots \int_{c^1 \in [\underline{c}^1, \bar{c}^1]} \cdots \int_{c^M \in [\underline{c}^M, \bar{c}^N]} 1 \Big/ \frac{\sum_{j=1}^{n} c^j w^j}{\sum_{j=1}^{n} c^j} \tag{23}$$

Its left and right end points y_l and y_r are obtained as the interval type-1 set, which can be written as follows [29,32–34]:

$$y_r = \frac{\sum_{j=1}^{N} \overline{c}^j \overline{w}^j}{\sum_{j=1}^{N} \overline{c}^j} = \frac{\sum_{j=1}^{R} \underline{c}^j \overline{w}^j + \sum_{j=R+1}^{N} \overline{c}^j \overline{w}^j}{\sum_{j=1}^{R} \underline{c}^j + \sum_{j=R+1}^{N} \overline{c}^j} \tag{24}$$

$$y_l = \frac{\sum_{j=1}^{N} \underline{c}^j \underline{w}^j}{\sum_{j=1}^{N} \underline{c}^j} = \frac{\sum_{j=1}^{L} \overline{c}^j \underline{w}^j + \sum_{j=L+1}^{N} \underline{c}^j \underline{w}^j}{\sum_{j=1}^{N} \overline{c}^j + \sum_{j=L+1}^{N} \underline{c}^j} \tag{25}$$

The corresponding weights of $\overline{c}$ and $\underline{c}$ are $\overline{w}$ and $\underline{w}$, respectively. We can obtain L and R from the following steps:

Phase I. Let the pre-computed value of $\overline{w}^J$ be in ascending order, i.e., $\overline{w}^1 \leq \overline{w}^2 \leq \ldots \leq \overline{w}^N$

Phase II. Compute y_r by initially setting $\overline{c}^j = (\overline{c}^j + \underline{c}^j)/2$ for $j = 1, 2, \ldots N$ and let $y'_r = y_r$

Phase III. Obtain $R(1 \leq R \leq N-1)$ and make $\overline{w}^R \leq y'_r \leq \overline{w}^{R+1}$

Phase IV. Find y_r as $\overline{c}^j = \underline{c}^j$ with $j \leq R$ and $\overline{c}^j = \overline{c}^j$ with $j > R$ then make $y''_r = y_r$

Phase V. If $y''_r \neq y_r$ then move to Phase VI, otherwise stop the process and let $y''_r \equiv y'_r$

Phase VI. Let $y'_r = y''_r$, move to Phase III.

The computation procedure of y_l is similar to y_r. In Phase III, obtain $L(1 \leq L \leq N-1)$ and make $\underline{w}^L \leq y_l^L \leq \underline{w}^{L+1}$. In Phase II, obtain the initial setting $\underline{c}^j = (\overline{c}^j + \underline{c}^j)/2$ for $j = 1, 2, ..., N$, and in Phase VI, obtain $\underline{y}_l$ with the specified conditions for $\underline{c}^j = \overline{c}^j$ and $j \leq L$, and $\underline{c}^j = \underline{c}^j$ and $j > L$. The defuzzified output value is calculated as the average of y_r and y_l.

$$y = y_r + y_l \tag{26}$$

The type-2 fuzzy CMAC updates its memory weights through a learning process aimed at minimizing the error between the desired and actual outputs. The associated learning rules are defined as follows:

$$\overline{W}_j^{(i)} = \overline{W}_j^{(i-1)} + \frac{\alpha_1}{m}(y_d - y)\overline{c}_j(x) / \sum_{j=1}^{N} \overline{c}_j(x) \tag{27}$$

$$\underline{w}_j^{(i)} = \underline{w}_j^{(i-1)} + \frac{\alpha_2}{m}(y_d - y)\underline{c}_j(x) / \sum_{j=1}^{N} \underline{c}_j(x) \tag{28}$$

where α_1 and α_2 denote the learning rates, m represents the floor size (also referred to as the generalization parameter), and y_d is the target output. In conventional type-2 fuzzy CMAC, weight updates rely on a fixed learning rate, which may render the learning process either too slow or excessively fast. To overcome this limitation, an adaptive learning rule is introduced where the learning rate is defined via a discrete-time Lyapunov function. The tracking error $e(t)$ is given in equation (9), and the corresponding Lyapunov function is formulated in equation (10). Its variation and the error difference are presented in equations (11) and (12), respectively. Using gradient descent, the resulting weight updates can be expressed as follows:

$$\Delta\overline{w}_j(t) = \frac{\alpha_1}{m}(y_d - y)\overline{c}_j(x) / \sum_{j=1}^{N} \overline{c}_j(x) \tag{29}$$

$$\Delta\underline{w}_j(t) = \frac{\alpha_2}{m}(y_d - y)\underline{c}_j(x) / \sum_{j=1}^{N} \underline{c}_j(x) \tag{30}$$

Since the derivative of the Lyapunov function with respect to $\overline{w}$ and $\underline{w}$ are

$$\begin{aligned} \frac{\partial V(t)}{\partial \overline{w}_j(t)} &= \frac{\partial V(t)}{\partial e(t)} \frac{\partial e(t)}{\partial \overline{w}_j(t)} = e(t) \frac{\partial e(t)}{\partial \overline{w}_j(t)} \\ &= -(y_d - y)\overline{c}_j(x) / \sum_{j=1}^{N} \overline{c}_j(x) \end{aligned} \tag{31}$$

$$\begin{aligned} \frac{\partial V(t)}{\partial \underline{w}_j(t)} &= \frac{\partial V(t)}{\partial e(t)} \frac{\partial e(t)}{\partial \underline{w}_j(t)} = e(t) \frac{\partial e(t)}{\partial \underline{w}_j(t)} = \underline{w}_j^{(i)} \\ &= -(y_d - y)\underline{c}_j(x) / \sum_{j=1}^{N} \underline{c}_j(x) \end{aligned} \tag{32}$$

