IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 15, 2019, accepted August 26, 2019, date of publication August 29, 2019, date of current version September 17, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2938273

Adaptive Neural Network Control and Optimal Path Planning of UAV Surveillance System With Energy Consumption Prediction

RONG-JONG WAI, (Senior Member, IEEE), AND ALEX S. PRASETIA
Department of Electronics and Computer Engineering, National Taiwan University of Science and Technology, Taipei 106, Taiwan

Corresponding author: Rong-Jong Wai (rjwai@mail.ntust.edu.tw)

This work was financially supported in part by the Ministry of Science and Technology of Taiwan under Grant MOST 108-2221-E-011-080-MY3, and in part by the E400 Department from Information and Communication Research Laboratories (ICL), Industrial Technology Research Institute (ITRI), Taiwan.

ABSTRACT A surveillance system is one of the most interesting research topics for an unmanned aerial vehicle (UAV). However, the problem of planning an energy-efficient path for the surveillance purpose while anticipating disturbances and predicting energy consumptions during the path tracking is still a challenging problem in recent years. The optimal path planning and the disturbance rejection control for a UAV surveillance system are investigated in this paper. A trained and tested energy consumption regression model is used to be the cost function of an optimal path planning scheme, which is designed from a clustered 3D real pilot flight pattern with the proposed K-agglomerative clustering method, and is processed via A-star and set-based particle-swarm-optimization (S-PSO) algorithm with adaptive weights. Moreover, an online adaptive neural network (ANN) controller with varied learning rates is designed to ensure the control stability while having a reliably fast disturbance rejection response. The effectiveness of the proposed framework is verified by numerical simulations and experimental results. By applying the proposed optimal path planning scheme, the energy consumption of the optimal path is only 72.3397 Wh while the average consumed energy of real pilot flight data is 96.593Wh. In addition, the proposed ANN control improves average root-mean-square error (RMSE) of horizontal and vertical tracking performance by 49.083% and 37.50% in comparison with a proportional-integral-differential (PID) control and a fuzzy control under the occurrence of external disturbances. According to all of the results, the combination of the proposed optimal path planning scheme and ANN controller can achieve an energy-efficient UAV surveillance systems with fast disturbance rejection response.

INDEX TERMS Unmanned aerial vehicle (UAV), optimal path planning, set-based particle-swarm-optimization (S-PSO), adaptive weights, adaptive neural network (ANN), varied learning rates.

I. INTRODUCTION

A surveillance system is one of the most interesting research topics for an unmanned aerial vehicle (UAV). An autonomous surveillance system using a UAV will face three major problems, which are energy consumption prediction, energy-efficient optimal path planning, and stable control design with disturbance anticipation property for the UAV to follow reference paths with low tracking error even in an area with a high chance of disturbance occurrence (e.g. windy area).

The associate editor coordinating the review of this manuscript and approving it for publication was Yongping Pan.

The energy consumption prediction of a UAV flight has been discussed in several researches. Methods to be used for solving this problem can be categorized into white-box and black-box methods. A white-box method was used by Liu *et al.* by designing a mathematical model for the UAV's power consumption with aerodynamic equations [1]. Abdilla *et al.* also did a white-box method by conducting a detail experiment to model a Li-Po battery, which then became the reference to model the duration of the quadrotor flight duration [2]. Another published paper with the white-box method are done by Bezzo *et al.* via the relation of power with the motor thrust [3]. In spite of high accuracy to be

achieved in [1]–[3], the white-box method in general will require detail physical parameters of a UAV, such that it will increase the complexity of the method. On the other hand, the black-box method can be easily used with several adjustments to increase the accuracy. The most comparable adjustment to be done in previous researches is the introduction of various inputs, which can represent several dynamics of a UAV. All of those works has been dicussed in our previous work that will be implemented in this surveillance systems as the cost function of the optimal path planning algorithm [4].

The optimal path planning problem is also an interesting research topic, which has been discussed by many researchers for various applications. Several methods that have been used can be classified into heuristic methods (e.g., Christofides [5] or Concorde [6]) and meta-heuristic methods (e.g., genetic algorithm [7] or discrete particle-swarm-optimization [8]). Although nowadays heuristic methods have been proved to get more efficient, the computation time still significantly depends on the problem size and thus will decrease its efficiency. On the other hand, meta-heuristic methods are more independent to problem size, which attracts researchers to apply this method and solve various node routing problem.

One of the meta-heuristic method that has been widely used to solve an optimal path planning problem is the particle swarm optimization (PSO). However, implementing the PSO to a discrete problem requires some adjustments that have been proposed by several researchers. Hadia *et al.* [8] proposed a swap operator to transform the particle's movement into a swap movement. A set-based particle-swarm-optimization (S-PSO) is proposed by Chen *et al.* [9] to have a more similar characteristic of the continuous-type PSO by transforming the movement in sequence of nodes into sets of paths with probability values. To have more improvements of the PSO algorithm in a discrete problem, Weng et al. [10] implemented a set-based comprehensive-learning PSO (S-CLPSO) for virtual machine placement problem. However, the inertia weight parameter in [9], [10] used a linearly dicreasing inertia over iterations. The performance of these methods will be analyzed and compared with the proposed adaptive inertia weight in this paper.

Applications of meta-heuristic and heuristic methods to the UAV routing problem have also been conducted by several researchers. Amorosi *et al.* [11] modified the genetic algorithm to form a decomposition-based approach to be constructed for a cellular network coverage problem with charging stations' batteries as constraints. Lim *et al.* [12] conducted a UAV surveilance system research as a travelling salesman problem with time window and solved it with a dynamic programming. Dorling *et al.* [13] implemented the simulated annealing method for vehicle routing of a drone delivery problem. Although those researches in [11]–[13] got good results, they did not include the actual flight data and thus, simulations of their flight did not have any accurate longitude, latitude, or altitude data. This matter is important to show the possibility of the method to be implemented in a real case.

An idea of including clustered actual flight data by the K-means clustering into the path planning problem has been proposed by Kwak and Sung [14] by consequently solving arc routing problems with the A-star path finding algorithm and node routing problems with the enhanced A-star algorithm to plan an optimal path of a UAV surveillance system. Although Kwak and Sung [14] had included pilot flight data, it did not include any energy consumption information. Moreover, the method in [14] can only be used for a two-dimensional problem, where the building and the surveillance points are assumed to be in the same altitude. This assumption may lead to unreliability of the method in [14] to the real case. Besides, there is a connectivity issue of using the K-means algorithm for a 3-dimensional energy-efficient path planning problem, and it will be discussed in this paper.

The third problem is the control design of a UAV with fast disturbance rejection response. The UAV itself can be classified into several types. The type used in this paper is a UAV with multi-rotors, which can be controlled from the thurst produced by motors. As for a nonlinear plant, a nonlinear control has been widely adopted to control a UAV. One of the nonlinear control is the backstepping control of quadrotors, which was investigated by Jiang *et al.* [15]. Another nonlinear control, a sliding-mode control, was also designed for a hexarotor by Busarakum and Srichatrapimuk [16]. However, nonlinear control always requires a vast detail in UAV dynamics that complicate the implementation of the control design process. To overcome this problem, Walid *et al.* [17] has implemented numerical simulations of a cascaded proportional-integral-differential (PID) controller for a quadrotor. The PID control scheme was also designed for a hexacopter by Moussid *et al.* [18]. Other than PID, a simple heuristic approach with fuzzy control was proposed by Fakurian *et al.* [19]. Although all of those controllers in [15]–[19] can be easily implemented and have remarkable results, PID and fuzzy control parameters still need to be tuned carefully as it is easier for a nonlinear plant to be unstable. Besides, disturbance responses of PID controllers are slow, which in turn will increase tracking errors. Although several adjustment mechanisms (e.g. disturbance observer) can be made to solve this problem, an additional observer design will complicate the design process with more detailed system dynamic to be identifed.

In recent years, many researches has been done to apply a neural network (NN) to the control field to deal with nonlinearities and uncertainties of the control system. A NN with an online learning framework has been proposed to solve this kind of problem on an ultrasonic motor servo-drive [20] and a permanent magnet syncronous motor servo-drive [21] by Lin and Wai. A fuzzy neural network (FNN) controller was also proposed by Wai *et al.* [22] to control a single-stage boost inverter. The NN control is believed to have an advantage of online learning to solve the afformentioned disturbance rejection problem.

In order to improve the aforementioned drawbacks, this paper proposes a UAV energy-consumption prediction

model, an optimal path-planning scheme, and a disturbance rejection control strategy for a UAV surveillance system. The prediction of energy consumption for each movement based on our previous framework in [4] is used as the cost function of the optimal path-planning scheme. The path is generated from cluster centers of 3D real pilot flight pattern data by the proposed K-agglomerative clustering method. The K-agglomerative clustering is proposed to solve the connectivity problem of K-means clustering. Moreover, the proposed adaptive-weight set-based particle-swarm-optimization (S-PSO) coupled with A-star algorithm is implemented to solve the path planning problem. In addition, an online adaptive neural network (ANN) controller with varied learning rates is investigated to overcome the disturbance rejection problem while minimizing the tracking error. The optimal path planning scheme is evaluated by comparing the predicted energy and the average energy consumed by real pilot flight data. Futhermore, numerical simulations of a UAV are built to evaluate and to compare the effectiveness of ANN controller in comparison with a traditional PID control and a fuzzy control under the occurrence of disturbances.

This paper is organized into six sections. Following the introduction, the mathematical model of a hexarotor is presented in Section II. In Section III, a newly-designed control scheme including an adaptive NN controller and a dynamic extension tracking controller are explained in detail. In Section IV, the optimal path planning composed of adaptive-weight set-based particle-swarm-optimization (S-PSO) coupled with A-star algorithm is designed. In Section V, numerical simulations and experimental results are provided to validate the effectiveness of the proposed framework, and performance comparisons with other scheme in previous researches are given to show the superiority of the proposed framework. Finally, some conclusions are drawn in Section VI.

II. MATHEMATICAL MODEL OF HEXAROTOR

The unmanned aerial vehicle (UAV) used in this study is the winning R&D 100 2018 conference award project to be called automatic police UAV patrol system (APUPS) [23] and is depicted in Fig. 1. The hexarotor is a vertical taking-off and landing (VTOL) type of a UAV. The thrust is produced by six rotors, which are placed around its center of mass with six degrees of freedom. Based on Walid *et al.* [17] and Moussid *et al.* [18], the modelling for rigid bodies could be derived with the Euler-Lagrange method. The body frame of a hexarotor is described in Fig. 2. The translational movements are in x-axis, y-axis, and z-axis, where the rotational movement in x-axis is roll (ϕ), rotational movement in y-axis is pitch (θ), and rotational movement in z-axis is yaw (ψ). Those movements are mapped in vectors $\boldsymbol{\xi} = [x, y, z]^T$ and $\boldsymbol{\varsigma} = [\phi, \theta, \psi]^T$. The rotation speeds of six motors are notated as $\Omega_i|_{i=1,2,\ldots,6}$.

The rotation matrix R of the hexarotor can be explained in (1), and the Euler-Lagrange equation of a hexarotor can be

FIGURE 1. APUPS.

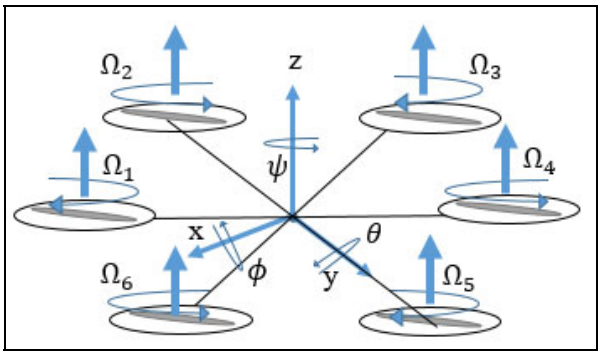

FIGURE 2. Body fixed frame of hexarotor.

