Jestr Journal of Engineering Science and Technology Review 7 (2) (2014) 66 – 74

JOURNAL OF
Engineering Science and Technology Review

Research Article

www.jestr.org

Backstepping Controller with Intelligent Parameters Selection for Stabilization of Quadrotor Helicopter

Mohd Ariffanan Mohd Basri*, Abdul Rashid Husain and Kumeresan A. Danapalasingam

Dept. of Control and Mechatronics, Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia.

Received 30 April 2014; Accepted 15 July 2014

Abstract

In this paper, the dynamic model of quadrotor helicopter has been mathematically formulated. Then, an intelligent backstepping controller (IBC) is designed for the quadrotor altitude and attitude stabilization in the existence of external disturbances and measurement noise. The designed controller consists of a backstepping controller which can automatically select its parameters on-line by a fuzzy supervisory mechanism. The stability criterion for the stabilization of the quadrotor is proven by the Lyapunov theorem. Several numerical simulations using the dynamic model of a four degree of freedom (DOF) quadrotor helicopter show the effectiveness of the approach. Besides, the simulation results indicate that the proposed design techniques can stabilize the quadrotor helicopter with better performance than established linear design techniques.

Keywords: Quadrotor helicopter, backstepping control, fuzzy supervisory.

1. Introduction

Recent interest in the utilization of unmanned aerial vehicles (UAVs) in a variety of civil and military applications has prompted the need for such systems to operate with increased levels of autonomy. The UAVs have shown applications in different areas including search and rescue (SAR), meteorological studies, infrastructure inspection, homeland security and traffic surveillance.

Rotating wing (or helicopter) UAVs have the advantage above fixed wing UAVs that they are able to perform vertical take-off and landing (VTOL), and hovering at a fixed point. One very successful design for smaller UAVs is a helicopter with four horizontal rotors with no tailrotor, or called quadrotor. Quadrotors have the advantage that they can be controlled by varying the speed of the rotors and thus fixed-pitch blades can be used which simplifies the design and control of the vehicle. Moreover, the use of four rotors allows each individual rotor to have a smaller diameter than the equivalent helicopter rotor, for a given vehicle size, allowing them to store less kinetic energy during flight. However, quadrotor helicopter are an underactuated multi-input and multi-output system which has nonlinear dynamic behavior such as high coupling degree and unknown nonlinearities. These features make the controlling of the quadrotor a very challenging problem.

Many methods have been proposed to control a quadrotor vehicle, such as linear quadratic regulator (LQR) control [1], proportional-integral-derivative (PID) control [2], fuzzy logic (FL) control [3], sliding mode control [4] and backstepping control [5-8]. The backstepping technique has been used for stabilization of quadrotor helicopter in this paper. The choice of backstepping control scheme is due to there is a significant volume of research on this particular nonlinear control approach. However, design of this controller essentially needs to select proper parameters in order to obtain a good performance response. The improper selection of the parameters leads to inappropriate responses or even may lead to instability of the system. The control parameters presented in the literature which make use of the backstepping method are variously selected.

Based on the literatures study [9-13], the FL based self tuning controller has been an effective tool for the self tuning control of many nonlinear systems. Moreover, the implementation of FL based self tuning controller does not involve developing any model or the usage of complex algorithms [14]. Thus, due to these advantages, FL is used to select the parameters for the backstepping controller of quadrotor systems. So far, this technique has not been employed to solve the problem outlined.

2. Quadrotor Systems Modeling

A. Quadrotor Description

The quadrotor helicopter, shown in Fig. 1, has four rotors to generate the propeller forces $F_{i=1,2,3,4}$. The four rotors can be thought of as two pairs, (1,3)@(front, back) and (2,4)@(left, right). One pair rotates clockwise, while the other rotates counter clockwise in order to balance the torques and produce yaw motion as needed. Yaw motion can be obtained from the difference in the counter torque between each pair of propellers, (1,3) and (2,4). When all four rotors are spinning with the same angular velocity the net yaw is zero, and a difference in velocities between the two pairs creates either positive or negative yaw motion. The up (down) motion can be achieved by increasing (decreasing) the rotor

* E-mail address: ariffanan@fke.utm.my
ISSN: 1791-2377

speeds altogether with the same magnitude. Forward (backward) motion which is related to the pitch, θ angle can be obtained by increasing the back (front) rotor thrust and decreasing the front (back) rotor thrust. Finally, a sideways motion which is related to the roll, ϕ angle can be achieved by increasing the left (right) rotor thrust and decreasing the right (left) rotor thrust. Fig. 2 shows the various movements of a quadrotor due to rotor speeds changes.

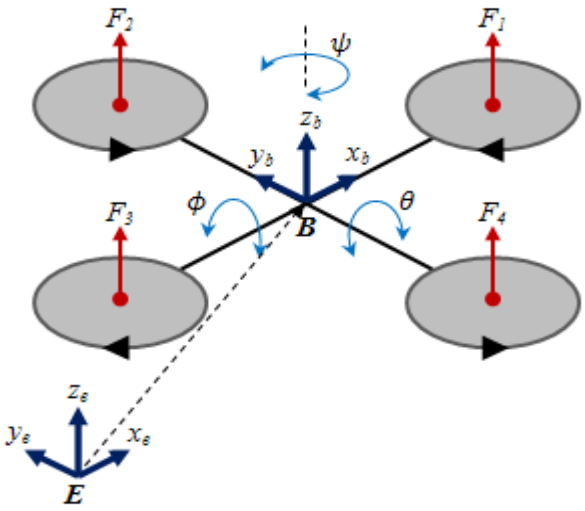

Fig. 1. Quadrotor helicopter configuration.

B. Quadrotor Kinematic Model

Let consider earth fixed frame $E = \{x_e, y_e, z_e\}$ and body fixed frame $B = \{x_b, y_b, z_b\}$, as seen in Fig. 1. Let $q = (x, y, z, \phi, \theta, \psi) \in R^6$ be the generalized coordinates for the quadrotor, where (x, y, z) denote the absolute position of the rotorcraft and (ϕ, θ, ψ) are the three Euler angles (roll, pitch and yaw) that describe the orientation of the aerial vehicle. Therefore, the model could be separated into two coordinate subsystems: translational and rotational. They are defined respectively by:

$$\xi = (x, y, z) \in R^3 \tag{1}$$

$$\eta = (\phi, \theta, \psi) \in R^3 \tag{2}$$

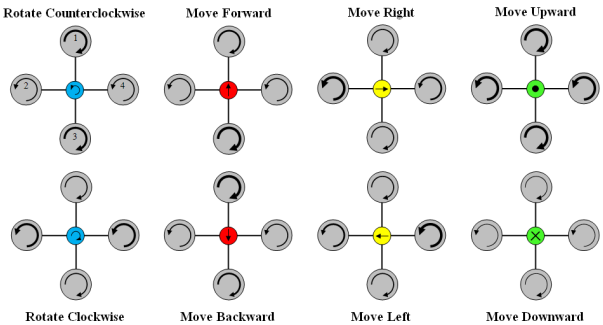

Fig. 2. The movements of a quadrotor: the arrow width is proportional to rotor speeds.