Thus, we can rewrite the weight change of $\overline{w}$ and $\underline{w}$ as

$$\Delta\overline{w}_j(t) = \frac{\alpha_1}{m}(y_d - y)\overline{c}_j(x)/\sum_{j=1}^{N}\overline{c}_j(x), j = 1, 2, ..., N \tag{33}$$

$$\Delta\underline{w}_j(t) = \frac{\alpha_2}{m}(y_d - y)\underline{c}_j(x)/\sum_{j=1}^{N}\underline{c}_j(x), j = 1, 2, ..., N \tag{34}$$

The matrices of weight change can be obtained as

$$\begin{aligned}\Delta\overline{W}(t) &= \begin{bmatrix} \Delta\overline{w}_1(t) \\ \Delta\overline{w}_2(t) \\ \vdots \\ \Delta\overline{w}_N(t) \end{bmatrix} = \frac{\alpha_1}{m}\begin{bmatrix} \overline{c}_1(x)/\sum_{j=1}^{N}\overline{c}_j(x) \\ \overline{c}_2(x)/\sum_{j=1}^{N}\overline{c}_j(x) \\ \vdots \\ \overline{c}_N(x)/\sum_{j=1}^{N}\overline{c}_j(x) \end{bmatrix}(y_d - y(t)) \\ &= \frac{\alpha_1}{m}(\overline{C}(x)/\sum_{j=1}^{N}\overline{c}_j(x))(y_d - y(t))\end{aligned} \tag{35}$$

$$\begin{aligned}\Delta\underline{W}(t) &= \begin{bmatrix} \Delta\underline{w}_1(t) \\ \Delta\underline{w}_2(t) \\ \vdots \\ \Delta\underline{w}_N(t) \end{bmatrix} = \frac{\alpha_2}{m}\begin{bmatrix} \underline{c}_1(x)/\sum_{j=1}^{N}\underline{c}_j(x) \\ \underline{c}_2(x)/\sum_{j=1}^{N}\underline{c}_j(x) \\ \vdots \\ \underline{c}_N(x)/\sum_{j=1}^{N}\underline{c}_j(x) \end{bmatrix}(y_d - y(t)) \\ &= \frac{\alpha_2}{m}(\underline{C}(x)/\sum_{j=1}^{N}\underline{c}_j(x))(y_d - y(t))\end{aligned} \tag{36}$$

From (21), (22), (27), and (28) we have

$$\begin{aligned}\frac{\partial e(t)}{\partial\overline{w}_j(t)} &= -\overline{c}_j(x)/\sum_{j=1}^{N}\overline{c}_j(x) \\ \frac{\partial e(t)}{\partial\overline{W}} &= -\overline{C}(x)^T/\sum_{j=1}^{N}\overline{c}_j(x)\end{aligned} \tag{37}$$

$$\begin{aligned}\frac{\partial e(t)}{\partial\underline{w}_j(t)} &= -\underline{c}_j(x)/\sum_{j=1}^{N}\underline{c}_j(x) \\ \frac{\partial e(t)}{\partial\underline{W}} &= -\underline{C}(x)^T/\sum_{j=1}^{N}\underline{c}_j(x)\end{aligned} \tag{38}$$

From (9) to (12) and (29) to (38), with respect to $\overline{W}$, we have

$$\begin{aligned}
&\Delta V = \tfrac{1}{2}\Big[e^2(t+1) - e^2(t)\Big] = \tfrac{1}{2}[e(t+1) - e(t)]\big[e(t+1) + e(t)\big] \\
&= \tfrac{1}{2}[\Delta e(t)]\big[2e(t) + \Delta e(t)\big] = \Delta e(t)\Big[e(t) + \tfrac{1}{2}\Delta e(t)\Big] \\
&= \left[\frac{\partial e(t)}{\partial \overline{W}}\frac{\alpha_1}{m}(\overline{C}(x)/\sum_{j=1}^{N}\overline{c}_j(x))e(t)\right] \\
&\left\{e(t) + \tfrac{1}{2}\left[\frac{\partial e(t)}{\partial \overline{W}}\frac{\alpha_1}{m}(\overline{C}(x)/\sum_{j=1}^{N}\overline{c}_j(x))e(t)\right]\right\}
\end{aligned} \tag{39}$$

Since $\frac{\partial e(t)}{\partial \overline{W}} = -\overline{C}(x)^T/\sum_{j=1}^{N}\overline{c}_j(x)$, we can rewrite (39) as

$$\begin{aligned}
&\Delta V = \left[-(\overline{C}(x)^T/\sum_{j=1}^{N}\overline{c}_j(x))\frac{\alpha_1}{m}(\overline{C}(x)/\sum_{j=1}^{N}\overline{c}_j(x))e(t)\right] \\
&\left\{e(t) + \tfrac{1}{2}\left[-(\overline{C}(x)^T/\sum_{j=1}^{N}\overline{c}_j(x))\frac{\alpha_1}{m}(\overline{C}(x)/\sum_{j=1}^{N}\overline{c}_j(x))e(t)\right]\right\} \\
&= \left[-\tfrac{1}{2}\frac{\alpha_1}{m}e(t)\frac{\overline{C}(x)^T\overline{C}(x)}{(\sum_{j=1}^{N}\overline{c}_j(x))^2}\right]\left[2e(t) - \frac{\alpha_1}{m}e(t)\frac{\|\overline{C}(x)^2\|}{(\sum_{j=1}^{N}\overline{c}_j(x))^2}\right] \\
&= \left[\frac{-1}{2}\frac{\alpha_1}{m}e^2(t)\frac{\|\overline{C}(x)^2\|}{(\sum_{j=1}^{N}\overline{c}_j(x))^2}\right]\left[2 - \frac{\alpha_1}{m}\frac{\|\overline{C}(x)^2\|}{(\sum_{j=1}^{N}\overline{c}_j(x))^2}\right]
\end{aligned} \tag{40}$$

