Evaluating the Quality of Intelligent Controllers for 3-DOF Delta Robot Control

Le Minh Thanh, Luong Hoai Thuong, Pham Thanh Tung
Vinh Long University of Technical Education, Vietnam
Email: thanhlm@vlute.edu.vn, {thuonglh, tungpt}@vlute.edu.vn

Cong-Thanh Pham
Viet Nam Aviation Academy, Vietnam
Email: thanhpc@vaa.edu.vn

Chi-Ngon Nguyen *
Can Tho University, Vietnam
Email: ncngon@ctu.edu.vn

***Abstract*—Delta robots have been successfully researched and manufactured in many countries. In this paper, the authors will research and compare many different controllers to control the delta robot so that it works stability when changing the working speed and load. The regression fuzzy neural network along with PID controller (RFNNC-PID) is used to observe the output error parameters of the robot through the identifier to update and adjust the optimal input parameters to control the robot, contributing error reduction of the closed-loop control system. The advantage of this controller is that it does not care about the robot's mathematical model and the RFNNC-PID controller has been successfully simulated by the authors in MATLAB/Simulink through the robot's trajectory control. The proposed controller will be compared to the single neuron PID controller and the traditional PID controller in MATLAB/Simulink. The simulation results show that the proposed controller is better than the single neuron PID controller and the traditional one with obtaining response time about 3.8 $\pm$0.1 (s) and without steady-state error.**

***Index Terms*—Delta robot, single neural PID, recurrent fuzzy neural network, Identifier, trajectory tracking.**

Symbol	**Unit**	**Meaning**
$\theta_1, \theta_2, \theta_3$	*rad/s*	Angles the upper leg of the robot
$\alpha_1, \alpha_2, \alpha_3$	*rad/s*	Passive angles determine the position of the connection points
R	*m*	Radius of the fixed plate
r	*m*	Radius of the moving plate
L_1	*m*	Length of upper arm
L_2	*m*	Length of upper arm
m_1	*kg*	Mass of upper arm
m_2	*kg*	Mass of lower arm
m_P	*kg*	Mass of the moving plate
O	*m*	The center of the fixed plate
P	*m*	The center of the moving plate
A_i	*m*	The connection point of the upper legs with the fixed plate
B_i	*m*	The connection point of the upper legs with the lower legs
D_i	*m*	The connection point of the lower legs with the moving plate

Abbreviation

PID	Proportional Integral Derivative
DOF	Degrees of freedom
RFNNC	Recurrent Fuzzy Neural Network Controller
RFNNI	Recurrent Fuzzy Neural Network Identifier

I. Introduction

Parallel robot is a kind of closed multi-loop mechanism and is widely used in industrial fields thanks to its high rigidity, high precision and outstanding weight and load ratio [1], [2]. Parallel Delta robot was adopted in many complex fields, for example: microelectronics [3], [4], medicine [5], [6], logistics intelligence [7], [8], and 3D printing [9], [10]. In those complex applications, the high-precision control of Delta parallel robots has become an important issue for researchers. One of these studies is the PD and LQR controllers presented by Joao Fabian [11]. In this paper, Joao Fabian used embedded NI my RIO hardware programmed in LabView software to compare the two PD controllers and LQR robot trajectory controllers. The nonlinear PID shown by the HAN [12] is used to replace increment scheduling with a nonlinear gain function by introducing a continuous dynamic nonlinear function to achieve better noise cancel-lation and better tracking, this is achieved by the synthesis of a function consists of a linear function close to zero error and a nonlinear function far from zero error. The third technique is using a PD controller with intelligent compensation that is used to solve the trap trajectory tracking for a parallel delta robot with three degrees of freedom presented by Ahmed Chemori [13]. Ahmed Chemori tested a PID controller that performs the

Manuscript received November 16, 2020; revised March 25, 2021.
(*) Corresponding author: Chi-Ngon Nguyen, Can Tho University, Email: ncngon@ctu.edu.vn

doi: 10.18178/ijmerr.10.10.542-552

reference trajectory tracking of a real delta robot with extremely high acceleration up to 100 G as shown in [14]

Recently, several controllers are proposed to control the parallel robot. In [15], R. Anoop and K. Achu designed and verified the performance of the adaptive PID driving a parallel Delta controller. In that work, the controller was able to track the desired trajectory without any discouragement. In [16], Zhang *et al.* studied the problem of dynamic control for redundantly actuated planer 2-DOF parallel manipulator. The work proposed an augmented PD controller based on the forward dynamic compensation control technique, which showed better performance when compared to conventional PD controls. In [17], Hussein Saied et al. proposed different model-based (augmented PD and adaptive feedforward with PD) controllers and non-model-based (PD, PID, and nonlinear PD) controllers for a 4-DOF parallel VELOCE robot. Experimental results indicated that the nonlinear control method can achieve a superior performance [18]. Su et al. developed a nonlinear proportional integral derivative algorithm in link space to achieve high precision tracking control for a general 6-DOF parallel manipulator.

The PID controllers have been successfully developed for 6-DOF robots. However, when changing robot's parameters such as load, input coupling and friction, the PID controllers are difficult to archive control criteria by their fixed parameters. So that, the neural networks and fuzzy logic are applied to improve controlling of Delta robot [19]. The RFNN controllers and identifiers have been developed and applied for nonlinear systems [20-24]. During control process, the RFNN identifier can estimate the object's sensitivity, called Jacobian information. That information is used for online training the RFNN controller. Therefore, by using RFNN, the control technique is flexible to adjust its parameters online adapting to the control conditions.

In this paper, the group of authors makes two main contributions. Firstly, the team set up the motion equation of the parallel delta robot (including the equations of the forward kinetics, the reverse kinetics, the delta robot dynamics) and successfully simulated the motion equations of the delta robots on MATLAB / Simulink. Secondly, the optimal RFNNC-PID is built and successfully simulated to control the trajectory of the delta robot on an ellipse curve and eight curve in MATLAB / Simulink and make sure this trajectory is always stable with the changes of control conditions.

This paper is organized consists of five sections. An introduction is Section I; Section II presents the construct mathematical models of the 3-DOF delta robot; trajectory tracking control of the 3-DOF delta robot is presented in Section III; compare the simulation results of the three controllers are presented in Section IV and Section V is the conclusion.

II. CONSTRUCT MATHEMATICAL MODELS OF THE 3-DOF DELTA ROBOT

A. *Kinetic Models of the 3-DOF Delta Robot*

The dynamics of the delta robot are shown in Fig. 1 which consists of two equal triangles (fixed plate and movable plate).

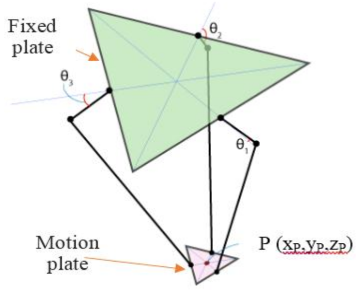


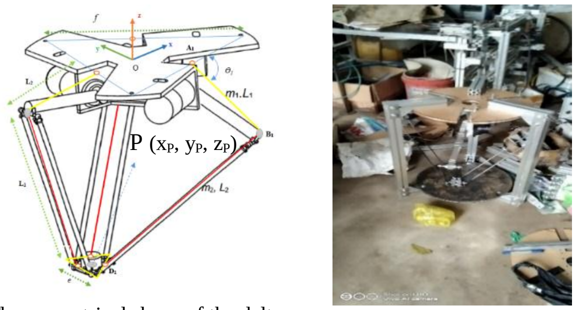
Figure 1. Rotation angle and position of delta robot.