expressed in (2)

$$R = \begin{bmatrix} c\theta\, c\psi & s\phi\, s\theta\, c\psi - c\phi\, s\psi & c\phi\, s\theta\, c\psi + s\phi\, s\psi \\ s\psi\, c\theta & s\phi\, s\theta\, s\psi + c\phi\, c\psi & c\phi\, s\theta\, s\psi - s\phi\, c\psi \\ -s\theta & s\phi\, c\theta & c\phi\, c\theta \end{bmatrix} \tag{1}$$

$$\begin{cases} \dfrac{d}{dt}\left(\dfrac{\delta L}{\delta \dot{\boldsymbol{q}}}\right) - \dfrac{\delta L}{\delta \boldsymbol{q}} = \begin{bmatrix} \sum F \\ \sum \tau \end{bmatrix} \\ L(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \Upsilon_p - \Upsilon_k \end{cases} \tag{2}$$

where $\boldsymbol{q} = [x, y, z, \phi, \theta, \psi]^T$ is the system state vector, F is the translational force, τ is the torque, Υ_p is the potential energy, and Υ_k is the kinetic energy. According to (2) the translational dynamic can be represented as follows:

$$\frac{d}{dt}\left(\frac{\delta L}{\delta \dot{\boldsymbol{q}}}\right) - \frac{\delta L}{\delta \boldsymbol{q}} = m\ddot{\boldsymbol{\xi}} = \boldsymbol{F_p} + \boldsymbol{F_g} \tag{3}$$

where $\boldsymbol{F_p} = R[\,0 \;\; 0 \;\; \sum_{i=1}^{6} b\,\Omega_i\,]^T$ is the thrust force vector, in which b is the thrust constant and Ω_i is the motor rotation speed; $\boldsymbol{F_g} = -[\,0 \;\; 0 \;\; m\,g\,]^T$ is the gravity force vector. In addition, the rotational dynamic can be expressed as follows:

$$\frac{d}{dt}\left(\frac{\delta L}{\delta \dot{\boldsymbol{q}}}\right) - \frac{\delta L}{\delta \boldsymbol{q}} = \boldsymbol{I}_t\, \ddot{\boldsymbol{\varsigma}} + \begin{bmatrix} \dot{\theta}\,\dot{\psi}(I_{zz} - I_{yy}) \\ \dot{\phi}\,\dot{\psi}(I_{xx} - I_{zz}) \\ \dot{\phi}\,\dot{\theta}(I_{yy} - I_{xx}) \end{bmatrix} = \boldsymbol{\tau_p} + \boldsymbol{\tau_{gh}} \tag{4}$$

where $\boldsymbol{I_t} = diag(I_{xx}, I_{yy}, I_{zz})$ is the inertia matrix; $\boldsymbol{\tau_p}$ is the actuated torque vector to be represented as follows:

$$\begin{aligned}\boldsymbol{\tau_p} &= \begin{bmatrix}\tau_{p\varphi}\\ \tau_{p\theta}\\ \tau_{p\psi}\end{bmatrix}\\ &= \begin{bmatrix}bl[-\Omega_2^2+\Omega_5^2+0.5(-\Omega_1^2-\Omega_3^2+\Omega_4^2+\Omega_6^2)]\\ bl\sqrt{3}(-\Omega_1^2+\Omega_3^2+\Omega_4^2-\Omega_6^2)/2\\ d(-\Omega_1^2+\Omega_2^2-\Omega_3^2+\Omega_4^2-\Omega_5^2+\Omega_6^2)\end{bmatrix}\end{aligned} \tag{5}$$

where d is the drag constant, l is the distance to center of gravity, and $\boldsymbol{\tau}_{\text{gh}} = -I_r\begin{bmatrix}\dot{\theta}\,\Omega_g & \dot{\phi}\,\Omega_g & 0\end{bmatrix}^T$ is the gyroscopic effect vector of the propeller, in which $\Omega_g = -\Omega_1+\Omega_2-\Omega_3+\Omega_4-\Omega_5+\Omega_6$, and I_r is the rotor inertia constant. From (3)-(5), the translation and rotation dynamics can be represented as follows:

$$\begin{cases}\ddot{x} = [(c\phi\, s\theta\, c\psi + s\phi\, s\psi)U_1]/m\\ \ddot{y} = [(c\phi\, s\theta\, s\psi - s\phi\, c\psi)U_1]/m\\ \ddot{z} = [(c\phi\, c\theta)U_1 - mg]/m\\ \ddot{\phi} = [U_2 + \dot{\theta}\,\dot{\psi}(I_{yy}-I_{zz}) - I_r\,\dot{\theta}\,\Omega_g]/I_{xx}\\ \ddot{\theta} = [U_3 + \dot{\phi}\,\dot{\psi}(I_{zz}-I_{xx}) - I_r\,\dot{\phi}\;\Omega_g]/I_{yy}\\ \ddot{\psi} = [U_4 + \dot{\phi}\,\dot{\theta}(I_{xx}-I_{yy})]/I_{zz}\end{cases} \tag{6}$$

According to (3) and (5), the relation of control inputs and motor speeds can be denoted as follows:

$$\begin{aligned}\begin{bmatrix}U_1\\U_2\\U_3\\U_4\end{bmatrix} &= \begin{bmatrix}b & b & b & b & b & b\\ -0.5bl & -bl & -0.5bl & 0.5bl & bl & 0.5bl\\ -\sqrt{3}bl/2 & 0 & \sqrt{3}bl/2 & \sqrt{3}bl/2 & 0 & -\sqrt{3}bl/2\\ -d & d & -d & d & -d & d\end{bmatrix}\\ &\times \begin{bmatrix}\Omega_1^2\\ \Omega_2^2\\ \Omega_3^2\\ \Omega_4^2\\ \Omega_5^2\\ \Omega_6^2\end{bmatrix}\end{aligned} \tag{7}$$

Therefore, Ω_i^2 can be obtained by the inverse of (7).

III. NEWLY-DESIGNED CONTROL SCHEME

A. NEURAL NETWORK CONTROL STRUCTURE

According to (6), a UAV is an under-actuated plant with six states and four control inputs. An adaptive neural network (ANN) controller with varied learning rates to control system states (z, ϕ, θ, and ψ) via U_1, U_2, U_3, and U_4; and a translational tracking control scheme with dynamic extensions to manipulate system states (x and y) are proposed in this paper. The control block diagram is depicted in Fig. 3. The variables of x_d, y_d, z_d, ϕ_d, θ_d, and ψ_d are the desired commands of system states x, y, z, ϕ, θ, and ψ, respectively. The transfer functions of the reference model in the outer loop

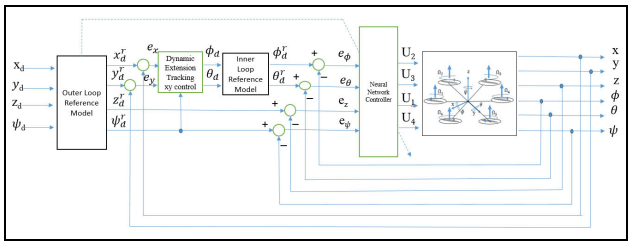

FIGURE 3. Framework of ANN control.

and the inner loop can be represented by (8a) and (8b), which are similar to low-pass filter types.

$$\frac{X_d^r}{X_d} = \frac{1}{s+\rho_1} \tag{8a}$$

$$\frac{\Phi_d^r}{\Phi_d} = \frac{1}{s+\rho_2} \tag{8b}$$

where X_d^r, X_d, Φ_d^r, and Φ_d^r are the Laplace transform of x_d^r, x_d, ϕ_d^r, and ϕ_d, respectively; s is the Laplace operator. The outer and inner loop structure is similar to a cascaded control structure, where the value of ρ_1 for the outer loop is set to be more than 10 times slower in comparison with the value of ρ_2 in the inner loop. Moreover, x_d^r, y_d^r, z_d^r, ϕ_d^r, θ_d^r and ψ_d^r, which are produced by reference models, are reference commands for system states x, y, z, ϕ, θ, and ψ, respectively. Tracking errors are defined as $e_x = x_d^r - x$, $e_y = y_d^r - y$, $e_z = z_d^r - z$, $e_\phi = \phi_d^r - \phi$, $e_\theta = \theta_d^r - \theta$, and $e_\psi = \psi_d^r - \psi$. In addition, U_1, U_2, U_3, and U_4 are control signals for z, ϕ, θ, and ψ, respectively.

A three-layer neural network structure is depicted in Fig. 4 [20]. The input signals are e_z, e_ϕ, e_θ, e_ψ, $\dot{e}_z$, $\dot{e}_\phi$, $\dot{e}_\theta$, and $\dot{e}_\psi$, which will be noted as x^i and explained in (9). The tracking error (e_z) and its derivative ($\dot{e}_z$) in the hidden layer is disconnected from the states of ϕ, θ, and ψ due to the difference of input range value. The angles of ϕ, θ, and ψ will be limited from -2π to 2π, but the value of z is from 0 to ∞. (O^i) is the output of activation function (f^i) of the input layer, which will be the input for the hidden layer and can be expressed in (10), where W^{ij} is the connective weight from the neuron (i) in the input layer to the neuron (j) in the hidden layer. The activation function (f^j) is a sigmoid function

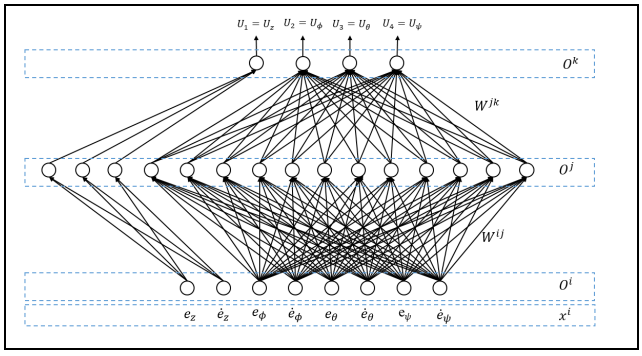

FIGURE 4. Neural network structure.

to mimic human brain, which will be used for the weight update calculation. The function in the output layer can be represented as (11), where W^{jk} is the connective weight from the neuron (j) in the hidden layer to the neuron (k) in the output layer. The output of the output layer is O^k, which is the output of the activation function (f^k) and is equal to the control signal (U_m), where $m = z, \phi, \theta$, or ψ.

Input layer:

$$net^i = x^i \qquad O^i = f^i(net^i) = net^i \tag{9}$$

Hidden layer:

$$net^j = \sum_i (W^{ij} O^i) \qquad O^j = f^j(net^j) = \frac{1}{1 + e^{-net^j}} \tag{10}$$

Output layer:

$$net^k = \sum_j (W^{jk} O^j) \qquad O^k = f^k(net^k) = net^k \tag{11}$$

B. ADAPTIVE NEURAL NETWORK CONTROL

The learning algorithms for connective weights are trained online with the energy functions (E) to be defined as follows:

$$E = \frac{1}{2} \sum_{m=z,\, \phi,\theta,\, \psi} e_m^2 \tag{12}$$

where the variable ($e_m|_{m=z,\, \phi,\, \theta \text{ or } \psi}$) is the tracking error at each system state. To minimize those values, the backpropagation algorithm is adopted to derive the update laws for connective weights. The corresponding update laws can be described in (13) with the learning rates (η^{ij} and η^{jk}).

$$\begin{aligned} W^{ij}(n+1) &= W^{ij}(n) - \eta^{ij} \frac{\partial E(n)}{\partial W^{ij}(n)} \\ W^{jk}(n+1) &= W^{jk}(n) - \eta^{jk} \frac{\partial E(n)}{\partial W^{jk}(n)} \end{aligned} \tag{13}$$

where n is the iteration number.

Then, chain rules are applied to the partial derivative mentioned in (13). Chain rules for the output layer can be expressed as (14), in which O^k is equal to U_1, U_2, U_3, and U_4.

$$\begin{aligned} \frac{\partial E(n)}{\partial W^{jk}(n)} &= \sum_m \frac{\partial E(n)}{\partial e_m(n)} \frac{\partial e_m(n)}{\partial m(n)} \frac{\partial m(n)}{\partial O^k(n)} \frac{\partial O^k(n)}{\partial W^{jk}(n)} \\ &= -\sum_m e_m(n) \frac{\partial m(n)}{\partial O^k(n)} O^j = -e_m(n) \frac{\partial m(n)}{\partial U_m(n)} O^j \end{aligned} \tag{14}$$

The term $e_m(n)\,[\partial m(n)/\partial U_m(n)]$ will be estimated with delta adaptation laws to be proposed in [21], and can be represented as follows:

$$e_m \frac{\partial m(n)}{\partial U_m(n)} = (e_m + \dot{e}_m) \tag{15}$$

Varied learning rates are designed to be adaptive by deriving a discrete-type Lyapunov function that is explained in [22]. A discrete-type Lyapunov function for the ANN controller is defined as follows:

$$V(n) = E(n) \tag{16}$$

The change of the Lyapunov function can be obtained as

$$\Delta V(n) = V(n+1) - V(n) = E(n+1) - E(n) \tag{17}$$

In order to prove the stability by assuring that $\Delta V(n) < 0$, the following term $E(n+1)$ should be smaller than $E(n)$:

$$\begin{aligned} &E(n+1) \\ &= E(n) + \Delta E(n) \\ &= E(n) + \left[\frac{\partial E(n)}{\partial W^{jk}(n)}\right] \Delta W^{jk}(n) + \left[\frac{\partial E(n)}{\partial W^{ij}}\right] \Delta W^{ij}(n) \\ &= E(n) - \eta^{jk} \sum_{j,k} \left(\frac{\partial E(n)}{\partial W^{jk}(n)}\right)^2 - \eta^{ij} \sum_{i,j,k} \left(\frac{\partial E(n)}{\partial W^{ij}(n)}\right)^2 \\ &= E(n) \left[1 - \frac{\eta^{jk}}{E(n)} \sum_{j,k} \left(\frac{\partial E(n)}{\partial W^{jk}(n)}\right)^2\right] \\ &\quad - E(n) \left[-\frac{\eta^{ij}}{E(n)} \sum_{i,j} \left(\frac{\partial E(n)}{\partial W^{ij}(n)}\right)^2\right] < E(n) \end{aligned} \tag{18}$$

Thus, varied learning rates (η^{ij}) and (η^{jk}) can be designed as

$$\eta^{jk} = \mu \frac{E(n)}{\sum_{j,k} \left(\frac{\partial E(n)}{\partial W^{jk}(n)}\right)^2 + \varepsilon} + \alpha \tag{19}$$

$$\eta^{ij} = \mu \frac{E(n)}{\sum_{i,j} \left(\frac{\partial E(n)}{\partial W^{ij}(n)}\right)^2 + \varepsilon} + \alpha \tag{20}$$

where ε is a small positive value to avoid dividing by zero; μ is a small positive constant to tune the learning speed; a threshold α is set to be 0.01 to keep the system on slowly learning to anticipate any disturbance. As a result, the term $\Delta E(n)$ will be less than zero, and with $E(n) > 0$, which implies $\Delta V(n) < 0$ and $V(n) > 0$. Therefore, tracking errors converge to zero as n tends to infinity.