Let $\left[2 - \frac{\alpha_1}{m}\frac{\|\overline{C}(x)^2\|}{\left(\sum_{j=1}^{N}\overline{c}_j(x)\right)^2}\right] > 0$ then $\Delta V < 0$, i.e., we can select the learning rate α_1 in the following range

$$\frac{2m(\sum_{j=1}^{N}\overline{c}_j(x))^2}{\|\overline{C}(x)^2\|} > \alpha_1 > 0 \tag{41}$$

From (9) to (12) and (29) to (38), with respect to $\underline{W}$, we have

$$\begin{aligned}
&\Delta V = \tfrac{1}{2}\Big[e^{2(t+1)-e^2(t)}\Big] = \tfrac{1}{2}\big[e\,(t+1) - e\,(t)\,\big]\big[e\,(t+1) + e\,(t)\,\big] \\
&= \tfrac{1}{2}\big[\Delta e\,(t)\,\big]\big[2e\,(t) + \Delta e\,(t)\,\big] = \Delta e\,(t)\,\Big[e\,(t) + \tfrac{1}{2}\Delta e\,(t)\,\Big] \\
&= \left[\frac{\partial e(t)}{\partial \underline{W}}\frac{\alpha_2}{m}(\underline{C}\,(x)\,/\sum_{j=1}^{N}\underline{c}_j\,(x))e\,(t)\right] \\
&\left\{e\,(t) + \tfrac{1}{2}\left[\frac{\partial e(t)}{\partial \underline{W}}(\frac{\alpha_2}{m})(\underline{C}\,(x)\,/\sum_{j=1}^{N}\underline{c}_j\,(x))e\,(t)\,\right]\right\}
\end{aligned} \tag{42}$$

Since $\frac{\partial e(t)}{\partial \underline{W}} = -\underline{C}(x)^T / \sum_{j=1}^{N} \underline{c}_j(x)$, we can rewrite (42) as

$$\begin{aligned}
\Delta V &= \Big[- (\underline{C}(x)^T / \sum_{j=1}^{N} \underline{c}_j(x)) \tfrac{\alpha_2}{m} (\underline{C}(x) / \sum_{j=1}^{N} \underline{c}_j(x)) e(t) \Big] \\
&\left\{ e(t) + \tfrac{1}{2} \Big[- (\underline{C}(x)^T / \sum_{j=1}^{N} \underline{c}_j(x)) \tfrac{\alpha_2}{m} (\underline{C}(x) / \sum_{j=1}^{N} \underline{c}_j(x)) e(t) \Big] \right\} \\
&= \left[- \tfrac{1}{2} \tfrac{\alpha_2}{m} e(t) \frac{\underline{C}(x)^T \underline{C}(x)}{(\sum_{j=1}^{N} \underline{c}_j(x))^2} \right] \left[2e(t) - \tfrac{\alpha_2}{m} e(t) \frac{\left\| \underline{C}(x)^2 \right\|}{(\sum_{j=1}^{N} \underline{c}_j(x))^2} \right] \\
&= \left[- \tfrac{1}{2} \tfrac{\alpha_2}{m} e^2(t) \frac{\left\| \underline{C}(x)^2 \right\|}{(\sum_{j=1}^{N} \underline{c}_j(x))^2} \right] \left[2 - \tfrac{\alpha_2}{m} \frac{\left\| \underline{C}(x)^2 \right\|}{(\sum_{j=1}^{N} \underline{c}_j(x))^2} \right]
\end{aligned} \tag{43}$$

Let $\left[2 - \frac{\alpha_2}{m} \frac{\|C(x)^2\|}{\left(\sum_{j=1}^{N} \underline{c}_j(x) \right)^2} \right] > 0$ then $\Delta V < 0$, i.e., we can select the learning rate α_2 in the following range

$$\frac{2m(\sum_{j=1}^{N} \underline{c}_j(x))^2}{\|\underline{C}(x)^2\|} > \alpha_2 > 0 \tag{44}$$

Since ΔV is negative definite, it follows that $\Delta V < 0$ for all $t > 0$, and $e(t) \to 0$ for $t \to \infty$. This ensures the convergence of the adaptive type-2 fuzzy CMAC learning process. Accordingly, the aircraft landing control system is locally asymptotically stable.

The aircraft system has four inputs—altitude, altitude rate, altitude command, and altitude rate command—which are fed to both the PID controller and the type-2 fuzzy CMAC. The pitch autopilot control signal, U, is computed as the sum of the PID output (U_{PID}) and the type-2 fuzzy CMAC output ($U_{fuzzyCMAC}$). While the PID controller ensures baseline stability, the type-2 fuzzy CMAC progressively enhances control performance, particularly under strong wind disturbances. The type-2 fuzzy CMAC operates in two phases: recall and learning. During recall, the CMAC uses $Y_d(k+1)$ and $Y(k)$ to access memory weights and generate an output $U_{fuzzyCMAC}$, where $Y(k)$ is the current aircraft output and $Y_d(k+1)$ the desired output at the next step (**Figure 8**). This output is combined with the PID signal to form the final control command U. In the learning phase (**Figure 9**), U serves as the desired output. The error between this signal and the actual system output ($U - U_{fuzzyCMAC}$) is calculated to update the memory weights at $Y(k)$ and $Y(k+1)$. Iterative updates reduce the error over time, allowing the type-2 fuzzy CMAC to effectively complement the PID controller and optimize overall aircraft landing performance.