The kinematic equations of the translational and rotational movements are obtained by means of the rotation R and transfer T matrices respectively. The expression of the rotation R and transfer T matrices can be found in Olfati-Saber [15], and defined accordingly by (3) and (4):

$$R = \begin{pmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{pmatrix} \tag{3}$$

$$T = \begin{pmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi/c\theta & c\phi/c\theta \end{pmatrix} \tag{4}$$

where $s(\cdot)$, $c(\cdot)$ and $t(\cdot)$ are abbreviations for $sin(\cdot)$, $cos(\cdot)$ and $tan(\cdot)$, respectively.

The translational kinematics can be written as:

$$\dot{\xi} = RV \tag{5}$$

where $\dot{\xi}$ and V are respectively the linear velocity vector w.r.t. the earth fixed frame E and body fixed frame B.

The rotational kinematics can be defined as follows:

$$\dot{\eta} = T\omega \tag{6}$$

where $\dot{\eta}$ and ω are the angular velocity vector w.r.t. the earth fixed frame E and body fixed frame B, respectively.

C. Quadrotor Dynamic Model

The dynamic model of quadrotor is derived from Newton-Euler approach. It can be useful to express the translational dynamic equations w.r.t. the earth fixed frame E and rotational dynamic equations w.r.t. the body fixed frame B.

Therefore, the translational dynamic equations of quadrotor can be written as follows:

$$m\ddot{\xi} = -mge_z + u_T R e_z \tag{7}$$

where m denotes the quadrotor mass, g the gravity acceleration, $e_z = (0,0,1)^T$ the unit vector expressed in the frame E and u_T the total thrust produced by the four rotors defined as follows:

$$u_T = \sum_{i=1}^{4} F_i = b\sum_{i=1}^{4} \Omega_i^2 \tag{8}$$

where F_i and Ω_i denote respectively, the thrust force and speed of the rotor i and b is the thrust factor.

The rotational dynamic equations of quadrotor can be written as follows:

$$I\dot{\omega} = -\omega \times I\omega - G_a + \tau \tag{9}$$

where I is the inertia matrix, $-\omega \times I\omega$ and G_a are the gyroscopic effect due to rigid body rotation and propeller orientation change respectively, while τ is the control torque obtained by varying the rotor speeds. G_a and τ are defined as:

$$G_a = \sum_{i=1}^{4} J_r(\omega \times e_z)(-1)^{i+1}\Omega_i \tag{10}$$

$$\tau = \begin{pmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{pmatrix} = \begin{pmatrix} lb(\Omega_4^2 - \Omega_2^2) \\ lb(\Omega_3^2 - \Omega_1^2) \\ d(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \end{pmatrix} \tag{11}$$

where J_r is the rotor inertia, l represent the distance from the rotors to the centre of mass and d is the drag factor.

Then, by recalling (7) and (9), the dynamic model of the quadrotor in terms of position (x, y, z) and rotation (ϕ, θ, ψ) is written as:

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -g \end{pmatrix} + \frac{1}{m} \begin{pmatrix} c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\phi s_\theta c_\psi - s_\phi c_\psi \\ c_\phi c_\theta \end{pmatrix} u_T \quad (12)$$

$$\begin{pmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = \begin{pmatrix} \dot{\theta}\dot{\psi}\left(\frac{I_{yy} - I_{zz}}{I_{xx}}\right) \\ \dot{\phi}\dot{\psi}\left(\frac{I_{zz} - I_{xx}}{I_{yy}}\right) \\ \dot{\theta}\dot{\phi}\left(\frac{I_{xx} - I_{yy}}{I_{zz}}\right) \end{pmatrix} - \begin{pmatrix} \frac{J_r}{I_{xx}}\dot{\theta}\Omega_d \\ -\frac{J_r}{I_{yy}}\dot{\phi}\Omega_d \\ 0 \end{pmatrix} + \begin{pmatrix} \frac{1}{I_{xx}}\tau_\phi \\ \frac{1}{I_{yy}}\tau_\theta \\ \frac{1}{I_{zz}}\tau_\psi \end{pmatrix} \quad (13)$$

Consequently, quadrotor is an underactuated system with six outputs $(x, y, z, \phi, \theta, \psi)$ and four control inputs ($u_T, \tau_\phi, \tau_\theta, \tau_\psi$).

Finally, the full dynamic model can be rearranged in the following form:

$$\ddot{x} = \left(c_\phi s_\theta c_\psi + s_\phi s_\psi\right)\frac{1}{m}u_1$$
$$\ddot{y} = \left(c_\phi s_\theta c_\psi - s_\phi c_\psi\right)\frac{1}{m}u_1$$
$$\ddot{z} = -g + \left(c_\phi c_\theta\right)\frac{1}{m}u_1$$
$$\ddot{\phi} = \dot{\theta}\dot{\psi}\left(\frac{I_{yy} - I_{zz}}{I_{xx}}\right) - \frac{J_r}{I_{xx}}\dot{\theta}\Omega_d + \frac{l}{I_{xx}}u_2$$
$$\ddot{\theta} = \dot{\phi}\dot{\psi}\left(\frac{I_{zz} - I_{xx}}{I_{yy}}\right) + \frac{J_r}{I_{yy}}\dot{\phi}\Omega_d + \frac{l}{I_{yy}}u_3$$
$$\ddot{\psi} = \dot{\theta}\dot{\phi}\left(\frac{I_{xx} - I_{yy}}{I_{zz}}\right) + \frac{1}{I_{zz}}u_4$$

with a renaming of the control inputs as:

$$u_1 = b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2)$$
$$u_2 = b(\Omega_4^2 - \Omega_2^2)$$
$$u_3 = b(\Omega_3^2 - \Omega_1^2)$$
$$u_4 = d(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \quad (15)$$

and the definition of disturbance:

$$\Omega_d = \Omega_2 + \Omega_4 - \Omega_1 - \Omega_3 \quad (16)$$

The control inputs can be rewritten in matrix form as:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} b & b & b & b \\ 0 & -b & 0 & b \\ -b & 0 & b & 0 \\ -d & d & -d & d \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \quad (17)$$