The common angles are $\theta_1, \theta_2, \theta_3$, the point P is the end point location with coordinates (x_P, y_P, z_P). To calculate the inverse kinetics of delta robot, we have to know in advance the position of the end point P(x_P, y_P, z_P), from there we go back to the opposite angle θ_1, θ_2, θ_3. Conversely, to find the forward dynamics of delta robot, we have to know the angles in advance, and then we find the endpoint position P (x_P, y_P, z_P)

1) *Reverse kinetics of delta robot*

The reverse kinetics in this paper require to obtain the desired angular position of the actuator given the desired endpoint of the effect in Cartesian space. This is a geometrical method, so weight and moment of inertia are not considered during modeling.

The physical geometry and the real robotic delta model are shown in Fig. 2



The geometrical shape of the delta robot

Realistic delta robot model made by authors

Figure 2. The physical geometry and the real robotic delta model.

In Fig. 2: *R* is the radius of the fixed plate, *r* is the radius of the moving plate, *f* represents the length of the equilateral triangle of the upper platform, *e* is the length of the equilateral side of the lower-moving disc, L_1 the upper leg length, L_2 length of the lower leg, $P(x_P, y_P, z_P)$ is the center of the low-moving disc, $D_1(x_{D_1}, y_{D_1}, z_{D_1})$ is the midpoint of the equilateral triangle of the lower-moving disc, and D_1 is the connection point of the lower leg to the motion plate, 0 (0,0,0) is the center of the top fixed plate, the point A_1 is the midpoint of the equilateral triangle of the top fixed plate, and A_1 is the connection point of the upper leg to the fixed plate. The angles θ_1, θ_2, θ_3, are the driving angles of the three mechanical arms, point B_1 is the intersection point between the upper leg and the lower leg of the delta robot. Due to the structure of the delta robot, A_1B_1 can only rotate around the *YZ* axis to form a circle with the center point A_1 and radius L_1. In contrast to A_1B_1, point D_1 is seen as a composite joint, meaning that D_1B_1 can rotate freely depending on point D_1 forming a sphere with center D_1 and radius L_2.

The intersections between a sphere and a circle are shown in Fig. 3

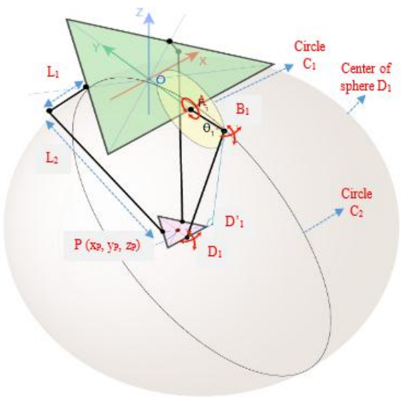


Figure 3. The geometric intersection between a sphere and a circle.

In the Fig. 3 points D'_1 is the projection of point D_1 on the *YZ* plane. The point B_1 can be found at the intersection point of two circles C_1 and C_2 with the centers are A_1 and D'_1 respectively, and the radius L_1 and D'_1B_1 respectively, if we find B_1 then we can Calculates the rotation angle of the robot and B_1 is found the following circle formulas

Equation of the circle C_1:

$$\left(x_{B_1}-y_{A_1}\right)+\left(y_{B_1}-y_{A_1}\right)^2+\left(z_{B_1}-z_{A_1}\right)^2=L_1^2 \tag{1}$$

With points $A_1\left(0,\dfrac{f}{2\sqrt{3}},0\right)$ and $B_1\left(0,y_{B_1},z_{B_1}\right)$ are found from Fig. 3, then deduced:

$$\left(y_{B_1}-\frac{f}{2\sqrt{3}}\right)^2+z_{B_1}^2=L_1^2$$

Equation of the circle C_2:

$$\left(y_{B_1}-y_{D'_1}\right)^2+\left(z_{B_1}-z_{D'_1}\right)^2=\left(D'_1B_1\right)^2=L_2^2-\left(D'_1D_1\right)^2 \tag{2}$$

With points $D'_1\left(x_P,y_P-\dfrac{e}{2\sqrt{3}},z_P\right)$ and $D'_1D_1=x_P$, then deduced:

$$\left(y_{B_1}-y_P+\frac{e}{2\sqrt{3}}\right)^2+\left(z_{B_1}-z_P\right)^2=L_2^2-x_P^2$$

We need to find the point $B_1\left(0,y_{B_1},z_{B_1}\right)$ through equations (1) and (2), and if we know in advance the location of the endpoint $P\left(x_P,y_P,z_P\right)$, then we find the angle θ_1 as shown in Fig. 4

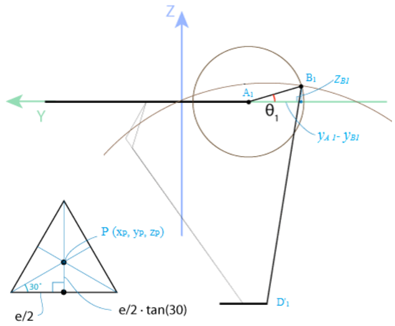


Figure 4. The YZ-axis and basic dimensions for finding angle θ_1.

From Fig. 4 we have:

$$\theta_1=\arctan\left[\frac{z_{B_1}}{\left(y_{A_1}-y_{B_1}\right)}\right] \tag{3}$$

Solving equations (1) and (2) and then substituting (3) we find the angle θ_1.

Due to the algebraic simplicity is a good choice of the reference frame: A_1B_1 joint moves in the *YZ*-axis, so we can completely ignore the *X* axis. To take advantage of this for the rest we find the angle θ_2 and θ_3, and since the symmetry of the delta robot is used, let's first rotate the system coordinates in the *XY* plane around the *Z*-axis through an angle of -120 degrees counterclockwise as shown in Fig. 5 to get the angle θ_2, and we continue to rotate the system coordinates in the *XY* plane around the *Z*-axis through an angle of +120 degrees clockwise as shown in Fig. 5 so we get the angle θ_3

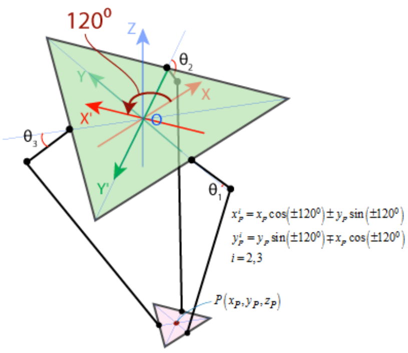


Figure 5. Symmetrical rotation and coordinates of delta robot.