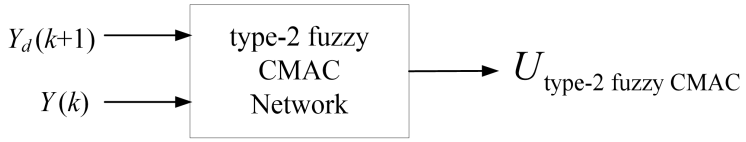

Figure 8. The control process of type-2 fuzzy CMAC.

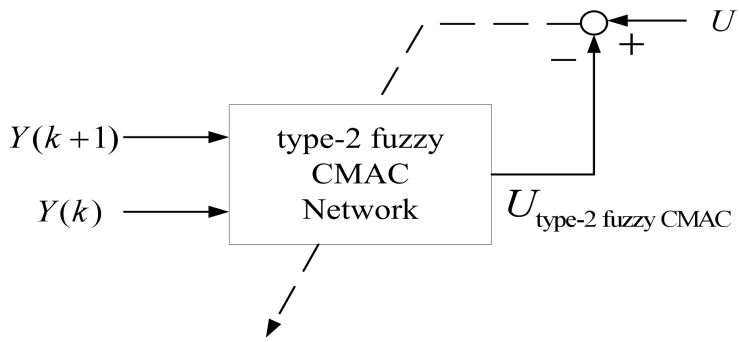

Figure 9. The learning process of type-2 fuzzy CMAC.

4. Simulations

The aircraft begins the Automatic Landing System operation at 500 ft altitude, 9,240 ft from the touchdown point, with a flight path angle of -3° and an airspeed of 234.7 ft/sec. A successful landing is defined by the following criteria:

$$-3 \leq \dot{h}(T) \text{ ft/sec} \leq 0, 200 \leq \dot{x}(T) \text{ ft/sec } \leq 270,$$

$$-300 \leq x(T) \text{ ft } \leq 1000, -10 \leq \theta(T) \text{ degree } \leq 5,$$

where T is the touchdown time, $\dot{h}(T)$ the vertical speed, $x(T)$ the horizontal position, $\dot{x}(T)$ the horizontal speed, and $\theta(T)$ the pitch angle at touchdown.

Table 2 presents the PID controller's performance under varying turbulence intensities [15]. With its original gains, the controller successfully guides the aircraft only at wind speeds up to 30 ft/sec. At this threshold, the aircraft attains a pitch angle of -0.17°, vertical speed of −2.19 ft/sec, horizontal velocity of 234.7 ft/sec, and a horizontal touchdown position of 844 ft (**Figures 10–12**). Beyond 30 ft/sec, the conventional controller fails to ensure a safe landing, highlighting the need for adaptive or fuzzy control strategies.

Table 2. PID control.

Wind (ft/sec)	Landing point (ft)	Vertical speed (ft/sec)	Pitch angle (degree)
0	797	−2.83	−1.41
10	910	−2.55	−0.85
20	809	−2.38	−0.59
30	844	−2.19	−0.17
40	1020	−1.72	0.44

Table 3 summarizes the type-1 fuzzy CMAC's performance under varying turbulence intensities. Unlike the conventional PID controller, this approach successfully guides the aircraft to a safe landing across a broader wind range of 10 ft/sec to 90 ft/sec [34].

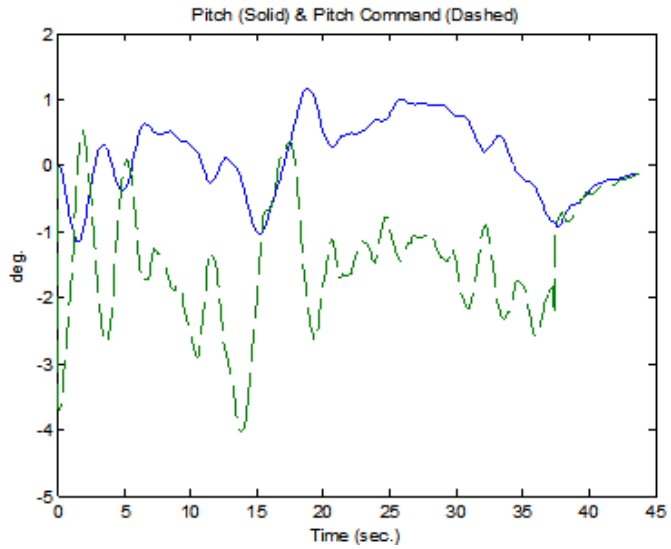

Figure 10. Pitch and pitch command under turbulence conditions of 30 ft/sec.

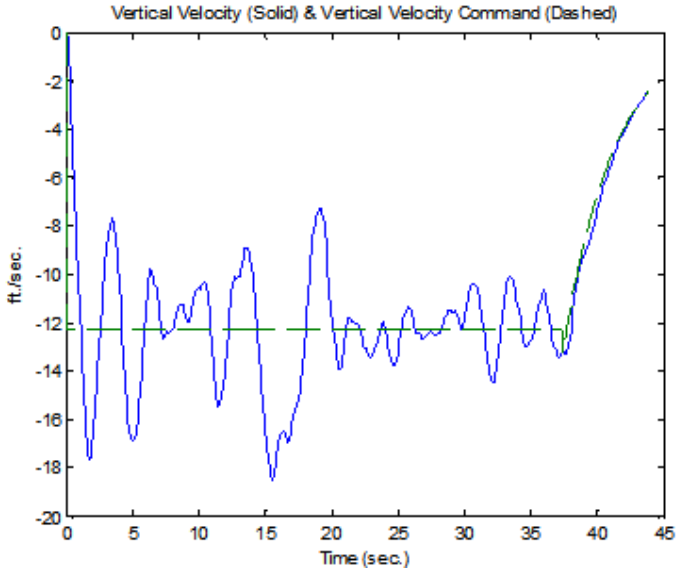

Figure 11. Vertical velocity and velocity command under turbulence conditions of 30 ft/sec.