Then, by inversing matrix (17) the rotor speeds can be calculated as:

$$\begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} = \begin{bmatrix} 0.25 & 0 & -0.5 & -0.25 \\ 0.25 & -0.5 & 0 & 0.25 \\ 0.25 & 0 & 0.5 & -0.25 \\ 0.25 & 0.5 & 0 & 0.25 \end{bmatrix} \begin{bmatrix} u_1/b \\ u_2/b \\ u_3/b \\ u_4/d \end{bmatrix} \quad (18)$$

3. Control System for Quadrotor

In this paper, for stabilizing control, the system is simplified into four DOF i.e. only the z-directional linear motion (altitude) and angular motion (attitude) are considered. For the design of the controller, the following state variables are defined:

$$x = \left[z\ \dot{z}\ \phi\ \dot{\phi}\ \theta\ \dot{\theta}\ \psi\ \dot{\psi}\right]^T = [x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7\ x_8]^T \quad (19)$$

The altitude and the rotational dynamics of quadrotor can be decomposed into four nonlinear subsystems, which are:

Altitude subsystem:

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = f_1(x) + g_1(x)u_1 \quad (20)$$

where

$$f_1(x) = -g$$
$$g_1(x) = c_\phi c_\theta \left(\frac{1}{m}\right)$$

Roll subsystem:

$$\dot{x}_3 = x_4$$
$$\dot{x}_4 = f_2(x) + g_2(x)u_2 \quad (21)$$

where

$$f_2(x) = \dot{\theta}\dot{\psi}\left(\frac{I_{yy} - I_{zz}}{I_{xx}}\right) - \frac{J_r}{I_{xx}}\dot{\theta}\Omega_d$$
$$g_2(x) = \frac{l}{I_{xx}}$$

Pitch subsystem:

$$\dot{x}_5 = x_6$$
$$\dot{x}_6 = f_3(x) + g_3(x)u_3 \quad (22)$$

where

$$f_3(x) = \dot{\phi}\dot{\psi}\left(\frac{I_{zz} - I_{xx}}{I_{yy}}\right) + \frac{J_r}{I_{yy}}\dot{\phi}\Omega_d$$
$$g_3(x) = \frac{l}{I_{yy}}$$

Yaw subsystem:

$$\dot{x}_7 = x_8$$
$$\dot{x}_8 = f_4(x) + g_4(x)u_4 \quad (23)$$

where

$$f_4(x) = \dot{\theta}\dot{\phi}\left(\frac{I_{xx} - I_{yy}}{I_{zz}}\right)$$

$$g_4(x) = \frac{1}{I_{zz}}$$

Thus the dynamic equations of the quadrotor can be decomposed into four single-input nonlinear subsystems in the form of:

$$x^{(n)} = f(x) + g(x)u, \quad n = 2 \tag{24}$$

where u is the input; $f(x)$and $g(x)$ are the nonlinear function.

A. Backstepping Control System

The control objective in this work is to design a suitable control law for the system (24) so that the state vector x of the quadrotor system can track a desired reference trajectory vector x_d.

The design of backstepping control for the quadrotor systems is described step-by-step as follows:

Step 1: Define the tracking error:

$$e_1 = x_d - x \tag{25}$$

where x_d is a desired trajectory specified by a reference model. Then the derivative of tracking error can be represented as:

$$\dot{e}_1 = \dot{x}_d - \dot{x} \tag{26}$$

The first Lyapunov function is chosen as:

$$V_1(e_1) = \frac{1}{2}e_1^2 \tag{27}$$

The derivative of V_1 is:

$$\dot{V}_1(e_1) = e_1\dot{e}_1 = e_1(\dot{x}_d - \dot{x}) \tag{28}$$

$\dot{x}$ can be viewed as a virtual control. The desired value of virtual control known as a stabilizing function can be defined as follows:

$$\alpha = \dot{x}_d + k_1e_1 \tag{29}$$

where k_1 is a positive constant.

By substituting the virtual control by its desired value, Eq. (28) then becomes:

$$\dot{V}_1(e_1) = -k_1e_1^2 \leq 0 \tag{30}$$

Step 2: The deviation of the virtual control from its desired value can be defined as:

$$e_2 = \alpha - \dot{x} = \dot{x}_d + k_1e_1 - \dot{x} \tag{31}$$

The derivative of e_2 is expressed as:

$$\dot{e}_2 = \dot{\alpha} - \ddot{x} = k_1\dot{e}_1 + \ddot{x}_d - f(x) - g(x)u \tag{32}$$

The second Lyapunov function is chosen as:

$$V_2(e_1, e_2) = \frac{1}{2}e_1^2 + \frac{1}{2}e_2^2 \tag{33}$$

Finding derivative of (33), yields:

$$\begin{aligned}\dot{V}_2(e_1, e_2) &= e_1\dot{e}_1 + e_2\dot{e}_2 \\ &= e_1(\dot{x}_d - \dot{x}) + e_2(\dot{\alpha} - \ddot{x}) \\ &= e_1(e_2 - k_1e_1) + e_2(k_1\dot{e}_1 + \ddot{x}_d - f(x) - g(x)u) \\ &= -k_1e_1^2 + e_2(e_1 + k_1\dot{e}_1 + \ddot{x}_d - f(x) - g(x)u)\end{aligned} \tag{34}$$

Step 3: For satisfying $\dot{V}_2(e_1, e_2) \leq 0$, the control input u is selected as:

$$u = \frac{1}{g(x)}(e_1 + k_1\dot{e}_1 + \ddot{x}_d - f(x) + k_2e_2) \tag{35}$$

where k_2 is a positive constant. The term k_2e_2 is added to stabilize the tracking error e_1.

Substituting (35) into (34), the following equation can be obtained:

$$\dot{V}_2(e_1, e_2) = -k_1e_1^2 - k_2e_2^2 = -E^TKE \leq 0 \tag{36}$$

where $E = [e_1\ e_2]^T$ and $K = diag(k_1, k_2)$.

Since $\dot{V}_2(e_1, e_2) \leq 0$, $\dot{V}_2(e_1, e_2)$ is negative semi-definite. Therefore, the control law in (35) will asymptotically stabilize the system.