Rotate anticlockwise at an angle -120^0 then $\theta_1\rightarrow\theta_2$; $A_1\rightarrow A'_1$ and $B_1\rightarrow B'_1$

Rotate clockwise at an angle +120^0 then $\theta_1\rightarrow\theta_3$; $A_1\rightarrow A''_1$ and $B1\rightarrow B''_1$

We have the following axis conversion formula:

$$\begin{aligned} x_P^i &= x_P\cos\left(\pm120^0\right)\pm y_P\sin\left(\pm120^0\right)\\ y_P^i &= y_P\sin\left(\pm120^0\right)\mp x_P\cos\left(\pm120^0\right)\\ z_P^i &= z_P \end{aligned} \tag{4}$$

With i = 2, 3

From the axis conversion formula we deduce the formula of angle θ_2 as follows:

$$\begin{aligned} x_P^2 &= x_P \cos\left(-120^0\right) - y_P \sin\left(-120^0\right) \\ y_P^2 &= y_P \sin\left(-120^0\right) + x_P \cos\left(-120^0\right) \\ z_P^2 &= z_P \end{aligned} \tag{5}$$

With i = 2, Similar to finding angle θ_1, we substitute (5) into (1) and (2) and solve the solution for solution, then we replace the solution we just found in (6) we find angle θ_2 :

$$\theta_2 = \arctan\left[\frac{z_{B'_1}}{\left(y_{A'_1} - y_{B'_1}\right)}\right] \tag{6}$$

We have the following formula for changing the axis of angle θ_3 :

$$\begin{aligned} x_P^3 &= x_P \cos\left(120^0\right) + y_P \sin\left(120^0\right) \\ y_P^3 &= y_P \sin\left(120^0\right) - x_P \cos\left(120^0\right) \\ z_P^3 &= z_P \end{aligned} \tag{7}$$

With i = 3

We substitute (7) into (1) and (2) and solve the solution for solution, then we replace the solution we just found in (8) we find angle θ_3

$$\theta_3 = \arctan\left[\frac{z_{B''_1}}{\left(y_{A''_1} - y_{B''_1}\right)}\right] \tag{8}$$

Constructing the inverse kinetic block of the delta robot in Simulink by giving the endpoint position $P\left(x_P, y_P, z_P\right)$, we will find the angles θ_1, θ_2, θ_3 of the delta robot through the formulas (3), (6) and (8).

2) The forward kinetics of delta robot

A forward motion is used to reach the end position using the upper leg angles. The representative geometry of a robot leg is shown in Fig. 6 [25]

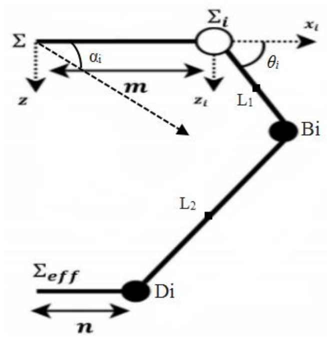


Figure 6. Geometrical parameters of a robotic delta leg.

From Fig. 7, we have a global variable like this:

$$\left\|\overline{\Sigma\Sigma_i}\right\| = m\,, \quad \left\|\overline{\Sigma_{eff}\, D_i}\right\| = n$$

$$\left\|\overline{B_i D_i}\right\| = L_2\,, \quad \left\|\overline{\Sigma_i\, B_i}\right\| = L_1$$

where point i ∈ {1, 2, 3} refers to pins on 1, 2, 3 and *Σ* is the global reference system, *Σi* is the reference system for the above pins, *Σeff* is the reference system for the end points, *Bi* is the connection point between the upper and lower legs, *Di* is the connection point between the lower leg and the fixed plate, θ_i are the angles of the upper legs and α_i are the angles separating each upper leg and the *XY* plane.

Lower leg length can be calculated using the following vector relation:

$$\overline{D_i B_i} = \overline{D_i \Sigma} - \overline{B_i \Sigma} \tag{9}$$

Equation (9) can be written in terms of the geometrical length of the robot and the position *Di* with due the correlation to reference *Σ*, as follows:

$$D_i B_i = R_{\Sigma_i}\begin{bmatrix} L_1\cos\left(\theta_i\right) \\ 0 \\ L_1 \sin\left(\theta_i\right)\end{bmatrix} + \begin{bmatrix} \Delta_n \cos\left(\alpha_i\right) \\ \Delta_n \sin\left(\alpha_i\right) \\ 0\end{bmatrix} - \begin{bmatrix} x_{D_i} \\ y_{D_i} \\ z_{D_i}\end{bmatrix} \tag{10}$$

Inside, Δ_n = m - n and R_{Σ_i} is the rotation matrix between the Σ_i and *Σ* of the reference system

The R_{Σ_i} contained in the equation (10) is bound by the following length:

$$\left\|D_i B_i\right\|^2 = L_2^2\text{, with } i = 1,2,3 \tag{11}$$

Equation (11) is rewritten as follows:

$$\left(x_{D_i} - x_{B_i}\right)^2 + \left(y_{D_i} - y_{B_i}\right)^2 + \left(z_{D_i} - z_{B_i}\right)^2 = L_2^2 \tag{12}$$

With *i* = 1, 2, 3

Substituting equation (12) for (10) gives the following formula:

$$\begin{aligned} x_{B_i} &= \left(\Delta_n + L_1\cos\left(\theta_i\right)\right)\times\cos\left(\alpha_i\right) \\ y_{B_i} &= \left(\Delta_n + L_1\cos\left(\theta_i\right)\right)\times\sin\left(\alpha_i\right) \\ z_{B_i} &= L_1 \sin\left(\alpha_i\right) \end{aligned} \tag{13}$$

From equation (13), if we give the angles θ_i and α_i then we will find the position $B_1\left(x_{B_1}, y_{B_1}, z_{B_1}\right)$ of delta robot. After finding the point $B_1\left(x_{B_1}, y_{B_1}, z_{B_1}\right)$ from equation (13), then we replace equation (12) solve equation finding for point $D_1\left(x_{D_1}, y_{D_1}, z_{D_1}\right)$, we deduce the end point location $P\left(x_P, y_P, z_P\right)$

Constructing the forward kinetic block of the delta robot in Simulink by giving the corners θ_1, θ_2, θ_3 of the delta robot, we will find the endpoint position $P\left(x_P, y_P, z_P\right)$ of the delta robot through the formulas (12) and (13).

B. Dynamic Model of a 3-DOF Delta Robot

The best model for a 3-DOF delta robot is a system of rigid bodies connected by joints. The parallelogram mechanisms that connect the driving links to the mobile platform are modeled as homogeneous rods with universal and spherical joints at two ends. By using this model, the robot is seen as a multibody system with seven bodies: three legs each leg having two links and the mobile platform. The calculating models are shown in Fig. 7 [21], [27].

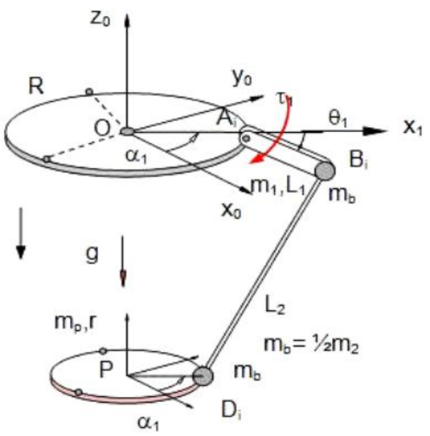


Figure 7. Calculating models of a 3-DOF delta robot [21].