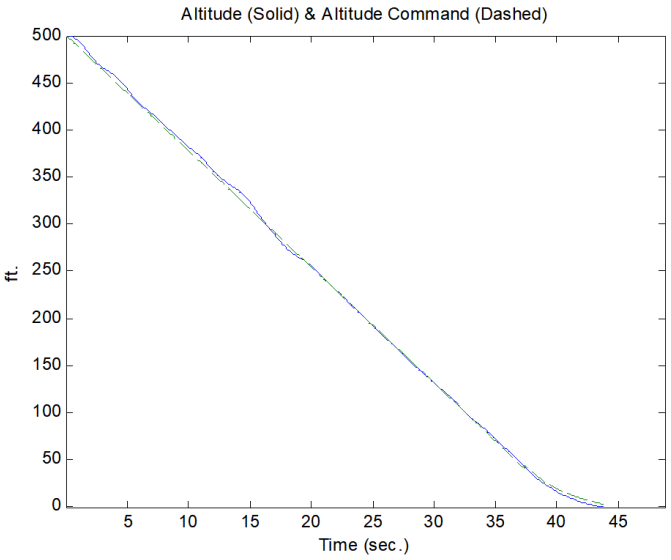

Figure 12. Altitude and altitude command under turbulence conditions of 30 ft/sec.

Table 3. Type-1 fuzzy CMAC control.

Wind (ft/sec)	Landing point (ft)	Vertical speed (ft/sec)	Pitch angle (degree)
10	656	−1.54	−0.88
20	574	−1.89	−0.41
30	856	−2.54	−0.18
40	656	−2.30	0.06
50	515	−2.21	0.68
60	422	−1.21	2.22
70	774	−1.64	1.37
80	468	−2.58	1.26
90	809	−2.45	2.27

Table 4 summarizes the performance of the adaptive type-1 fuzzy CMAC with an adaptive learning rate. Using the original pitch autopilot parameters ($\alpha = \frac{m(\Sigma C(x))^2}{\|C(x)^2\|}$, $m = 10$ blocks), this controller successfully guides the aircraft under wind conditions up to 100 ft/sec, outperforming both the standard type-1 fuzzy CMAC and the conventional PID controller (**Figures 13–15**).

Table 4. Adaptive type-1 fuzzy CMAC control.

Wind (ft/sec)	Landing point (ft)	Vertical speed (ft/sec)	Pitch angle (degree)
10	844	−2.51	−0.90
20	879	−2.29	−0.55
30	70	−2.84	−0.09
40	774	−2.09	0.10
50	738	−2.44	0.26
60	656	−2.78	0.87
70	551	−2.36	1.51
80	480	−2.05	1.95
90	433	−1.80	2.39
100	621	−2.76	2.30

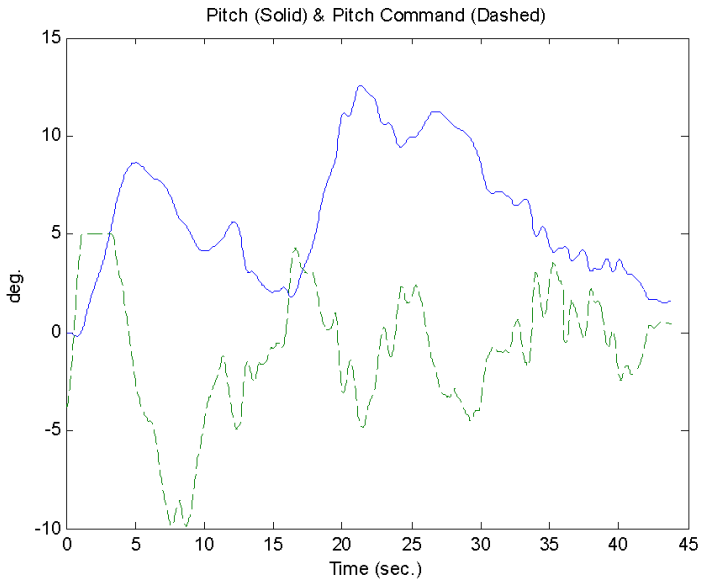

Figure 13. Pitch and pitch command under turbulence conditions of 100 ft/sec.

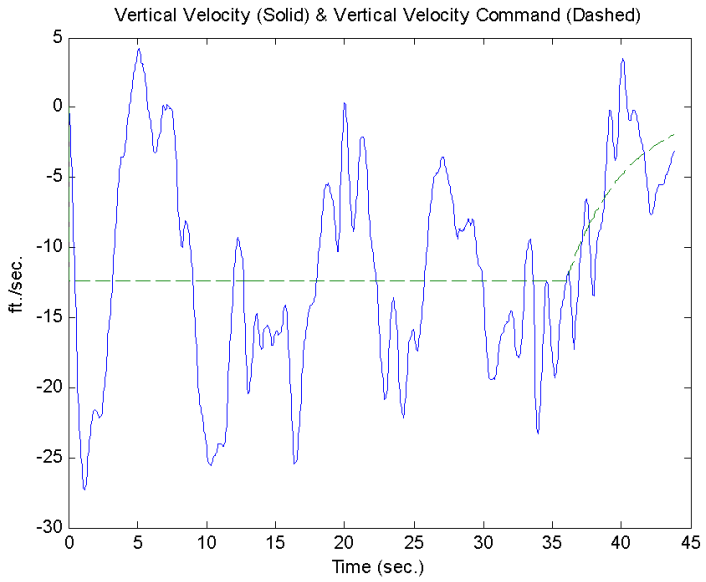

Figure 14. Vertical velocity and velocity command under turbulence conditions of 100 ft/sec.