C. TRANSLATIONAL TRACKING CONTROL

With stabilized states (z, ϕ, θ, and ψ), a translational tracking control scheme is designed to manipulate system states (x and y) via θ and ϕ. The nonlinear tracking control efforts (U_x and U_y) for system states (x and y) can be derived from the first Lyapunnov function defined in (21) and its first derivative in (22).

$$V_1 = \frac{1}{2} e_x^2 + \frac{1}{2} e_y^2 = \frac{1}{2}(x_d^r - x)^2 + \frac{1}{2}(y_d^r - y)^2 \tag{21}$$

$$\dot{V}_1 = e_x \dot{e}_x + e_y \dot{e}_y = e_x(\dot{x}_d^r - \dot{x}) + e_y(\dot{y}_d^r - \dot{y}) \tag{22}$$

By extending the dynamics as $\dot{x}^* = \dot{x}_d^r + K_{px} e_x$ and $y^* = \dot{y}_d^r + K_{py} e_y$ with positive constants (K_{px} and K_{py}), two new virtual command errors (v_x and v_y) can be defined in (23) and its first derivative can be obtained as (24).

$$\begin{aligned} v_x &= \dot{x}^* - \dot{x} = \dot{x}_d^r + K_{px} e_x - \dot{x} \\ v_y &= \dot{y}^* - \dot{y} = \dot{y}_d^r + K_{py} e_y - \dot{y} \end{aligned} \tag{23}$$

$$\begin{aligned} \dot{v}_x &= \ddot{x}^* - \ddot{x} = \ddot{x}_d^r + K_{px} \dot{e}_x - U_x \\ \dot{v}_y &= \ddot{y}^* - \ddot{y} = \ddot{y}_d^r + K_{py} \dot{e}_y - U_y \end{aligned} \tag{24}$$

The second Lyapunnov function with virtual inputs ($\dot{x}^*$ and $\dot{y}^*$) can be designed as

$$V_2 = V_1 + \frac{1}{2}v_x^2 + \frac{1}{2}v_y^2 \tag{25}$$

By taking the derivative of (25) with respect to time, one can obtain

$$\begin{aligned}\dot{V}_2 &= \dot{V}_1 + v_x\dot{v}_x + v_y\dot{v}_y \\ &= e_x(\dot{x}_d^r - \dot{x} + \dot{x}^* - \dot{x}^*) + v_x(\ddot{x}_d^r + K_{px}\dot{e}_x - U_x) \\ &\quad + e_y(\dot{y}_d^r - \dot{y} + \dot{y}^* - \dot{y}^*) + v_y(\ddot{y}_d^r + K_{py}\dot{e}_y - U_y) \\ &= e_x[\dot{x}_d^r - \dot{x} + \dot{x}^* - \dot{x}_d^r - K_{px}e_x] \\ &\quad + v_x(\ddot{x}_d^r + K_{px}\dot{e}_x - U_x) \\ &\quad + e_y[\dot{y}_d^r - \dot{y} + \dot{y}^* - \dot{y}_d^r - K_{py}e_y] \\ &\quad + v_y(\ddot{y}_d^r + K_{py}\dot{e}_y - U_y) \\ &= -K_{px}e_x^2 + e_xv_x + v_x(\ddot{x}_d^r + K_{px}\dot{e}_x - U_x) \\ &\quad - K_{py}e_y^2 + e_yv_y + v_y(\ddot{y}_d^r + K_{py}\dot{e}_y - U_y)\end{aligned} \tag{26}$$

By designing U_x and U_y in (27), it can guarantee the facts of $V_2 > 0$ and $\dot{V}_2 = -K_{px}e_x^2 - K_{py}e_y^2 - K_{dx}v_x^2 - K_{dy}v_y^2 \leq 0$. It can imply that the errors (e_x, e_y, v_x, v_y) will converge to zero as time tends to infinity.

$$\begin{aligned}U_x &= \ddot{x}_d^r + K_{px}\dot{e}_x + e_x + K_{dx}v_x \\ U_y &= \ddot{y}_d^r + K_{py}\dot{e}_y + e_y + K_{dy}v_y\end{aligned} \tag{27}$$

where K_{dx} and K_{dy} are positive coefficients.

According to (27), desired angle commands (ϕ_d and θ_d) can be calculated by

$$\begin{aligned}\phi_d &= -\frac{m}{U_1}[-\sin(\psi)\,U_x + \cos(\psi)\,U_y] \\ \theta_d &= \frac{m}{U_1\cos\phi}[\cos(\psi)\,U_x + \sin(\psi)\,U_y]\end{aligned} \tag{28}$$

It should be noted that based on the frame model in Fig. 2, the positive direction of x requires a positive rotation direction of θ, while a positive direction of y requires a negative rotation direction of ϕ. The terms (m/U_1 and $m/(U_1\cos\phi)$) in (28) are used to cancel the signal U_1 in (6).

IV. OPTIMAL PATH PLANNING

A combination of the A-star path-finding algorithm and the set-based particle swarm optimization (S-PSO) with adaptive inertia weight to solve the UAV optimal path-planning problem is presented in this section. The objective function is discussed in the first subsection. The second subsection describes the data collection process. The third subsection expresses the proposed k-agglomerative clustering procedure, and explains how to overcome the limitation of K-means clustering. Finally, the fourth subsection presents the optimal path-planning scheme via A-star and S-PSO algorithms with adaptive inertia weight.

A. OBJECTIVE FUNCTION DEFINITION

A UAV surveillance system can be represented as a graph $G(A, H)$, where A is a set of vertices to be represented in latitude, longitude, and altitude coordinates from the system, and H is a set of edges to express the connection between vertices. There are two types of vertices, which are $a^u \in A^u$ for imaging points, and $a^r \in A^r$ for non-imaging points. Imaging points are coordinates, where the UAV has to take surveillance video, and non-imaging points are coordinates, where the UAV does not take any surveillance video. The objective of the path planning is to find a tour from the take-off point with the smallest cost to visit each imaging points exactly once, and then go to landing point. The tour can be represented as permutation (o) of imaging points $\{a_1^u, a_2^u, \ldots, a_{n_u}^u\}$, where n_u is number of imaging points. The costs of edges between imaging points in the tour are the output of energy prediction model discussed in our previous work [4] for missions to move from the coordinate (a_i) to the coordinate (a_j), which will be noted as $P(a_i, a_j)$. The objective function can be defined as

$$\begin{aligned}\beta = \arg\min_{o}[&\sum_{i\in o\,<n_u} P(a_i^u, a_{i+1}^u) + \sum_{i=1}^{n_u} P(a_i^u) \\ &+ P(a^{TO}, a_1^u) + P(a_{n_u}^u, a^L)]\end{aligned} \tag{29}$$

where $P(a_i^u, a_{i+1}^u)$ is a predicted energy consumption to move through edges that connect the imaging point (a_i^u) to the imaging point (a_{i+1}^u); $P(a^{TO}, a_1^u)$ is the predicted energy consumption to move through the edge that connects the take-off point (a^{TO}) to the first imaging point (a_1^u) of the chosen sequence; $P(a_{n_u}^u, a^L)$ is the predicted energy consumption to move through the edge that connects the last imaging point ($a_{n_u}^u$) of the chosen sequence to the landing point (a^L), and $P(a_i^u)$ is the predicted consumed energy to take videos in the imaging point (a_i^u). This objective function is implemented to solve the path-planning problem of real pilot flight data and simulation data. The process is done in three consecutive steps including the data collection, the cluster analysis, and the optimal path planning. The corresponding process is depicted in Fig. 5.

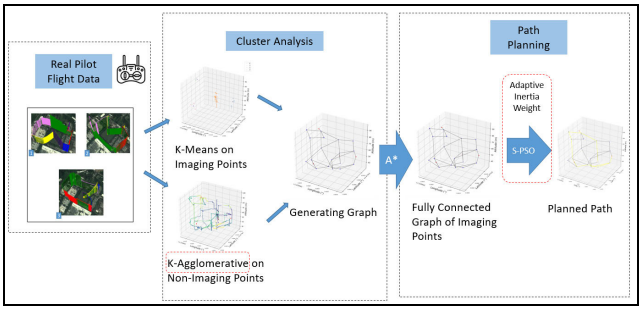

FIGURE 5. Optimal path-planning process.

B. PILOT FLIGHT DATA COLLECTION

Collection processes of the pilot flight data are done by recording longitude, latitude, altitude, and heading of several flight patterns. Mission planner in [24] and Arducopter firmware in [25] are installed in the UAV to conduct the

proposed method. Due to unavailability of camera angle data in the corresponding UAV, the camera angle is set to be fixed before the data is taken. Therefore, the representation of the camera angle can only be represented as the combination of altitude and heading. A python code to record the log of longitude, latitude, altitude, and heading values from the mission planner with the HTTP protocol (http://127.0.0.1:56781/) at every 0.5s is created. The value of $u = 1$ is automatically set when the camera is on, and the value of $u = 0$ is automatically set when the camera is off. By running this code while letting the pilot to fly the UAV for completing surveillance missions, imaging points and non-imaging points are collected at the same time. Those data with $u = 1$ are imaging points while data with $u = 0$ are non-imaging points. By including altitude in the recording data, it will open capability to solve a 3D path-planning problem, which will be more reliable in comparison with the proposed method by Kwak and Sung [14].

Imaging points are locations to take surveillance videos, and those locations are depicted in Fig. 6. Points A, B, C, and D have different longitude, latitude, altitude, and heading with the yellow circles as their survey objects. Video taking on the points (A, B, and D) are done by hovering and turning the heading and altitude to the required position and orientation to have a relatively the same video of the objects for 10 seconds. The point C has a consecutively horizontal right - vertical down - horizontal left movement while maintaining the same heading and is depicted in Fig. 7.

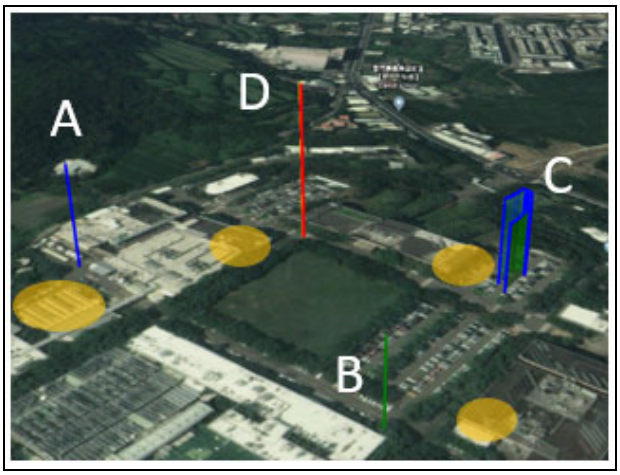

FIGURE 6. Imaging points in real pilot flight data.

Non-imaging points are locations, where the pilot flew the UAV to move from the take-off point, through all the imaging points, then go to the landing point. These points are shown in Fig. 8, and cluster centers of these points will be arcs that connect imaging points after the clustering. The pilot flew the UAV three times for three different flight patterns, where the first location point of each flight pattern data is defined as a take-off point (a^{TO}), and the last one is defined as the landing point (a^{L}).

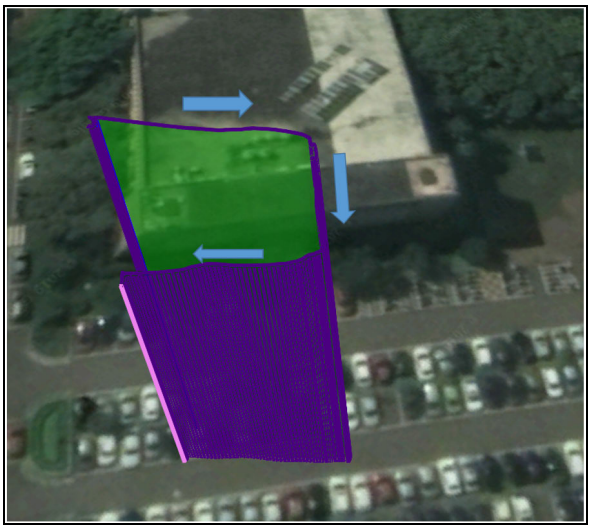

FIGURE 7. Imaging point C in real pilot flight data.