B. Fuzzy-Based Backstepping Control System

In order to guarantee the system stability and convergence of tracking error, backstepping control parameters $k_{i=1,2}$ (for each subsystem) need to be selected properly. In conventional backstepping method, these parameters are selected by trial-and-error, which is rather time consuming to search a special suited solution in such a large search space that includes all feasible solutions. To overcome this drawback, this paper adopts the FL for selecting the optimal value of the backstepping control parameters. The structure of the proposed IBC is as shown in Fig. 3. Since the proposed IBC aims to improve the control performance yielded by a backstepping controller, it keeps the simple structure of the backstepping controller.

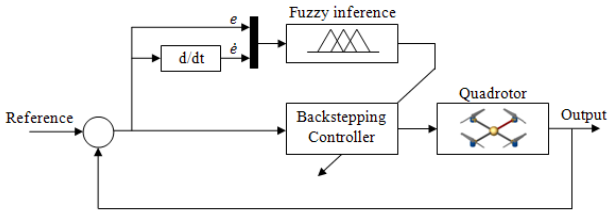

Fig. 3.Structure of the intelligent backstepping controller.

From the fuzzy structure, there are two inputs to the fuzzy inference: error, e and derivative of error, $\dot{e}$ and two outputs for each backstepping control parameters subsystem. Sugeno model is applied as structure of fuzzy inference to obtain the best parameters value. The structure of the fuzzy inference block in Fig. 3 is shown in Fig. 4.

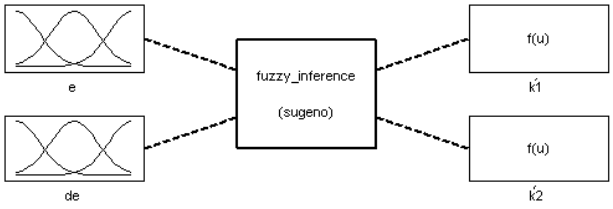

Fig.4. Structure of the fuzzy inference block.

The steps involved in the design of a fuzzy supervisory system include the normalization of the parameters, fuzzification of the inputs and outputs, development of rule base and the defuzzification process. It can be described step-by-step as follows:

Step 1: Normalization of the control parameters

Suppose the variable ranges of the parameters k_i are $[k_{i\,min}, k_{i\,max}]$. The range of each parameter is determined based on the numerical experiments on backstepping controller. In this case, the range of each parameter is provided as $k_i \in [1,50]$. The parameters must be normalized over the interval [0, 1] as follows:

$$k_i' = \frac{k_i - k_{i\,min}}{k_{i\,max} - k_{i\,min}} = \frac{k_i - 1}{50 - 1} \quad (37)$$

Therefore: $k_i = 49k_i' + 1$

Step 2: Fuzzification of the inputs and outputs

In this paper, the triangular shape input memberships as shown in Fig. 5 and 6 are considered. The range of inputs (-1 until 1) is divided into five sets of triangular shapes which are NB (negative big), NS (negative small), ZE (zero), PS (positive small) and PB (positive big). The output memberships are represented by five Sugeno-type singleton values that range between 0 to 1 as shown in Fig. 7.

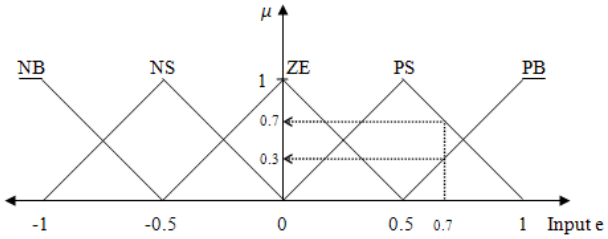

Fig. 5. The membership functions for the input e.

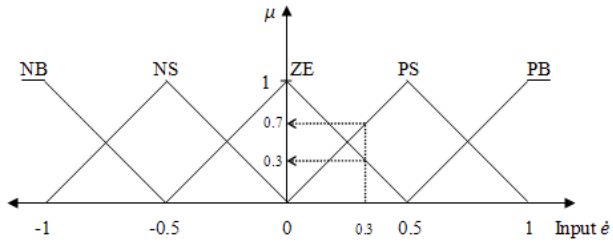

Fig. 6. The membership functions for the input $\dot{e}$.

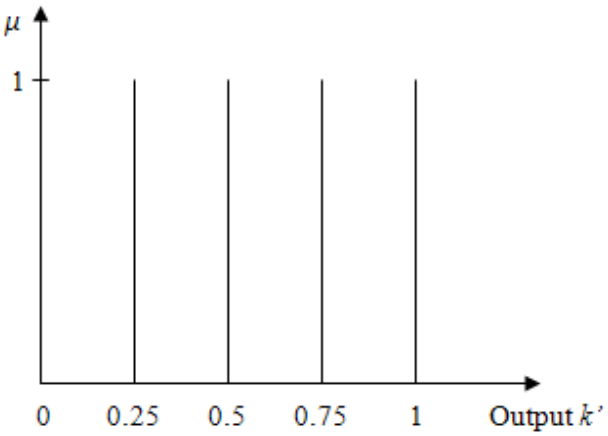

Fig. 7. The membership functions for the output k_i'.

Step 3: Development of the fuzzy rule base

A typical rule in a Sugeno fuzzy model has the form:

If (Input 1 is *x*) *and* (Input 2 is *y*) *then* (Output is $z = ax + by + c$)

If the output membership functions are constant as employed in this work, then $a = b = 0$. The fuzzy rule base is constructed by using several *if-then* statements and premise and consequent of each statement which are fuzzy propositions. The *if*-part of the rule "Input 1 is *x*" and "Input 2 is *y*" are called the *antecedent* or premise, while the *then*-part of the rule "Output is *z*" is called the *consequent.* For a FL with two inputs and five linguistic values for each input, there are $5^2 = 25$ possible rules that connect the inputs and the singleton output as given in Table 1.

Table 1. Rules of the fuzzy inference.

		e				
		NB	NS	ZE	PS	PB
ė	NB	0	0	0.25	0.25	0.5
	NS	0	0.25	0.25	0.5	0.75
	ZE	0.25	0.25	0.5	**0.75**	**0.75**
	PS	0.25	0.5	0.75	**0.75**	**1**
	PB	0.5	0.75	0.75	1	1

Step 4: Fuzzy inference system

The fuzzy inference system is the heart of a FL. It acts as the bridge between the fuzzification input stage and defuzzification output stage. The fuzzy inference system can be divided into three process elements:

i) Application of the fuzzy operator (AND or OR) in the antecedent.