Putting θ_i, i = 1, 2, 3 be the driving angles of the actuated links; α_1, i = 1, 2, 3 be the passive angles that determine the position of the connecting rods; and $P\left(x_P, y_P, z_P\right)^T$ be the position of the center of mass of the mobile platform. Hence, the position of robot is determined by the generalized coordinates:

$$q = \left[\theta_1\ \theta_2\ \theta_3\ x_P\ y_P\ z_P\right]$$

Set the differential equation of motion of the 3-DOF delta robot

The motion equation is established by using the Lagrange formula:

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_k}\right) - \frac{\partial T}{\partial q_k} = Q_k - \sum_{i=1}^{r} \lambda_i \frac{\partial f_i}{\partial q_k} (k = 1,2, \ldots., m) \quad (14)$$

where q_k is the extrapolation coordinates of the robot, f_i is the linking equations, Q_k is the extrapolation force, λ_i is the Lagrange factor. With this model, the vector of extrapolation coordinates $\theta \in R^6$ and the number of associated equations is three, so m = 6, r = 3. We divide the forces acting on the robot into potential forces and forces without potential energy, the extrapolation force Q_k is calculated as follow

$$Q_k = -\frac{\partial \pi}{\partial q_k} + Q_k^{np} \quad (15)$$

In it Q_k^{np} are extrapolation forces corresponding to forces that are not possible. Virtual power of extrapolation is not so:

$$\delta A = \tau_1 \delta\theta_1 + \tau_2 \delta\theta_2 + \tau_3 \delta\theta_3 \quad (16)$$

So we have: $Q_1^{np} = \tau_1, Q_2^{np} = \tau_2, Q_3^{np} = \tau_3$, cases $Q_k^{np} = 0$ with k = 4, 5, 6. Substituting kinetic expressions, potentials and equations into equation (14), we get the motion equation of the robot as the system. The differential equation-algebra is as follows:

$$\begin{aligned}(I_{Iy} + m_b l_1^2)\ddot{\theta}_1 &= g l_1 \left(\tfrac{1}{2} m_1 + m_b\right) \cos\theta_1 + \tau_1 \\ &-2\lambda_1 l_1 \begin{pmatrix} \sin\theta_1 (R-r) - \cos\alpha_1 \sin\theta_1 x_p \\ - \sin\alpha_1 \sin\theta_1 y_p - \cos\theta_1 z_p \end{pmatrix}\end{aligned} \quad (17)$$

$$\begin{aligned}(I_{Iy} + m_b l_1^2)\ddot{\theta}_2 &= g l_1 \left(\tfrac{1}{2} m_1 + m_b\right) \cos\theta_2 + \tau_2 \\ &-2\lambda_1 l_1 \begin{pmatrix} \sin\theta_2 (R-r) - \cos\alpha_2 \sin\theta_2 x_p \\ - \sin\alpha_2 \sin\theta_2 y_p - \cos\theta_2 z_p \end{pmatrix}\end{aligned} \quad (18)$$

$$\begin{aligned}(I_{Iy} + m_b l_1^2)\ddot{\theta}_3 &= g l_1 \left(\tfrac{1}{2} m_1 + m_b\right) \cos\theta_3 + \tau_3 \\ &-2\lambda_3 l_1 \begin{pmatrix} \sin\theta_3 (R-r) - \cos\alpha_3 \sin\theta_3 x_p \\ - \sin\alpha_3 \sin\theta_3 y_p - \cos\theta_3 z_p \end{pmatrix}\end{aligned} \quad (19)$$

$$\begin{aligned}&(m_p + 3m_b)\ddot{x}_p = \\ &\quad -2\lambda_1\left(\cos\alpha_1 (R-r) + l_1 \cos\alpha_1 \cos\theta_1 - x_p\right) \\ &\quad -2\lambda_2\left(\cos\alpha_2 (R-r) + l_1 \cos\alpha_2 \cos\theta_2 - x_p\right) \\ &\quad -2\lambda_3\left(\cos\alpha_3 (R-r) + l_1 \cos\alpha_3 \cos\theta_3 - x_p\right)\end{aligned} \quad (20)$$

$$\begin{aligned}&(m_p + 3m_b)\ddot{y}_p = \\ &\quad -2\lambda_1\left(\sin\alpha_1 (R-r) + l_1 \sin\alpha_1 \cos\theta_1 - y_p\right) \\ &\quad -2\lambda_2\left(\sin\alpha_2 (R-r) + l_1 \sin\alpha_2 \cos\theta_2 - y_p\right) \\ &\quad -2\lambda_3\left(\sin\alpha_3 (R-r) + l_1 \sin\alpha_3 \cos\theta_3 - y_p\right)\end{aligned} \quad (21)$$

$$\begin{aligned}&(m_p + 3m_b)\ddot{z}_p = \\ &\quad -(3m_b + m_p)g + 2\lambda_1\left(z_p + l_1 \sin\theta_1\right) \\ &\quad +2\lambda_2\left(z_p + l_1 \sin\theta_2\right) + 2\lambda_3\left(z_p + l_1 \sin\theta_3\right)\end{aligned} \quad (22)$$

$$\begin{aligned}&l_1^2 - \left(\cos\alpha_1 (R-r) + l_1 \cos\alpha_1 \cos\theta_1 - x_p\right)^2 \\ &\quad - \left(\sin\alpha_1 (R-r) + l_1 \sin\alpha_1 \cos\theta_1 - y_p\right)^2 \\ &\quad - \left(l_1 \sin\theta_1 + z_p\right)^2 = 0\end{aligned} \quad (23)$$

$$\begin{aligned}&l_2^2 - \left(\cos\alpha_2 (R-r) + l_1 \cos\alpha_2 \cos\theta_2 - x_p\right)^2 \\ &\quad - \left(\sin\alpha_2 (R-r) + l_1 \sin\alpha_2 \cos\theta_2 - y_p\right)^2 \\ &\quad - \left(l_1 \sin\theta_2 + z_p\right)^2 = 0\end{aligned} \quad (24)$$

$$\begin{aligned}&l_3^2 - \left(\cos\alpha_3 (R-r) + l_1 \cos\alpha_3 \cos\theta_3 - x_p\right)^2 \\ &\quad - \left(\sin\alpha_3 (R-r) + l_1 \sin\alpha_3 \cos\theta_3 - y_p\right)^2 \\ &\quad - \left(l_1 \sin\theta_3 + z_p\right)^2 = 0\end{aligned} \quad (25)$$

From the motion equations of the parallel robot (17) to (25), according to the proposal [21], [26], [27] we will construct the model of the 3-DOF Delta robot in MATLAB/Simulink.

III. Trajectory Tracking Control of the 3-DOF Delta Robot

A. Delta Robot Control Using a Single Neuron PID

The control system for a robot arm is presented in Fig. 8

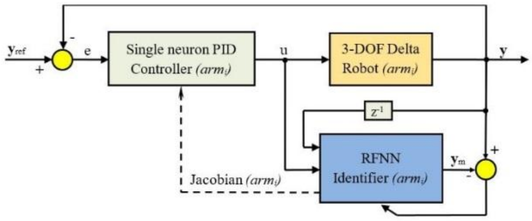


Figure 8. Controller structure [21].

The reference signal yref is the rotation angle of the upper three arms of the robot is sent to three adders giving three error signals e_1, e_2, e_3 to three PID controllers one neuron, the output of three the controller was inserted into Tau_1, Tau_2, Tau_3 of the 3-DOF Delta Robot. At the same time, three RFNN identifiers will observe three outputs of three single-neuron PID controllers and three output signals y (theta1, theta2, theta3) of delta robot, each RFNN identifier will train the parameter Jacobian returned three single-neuron PID controllers to continuously update the K_P, K_D, and K_I parameters of the three controllers to control the delta robot following the reference signal y_{ref} presented by the authors in [21]

B. The 3-DOF Delta Robot Control Using RFNNC-PID

1) The 3-DOF delta robot RFNN identifier

The identifier is shown in Fig. 9 that consists of 4 layers, with a 2-node input layer, a 10-node fuzzy layer, a 25-node fuzzy rule, and a 1-node output layer [28]-[32].