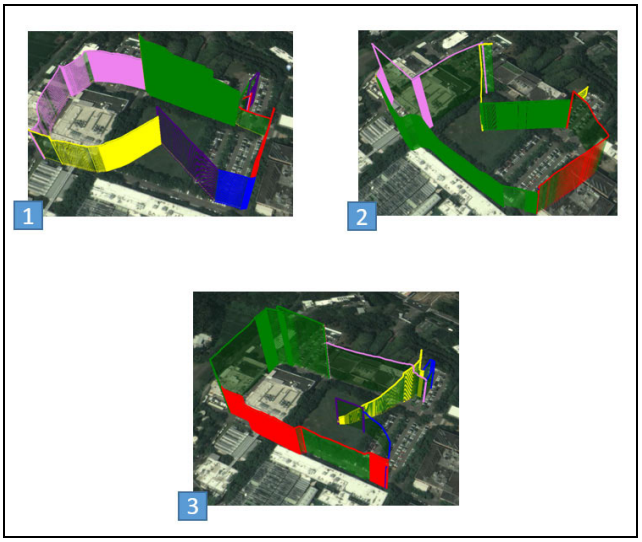

FIGURE 8. Non-imaging points in real pilot flight data.

C. K-AGGLOMERATIVE CLUSTERING

Clustering is one of unsupervised learning method to group a set of objects with more similar characteristic than the other group. In this paper, this characteristic is the distance between non-imaging points or imaging points. The purpose of this process is to decrease the number of recorded coordinates points from the data collection process. This process is necessary to simplify the path-planning problem. By this way, it will shorten the computational time of the proposed path-planning algorithm.

The collected imaging points and non-imaging points are then clustered. With different values for u, the data can be easily separated between imaging points and non-imaging points. Afterwards, each of the separated data will be clustered among themselves for all flight patterns that have been recorded in Section IV-B. Imaging points will be clustered with the K-means clustering algorithm while non-imaging

points will be clustered with the proposed K-agglomerative clustering. This is due to the necessity of non-imaging points to be clustered by an algorithm that can include a connectivity constrain, which the K-means clustering algorithm is unable. On the other hand, it is important for the clustering process of imaging points to have cluster centers as close as possible to the real data in order to have a similar image with the real data, which the K-means clustering algorithm has this ability.

The K-agglomerative clustering is proposed in this research as the combination of the K-means to define the number of cluster, and the agglomerative to do the clustering. The agglomerative clustering is one type of hierarchical clustering [26]. This method is a bottom-up approach, where the clustering process will be started from all points. Then, it will be clustered with 1 closest distance point in every iteration until there is only 1 cluster. The dendrogram illustration of the agglomerative clustering is depicted in Fig. 9. One of the problem of implementing the agglomerative as a hierarchical approach of cluster analysis is to define the number of the cluster (K).

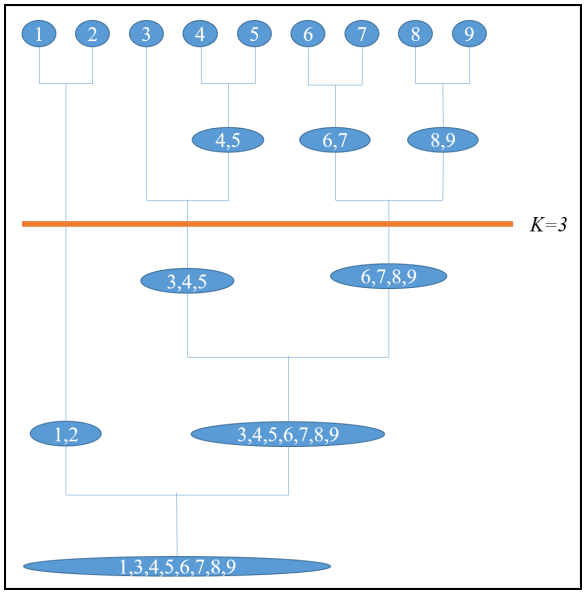

FIGURE 9. Dendrogram illustration of agglomerative clustering.

To define the number of cluster (K), a similar mechanism with the K-means is implemented. The objective function of this mechanism is to minimize the function of $\mho(n)$ in (30), which is sum of distances of normalized point coordinate $D_a|_{a=1,2,\ldots,n_a}$ to its closest centroid $\zeta_c|_{c=1,2,\ldots,K}$.

$$\mho(n) = \arg\min_{S} \sum_{c=1}^{K} \sum_{D_a \in S_c(n)} \left(\|D_a - \zeta_c(n)\|^2 \right) \tag{30}$$

$$S_i(n) = \left\{ D_a : \|D_a - \zeta_i(n)\|^2 \le \|D_a - \zeta_c\|^2 \, \forall c, \; 1 \le c \le K \right\} \tag{31}$$

The algorithm starts by randomly placing ζ_c into the coordinate space of D_a as many as K. Then, each D_a is assigned to its closest ζ_c. Moreover, $\mho$ in (30) can be calculated and evaluated. If the value of $\mho(n+1) - \mho(n)$ is not smaller than or equal to a preset constant value ($\varphi = 0.0001$), the position of the centroid (ζ_c) is updated by averaging all the D_a position, which is assigned to it. The relation between $\mho$ and K of non-imaging points can be plotted in Fig. 10.

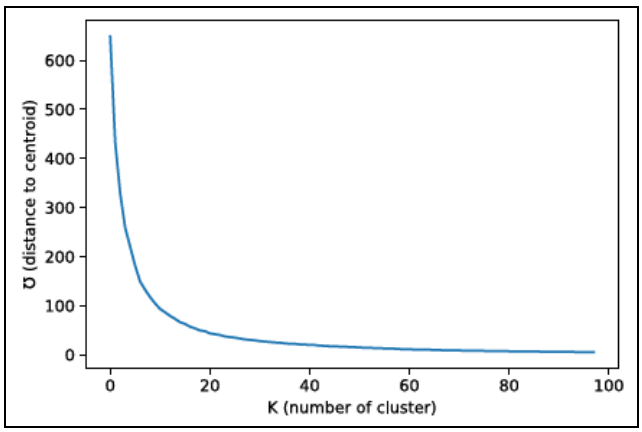

FIGURE 10. Relation between $\mho$ and K.

Figure 10 shows an elbow curve characteristic, where the *knee* can be found by implementing the kneedle algorithm by Satopaa *et al.* [27] to the relation $\mho$ of and K. Then, the value of K is used to be the number of cluster parameter to implement the agglomerative clustering.

The agglomerative clustering also has an advantage to solve this kind of data, which is connectivity constraint to be proposed by Li [28]. The connectivity constraint can guarantee that data in one cluster are data that continuously connected in the recording phase. This part is important because the pilot could have avoided an obstacle in two different flight pattern by flying roundabout the obstacle and clustering without connectivity constraint (e.g., with K-means) will cluster them into one cluster with the centroid on the obstacle itself. The result and comparison of the proposed K-agglomerative clustering and the K-means for this problem will be discussed in the Section V.

An undirected graph $G(A, H)$ can be generated from clustered imaging points and non-imaging points with cluster center of clustered imaging points by K-means as set of vertices A^u, and medians of clustered non imaging points by K-Agglomerative as set of vertices A^r. The connectivity between imaging points and non-imaging points is generated based on the recorded data index, where an edge ($h_{r,u}$) from the set of edge H between one median of clustered non-imaging points and one cluster center of imaging point is generated if the index of at least one of each cluster member is sequentially connected. The same way goes for the edge ($h_{r,r}$) between clustered non-imaging points and non-imaging points, the edge ($h_{u,r}$) between imaging-points and non-imaging points, and the edge ($h_{u,u}$) between imaging-points and imaging points. Finally, the edge (h_{to}) from the take-off point to clustered non-imaging points or to clustered

imaging points, and the edge (h_l) from clustered non-imaging points or from clustered imaging points to the landing point are generated by the same way. The existences of these edges will represent the connection between cluster centers of imaging points or non-imaging points, which longitude, latitude, and altitude information will be the inputs for the energy consumption prediction as a mission to move from one cluster center to the other cluster center.

In this step, the term $(\sum_{i=1}^{n_u} P(a_i^u))$ in (29) is also predicted using the energy consumption prediction from [4]. The energy consumptions for survey points (A, B, and D) of real pilot flight data are predicted as 10 seconds hover missions. Moreover, the required energy to cover the survey point (C) in Fig. 7, which is an imaging point that is composed of horizontal-vertical-horizontal movements while heading to one direction, is predicted by redoing the clustering to these points with four number of clusters. Then, the UAV can be set to fly through these four cluster centers while turning the camera direction to the cluster center of heading and taking video at the same time. The comparison of K-means and K-agglomerative clustering algorithms in real pilot flight data is depicted in Fig. 11. One can evaluate the objective function in (29) from these results in Fig. 11, and one can obtain the values of 0.9775 and 1.0312 for the K-means clustering algorithm and the K-agglomerative clustering algorithm, respectively. Because it would be better to have a smaller amount of $\mho(n)$, the K-means clustering algorithm is chosen to cluster imaging points.

D. A-STAR AND ADAPTIVE-WEIGHT S-PSO PATH PLANNING

An A-star algorithm, which is an arc routing algorithm to find a shortest path, is implemented to simplify the graph G to be a node routing problem with one vehicle (i.e., the travelling salesman problem). This method is implemented by running an A-star algorithm to find the shortest path to move from one $a^u \in A^u$ to all member of A^u through $a^r \in A^r$. The A-star algorithm is also used to find shortest paths to move from take-off vertices a^{TO} to all $a^u \in A^u$ through $a^r \in A^r$ and to find shortest paths to move from all $a^u \in A^u$ to landing vertices a^L through $a^r \in A^r$. Then, a new fully connected graph $G^f(\{a^u, a^{TO}, a^L\}, H^f)$ is generated as the result of the A-star algorithm, where H^f is the set of edges that is generated by A-star algorithm as shortest paths through edges $h_{r,r}$ and vertices $a^r \in A^r$. This fully connected graph then will be solved by the S-PSO algorithm.

A PSO algorithm is a meta-heuristic optimization algorithm that inspired by social behavior of bird flocking or fish schooling. The term particle in this algorithm is used to visualize the individual optimization of each individual bird during the flocking while maintaining communication with the best optimized value the group can have. In general, a PSO algorithm can be expressed in (32) and (33). The position of particle (i) represented in (32) is a sequence combination of imaging points (o_i) to minimize predicted energy consumption (β_i) from (29), where λ is the iteration

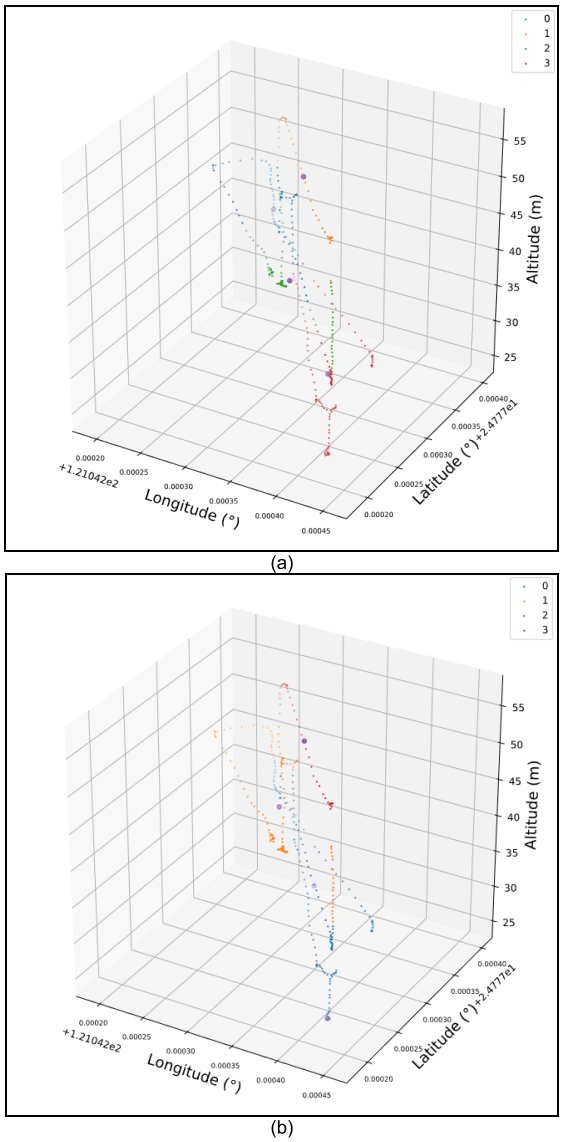

FIGURE 11. Clustering of moving survey points C in real pilot flight data: (a) K-means clustering result; (b) K-agglomerative clustering result.

number. The velocity update for particle i ($v_i(\lambda)$) is expressed in (33), where $\omega_i(\lambda)$ is the inertia weight, c_1 is the exploitation constant, c_2 is the exploration constant, o_i^* is the best position of particle (i), o^* is the global best position of the group, and two random operator (r_1 and r_2) with values between 0 to 1.

$$o_i(\lambda+1) = o_i(\lambda) + v_i(\lambda) \tag{32}$$

$$v_i(\lambda+1) = \omega_i(\lambda)v_i(\lambda) + c_1\, r_1\, (o_i^* - o_i) + c_2\, r_2\, (o^* - o_i) \tag{33}$$

The set-based PSO (S-PSO) was first investigated by Chen *et al.* [9]. The purpose of this method is to tackle the ineffectiveness of PSO to solve a discrete problem. Thus, this method can be explored as a method to bring any upgrade in the PSO from a continuous problem to a discrete problem.