If the antecedent of a given rule has more than one part, the fuzzy operator is applied to obtain one number that represents the result of the antecedent for that rule. In Matlab's FL toolbox, two built-in AND methods are supported: *min* (minimum) and *prod* (product). Two built-in OR methods are also supported: *max* (maximum), and *probor* (probabilistic OR). The probabilistic OR method is calculated according to: $probor(a, b) = a + b - ab$. Any premise with a value greater than zero means that its corresponding rule is active, or has "fired" in FL terminology.

ii) Implication from the antecedent to the consequent.

Implication refers to the process of shaping the fuzzy set in the consequent based on the results of the antecedent. The input for the implication process is a single number given by the antecedent, and the output is a fuzzy set. Two built-in methods are supported, and they are the same functions that are used by the AND method: *min* (minimum), which truncates the output fuzzy set, and *prod* (product), which scales the output fuzzy set.

iii) Aggregation of the consequents across the rules.

Aggregation is the process by which the fuzzy sets that represent the outputs of each rule are combined into a single

fuzzy set. The input of the aggregation process is the list of truncated output functions returned by the implication process for each rule. The sets are combined by calculating the union of the implied membership functions. Three built-in methods are supported: *max* (maximum), *probor* (probabilistic OR) and *sum* (simply the sum of each rule's output set). The output of the aggregation process is one fuzzy set, which is the input to the defuzzification process.

As an example of the inference process, consider the error, e and derivative of error, $\dot{e}$ membership function introduced earlier. The 0.7 error has a membership of 0.7 in the PS (positive small) set, 0.3 in the PB (positive big) set and 0 in all other sets. Meanwhile, the 0.3 derivative of error has a membership of 0.7 in the PS (positive small) set, 0.3 in the ZE (zero) set and 0 in all other sets. Firstly, the fuzzy operator is applied to each rule since the antecedent has two parts (error, e and derivative of error, $\dot{e}$). In this case, consider the fuzzy AND operation to be the product of the two antecedent values. The two different pieces of the antecedent (e.g. e is PB and $\dot{e}$ is PS) yielded the fuzzy membership values 0.3 and 0.7 respectively. The fuzzy AND operator simply products the two values, result in the firing strength 0.21, and the fuzzy operation for that particular rule is complete. Secondly, the implication process is implemented to reshape the consequent using the firing strength value given by the antecedent. The output membership functions are modified by the AND operation. In this example the *min* operator has been chosen for the AND operation, so that the output singleton membership functions are truncated at the consequent. This can be seen in Fig. 8. The process one and two are repeated for the other rules. Finally, once the fuzzy sets that represent the outputs of each rule are obtained, the aggregation process is implemented to combine the fuzzy sets into a single fuzzy set. This is achieved by calculating the union of the outputs of each rule using *max* method.

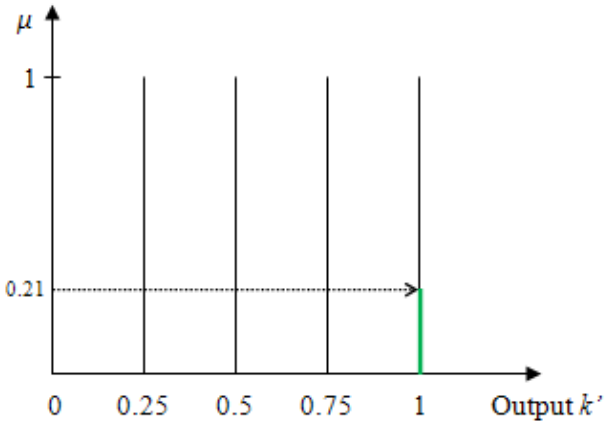

Fig. 8. Membership functions for output of the rule (e is PB and $\dot{e}$ is PS then k' is 1), truncated at consequent.

Step 5: Defuzzification of the output fuzzy set

Since a Sugeno inference engine with singleton output membership functions was employed, the centroid defuzzification method is used to convert the aggregated fuzzy set to a crisp output value. This work computes the weighted average of the membership function or the center of gravity (COG) of the aggregated membership function:

$$Output, k' = \frac{\sum_{i=1}^{n} w_i c_i}{\sum_{i=1}^{n} w_i} \quad (38)$$

where w_i is firing strength value for ith rule, c_i is the corresponding singleton value and n is number of rules.

Continuing with the aforementioned example, the defuzzification process computes the weighted average of the aggregated membership function. Since the error value (0.7) is a member of PS and PB membership functions, then NB, NS and ZE will have zero membership degree value. Simultaneously, 0.3 value of the error rate is a member of ZE and PS membership functions, then NB, NS and PB will have zero membership degree value. Since AND operation is used to connect the two antecedent and *prod* (product) method is chosen, thus only rules with both of the antecedent have a value greater than zero will active or fire. Hence, in this case only four rules will be fired. The bold output membership functions showed in Table 1 indicate the active rules. The corresponding active rules in *if-then* statements are listed as follows:

If e is PS *and* $\dot{e}$ is ZE *then* k' is 0.75; $(weighted\ value = 0.7\times0.3 = 0.21)$

If e is PS *and* $\dot{e}$ is PS *then* k' is 0.75; $(weighted\ value = 0.7\times0.7 = 0.49)$

If e is PB *and* $\dot{e}$ is ZE *then* k' is 0.75; $(weighted\ value = 0.3\times0.3 = 0.09)$

If e is PB *and* $\dot{e}$ is PS *then* k' is 1; $(weighted\ value = 0.3\times0.7 = 0.21)$

Therefore, Eq. (38) can be written as:

$$Output, k' = \frac{(0.21\times0.75) + (0.49\times0.75) + (0.09\times0.75) + (0.21\times1)}{0.21 + 0.49 + 0.09 + 0.21} = 0.8$$

4. Simulation Model

As a precursor for developing a model based control design, the simulation environment of the quadrotor mathematical model is developed. The simulation model provides a platform suitable for control design of quadrotor systems to be used for control algorithm development and verification, before working with a real experimental system. For modeling and simulating the quadrotor dynamic model a few steps design procedure need to be done. Firstly, the differential equations in (20)-(23) of the quadrotor systems are modeled using Matlab Simulink. The parameters values used in the simulation are taken from Voos [16], as listed in Table 2. Fig. 9 depicts the simulink model used for the simulation studies. Initially, the numerical solution of the model is solved using the *ode45* variable-step solver, with a relative tolerance of 0.001 (the default). A variable-step solver can shorten the simulation time significantly because it can dynamically adjust the step size as necessary and thus reduce the number of steps. Even this solver can give a desired level of accuracy, but it is not useful for real-time simulation. Thus, the solver options used in this paper are configured as fixed-step type.