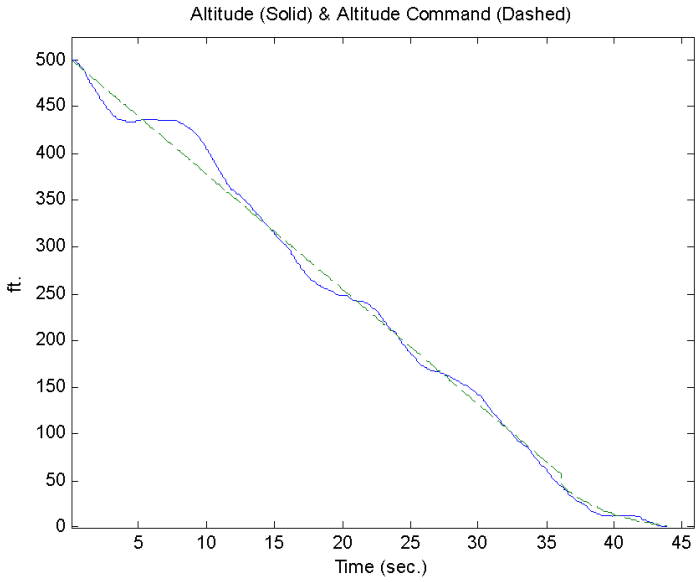

Figure 15. Altitude and altitude command under turbulence conditions of 100 ft/sec.

Table 5 presents the results for the type-2 fuzzy CMAC, and **Table 6** shows the performance of the adaptive type-2 fuzzy CMAC. As shown in **Figures 16** and **17**, the adaptive controller significantly outperforms previous approaches, successfully guiding the aircraft under wind conditions up to 160 ft/sec.

Table 5. Type-2 fuzzy CMAC control.

Wind (ft/sec)	Landing point (ft)	Vertical speed (ft/sec)	Pitch angle (degree)
30	703	−2.30	0.05
50	668	−2.19	0.19
70	644	−2.43	0.77
90	562	−2.03	1.88
110	691	−1.82	2.85
130	609	−1.70	3.64

Table 6. Adaptive type-2 fuzzy CMAC control.

Wind (ft/sec)	Landing point (ft)	Vertical speed (ft/sec)	Pitch angle (degree)
30	933	−2.11	−0.33
50	802	−1.88	0.85
70	575	−1.85	1.63
90	184	−2.62	2.85
110	275	−2.58	3.66
130	493	−2.32	3.81
150	366	−2.90	3.21
160	980	−2.88	2.66

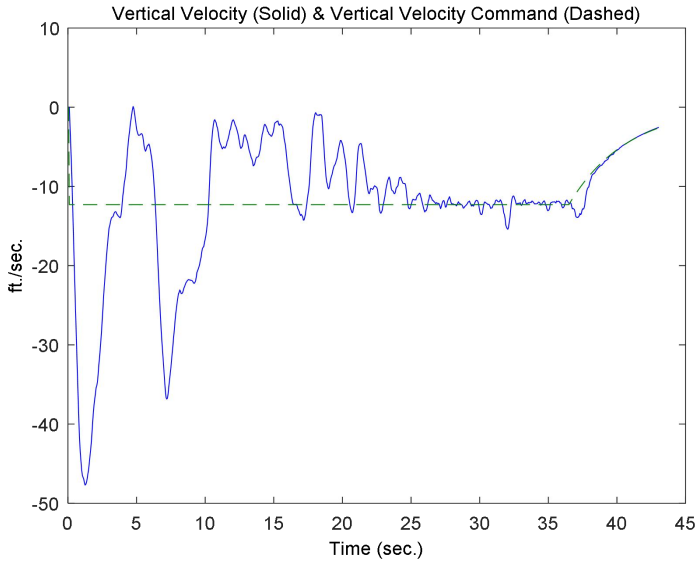

Figure 16. Vertical velocity and velocity command under turbulence conditions of 160 ft/sec.

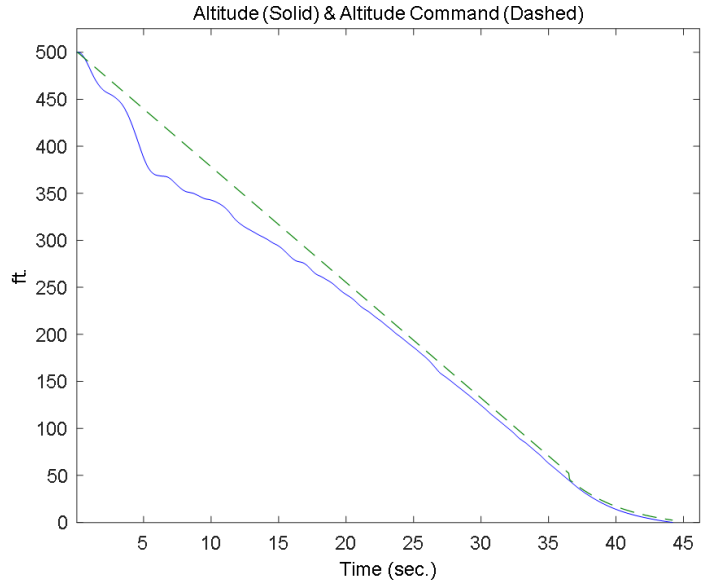

Figure 17. Altitude and altitude command under turbulence conditions of 160 ft/sec.