The difference that is offered by the S-PSO is the representation of particle's velocity to be a set with possibilities given

by

$$v_i = \{h^f / p(h^f) | h^f \in H^f\} \tag{34}$$

where each element $h^f \in H^f$ has a possibility $p(h^f) \in [0, 1]$ in v_i. Using this definition, the fully connected graph (G^f), as the output of the A-star algorithm, can be represented as the particle's velocity.

In this paper, a method to define an adaptive inertia weight of the S-PSO for particle (i) is proposed, implemented, and compared to other method, and is explained in (35) and (36).

$$\omega_i(\lambda) = \omega_i^0 - (\omega_i^0 - \omega_i^{\lambda_n}) \times 2[\sigma(-\frac{\lambda}{0.2\lambda_n}) - 0.5] + \delta \tag{35}$$

$$\delta = \begin{cases} 0.1 \times rand, & if\ o_i(\lambda) = o_i^* \\ -0.1 \times rand, & if\ o_i(\lambda) \neq o_i^* \end{cases} \tag{36}$$

where ω_i^0 is the initial inertia weight, $\omega_i^{\lambda_n}$ is the desired final inertia weight value in the last iteration number, σ is a sigmoid function, λ is the iteration number, and λ_n is the total iteration instant. In (36), the value of δ is defined as the function of the random operator, which is a random float number between 0 and 1. In the proposed adaptive inertia weight, a positive value of δ will be given if the position of the particle at iteration λ ($o_i(\lambda)$) is its best position (o_i^*). Otherwise, it will be negative if $o_i(\lambda)$ is not equal to o_i^*. When the values of $\omega_i^0 = 0.9$. and $\omega_i^{\lambda_n} = 0.4$ are selected, the plot of $\omega(\lambda)$ is depicted in Fig. 12.

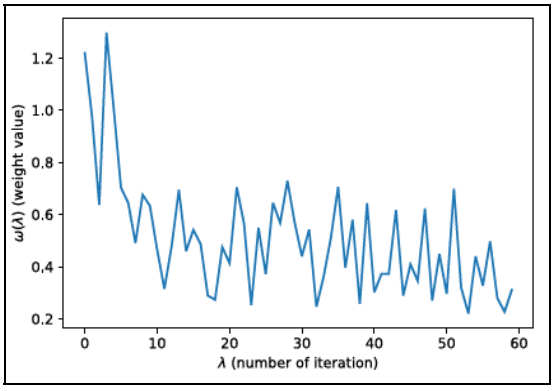

FIGURE 12. Inertia weight $\omega(\lambda)$.

V. RESULT DISCUSSION

A. CONTROL SYSTEM DESIGN

The detailed parameters of the APUPS can not be obtained from the manufacturer. Therefore, parameters in Moussid *et al.* [18] are used in this paper as numerical simulations. The hexarotor parameters are summarized in Table 1.

By using the parameters in Table 1, numerical simulations of proportional-integral-differential (PID) control in [18], fuzzy control in [19], and the proposed ANN control are created in MATLAB software with the sampling time ($t_s = 0.001\,s$) and the occurrence of a disturbance ($-30\,N$) at $35s < t < 45s$ for U_1. The parameters of PID, fuzzy, and

TABLE 1. Hexarotor parameters.

Notation	Description	Value
I_{xx}	Moment of inertia about body fram's x-axis	7.5×10^{-3} kg m^2
I_{yy}	Moment of inertia about body fram's y-axis	7.5×10^{-3} kg m^2
I_{zz}	Moment of inertia about body fram's z-axis	1.3×10^{-2} kg m^2
b	Thrust constant	3.13×10^{-5} N s^2
d	Drag constant	7.5×10^{-7} Nms
I_r	Inertia of each rotor	7.5×10^{-7} kg m^2
m	Mass	0.65 kg
l	Distance to center of gravity	0.23 m
g	Gravity constant	9.8 m/s^2

ANN controllers are provided as follows:

$$\begin{aligned}
&\rho_1 = 6, \quad \rho_2 = 0.09, \\
&K_{px} = K_{py} = 0.41, \quad K_{dx} = K_{dy} = 0.15, K_{ix} = K_{iy} = 0, \\
&K_{pz} = 15, \quad K_{dz} = 5, K_{iz} = 10, K_{p\phi} = K_{p\theta} = 200, \\
&K_{d\phi} = K_{d\theta} = 6.2, \quad K_{i\phi} = K_{i\theta} = 0.048, K_{p\psi} = 70, \\
&K_{d\psi} = 4, \quad K_{i\psi} = 5, \\
&K_{ez} = 10, \quad K_{\dot{e}z} = 5, K_{e\phi} = K_{e\theta} = K_{e\psi} = 2, \\
&K_{\dot{e}\phi} = K_{\dot{e}\theta} = K_{\dot{e}\psi} = 1, \quad K_{Uz} = 3.9, \\
&K_{U\phi} = K_{U\theta} = K_{U\psi} = 1 \\
&\mu_z = 0.1, \quad \mu_{\phi\theta\psi} = 1, \varepsilon_z = \varepsilon_{\phi\theta\psi} = 1, \\
&\alpha_z = \alpha_{\phi\theta\psi} = 0.01
\end{aligned} \tag{37}$$

where ρ_1 and ρ_2 are time constant of reference model in outer loop and in inner loop, respectively; K_{px}, K_{py}, K_{dx}, and K_{dy} are proportional and derivative gains for nonlinear tracking control of systems states (x and y) that are constructed according to (27), tuned and chosen to get the best transient control performance in numerical simulations while maintaining the stability, and implemented to PID, fuzzy, and ANN; K_{pz}, $K_{p\phi}$, $K_{p\theta}$, $K_{p\psi}$, K_{dz}, $K_{d\phi}$, $K_{d\theta}$, $K_{d\psi}$, K_{iz}, $K_{i\phi}$, $K_{i\theta}$, and $K_{i\psi}$ are PID controller parameters for system states (z, ϕ, θ, and ψ) that are constructed according to control framework in [18] and are chosen to have the best transient performances that match aforementioned hexacopter parameters in Table 1 while ensuring the stability. A fuzzy control framework proposed in [19] is designed to only control systems states (z, ϕ, θ, and ψ). Therefore, by implementing (27), the other parameters to be tuned are K_{ez}, $K_{e\phi}$, $K_{e\theta}$, and $K_{e\psi}$, which are error gains and are tuned to adjust the steady-state response. $K_{\dot{e}z}$, $K_{\dot{e}\phi}$, $K_{\dot{e}\theta}$, and $K_{\dot{e}\psi}$ are the first-derivative error gains and are tuned to adjust the damping characteristic of the transient; K_{Uz}, $K_{U\phi}$, $K_{U\theta}$, and $K_{U\psi}$ are control output gains and are also tuned to adjust the steady state and the stability of system states (z, ϕ, θ, and ψ). The proposed ANN control framework is implemented according to Figs. 3 and 4. Moreover, the tuned parameters from (19) and (20) are μ_z, ε_z, and α_z for the ANN of the system state (z), and $\mu_{\phi\theta\psi}$,

$\varepsilon_{\phi\,\theta\,\psi}$, and $\alpha_{\phi\,\theta\,\psi}$ for the ANN of system states (ϕ, θ, and ψ). The value of μ_z are tuned to be slower than the inner loop $\mu_{\phi\,\theta\,\psi}$ to have the outer loop at least ten times slower learning rates in comparison with the inner loop. The values of ε_z and $\varepsilon_{\phi\,\theta\,\psi}$ are chosen to prevent dividing by zero in (19) and (20) if the partial derivative in the denominator is equal to zero. The values of α_z and $\alpha_{\phi\,\theta\,\psi}$ are selected to have the ANN to be constantly learning to anticipate any disturbance.

To examine and compare the control performance, the following root-mean-square-error (RMSE) values of states tracking responses are defined:

$$RMSE(q_j) = \frac{1}{T}\sqrt{\sum_{n=1}^{T} e_j^2(n)} \tag{38}$$

where q_j and e_j indicate the elements of the system state vector $\boldsymbol{q} = [x, y, z, \phi, \theta, \psi]^T$ and the corresponding error state vector $\boldsymbol{e} = [e_x, e_y, e_z, e_\phi, e_\theta, e_\psi]^T$; T is the total sampling instant; n is the iteration number.

Numerical simulations of the PID control in [18], the fuzzy control in [19], and the proposed ANN control are depicted in Figs. 13-16, respectively. The translational movements in 3D illustration of PID, fuzzy, and ANN controllers are depicted in Fig. 13. Figure 14 clearly shows that the given disturbance affected the system controlled by the PID control to have a 3.5m error when the disturbance is given, and a -3.488m error when the disturbance is removed for z-axis responses. Figure 14 shows that with a fine parameter tuning, the fuzzy control can significantly minimize errors from the given disturbance to be 0.831m and -0.0360m. However, it can also clearly be seen that the steady state errors of all the states are slightly bigger, which will give a bad impact to the RMSE values. Figure 16 shows that the ANN controller can minimize errors from the given disturbance to be 1.399m and -0.3060m, which is slightly worse than the fuzzy control. But, the proposed ANN controller has a better steady-state

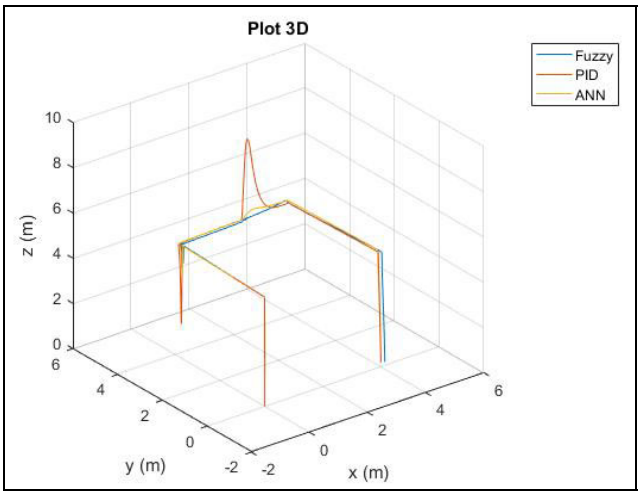

FIGURE 13. Translational movement in 3D of PID, fuzzy, and ANN controllers.

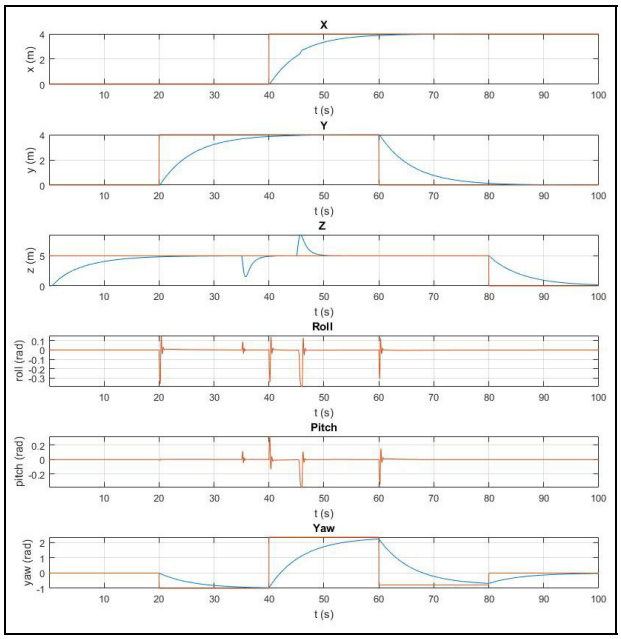

FIGURE 14. Response of PID controller.

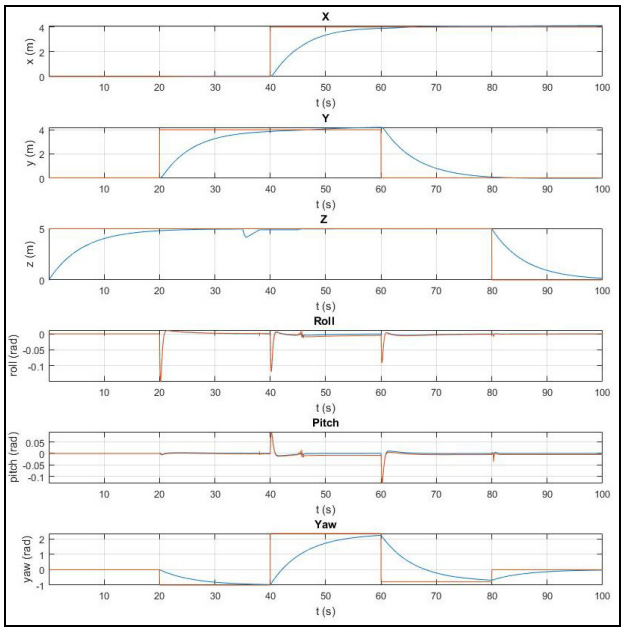

FIGURE 15. Response of fuzzy controller.

response. Moreover, varied learning rates of the ANN to be updated according to (19) and (20) for this scenario are depicted in Fig. 17. It is obvious that the disturbance affects the learning rate for the system state of z at 35s and 45s.

