iCT Focus

https://www.ictfocus.org

JOURNAL ARTICLE OPEN ACCESS

Quadcopter Stabilization using Neural Network Model from Collected Data of PID Controller

Bat-Erdene Gotov[1], Tengis Tserendondog[2], Lodoiravsal Choimaa[3], Batmunkh Amar[4]

[1]Department of General Science, Mongolian National University of Medical Sciences, Ulaanbaatar, Mongolia,

[2,4]Department of Electronics, Mongolian National University of Science and Technology, Ulaanbaatar, Mongolia

[3]Department of Machine Intelligence, The National University of Mongolia, Ulaanbaatar, Mongolia

e-mail: tengis@must.edu.mn

Abstract

There are a lot of methods for the stabilization of quadcopters and the newest are based on AI. A neural network is a simplified model that imitates the human brain's processes. In the research paper, we present a neural network control model for quadcopter stabilization. A single hidden layer network model was estimated to investigate the dynamics of the UAV. A control system with a classical PID controller was used to train the neural network model. This method is used for examining how the neural network imitates the stabilization of the quadcopter in real flight mode. The novelty of the work was to design of small size 3 layers NN model that runs in real-time in a quadcopter. The PID and machine learning controllers' operation results were compared to each other and shown in the experiment.

Keywords: artificial intelligence; control system; quadrotor; neural network; unmanned aerial vehicle

Received: August 08, 2022.

Accepted: September 05, 2022.

Published: September 29, 2022.

Corresponding author: Tengis Tserendondog

1. INTRODUCTION

In recent years, quadrotors (UAV, quadcopter) emerged widely in commercial and army areas [1, 2, 3]. There are many applications of quadcopter UAVs, such as life-saving operations, surveillance, aerial photography for mapping, an inspection of power lines, traffic monitoring in city areas, crop monitoring and spraying, border patrol, search operations for missing persons and natural disasters [4, 5, 6, 7].

Most quadcopters apply the classical control method, so they are driven by proportional-integral-derivative-PID controllers [1-10]. PID controller method and its usability in different systems are well described in many research papers [1-10]. There are several other algorithms that control quadcopters, such as LQR, Backstepping, Sliding mode, Linearized Feedback, Fuzzy logic, Optimal, L1, H∞, and Adaptive control. A quadcopter control algorithm is compared in [11-15]. Researchers search for more advantageous methods of control, like artificial intelligence algorithms, for example, neural network controllers. Compared to the classical approach, the main advantage of a neural network is the controller's nonlinear character. In this research work, the back-propagation neural network has been used to learn the integrated quadcopter dynamics model. The neural network was trained by a control system with a classical PID controller [16, 17, 18, 19].

In machine learning like supervised learning, each training sample is a pair consisting of input data in our case the angles of the quadcopter, and the desired output data in our case the rotational speed of the propeller. After training based on collected data, the controller is expected to make correct predictions for untrained data.

2. MATHEMATICAL MODEL OF QUADROTOR

This part presents the mathematical model of the quadcopter.

2.1. Quadcopter model

The quadcopter has a "CROSS type" flying configuration (Fig. 1), with two opposite motors rotating CW and the other two motors rotating CCW to equilibrium the torque [20, 21]. The three angles, such as roll, yaw, pitch, and push-up operations, are controlled by modifying the thrusts of the motors using a PWM signal to give the necessary output. Consider a quadcopter UAV with six DOF, and the dynamic of the quadcopter can be separated into two subsystems which include a translational subsystem and a rotational subsystem [20-24].

The quadcopter must be provided with a good inertial measuring sensor to obtain the necessary measurements. This sensor gives the angular rates and accelerations that can be used to calculate velocities and angles [6]. We examine an inertial field and a body-fixed field whose origin is in the center of mass of the quadcopter, as shown in Fig. 1

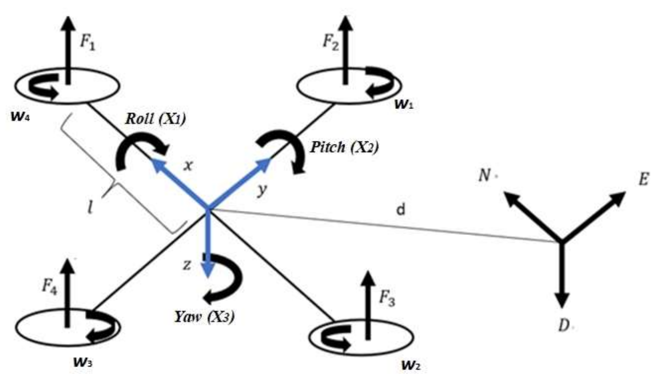


Figure 1. Quadcopter field.