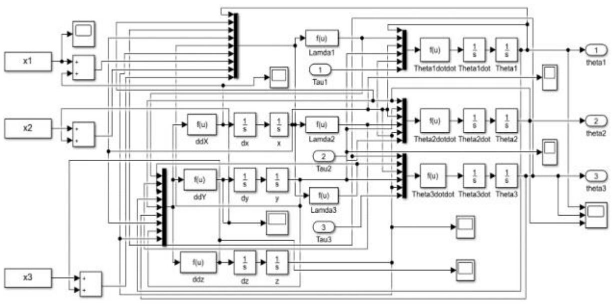


Figure 9. Delta robot model in MATLAB/Simulink.

Layer 1 (Input layer): This layer takes the input variables and passes the input values to the next layer. Feedback lines are added in this layer to embed time relations into the network. The output of layer 1 is represented as (26):

$$O_i^1(k) = x_i^1(k) + \theta_i^1 O_i^k(k-1), \quad i = 1,2 \qquad (26)$$

With θ_i^1 is the connection weight at the current time k. The input of the corresponding RFNN is the current control signal and the past output of the response:

$$x_1^1(k) = u(k), x_2^1(k) = y(k-1) \qquad (27)$$

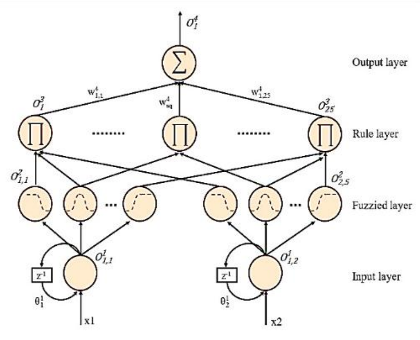


Figure 10. Structure of four-layer RFNN.

Layer 2 (Fuzzy layer): This layer consists of (2x5) nodes, each node representing a related function of the Gaussian form with mean value m_{ij} and standard deviation σ_{ij}, and defined as (28).

$$O_{ij}^2(k) = \exp\left\{ -\frac{\left(O_i^1(k) - m_{ij}\right)^2}{\left(\sigma_{ij}\right)^2} \right\}, i = 1,2;\ j = 1,2,...,5 \qquad (28)$$

At each node on the fuzzy layer, there are 2 parameters that are automatically adjusted during the online training of the RFNN identifier, that is m_{ij} and σ_{ij}.

Layer 3 (Rule layer): This layer provides fuzzy inferences. Each node corresponds to a fuzzy rule. The link before each node represents the prerequisites of the respective rule. The q_{th} the fuzzy law can be described:

$$O_q^3(k) = \prod_i O_{iq_i}^2(k),\ i = 1,2,...,5; q_i = 1,2,...,5 \qquad (29)$$

Layer 4 (Output layer): This layer includes 1 linear neuron with the defined output as follows:

$$O_i^4(k) = \sum_j w_{ij}^4 O_j^3(k),\ i = 1;\ j = 1,2,...,25 \qquad (30)$$

where w_{ij}^4 is the connecting weights from 3rd layer to 4th layer. The output of this layer is also the output of the RFNN:

$$O_1^4(k) = y_m(k) = \hat{f}[x_1(k), x_2(k)] = \hat{f}[u(k), y(k-1)] \qquad (31)$$

The performance RFNN identifier training is based on a cost function in (32) [20]:

$$E_I(k) = \frac{1}{2}\left[y(k) - y_m(k)\right]^2 = \frac{1}{2}\left[y(k) - O_{I1}^4(k)\right]^2 \qquad (32)$$

The RFNN's weights are adjusted according to (33) [20]:

$$\begin{aligned} W_I(k+1) &= W_I(k) + \Delta W_I(k) \\ &= W_I(k) + \eta_I\left(-\frac{\partial E_I(k)}{\partial W_I}\right) \end{aligned} \qquad (33)$$

where, η_I is the learning rate, $W_I=[\theta_I, m_I, \sigma_I, w_I]^T$ is the weight vector of RFNN updated during training the RFNN. And subscript I presents the RFNN identifier, called RFNNI.

Given $e_I(k)=y(k)-y_m(k)$ is the error between plant's output and RFNN's output, then the gradient $E_I(.)$ with respect to W_I is determined as (34):

$$\frac{\partial E_I(k)}{\partial W_I}=-e_I(k)\frac{\partial y_m(k)}{\partial W_I}=-e_I(k)\frac{\partial O_{I1}^4(k)}{\partial W_I} \tag{34}$$

The weight of each RFNNI network layer is updated as follows [21], [32]:

$$w_{Iij}^4(k+1)=w_{Iij}^4(k)+\eta_I^{w_I}e_I(k)O_{Ii}^3 \tag{35}$$

$$m_{Iij}(k+1)=m_{Iij}(k)+\eta_I^{m_I}\sum_k e_I(k)w_{Iik}^4O_{Ik}^3\frac{2\left[O_{Iij}^1(k)-m_{Iij}\right]}{\left(\sigma_{Iij}\right)^2} \tag{36}$$

$$\sigma_{Iij}(k+1)=\sigma_{Iij}(k)+\eta_I^{\sigma_I}\sum_k e_I(k)w_{Iik}^4O_{Ik}^3\frac{2\left[O_{Iij}^1(k)-m_{Iij}\right]^2}{\left(\sigma_{Iij}\right)^3} \tag{37}$$

$$\theta_{Ii}^1(k+1)=\theta_{Ii}^1(k)+\eta_I^{\theta_I}\sum_k e_I(k)w_{Iik}^4O_{Ik}^3\frac{(-2)\left[O_{Iij}^1(k)-m_{Iij}\right]O_{Iij}^1(k-1)}{\left(\sigma_{Iij}\right)^2} \tag{38}$$

In addition, to estimate the output of the model $y_m(k)$, the RFNNI must also estimate Jacobian information $\partial y(k)/\partial u(k)$, determined as follows [21], [31], [32]:

$$\frac{\partial y(k)}{\partial u(k)}=\sum_q w_{Iij}^4\left\{\sum_s\frac{\partial O_{Iq}^3}{\partial O_{Iqs}^2}\frac{(-2)\left[O_{Iij}^1(k)-m_{Iij}\right]}{\left(\sigma_{Iij}\right)^2}\right\} \tag{39}$$

2) Building controller of RFNNC-PID

In this part, for controlling a robot arm, a RFNN controller (RFNNC) is combined with a traditional PID controller with its structure is shown in Fig. 11 [21]-[23].

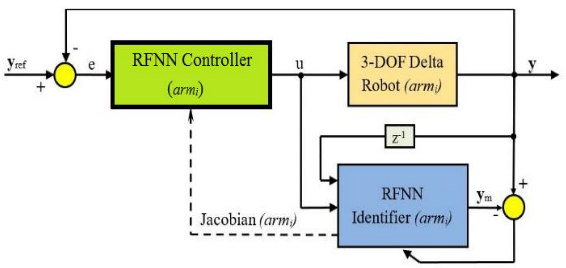


Figure 11 Control system based on RFNNs.

In Fig. 11, the RFNNI is used to identify the model of the robot, based on *y(k-1)* and *u(k)*. That makes the RFNN model is simple and decreases number of neurons. Through training, the RFNNI estimates the output trajectories of the delta robot by (30).