The performance comparisons of the PID control in [18], the fuzzy control in [19], and the proposed ANN control are summarized in Table 2. As can be seen from Table 2, the proposed ANN control can achieve 49.083% and 30.433% improvement of average horizontal and vertical tracking performance, and 44.44% and 37.50% fewer number of parameters to be tuned in comparison with PID and fuzzy controllers,

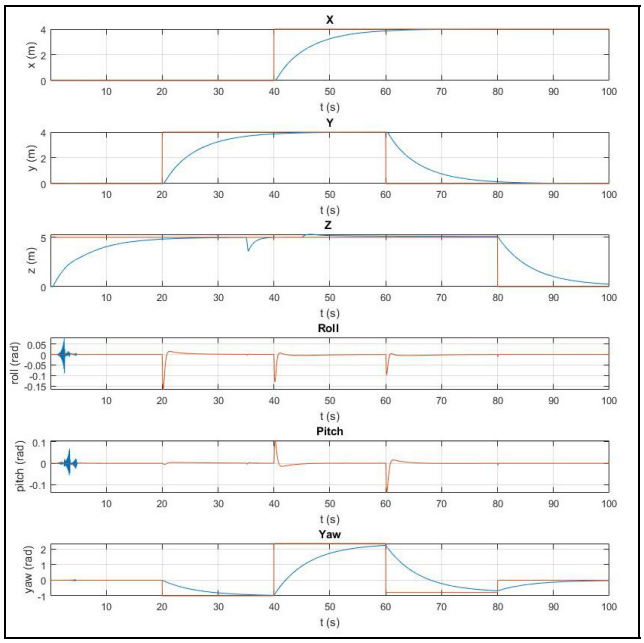

FIGURE 16. Response of ANN controller.

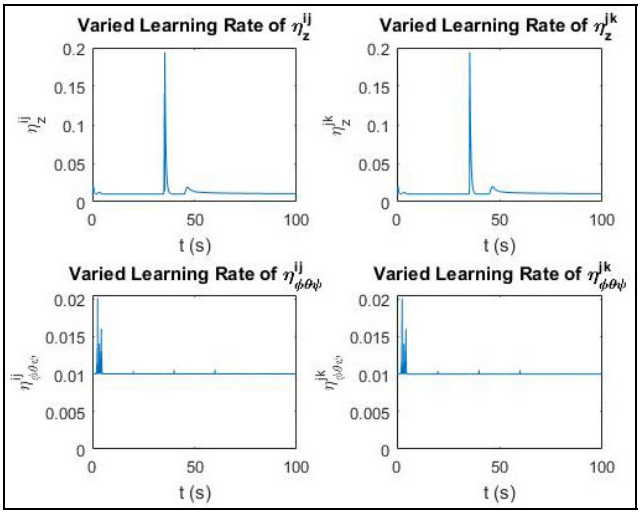

FIGURE 17. Varied learning rate.

respectively. Although non-pretrained results of the ANN control in Fig. 16 only achieve slightly better average angle control results compared to fuzzy controller, which is 1.33%. A pre-training process is believed to be able to tackle this problem. Moreover, the proposed ANN control is the only method with learning ability and Lyapunov stability analysis in comparison with PID and fuzzy controllers.

Advantages of the proposed ANN controller with varied learning rates in comparison with conventional methods are summarized as follows. 1) The disturbance anticipation has a faster response with smaller errors. 2) The ANN has more robust control structure with adaptive and online learning capability. 3) It also has better steady-state responses in comparison with conventional methods. 4) It has fewer parameter to be tuned with guaranteed stability, which means a better usability in comparison with conventional methods.

TABLE 2. Performance comparisons of PID, FUZZY, and ANN controllers.

	PID control in [18]	Fuzzy control in [19]	The proposed ANN
RMSE(x)	0.0282m	0.0697m	0.0176m
RMSE(y)	0.0410m	0.0947m	0.0251m
RMSE(z)	0.5141m	0.0954m	0.1497m
RMSE(ϕ)	0.0001rad	0.0022rad	0.0031rad
RMSE(θ)	0.0001rad	0.0046rad	0.0027rad
RMSE(ψ)	0.0001rad	0.0029rad	0.0029rad
Parameters number to be tuned	18	16	**10**
Lyapunnov stability analysis	Not included	Not included	**Included**
Learning ability	None	None	**Online learning**

B. OPTIMAL PATH PLANNING

To evaluate the path-planning performance, the method is implemented via numerical simulations and real pilot flight data. The parameters for the S-PSO in (35) are given as follows:

$$\lambda_{n1} = 60, \quad \lambda_{n2} = 20 \tag{39}$$

where λ_{n1} is the number of iteration for the simulation data, and λ_{n2} is the number of iteration for the implementation data. Moreover, the numbers of particles are 15 and 3 for simulation and implementation data, respectively.

1) SIMULATION

The simulation is conducted via the software in the loop simulator by ArduPilot [29]. This program let the user move the UAV in the mission planner with joystick as a simulation and retrieve the GPS data. Thus, the logger code in python can be used to log the data from the HTTP connection to the mission planner.

Imaging points and the flight pattern of simulation data are depicted in Figs. 18 and 19, respectively. The number of imaging points in the simulation data are six, which includes the horizontal-vertical-horizontal imaging movement in the survey point E. The detailed imaging points are depicted in Fig. 20.

The comparison of clustering results via the K-means and the proposed K-agglomerative clustering is depicted in Fig. 21. The blue arrows in the K-means clustering results are pointing to centroids that are not from the data. This is the disadvantage of the traditional K-means clustering. The result of the proposed K-agglomerative clustering is a graph that will be the input to the optimal path planning process, and is depicted in Fig. 22.

The shortest paths are then calculated with the A-star algorithm, which will make the system become a fully connected graph. Afterward, the fully connected graph will be the input

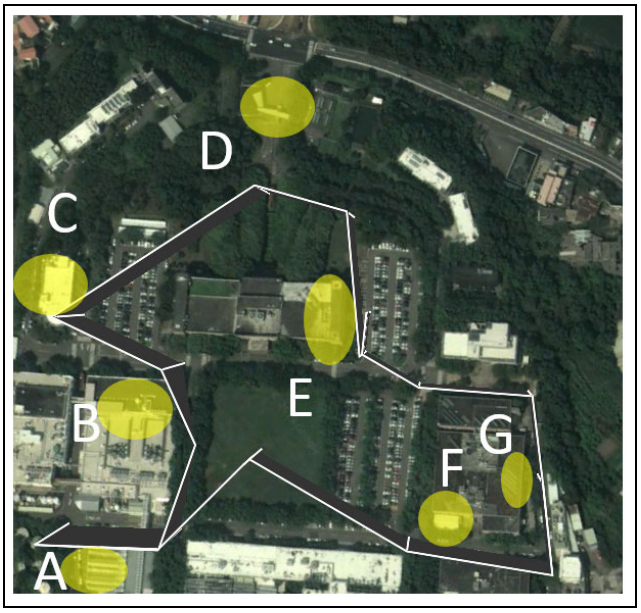

FIGURE 18. Imaging points of simulation data.

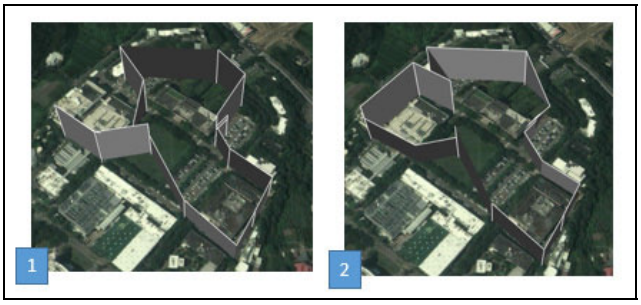

FIGURE 19. Flight patterns of simulation data.

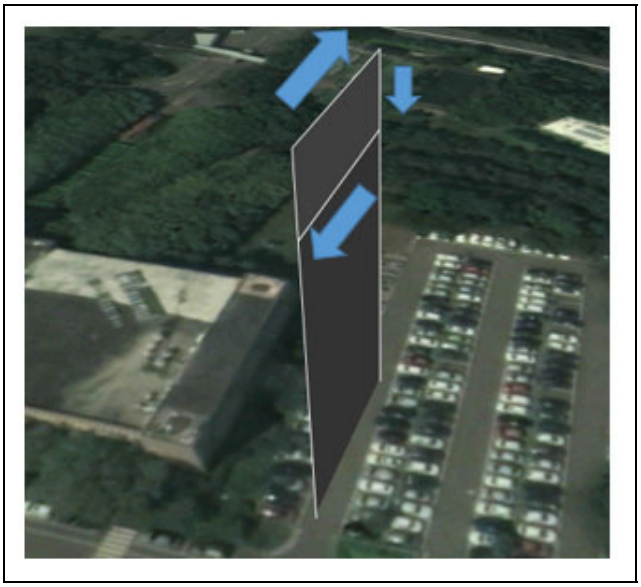

FIGURE 20. Imaging point E of simulation data.

for the path planning. Moreover, the term $(\sum_{i=1}^{n_u} P(a_i^u))$ of the simulation data is predicted by inputting six hovering missions of survey points (A, B, C, D, F, and G) of simulation data

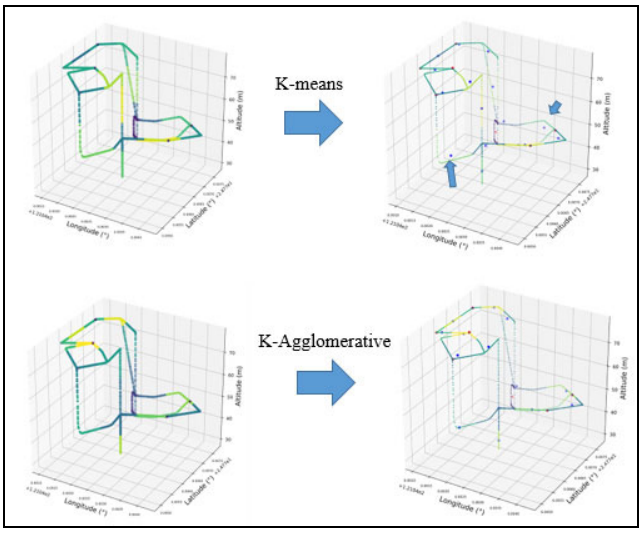

FIGURE 21. Performance comparison of K-means and K-agglomerative clustering on simulation data.

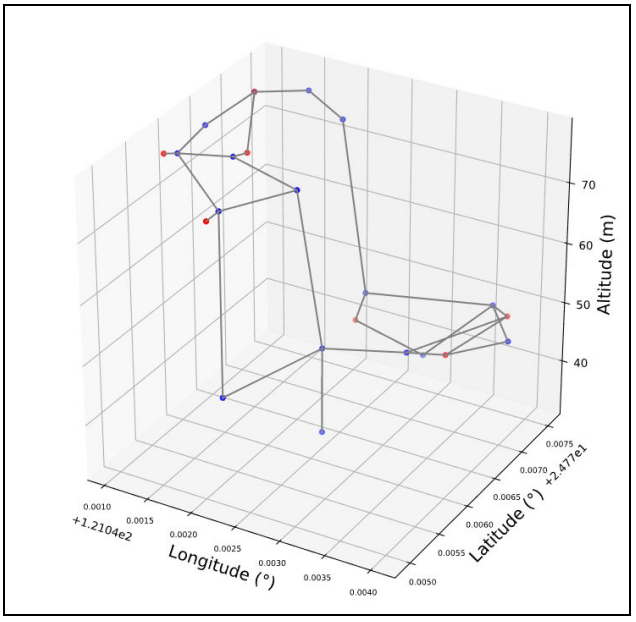

FIGURE 22. Graph from K-agglomerative of simulation data.

and inputting one moving mission for survey points (E). The moving mission of survey points (E) is depicted in Fig. 23. The total energy requirement for taking videos in these imaging points of simulation data is 16.585 Wh, which will be used to calculate the objective function in (29).

The performance of the proposed path-planning method is compared with the ones of the PSO-Swap operator in [8], the S-PSO in [9], the S-comprehensive learning PSO (S-CLPSO) in [10], and the S-PSO with chaotic inertia weight in [30]. The inertia weights for previous methods in [8]–[10], [30] are exactly the same as the lists in the references. These methods are reconstructed for this comparison in Python code. All of these five methods are used in five times with the aforementioned parameters, and the comparison of the objective function in (29) are summarized

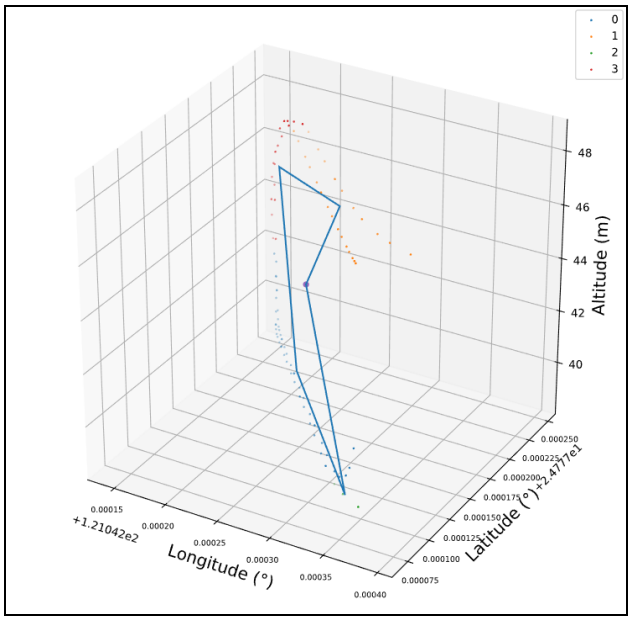

FIGURE 23. Moving missions for survey points E of simulation data.