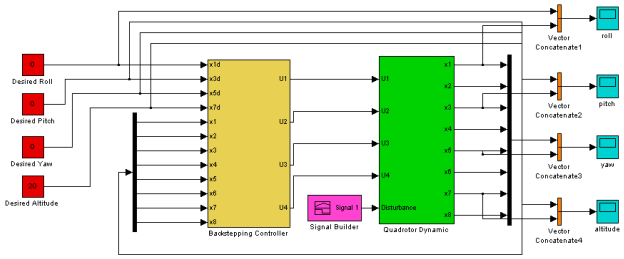

Fig. 9. The overall Simulink model of the quadrotor helicopter and the control system

Table 2. Parameters of the quadrotor.

Parameter	Description	Value	Units
g	Gravity	9.81	m/s^2
m	Mass	0.5	kg
l	Distance	0.2	m
I_{xx}	Roll inertia	4.85×10^{-3}	$kg \cdot m^2$
I_{yy}	Pitch inertia	4.85×10^{-3}	$kg \cdot m^2$
I_{zz}	Yaw inertia	8.81×10^{-3}	$kg \cdot m^2$
b	Thrust factor	2.92×10^{-6}	
d	Drag factor	1.12×10^{-7}	

5. Simulation Results

In this section, the performance of the proposed approach is evaluated. Four simulation experiments have been performed on the quadrotor. In the first experiment, the simulation results of the proposed controller in a stabilizing problem are given. In the second, the performance of the scheme is investigated in attitude tracking problem. In the third, the disturbance rejection performance in attitude stabilization around zero is shown in order to demonstrate the effectiveness of the designed controller. Finally, to assess the performance of the control technique under the influence of noise measurements, an experiment of attitude stabilization is accomplished.

A. Simulation experiment 1: stabilizing problem

In this simulation experiment, the control objectives are to reach and maintain quadrotor at a certain desired altitude/attitude, such that the vehicle can hover at a fixed point. The desired altitude/attitude is given by $x_d = [z_d, \phi_d, \theta_d, \psi_d] = [20, 0, 0, 0]^T$. The initial states are given by $z = 0, \phi = 0.2, \theta = 0.2$ and $\psi = 0.2$. Simulation results show the control design is able to stabilize the quadrotor in hover mode. Under the proposed fuzzy-based IBC, it can be observed that the altitude/attitude of the quadrotor can be maintained at the desired altitude/attitude, that is, the hovering flight is stable as shown in Fig. 10.

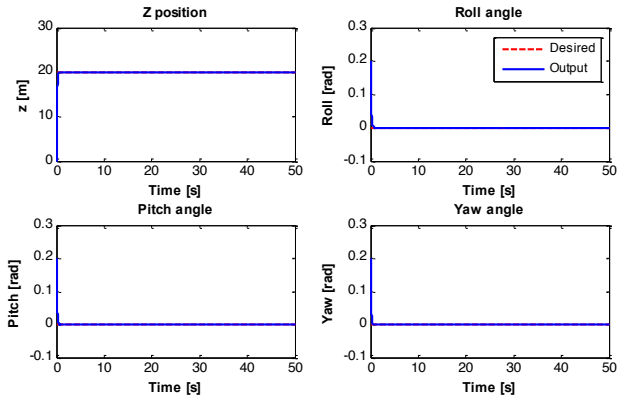

Fig. 10. Altitude/attitude of the hovering quadrotor using IBC.

As aforementioned, the improper selection of the backstepping control parameters leads to inappropriate responses of the system. Results from Fig. 11 are evidence that the poorly defined of backstepping control parameters will degrade the performance response of the system.

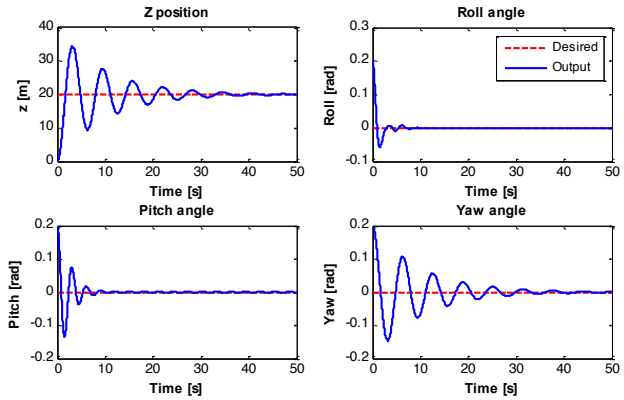

Fig. 11. Altitude/attitude of the hovering quadrotor using backstepping control with improper parameters.

For the aim of the comparison of the control performance, the proportional-derivative (PD) control is used to control the quadrotor. The parameters of PD controller are heuristically selected as $k_p = 0.2$ and $k_d = 0.7$. Fig. 12 shows the simulation results of the PD control to perform stabilization where it can be seen the settling time with PD controller is rather large and having a small overshoot, but in contrast, the transient response is faster and non-overshooting using the IBC.

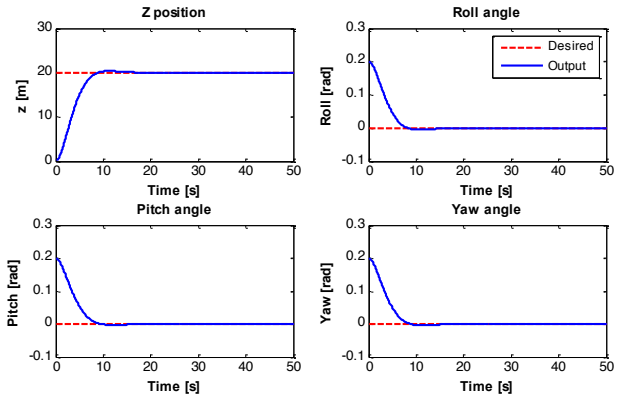

Fig. 12. Altitude/attitude of the hovering quadrotor using PD control.