The aircraft system used identical simulation parameters for all tests. For each wind strength, ten randomly generated turbulence profiles were applied, and the

controller was required to successfully handle all profiles to demonstrate robustness.

5. Conclusions

This paper explores the use of intelligent control techniques in aircraft automatic landing systems, emphasizing both the conventional fuzzy CMAC and an enhanced version with an adaptive learning rule. In standard CMAC architectures, weight updates are governed by a fixed learning rate, which limits learning efficiency: a small rate slows adaptation, whereas a large rate may cause instability. To overcome this issue, we propose an adaptive learning rate that adjusts dynamically during the weight update process, enhancing overall learning performance. Simulation results confirm the proposed system's superior tracking ability and adaptability to environmental disturbances. The type-2 fuzzy CMAC demonstrates improved performance over both the conventional PID controller and the type-1 fuzzy CMAC, effectively rejecting disturbances under turbulence conditions up to 130 ft/sec. Furthermore, the adaptive type-2 fuzzy CMAC achieves greater robustness, enabling safe landings under turbulence levels reaching 160 ft/sec. System stability is verified through gradient descent and Lyapunov-based analyses. Although the proposed approach introduces higher computational complexity compared with fixed-rate models, the training is conducted offline, which greatly reduces online computation and ensures real-time feasibility. Future research may include hardware-in-the-loop validation, while large language models (LLMs) offer promising opportunities for autonomous navigation reasoning in subsequent studies [35].

Author contributions: Conceptualization, T-CY and J-GJ; methodology, T-CY and J-GJ; software, T-CY; validation, T-CY and J-GJ; formal analysis, T-CY and J-GJ; investigation, T-CY and J-GJ; resources, J-GJ; data curation, T-CY and J-GJ; writing—original draft preparation, T-CY; writing—review and editing, J-GJ; visualization, J-GJ; supervision, J-GJ; project administration, J-GJ; funding acquisition, J-GJ. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Science Council, Taiwan, under Grant NSC 97-2221-E- 019-025.

Conflict of interest: The authors declare no conflict of interest.

References

1. Aircraft Accident Statistics. Available online: https://www.planecrashinfo.com/2010/2010.htm (accessed on 13 August 2025).
2. Buschek H, Calise AJ. Uncertainty modeling and fixed-order controller design for a hypersonic vehicle model. Journal of Guidance, Control, and Dynamics. 1997; 20(1): 42–48. doi: 10.2514/2.4031
3. Federal Aviation Administration. Automatic Landing Systems. Available online: https://www.faa.gov/documentLibrary/media/Advisory_Circular/AC_120-118.pdf (accessed on 13 August 2025).
4. Perrusquia A, Yu W. Discrete-time H2 neural control using reinforcement learning. IEEE Transactions on Neural Networks and Learning Systems. 2021; 32(11): 4879–4889. doi: 10.1109/TNNLS.2020.3026010
5. Chen B, Lin C. Finite-time stabilization-based adaptive fuzzy control design. IEEE Transactions on Fuzzy Systems. 2021; 29(8): 2438–2443. doi: 10.1109/TFUZZ.2020.2991153