TABLE 3. Comparison of path planning on real pilot flight data.

Algorithm	Best (Wh)	Worst (Wh)	Mean Energy (Wh)	Mean Duration (s)
PSO–Swap in [8]	78.6414	89.0007	85.0028	0.0173
S-PSO in [9]	78.6414	89.6598	82.884	5.0583
S-CLPSO in [10]	78.9814	89.8656	85.8609	3.2147
S-PSO with chaotic inertia weight in [30]	78.9814	85.0715	82.2112	5.8839
S-PSO with adaptive weight	78.6414	85.0715	80.153	5.4970

in Table 3. The results clearly show that the proposed S-PSO with adaptive inertia weights can plan a pattern with the lowest mean predicted energy with 5.23% lower average in comparison with other methods in [8]–[10], [30].

2) REAL PILOT FLIGHT DATA

The average energy consumed for those patterns in Figs. 6-8 is 96.593Wh. The experiments are done in ITRI headquarter, Hsinchu, Taiwan. The performances of the K-means and the proposed K-agglomerative clustering of the recorded data are depicted in Fig. 24. The problem in the K-means clustering can be solved by using the proposed K-agglomerative clustering to ensure the connectivity before defining a cluster.

The term $\sum_{i=1}^{n_u} P(a_i^u)$ of the real pilot flight data is predicted by inputting three hovering missions of survey points (A, B, and D) of real pilot flight data, and inputting one moving mission of survey points (C), which cluster is depicted in Fig. 11. The moving missions of survey points (C) are depicted in Fig. 26. The total predicted required energy for taking videos in these imaging points of real pilot flight data is 19.70 Wh. This value is higher than the one in simulation data because the real pilot flight data has a higher altitude

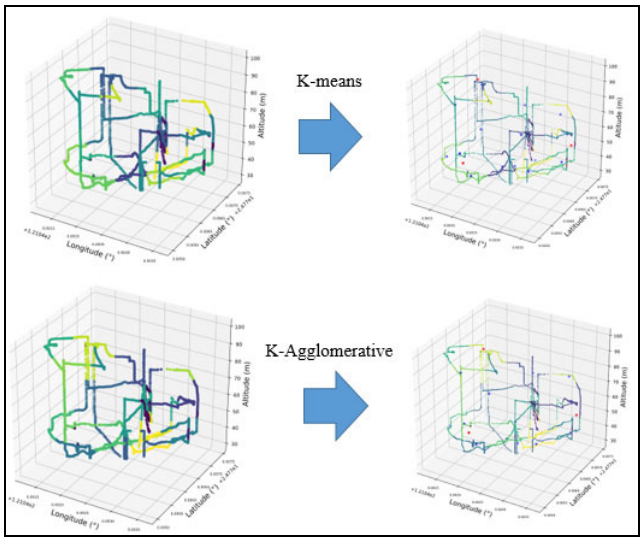

FIGURE 24. Performance comparison of K-means and K-agglomerative on real pilot flight data.

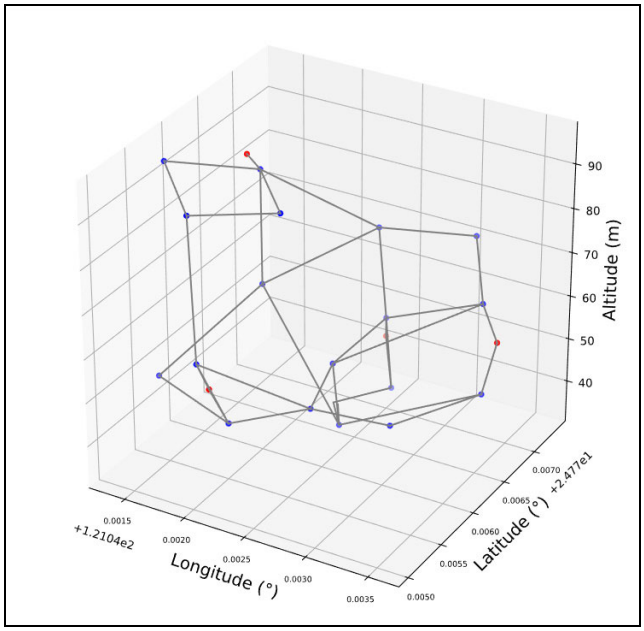

FIGURE 25. Graph from K-agglomerative of real pilot flight data.

than the one in the simulation data. This means that it requires more vertical movement, which will consume more energy than the horizontal movement. Later, this value will be used to calculate the objective function in (29).

By doing the same comparison in Table 3, the performance of the proposed S-PSO with adaptive inertia weight via real pilot flight data in comparison with other methods in [8]–[10], [30] is summarized in Table 4 with the aforementioned parameters and five times run. These methods are reconstructed for this comparison in Python code. The proposed S-PSO with adaptive inertia weight can plan an optimal path with 2.75% mean predicted energy in average compare to other methods in [8]–[10], [30] and is the third fastest

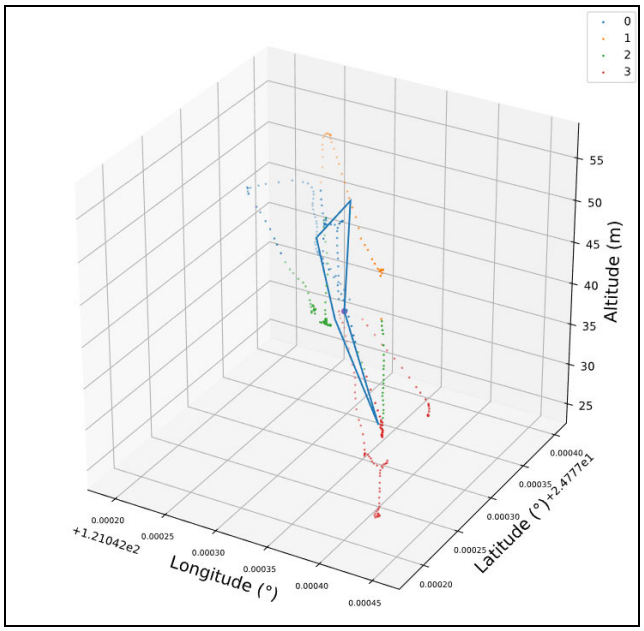

FIGURE 26. Moving missions of survey points C of real pilot flight data.

TABLE 4. Comparison of path planning on real pilot flight data.

Algorithm	Best (Wh)	Worst (Wh)	Mean Energy (Wh)	Mean Duration (s)
PSO–Swap in [8]	76.5024	76.7728	76.348	0.006
S-PSO in [9]	72.3339	76.0502	73.1762	0.1755
S-CLPSO in [10]	72.3339	74.6485	73.6329	0.1183
S-PSO with chaotic inertia weight in [30]	72.3482	74.1852	73.083	0.1948
S-PSO with adaptive weight	72.3339	72.3482	72.3397	0.1707

mean calculation duration. Although the mean duration of simulation and real pilot flight data by the proposed strategy are high, the proposed method shows consistency on giving a better objective value.

The advantages of the proposed S-PSO with adaptive inertia weight compared to the PSO-Swap operator in [8], the S-PSO in [9], and the S-CLPSO in [10] with linearly decreased inertia weight; and S-PSO with chaotic inertia weight in [30] are summarized as follow. 1) The proposed method increases the efficiency of the S-PSO by slightly decrease the mean duration in comparison with the S-PSO with linearly decreased inertia weight. 2) The proposed adaptive inertia weight has capability to give a response to a wrong position of the particle by decreasing the inertia weight (increase global search) for the next iteration and do the otherwise when the position is the best position. 3) The accuracy of the prediction for the aforementioned scenario is proved to be more consistently be able to choose the path with the lowest energy consumption prediction.

3) PLANNED PATH CONTROL RESPONSE

The planned path then is used to do the final test, which are control responses for the path following process.

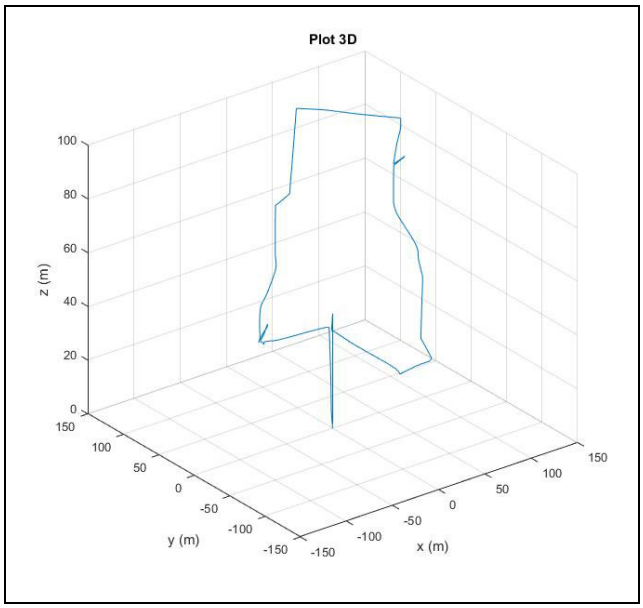

FIGURE 27. Result of ANN controller for planned path.

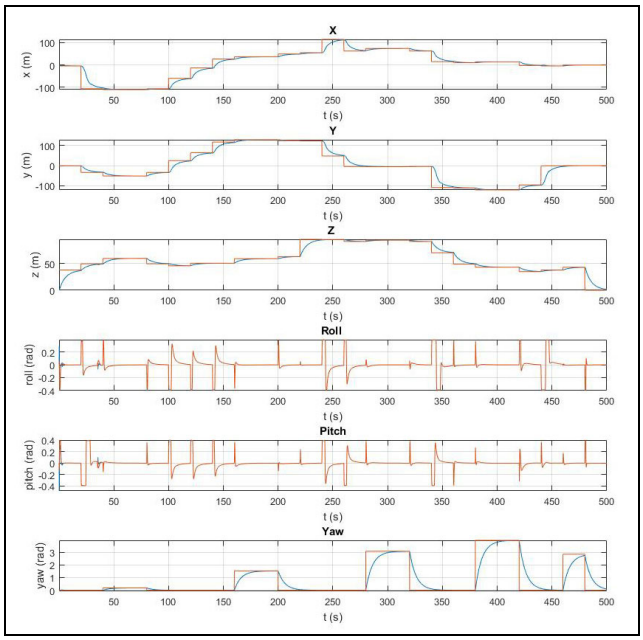

FIGURE 28. Responses of ANN controller for planned path.

A disturbance $(-30\,N)$ at $35s < t < 45s$ for U_1 is also given to examined the performance of the disturbance rejection control of the proposed ANN controller. The results are depicted in Figs. 27 and 28, and the corresponding RMSE values are 2.1578m, 1.8678m, 0.0458m, 0.0020rad, 0.0024rad, and 0.0002rad for system states (x, y, z, ϕ, θ, and ψ) with respect to individual reference commands (x_d^r, y_d^r, z_d^r, ϕ_d^r, θ_d^r and ψ_d^r), respectively. The total predicted power consumption of the planned path is 72.3339 Wh. This means that the UAV will consume 24.2591Wh less energy compared to the average of the pilot flight data.

Comparisons of imaging points from real pilot flight data and imaging points from the planned path are depicted in Fig. 29. It can clearly be seen that the image on cluster centers are similar to the real pilot flight data.