B. Simulation experiment 2: tracking problem

In this simulation experiment, the performance of the proposed control approach is investigated in attitude tracking problem of the quadrotor. The periodic sinusoidal functions are used as a reference to the attitude angles and the response is shown in Fig. 13. As it can be seen, the attitude angles tracks the desired reference trajectories smoothly. The results also show that, the IBC can give a very small tracking error which is indicating a good tracking performance. In addition, for the purpose of comparison, the conventional PD control is adopted. From the simulation results shown in Fig. 14, it is noted that by using the PD control system the attitude angles are unable to track the desired reference trajectories accurately. Obviously, the

tracking performance of the proposed method is better than PD control.

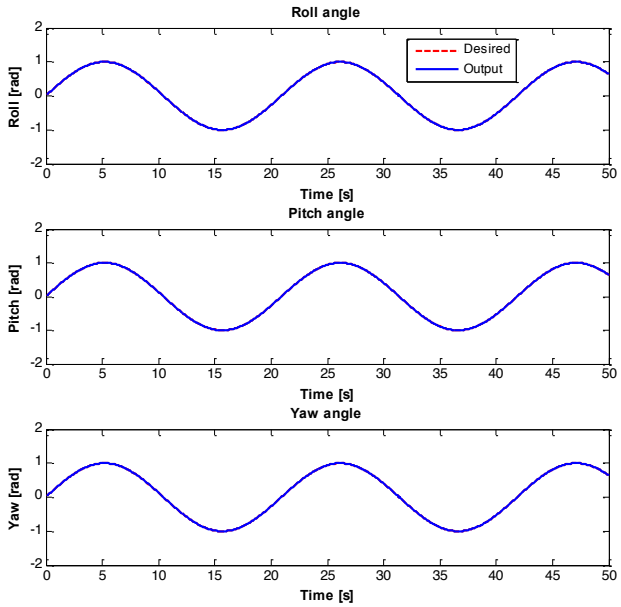

Fig. 13. Attitude tracking of the quadrotor using IBC.

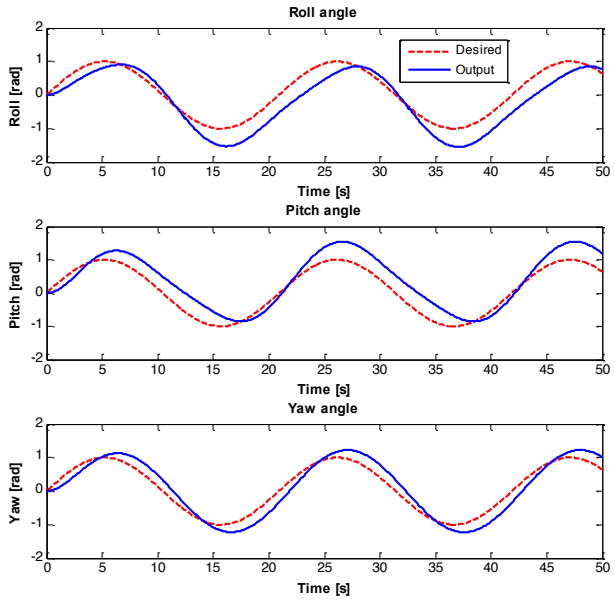

Fig. 14. Attitude tracking of the quadrotor using PD control.

C. Simulation experiment 3: disturbance rejection

To further highlight the advantage of the proposed control structure, the simulation experiment of disturbance rejection in attitude stabilization around zero is carried out. In this case, the impulse type disturbance as illustrated in Fig. 15 is considered. The attitude angles are externally disturbed at time instants 10s for the roll, 20s for the pitch and 30s for the yaw angle. The disturbed attitude angles response is shown in Fig. 16. It can be noted that the attitude angles converge to zero set-point rapidly as the disturbances are exerted, and hence the quadrotor helicopter can be stable.

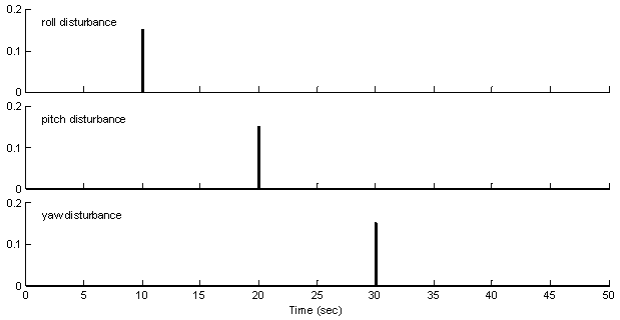

Fig. 15. Impulse type disturbances

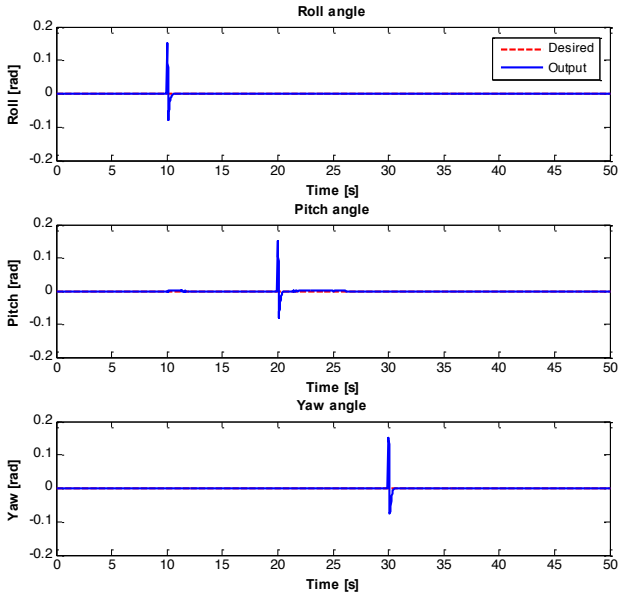

Fig. 16. Disturbance rejection in attitude stabilization around zero using IBC.

D. Simulation experiment 4: measurement noise condition

In order to assess the performance of the control technique under the influence of measurement noise, the measured output angles are added with noise. Fig. 17 shows the Gaussian noise with zero mean and standard deviation of 0.1 added into the system. Fig. 18 represents the response of quadrotor attitude stabilization around zero with state measurements are corrupted by noise. As it can be clearly seen, a satisfactory performance response with small deviation of roll, pitch and yaw angle from zero is obtained, which confirms the effectiveness of the proposed controller against measurement noise.