6. Langazane SN, Saha AK. Effects of particle swarm optimization and genetic algorithm control parameters on overcurrent relay selectivity and speed. IEEE Access. 2022; 10: 4550–4567. doi: 10.1109/ACCESS.2022.3140679
7. Salimi-Badr A, Ebadzadeh MM. A novel self-organizing fuzzy neural network to learn and mimic habitual sequential tasks. IEEE Transactions on Cybernetics. 2022; 52(1): 323–332. doi: 10.1109/TCYB.2020.2984646
8. Innocenti M, Pollini L, Turra D. A fuzzy approach to the guidance of unmanned air vehicles tracking moving targets. IEEE Transactions on Control Systems Technology. 2008; 16(6): 1125–1137. doi: 10.1109/TCST.2008.917224
9. Jinhui Z, Yuanqing X. Design of static output feedback sliding mode control for uncertain linear systems. IEEE Transactions on Industrial Electronics. 2010; 57(6): 2161–2170. doi: 10.1109/TIE.2009.2033485
10. Liang Y, Chen X, Xu R. Research on longitudinal landing track control technology of carrier-based aircraft. In: Proceedings of the 2020 Chinese Control And Decision Conference (CCDC); 22 August 2020; Hefei, China. pp. 3039–3044. doi: 10.1109/CCDC49329.2020.9164274
11. Akmeliawati R, Mareels IMY. Nonlinear energy-based control method for aircraft automatic landing systems. IEEE Transactions on Control Systems Technology. 2010; 18(4): 871–884. doi: 10.1109/TCST.2009.2030788
12. Jorgensen CC, Schley C. A neural network baseline problem for control of aircraft flare and touchdown. In: Miller WT, Sutton RS, Werbos PJ (editors). Neural Networks for Control. The MIT Press; 1991. pp. 403–426. doi: 10.7551/mitpress/4939.003.0023
13. Liu Z, Yang T, Li Zhongjian, et al. Automatic landing trajectory control of aircraft based on Iterative-LQR algorithm. In: Proceedings of the 2022 9th International Forum on Electrical Engineering and Automation (IFEEA); 4 November 2022; Zhuhai, China. pp. 925–929. doi: 10.1109/IFEEA57288.2022.10038260
14. Wu Q, Zhu Q. Longitudinal direct lift control of carrier-based aircraft automatic landing based on NTSM-LADRC. In: Proceedings of the 14th Asian Control Conference; 5 July 2024; Dalian, China. pp. 1980–1986.
15. Juang J-G, Cheng K-C. Application of neural networks to disturbances encountered landing control. IEEE Transactions on Intelligent Transportation Systems. 2006; 7(4): 582–588. doi: 10.1109/TITS.2006.884885
16. Fang X, Jiang J, Chen W-H. Model predictive control with wind preview for aircraft forced landing. IEEE Transactions on Aerospace and Electronic Systems. 2023; 59(4): 3995–4004. doi: 10.1109/TAES.2023.3235321
17. Chen C, Yong K. Spatial prescribed performance-based carrier landing control for aircraft under external disturbances. In: Proceedings of the 2023 7th International Symposium on Computer Science and Intelligent Control (ISCSIC); 27 October 2023; Nanjing, China. pp. 201–204. doi: 10.1109/ISCSIC60498.2023.00049
18. Albus JS. Data storage in the cerebellar model articulation controller (CMAC). Journal of Dynamic Systems, Measurement, and Control. 1975; 97(3): 228–233. doi: 10.1115/1.3426923
19. Groza B, Popa L, Murvay P-S. Highly efficient authentication for CAN by identifier reallocation with ordered CMACs. IEEE Transactions on Vehicular Technology. 2020; 69(6): 6129–6140. doi: 10.1109/TVT.2020.2990954
20. Zhong Z, Lam H-K, Ying H, et al. CMAC-based SMC for uncertain descriptor systems using reachable set learning. IEEE Transactions on Systems, Man, and Cybernetics: Systems. 2024; 54(2): 693–703. doi: 10.1109/TSMC.2023.3311540
21. Lee KS. Performance analysis for secure communication based on CMAC. In: Proceedings of the 2022 22nd International Conference on Control, Automation and Systems (ICCAS); 27 November 2022; Jeju, South Korea. pp. 378–381. doi: 10.23919/ICCAS55662.2022.10003867
22. Li Z, Hiraoka K, Yamamoto T. Application of a database-driven PID controller using a CMAC memory in a hydraulic system. In: Proceedings of the IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society; 19 October 2023; Singapore. pp. 1–6. doi: 10.1109/IECON51785.2023.10311646
23. Kai Z, Feng Q. Fuzzy CMAC and its application. In: Proceedings of the 3rd World Congress on Intelligent Control and Automation (Cat. No.00EX393); June 2000; Hefei, China. pp. 944–947. doi: 10.1109/WCICA.2000.863372
24. Zadeh LA. Fuzzy sets. Information and Control. 1965; 8(3): 338–353. doi: 10.1016/S0019-9958(65)90241-X
25. Karnik NN, Mendel JM, Qilian L. Type-2 fuzzy logic systems. IEEE Transactions on Fuzzy Systems. 1999; 7(6): 643–658. doi: 10.1109/91.811231
26. Peng Y-F, Wai R-J, Lin C-M. Implementation of LLCC-resonant driving circuit and adaptive CMAC neural network control for linear piezoelectric ceramic motor. IEEE Transactions on Industrial Electronics. 2004; 51(1): 35–48. doi: 10.1109/TIE.2003.822078
27. Chung C-M, Hsu C-F, Lin C-M, et al. Design of a robust adaptive CMAC system for BLDC motors with PI type parameter adaptation. In: Proceedings of the 2008 International Conference on Machine Learning and Cybernetics; 7 July 2008; Kunming, China. pp. 3827–3832. doi: 10.1109/ICMLC.2008.4621072
28. Flight Safety Foundation-Taiwan. Available online: https://www.caa.gov.tw/article.aspx?a=1665&lang=2 (accessed

on 13 August 2025).
29. Yang T-C, Juang J-G. Application of adaptive type-2 fuzzy CMAC to automatic landing system. In: Proceedings of the 2010 International Symposium on Computational Intelligence and Design; 10 October 2010; Hangzhou, China. pp. 221–224. doi: 10.1109/ISCID.2010.146
30. Juang J-G, Chiou H-K, Lee C-L. Intelligent computation and DSP-based landing control. Journal of Marine Science and Technology. 2017; 25(4). doi: 10.6119/JMST-017-0329-2
31. Juang J-G, Yu C-L, Lin C-M, et al. Real-time image recognition and path tracking of a wheeled mobile robot for taking an elevator. Acta Polytechnica Hungarica. 2013; 10(6). doi: 10.12700/APH.10.06.2013.6.1
32. Liang Q, Mendel JM. Interval type-2 fuzzy logic systems: theory and design. IEEE Transactions on Fuzzy Systems. 2000; 8(5): 535–550. doi: 10.1109/91.873577
33. Lee C-L, Juang J-G. Aircraft landing control in wind shear condition. In: Proceedings of the 2011 International Conference on Machine Learning and Cybernetics; 17 July 2011; Guilin, China. pp. 1180–1185. doi: 10.1109/ICMLC.2011.6016885
34. Juang J-G, Lee C-L. Applications of cerebellar model articulation controllers to intelligent landing system. Journal of Universal Computer Science. 2009; 15(13): 2586–2607. Available online: http://scholars.ntou.edu.tw/handle/123456789/4992
35. Zou Y, Xu Z, Wang T, et al. Generative AI-driven dynamic information prioritization for enhanced autonomous driving. IEEE Transactions on Intelligent Transportation Systems. 2025; 1–14. doi: 10.1109/TITS.2025.3552795