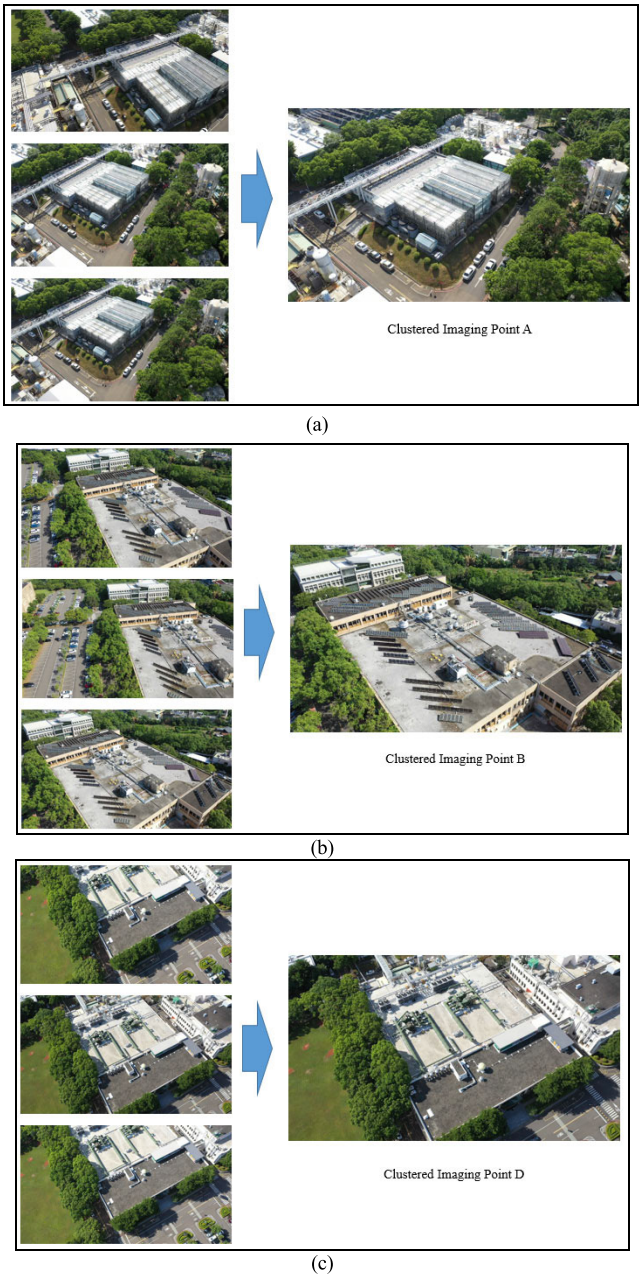

FIGURE 29. Imaging points comparisons: (a) Comparisons of imaging point A of real pilot data and its cluster center; (b) Comparisons of imaging point B of real pilot data and its cluster center; (c) Comparisons of imaging point D of real pilot data and its cluster center.

VI. CONCLUSION

This paper has been successfully designed a complete UAV surveillance system, and discussed from the low-level controller to the optimal path-planning solution. According to the root-mean-square-error (RMSE) comparisons, the proposed adaptive neural network (ANN) controller can achieve 49.083% and 30.433% improvement of average horizontal and vertical tracking performance, and 44.44% and 37.50% fewer number of parameters to be tuned in comparison with proportional-integral-differential (PID) and fuzzy controllers, respectively. Moreover, the learning behaviour of the proposed ANN control can make the control structure easier to be used for other UAVs with different parameters without tuning control parameters as many as what it is in PID or fuzzy controller with better disturbance anticipation. In addition, the combination of the proposed K-agglomerative clustering, the set-based particle-swarm-optimization (S-PSO) with adaptive weight, and the A-star algorithm can plan a path with predicted energy 21.28Wh less then actual energy consumed by pilot flight. The main contributions of this study includes an ANN framework for the disturbance rejection control, a mission-based energy consumption prediction to give an insight of the flight duration limitation, and an optimal path planning with K-agglomerative clustering and adaptive inertia-weight S-PSO to plan an energy efficient path.

REFERENCES

[1] Z. Liu, R. Sengupta, and A. Kurzhanskiy, "A power consumption model for multi-rotor small unmanned aircraft systems," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Miami, FL, USA, Jun. 2017, pp. 310–315.

[2] A. Abdilla, A. Richards, and S. Burrow, "Power and endurance modelling of battery-powered rotorcraft," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep./Oct. 2015, pp. 675–680.

[3] N. Bezzo, K. Mohta, C. Nowzari, I. Lee, V. Kumar, and G. Pappas, "Online planning for energy-efficient and disturbance-aware UAV operations," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 5027–5033.

[4] A. S. Prasetia, R.-J. Wai, Y.-L. Wen, and Y.-K. Wang, "Mission-based energy consumption prediction of multirotor UAV," *IEEE Access*, vol. 7, pp. 33055–33063, 2019.

[5] M. Dorigo and M. Birattari, *Ant Colony Optimization*. Boston, MA, USA: Springer, 2010.

[6] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook. (Mar. 1, 2015). *Concorde TSP Solver. Traveling Salesman Problem*. [Online]. Available: https://www.math.uwaterloo.ca/tsp/concorde

[7] S. S. Juneja, P. Saraswat, K. Singh, J. Sharma, R. Majumdar, and S. Chowdhary, "Travelling salesman problem optimization using genetic algorithm," in *Proc. Amity Int. Conf. Artif. Intell. (AICAI)*, Feb. 2019, pp. 264–268.

[8] S. K. Hadia, A. H. Joshi, C. K. Patel, and Y. P. Kosta, "Solving city routing issue with particle swarm optimization," *Int. J. Comput. Appl.*, vol. 47, no. 15, pp. 27–30, 2012.

[9] W.-N. Chen, J. Zhang, H. S. H. Chung, W.-L. Zhong, W.-G. Wu, and Y.-H. Shi, "A novel set-based particle swarm optimization method for discrete optimization problems," *IEEE Trans. Evol. Comput.*, vol. 14, no. 2, pp. 278–300, Apr. 2010.

[10] Y. Weng, W.-N. Chen, A. Song, and J. Zhang, "Set-based comprehensive learning particle swarm optimization for virtual machine placement problem," in *Proc. 9th Int. Conf. Intell. Control Inf. Process. (ICICIP)*, Nov. 2018, pp. 243–250.

[11] L. Amorosi, L. Chiaraviglio, and J. Galán-Jiménez, "Optimal energy management of UAV-based cellular networks powered by solar panels and batteries: Formulation and solutions," *IEEE Access*, vol. 7, pp. 53698–53717, 2019.

[12] C.-W. Lim, S. Park, C.-K. Ryoo, K. Choi, and J.-H. Cho, "A path planning algorithm for surveillance UAVs with timing mission constrains," in *Proc. Int. Conf. Control, Autom., Syst. (ICCAS)*, Oct. 2010, pp. 2371–2375.

[13] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems for drone delivery," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 1, pp. 70–85, Jan. 2017.

[14] J. Kwak and Y. Sung, "Autonomous UAV flight control for GPS-based navigation," *IEEE Access*, vol. 6, pp. 37947–37955, 2018.

[15] T. Jiang, D. Lin, and T. Song, "Finite-time backstepping control for quadrotors with disturbances and input constraints," *IEEE Access*, vol. 6, pp. 62037–62049, 2018.
[16] S. Busarakum and V. Srichatrapimuk, "The design of sliding mode control of a hexarotor," in *Proc. IEEE Conf. Syst., Process Control (ICSPC)*, Dec. 2014, pp. 47–52.
[17] M. Walid, N. Slaheddine, A. Mohamed, and B. Lamjed, "Modeling and control of a quadrotor UAV," in *Proc. 15th Int. Conf. Sci. Techn. Autom. Control Comput. Eng. (STA)*, Dec. 2014, pp. 343–348.
[18] M. Moussid, A. Sayouti, and H. Medromi, "Dynamic modeling and control of a hexarotor using linear and nonlinear methods," *Int. J. Appl. Inf. Syst.*, vol. 9, no. 5, pp. 9–17, 2015.
[19] F. Fakurian, M. B. Menhaj, and A. Mohammadi, "Design of a fuzzy controller by minimum controlling inputs for a quadrotor," in *Proc. 2nd RSI/ISM Int. Conf. Robot. Mechatronics (ICRoM)*, Oct. 2014, pp. 619–624.
[20] F.-J. Lin, W.-J. Hwang, and R.-J. Wai, "Ultrasonic motor servo-drive with online trained neural-network model-following controller," *IEE Proc.-Electr. Power Appl.*, vol. 145, no. 2, pp. 105–110, Mar. 1998.
[21] F.-J. Lin and R.-J. Wai, "Hybrid controller using a neural network for a PM synchronous servo-motor drive," *IEE Proc.-Electr. Power Appl.*, vol. 145, no. 3, pp. 223–230, May 1998.
[22] R. J. Wai, M. W. Chen, and Y. K. Liu, "Design of adaptive control and fuzzy neural network control for single-stage boost inverter," *IEEE Trans. Ind. Electron.*, vol. 62, no. 9, pp. 5434–5445, Sep. 2015.
[23] (2018). *RD 100 Conference, Award Winners Finalists*. [Online]. Available: https://www.rd100conference.com/ awards/ winners-finalists/year/2018/
[24] ArduPilot Dev Team. (2016). *Mission Planner, Mission Planner Home*. [Online]. Available: http://ardupilot.org/planner/index.html
[25] ArduPilot Dev Team. (2016). *Ardu Copter, Copter Home*. [Online]. Available: http://ardupilot.org/copter/
[26] L. Rokach and O. Maimon, "Clustering methods," in *Data Mining and Knowledge Discovery Handbook*. Boston, MA, USA: Springer, 2005, pp. 321–352.
[27] V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan, "Finding a 'Kneedle' in a haystack: Detecting knee points in system behavior," in *Proc. 31st Int. Conf. Distrib. Comput. Syst. Workshops*, Jun. 2011, pp. 166–171.
[28] J. Li, "Agglomerative connectivity constrained clustering for image segmentation," *Stat. Anal. Data Mining, ASA Data Sci. J.*, vol. 4, no. 1, pp. 84–99, 2011.
[29] ArduPilot Dev Team. (2019). *SITL Simulator (Software in the Loop)*. Accessed: Jun. 18, 2019. [Online]. Available: http://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html
[30] J. C. Bansal, P. K. Singh, M. Saraswat, A. Verma, S. S. Jadon, and A. Abraham, "Inertia weight strategies in particle swarm optimization," in *Proc. IEEE 3rd World Congr. Nature Biol. Inspired Comput.*, Oct. 2011, pp. 633–640.

RONG-JONG WAI (M'99–SM'05) was born in Tainan, Taiwan, in 1974. He received the B.S. degree in electrical engineering and the Ph.D. degree in electronic engineering from Chung Yuan Christian University, Chung Li, Taiwan, in 1996 and 1999, respectively.

From 1998 to 2015, he was with Yuan Ze University, Chung Li, where he was the Dean of general affairs, from 2008 to 2013, and the Chairman of the Department of Electrical Engineering, from 2014 to 2015. Since 2015, he has been with the National Taiwan University of Science and Technology, Taipei, Taiwan, where he is currently a Distinguished Professor, the Dean of general affairs, and the Director of the Energy Technology and Mechatronics Laboratory. He is a chapter-author of *Intelligent Adaptive Control: Industrial Applications in the Applied Computational Intelligence Set* (Boca Raton, FL: CRC Press, 1998) and the coauthor of *Drive and Intelligent Control of Ultrasonic Motor* (Tai-chung, Taiwan: Tsang-Hai, 1999), *Electric Control* (Tai-chung, Taiwan: Tsang-Hai, 2002), and *Fuel Cell: New Generation Energy* (Tai-chung, Taiwan: Tsang-Hai, 2004). He has authored more than 170 conference articles and over 180 international journal articles. He has 57 inventive patents. His current research interests include power electronics, motor servo drives, mechatronics, energy technology, and control theory applications. The outstanding achievement of his research is for the contributions to real-time intelligent control in practical applications and high-efficiency power converters in energy technology.

Dr. Wai is a Fellow of the Institution of Engineering and Technology, U.K. He received the Excellent Research Award, in 2000, the Wu Ta-You Medal, and Young Researcher Award from the National Science Council, China, in 2003. In addition, he was a recipient of the Outstanding Research Award from the Yuan Ze University, China, in 2003 and 2007; the Excellent Young Electrical Engineering Award and the Outstanding Electrical Engineering Professor Award from the Chinese Electrical Engineering Society, China, in 2004 and 2010, respectively; the Outstanding Professor Award from the Far Eastern Y. Z. Hsu Science and Technology Memorial Foundation, China, in 2004 and 2008; the International Professional of the Year Award from the International Biographical Centre, U.K. in 2005; the Young Automatic Control Engineering Award from the Chinese Automatic Control Society, China, in 2005; the Yuan-Ze Chair Professor Award from the Far Eastern Y. Z. Hsu Science and Technology Memorial Foundation, China, in 2007, 2010, and 2013; the Electric Category-Invent Silver Medal Award, in 2007; the Electronic Category-Invent Gold and Silver Medal Awards, in 2008; the Environmental Protection Category-Invent Gold Medal Award, in 2008; the Most Environmental Friendly Award, in 2008; the Power Category-Invent Bronze Medal Award, in 2012; and the Electronic Category-Invent Gold and Silver Medal Awards from the International Invention Show and Technomart, Taipei, in 2015; the University Industrial Economic Contribution Award from the Ministry of Economic Affairs, in 2010; the Ten Outstanding Young Award from the Ten Outstanding Young Person's Foundation, in 2012; the Taiwan Top 100 MVP Managers Award from *MANAGER Today* magazine, in 2012; the Outstanding Engineering Professor Award from the Chinese Institute of Engineers, in 2013; the Green Technology Category-Scientific Paper Award from the Far Eastern Y. Z. Hsu Science and Technology Memorial Foundation, in 2014; the Scopus Young Researcher Lead Award-Computer Science from Taiwan Elsevier, in 2014; the Outstanding Research Award from the National Taiwan University of Science and Technology, in 2016 and 2018; and the Most Cited Researchers Award, in 2016 (Field: Electrical and Electronics Engineering).

ALEX S. PRASETIA was born in Negara, Bali, Indonesia, in 1995. He received the B.S. degree in electrical engineering from the Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia, in 2017, and the M.S. degree in electronics and computer engineering from the National Taiwan University of Science and Technology, Taiwan, in 2019. His current research interests include control system, power electronics, and machine learning.