Training performance criterion is defined as (40) [20]:

$$\begin{aligned}E_C(k)&=\frac{1}{2}\left[u_{rfnnc}(k)-u(k)\right]^2\\&=\frac{1}{2}\left[u_{rfnnc}(k)-O_{C1}^4(k)\right]^2\end{aligned} \tag{40}$$

where $u(k)$ is the s^{th} input of the plant, $u_{rfnnc}(k)$ is the s^{th} output of RFNNC. After the initialization process, a gradient-descent-based back-propagation algorithm was employed to adjust the controller parameters (40):

$$\begin{aligned}W_C(k+1)&=W_C(k)+\Delta W_C(k)=\\&W_C(k)+\eta_C\left(-\frac{\partial E_C(k)}{\partial W_C}\right)\end{aligned} \tag{41}$$

In which, η_c is the learning rate, $W_C=[\theta_C, m_C, \sigma_C, w_C]^T$ is the weight vector of the RFNNC updated during training. Subscript *C* presents the RFNNC.

Given $e_C(k)=urfnn(k)-u(k)$, the gradient $E_C(.)$ with respect to W_C is defined as (42):

$$\frac{\partial E_C(k)}{\partial W_C}=-e_C(k)\frac{\partial u_{rfnnc}(k)}{\partial W_C}=-e_C(k)\frac{\partial O_1^4(k)}{\partial W_C} \tag{42}$$

The weight of each RFNNC network layer is updated as follows [21], [30]:

$$\begin{aligned}w_{Cij}^4(k+1)&=w_{Cij}^4(k)+\eta_C^{w}\left(-\frac{\partial E_C(k)}{\partial w_{Cij}^4}\right)\\&=w_{Cij}^4(k)+\eta_C^{w_C}u_{pid}(k)e_C(k)O_{Ci}^3\end{aligned} \tag{43}$$

$$\begin{aligned}m_{Cij}(k+1)&=m_{Cij}(k)+\eta_C^{m}\left(-\frac{\partial E_C(k)}{\partial m_{Cij}}\right)\\&=m_{Cij}(k)+\eta_C^{m_C}\sum_k e_C(k)w_{Cik}^4O_{Ck}^3\frac{2\left[O_{Cij}^1(k)-m_{Cij}\right]}{\left(\sigma_{Cij}\right)^2}\end{aligned} \tag{44}$$

$$\begin{aligned}\sigma_{Cij}(k+1)&=\sigma C_{ij}(k)+\eta_C^{\sigma}\left(-\frac{\partial E_C(k)}{\partial \sigma_{Cij}}\right)\\&=\sigma_{Cij}(k)+\eta_C^{\sigma_C}\sum_k e_C(k)w_{Cik}^4O_{Ck}^3\frac{2\left[O_{Cij}^1(k)-m_{Cij}\right]^2}{\left(\sigma_{Cij}\right)^3}\end{aligned} \tag{45}$$

$$\begin{aligned}\theta_{Ci}^1(k+1)&=\theta_{Ci}^1(k)+\eta_C^{\theta}\left(-\frac{\partial E_C(k)}{\partial \theta_{Ci}^1}\right)=\theta_{Ci}^1(k)+\\&\eta^{\theta_C}\sum_k e_C(k)w_{Cik}^4O_{Ck}^3\frac{(-2)\left[O_{Cij}^1(k)-m_{Cij}\right]O_{Cij}^1(k-1)}{\left(\sigma_{Cij}\right)^2}\end{aligned} \tag{46}$$

IV. Compare the Simulation Results of the Three Controllers

A. Simulation Parameters of Three Controllers

The specifications of the real robot that the authors have built are shown in Table I.

TABLE I. DELTA ROBOT MECHANICAL SPECIFICATIONS

Symbol	Value	Unit
α_1	0	rad/s
α_2	$2\pi/3$	rad/s
α_3	$4\pi/3$	rad/s
R	138.9	mm
r	25	mm
f	481.07	mm
e	43.3	mm
L_1	250	mm
L_2	544	mm
m_1	0.258	kg
$m_2=2m_b$	0.088	kg
m_p	0.044	kg

The parameters of the Single Neural PID Algorithms, and Recurrent Fuzzy Neural Network Controller, and Recurrent Fuzzy Neural Network Identifier are randomly initialized.

B. Simulation Results

The MATLAB/Simulink control system for a 3-DOF delta robot with the desired trajectory as an ellipse curve (47).

$$\begin{aligned} x_d &= 0.19\sin(\pi t)+0.3 \\ y_d &= 0.12\cos(\pi t)+0.2 \\ z_d &= -0.8 \end{aligned} \tag{47}$$

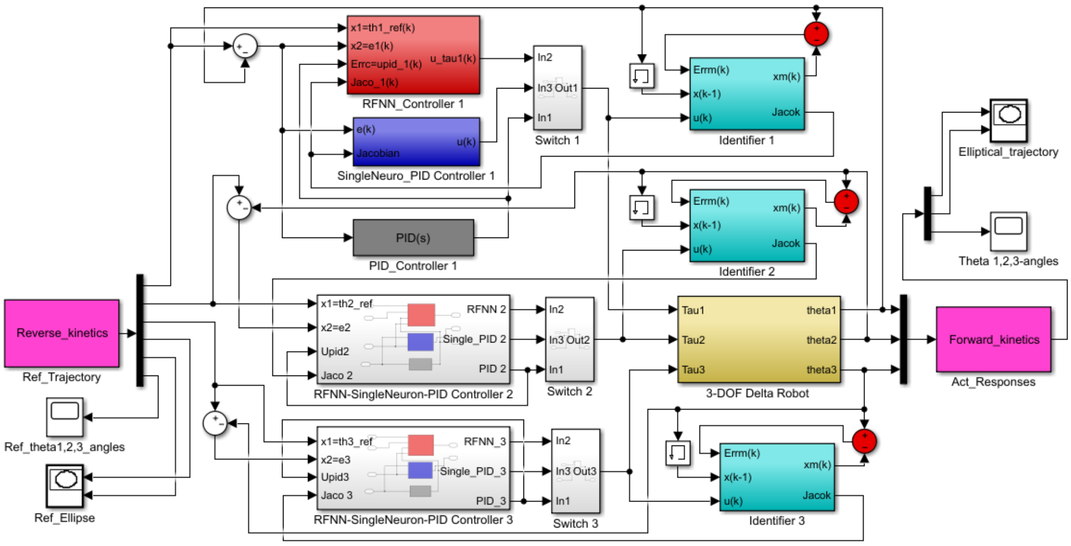


Figure 12. The 3-DOF Delta Robot controller in MATLAB/Simulink.

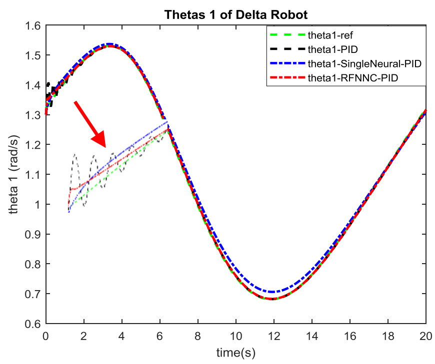


Figure 13. The angles of theta 1.

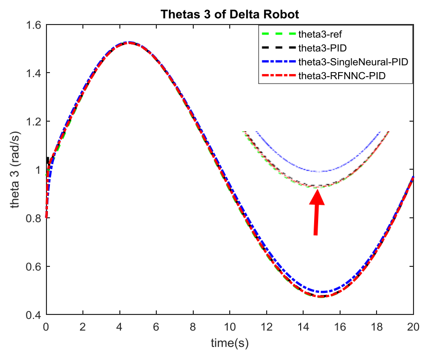


Figure 15. The angles of theta 3.