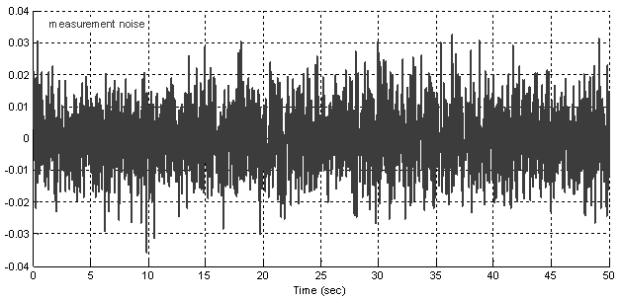

Fig. 17. Measurement noise

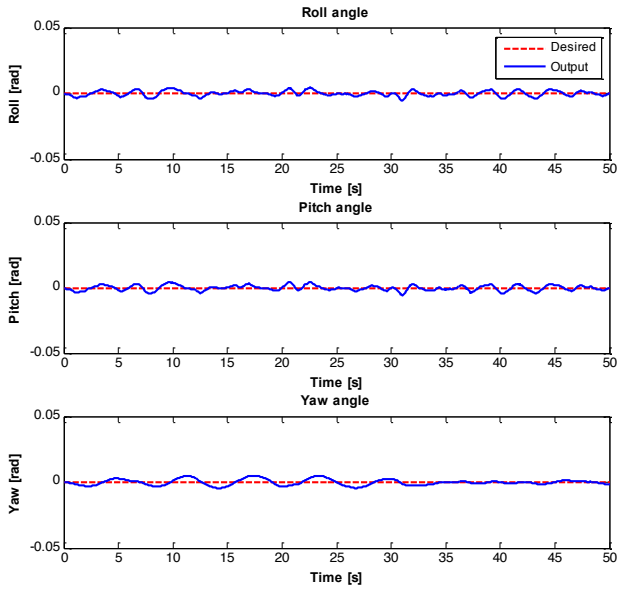

Fig. 18. Attitude stabilization with noise measurements using IBC.

From the simulation results, the potential and performances of the proposed control scheme can be clearly seen. Since the proposed control structure captures the dynamic response of controlled system, the IBC will achieve satisfactory control performance for the quadrotor system. Furthermore, it is obvious that the transient and tracking performances of the proposed method are better than PD control. Therefore, the proposed control scheme is suitable for stabilization of quadrotor helicopter.

6. Conclusions

In this paper, the application of an intelligent backstepping controller to control the altitude and attitude of a quadrotor helicopter is successfully demonstrated. First, the mathematical model of the quadrotor is introduced. Then, the proposed intelligent backstepping controller which can automatically select the controller parameters based on the fuzzy logic method is developed. The backstepping control design is derived based on Lyapunov function, so that the stability of the system can be guaranteed. Finally, the proposed control scheme is applied to autonomous hovering quadrotor helicopter. Several simulation results show that high-precision transient response can be achieved by using the proposed control system.

Acknowledgement

The authors would like to express great appreciation to Universiti Teknologi Malaysia for providing great support during this work and acknowledges financial support from the Ministry of Higher Education Malaysia.

References

1. P. Castillo, R. Lozano, A. Dzul, "Stabilization of a mini rotorcraft with four rotors", In Proc. of The IEEE Control Systems Magazine, vol. 25, no. 6, pp. 45-55, 2005.
2. A.L. Salih, M. Moghavvemi, H.A.F. Mohamed, K.S. Gaeid, "Modelling and PID controller design for a quadrotor unmanned air vehicle", In *Proc. of The IEEE International Conference on Automation Quality and Testing Robotics (AQTR)*, pp. 1-5, 2010.
3. M. Santos, V. López, F. Morata, "Intelligent fuzzy controller of a quadrotor", In *Proc. of The IEEE International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, pp. 141-146, 2010.
4. R. Xu, U. Ozguner, "Sliding mode control of a quadrotor helicopter", In *Proc. of The IEEE Conference on Decision and Control*, pp. 4957-4962, 2006.
5. T. Madani, A. Benallegue, "Backstepping Control for a Quadrotor Helicopter", In *Proc. of The IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3255-3260, 2006.
6. H. Bouadi, M. Bouchoucha, M. Tadjine, "Modelling and Stabilizing Control Laws Design Based on Backstepping for an UAV Type-Quadrotor", In *Proc. of the IFAC Symposium on IAV*, Toulouse, France, 2007.
7. G.V. Raffo, M.G. Ortega, F.R. Rubio, "Backstepping/nonlinear H∞ control for path tracking of a quadrotor unmanned aerial vehicle", In *Proc. of The American Control Conference*, pp. 3356-3361, 2008.
8. G. Regula, B. Lantos, "Backstepping-based control design with state estimation and path tracking to indoor quadrotor helicopter", *Period. Polytech. Electr. Eng*, pp. 1-10, 2010.
9. M.A. Llama, R. Kelly, V. Santibanez, "A stable motion control system for manipulators via fuzzy self-tuning", *Fuzzy sets and systems*, vol. 124, no. 2, pp. 133-154, 2001.
10. R. Babuska, J. Oosterhoff, A. Oudshoorn, P. Bruijn, "Fuzzy self-tuning PI control of pH in fermentation", *Engineering Applications of Artificial Intelligence*, vol. 15, no. 1, pp. 3-15, 2002.
11. T. Ahn, Y. Kwon, H. Kang, "Drive of induction motors using a pseudo-on-line fuzzy-PID controller based on genetic algorithm", *Trans. on Control, Automation and Systems Engineering*, vol. 2, no. 2, pp. 85-91, 2000.
12. Z. Sun, R. Xing, C. Zhao, W. Huang, "Fuzzy auto-tuning PID control of multiple joint robot driven by ultrasonic motors", *Ultrasonics*, vol. 46, no. 4, pp. 303-312, 2007.
13. S. Soyguder, M. Karakose, H. Alli, "Design and simulation of self-tuning PID-type fuzzy adaptive control for an expert HVAC system", *Expert Systems with Applications*, vol. 36, no. 3, pp. 4566-4573, 2009.
14. M. Venmathi, K. Sujatha, E.S. Percis, A. Nalini, "Fuzzy and ANN tuning of PI controller for level process station", *International Journal on Computational Intelligence*, vol. 1, no. 2, pp. 3-21, 2010.
15. R. Olfati-Saber, "Nonlinear control of underactuated mechanical systems with application to robotics and aerospace vehicles", *Ph.D. thesis*, Massachusetts Institute of Technology, 2000.
16. H. Voos, "Nonlinear control of a quadrotor micro-UAV using feedback-linearization", In *Proc. of The IEEE International Conference on Mechatronics*, pp. 1-6, 2009.