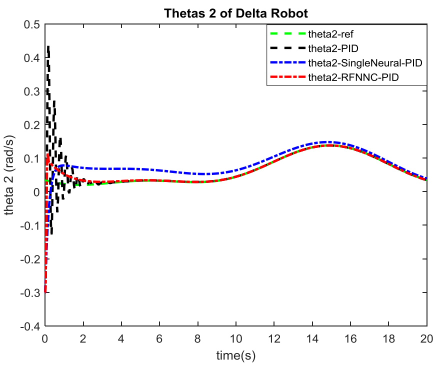


Figure 14. The angles of theta 2.

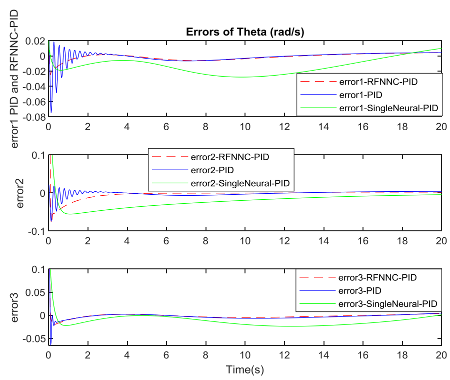


Figure 16. Errors of responses.

Fig. 13, Fig. 14, Fig. 15 and Fig. 16 comparisons angles theta of three traditional PID, Single-Neuron PID controller and RFNN-PID controller.

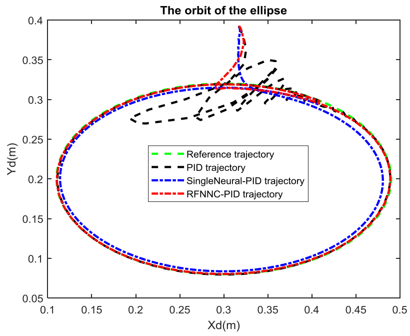


Figure 17. Trajectory tracking of traditional PID and Single Neuron PID and RFNN-PID controller.

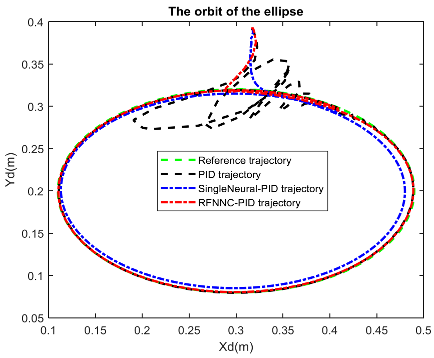


Figure 18. Responses when changing load from 4.71 Kg to 7.05 Kg.

The response of the traditional PID, Single neuron PID and the RFNN-PID controllers are presented in Fig. 13 – Fig. 18 including load changed. Simulation results show that the RFNN-PID controller is better than Single neuron PID controller, with the setting time is about 3. 8±0. 1 seconds, and the steady-state error is eliminated.

The MATLAB/Simulink control system for a 3-DOF delta robot with the desired trajectory as a Fig. 8 is shown (48).

$$\begin{aligned} x_d &= 0.15\sin(2\pi t)+0.3 \\ y_d &= 0.15\sin(2\pi t)*\cos(2\pi t)+0.\ 2 \\ z_d &= -0.7 \end{aligned} \tag{48}$$

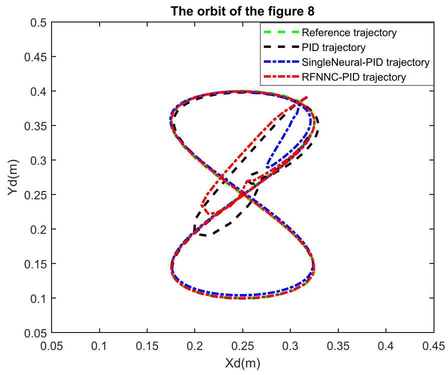


Figure 19. Trajectory tracking of the Fig. 8.

And the control criteria of PID controller, Single Neuron PID and RFNNC-PID controller are presented in Table II. The results show that the proposed controller must be better than the Single Neuron PID controller and traditional PID controller.

TABLE II. SYSTEM CONTROL QUALITY STANDARDS

Response	**PID**			**Single Neuron PID**			**RFNN-PID**		
	Rise time (s)	Overshoot (%)	Settling time (s)	Rise time (s)	Overshoot (%)	Settling time (s)	Rise time (s)	Overshoot (%)	Settling time (s)
Theta 1	3.134	1.873	5.6±0.1	3.119	1.799	4.2±0.1	2.942	0.299	3.9±0.1
Theta 2	2.838	1.882	6.9±0.1	2.889	1.992	3.6±0.1	3.346	1.953	3.8±0.1
Theta 3	2.698	1.859	4.2±0.1	2.705	1.973	3.9±0.1	2.447	0.409	3.6±0.1

V. CONCLUSION

In this paper, the RFNN-PID controller is proposed to control the 3-DOF delta robot. This controller guarantees the real trajectory converges the reference trajectory with finite time. The simulation results show that the proposed controller is better than the single neuron PID controller and the traditional PID controller. The proposed RFNNC-PID algorithm is stable while changing the control conditions such as the speed and load of the delta robot increasing. In further work, the proposed controller will be experimenting with the real robot model.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Mr. Le Minh Thanh, first author, is a PhD student under supervising of Assoc. Prof. Dr. Chi-Ngon Nguyen (last and corresponding author), who has prepared the manuscript. Mr. Luong Hoai Thuong and Mr. Pham Thanh Tung, 2nd and 3rd authors have contributed on model simulation. Dr. Cong-Thanh Pham, 4th author, has contributed on writing correction. Assoc. Prof. Dr. Chi-Ngon Nguyen is chief of research group, who has supervised for this study and finalized this paper.

REFERENCES

[1] G. Gao, M. Ye, and M. Zhang, "Synchronous robust sliding mode control of a parallel robot for automobile electro-coating conveying," *IEEE Access*, vol. 7, pp. 85838-85847, 2019.

[2] S. Qian, B. Zi, D. Wang, and Y. Li, “Development of modular cable-drivenn parallel robotic systems,” *IEEE Access*, vol. 7, pp. 5541–5553, 2019.

[3] J. E. Correa, J. Toombs, N. Toombs, and P. M. Ferreira, “Laminated micromachine: Design and fabrication of a flexure-based Delta Robot,” *J. Manuf. Processes*, vol. 24, pp. 370–375, Oct. 2016.

[4] X. J. Liu, J. I. Jeong, and J. Kim, “A three translational DOFs parallel cube manipulator,” *Robotica*, vol. 21, no. 6, pp. 645–653, Dec. 2003.

[5] K. C. Olds, “Global indices for kinematic and force transmission performance in parallel robots,” *IEEE Trans. Robot.*, vol. 31, no. 2, pp. 494–500, Apr. 2015.

[6] G. Yedukondalu, A. Srinath, and J. S. Kumar, “Mechanical chest compression with a medical parallel manipulator for cardiopulmonary resuscitation,” *Int. J. Med. Robot. Comput. Assist. Surgery*, vol. 11, no. 4, pp. 448–457, Oct. 2014.

[7] G. Borchert, M. Battistelli, G. Runge, and A. Raatz, “Analysis of the mass distribution of a functionally extended delta robot,” *Robot. Comput-Integr. Manuf.*, vol. 31, pp. 111–120, Feb. 2015.

[8] R. Kelaiaia, “Improving the pose accuracy of the Delta robot in machining operations,” *Int. J. Adv. Manuf. Technol*, vol. 91, no. 5–8, pp. 2205–2215, July 2017.

[9] E. Rodriguez, C. Riaño, A. Alvares, and R. Bonnard, “Design and dimensional synthesis of a linear delta robot with single legs for additive manufacturing,” *J. Brazilian Soc. Mech. Sci. Eng.*, vol. 41, no. 11, p. 536, Nov. 2019.

[10] K. He, Z. Yang, Y. Bai, J. Long, and C. Li, “Intelligent fault diagnosis of Delta 3D printers using attitude sensors based on support vector machines,” *Sensors*, vol. 18, no. 4, p. 1298, Apr. 2018.

[11] J. Fabian, C. Monterrey and R. Canahuire, “Trajectory tracking control of a 3 DOF delta robot: a PD and LQR comparison,” in *Proc. 2016 IEEE XXIII. Inter. Congress on Electronics, Electrical Engineering and Computing (INTERCON)*, Piura, pp. 1-5, 2016.

[12] J. Han, “From PID to active disturbance rejection control,” *IEEE Transactions on Industrial Electronics*, vol. 56, no. 3, pp. 900-906, 2009.

[13] J. M. E. Hernández, H. Aguilar-Sierra, O. Aguilar-Mejía, A. Chemori, J. Arroyo-Núñez, “An intelligent compensation through B-spline neural network for a delta parallel robot,” in *Proc. 6th Inter. Conf. on Control, Decision and Information Technologies (CoDIT)*, Paris, France, pp. 361-366, 2019.

[14] A. Chemori, G. S. Natal, F. Pierrot, “Control of parallel robots: Towards very high accelerations. SSD,” *Systems, Signals and Devices*, Mar. 2013, Hammamet, Tunisia. pp. 8, ffirmm-00809514 2013.

[15] R. Anoop and K. Achu, “Control technique for parallel manipulator using PID,” *Inter. J. of Engineering Research & Technology*, vol. 5, no. 7, pp. 56–59, 2016.

[16] Y. X. Zhang, S. Cong, W. W. Shang, Z. X. Li, and S. L. Jiang, “Modeling, identification and control of a redundant planar 2-DOF parallel manipulator,” *Inter. J. of Control, Automation, and Systems*, vol. 5, no. 5, pp. 559–569, 2007.

[17] H. Saied, A. Chemori, M. El Rafei, C. Francis, and F. Pierrot, “From non-model-based to model-based control of PKMS: a comparative study,” in *Proc. 1st Inter. Congress for the Advan. of Mechanism, Machine, Robotics and Mechatronics Sciences*, pp.50–64, Beirut Lebanon, 2017.

[18] Y. X. Su, B. Y. Duan, and C. H. Zheng, “Nonlinear PID control of a six-DOF parallel manipulator,” in *IEEE Proc. Control Theory and Applications*, vol. 151, no. 1, pp. 95–102, 2004.

[19] W. Widhiada, T. G. T. Nindhia, and N. Budiarsa, “Robust control for the motion five fingered robot gripper,” *Inter. J. Mecha. Eng. and Robotics Research*, vol. 4, no. 3, pp. 226-232, 2015.

[20] J. K. Liu, “Radial Basis Function (RBF) neural network control for mechanical systems,” *Design, Analysis and Matlab Simulation*. *Springer-Verlag Berlin Heidelberg*, pp. 55-69, 2013.

[21] L. M. Thanh, L. H. Thuong, P. Th. Loc, C. N. Nguyen, “Delta robot control using single neuron PID algorithms based on recurrent fuzzy neural network Identifiers,” *Inter. J. of Mechanical Eng. and Robotics Research*, vol. 9, no. 10, 2020.

[22] S. Slama, A. Errachdi, and M. Benrejeb, “Adaptive PID controller based on neural networks for MIMO nonlinear systems,” *J. of Theoretical and Applied Information Tech*., vol. 97, no. 2, pp. 361–371, 2019.

[23] W. Sun, Y. N. Wang, “A recurrent fuzzy neural network based adaptive control and its application on robotic tracking control,” *Neural Information Processing-Letters and Reviews*, vol. 5, no. 1, 2004.

[24] H. Hasanpour, M. H. Beni, and M. Askari, “Adaptive PID control based on RBF NN for quadrotor,” *Inter. Research J. of Applied and Basic Sciences*, vol. 11, no. 2, pp. 177–186, 2017.

[25] J. Fabian, C. Monterrey, and R. Canahuire, "Trajectory tracking control of a 3 DOF delta robot: A PD and LQR comparison," in *Proc. 2016 IEEE XXIII International Congress on Electronics, Electrical Engineering and Computing (INTERCON)*, Piura, 2016, pp. 1-5.

[26] J. Merlet, *Parallel Robots*, the Netherlands: Kluwer Academic Publishers (2000).

[27] N. D. Dung, “Reverse dynamics of parallel delta space robots,” PhD dissertation on Mechanical Engineering,” *Vietnam National Library*, 2018.

[28] H. Hasanpour, M. H. Beni, and M. Askari: “Adaptive PID control based on RBF NN for quadrotor,” *Inter. Research J. of Applied and Basic Sciences*, vol. 11, no. 2, pp. 177–186, 2017.

[29] S. Slama, A. Errachdi, and M. Benrejeb, “Neural adaptive PID and neural indirect adaptive control switch controller for nonlinear MIMO systems,” *Mathematical Problems in Engineering*, vol. 2019.

[30] C. H. Lee and C. C. Teng, “Identification and control of dynamic systems using recurrent fuzzy neural networks,” *IEEE Transaction on Fuzzy Systems*, vol. 8, no.4, pp. 349-366, 2000.

[31] S. Wei, Z. Lujin, Z. Jinhai, and M. Siyi, “Adaptive control based on neural network,” *Adaptive Control*, Kwanho You (Ed.), InTech 2009.

Le Minh Thanh received a Bachelor of Engineering degree in Electrical and Electronic Engineering at Mekong University in 2006, a Master's degree in Automation at the University of Transport in Ho Chi Minh City in 2011. He is now a lecturer of the Faculty Electrical – Electronics, Vinh Long University of Technical Education.

Luong Hoai Thuong received a Bachelor of Engineering in Control Engineering from Can Tho University in 2009, a Master's degree in Electronic Engineering at Ho Chi Minh City University of Technical Education in 2015. He is currently a lecturer in the Faculty of Electrical - Electronics, Vinh Long University of Technical Education.

Cong-Thanh Pham received the Bachelor & M.Sc. degree in automation and control engineering from the Ho Chi Minh City University of Technology, Vietnam, 2002 & 2006, the Ph.D. received degree at Department of Control Science and Engineering, Huazhong University of Science and Technology (HUST), Wuhan 430074, China, 2014. He is currently with the Faculty of Automatiom Control, Viet Nam Aviation Academy, Ho Chi Minh City. His current research interests include AC motor, Pes

Thanh Tung Pham received degree in Electrical and Electronic Engineering at Mekong University in 2004, a Master's degree in Automation at Ho Chi Minh City University of Transport in 2010. The degree of Ph.D. was award by the Ho Chi Minh City University of Transport, Vietnam, in 2019. Nowadays, he has worked at Vinh Long University of Technical Education.

Chi-Ngon Nguyen received his B.S. and M.S. degree in Electrical Engineering from Can Tho University and Ho Chi Minh City University of Technology, in 1996 and 2001, respectively. The degree of Ph.D. was award by the University of Rostock, Germany, in 2007.

Since 1996, he has worked at the Can Tho University. Currently, he is an associate professor in automation of the Department of Automation Technology. He is working as a position of Dean of the College of Engineering Technology at the Can Tho University. His research interests are intelligent control, medical control, pattern recognition, classifications, speech recognition and computer vision.