2.2. Quadcopter equations

Using the Newton-Euler method [23, 24] the rotational formulas of motion are generated in the body field shown in equation (1).

$$J\dot{\omega} + \omega \times J\omega + M_g = M_b \tag{1}$$

Here, J-quadcopter's diagonal inertia matrix, ω-angular velocities, M_g-gyroscopic moments due to motors inertia, and M_b-moments acting on the quadcopter. Overall moments acting on the quadcopter become:

$$M_b = \begin{vmatrix} lU_2 \\ lU_3 \\ lU_4 \end{vmatrix} = \begin{bmatrix} lK_f(\omega_1^2 - \omega_3^2) \\ lK_f(\omega_2^2 - \omega_4^2) \\ K_m(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} \tag{2}$$

Here K_f and K_m are the aerodynamic force and moments constant, respectively, ω_i is the angular velocity of *the i-th* motor. Each motor produces a moment M_i in the opposite direction to the direction of rotation of the corresponding rotor *i* and provides an upwards thrust force F_i. Substituting formula (2) into the rotational formula of motion (1), the following relation can be formulated:

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \ddot{\varphi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\varphi} \\ \dot{\varphi} \\ \dot{\varphi} \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{\varphi} \\ \dot{\varphi} \\ \dot{\varphi} \end{bmatrix} + \begin{bmatrix} \dot{\varphi} \\ \dot{\varphi} \\ \dot{\varphi} \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ J_r\omega_r \end{bmatrix} = \begin{bmatrix} lU_2 \\ lU_3 \\ U_4 \end{bmatrix} \tag{3}$$

Rewriting the last formula to get the angular accelerations in terms of the other variables:

$$\ddot{\varphi} = \frac{l}{I_{xx}} U_2 - \frac{J_r}{I_{xx}} \dot{\theta}\omega_r + \frac{I_{yy}}{I_{xx}} \dot{\psi}\dot{\theta} - \frac{I_{zz}}{I_{xx}} \dot{\theta}\dot{\psi} \tag{4}$$

$$\ddot{\theta} = \frac{l}{I_{yy}} U_3 - \frac{J_r}{I_{yy}} \dot{\varphi}\omega_r + \frac{I_{zz}}{I_{yy}} \dot{\varphi}\dot{\theta} - \frac{I_{xx}}{I_{yy}} \dot{\theta}\dot{\psi} \tag{5}$$

$$\ddot{\psi} = \frac{1}{I_{zz}} U_4 - \frac{I_{xx}}{I_{zz}} \dot{\theta}\varphi + \frac{I_{yy}}{I_{zz}} \dot{\varphi}\dot{\theta} \tag{6}$$

Based on Newton's second law the quadcopter translation formulas of motion are derived, and they are obtained from the Earth's inertial field:

$$m\ddot{r} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + RF_b \tag{7}$$

Here m -a mass of UAV, $r = [x\ y\ z]^T$ aerial vehicle's distance from the inertial field, F_b-non-gravitational forces acting on the aerial vehicle in the body field and g-gravitational acceleration. Insert that into the translational formula of motion (7) and expanding the terms, we obtain:

$$m\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + \begin{bmatrix} c\psi c\theta & c\psi s\varphi s\theta & s\varphi s\psi + c\varphi c\psi s\theta \\ c\theta s\psi & c\theta c\psi + s\varphi s\psi s\theta & c\varphi s\psi s\theta - c\psi s\theta \\ -s\theta & c\theta s\varphi & c\varphi c\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -U_1 \end{bmatrix} \tag{8}$$

To get the accelerations in terms of the other variables we rewrite formula (8).

$$\ddot{x} = \frac{-U_1}{m}(\sin\varphi \sin\psi + \cos\varphi \cos\psi \sin\theta) \tag{9}$$

$$\ddot{y} = \frac{-U_1}{m}(\cos\varphi \sin\psi \sin\theta - \cos\psi \sin\varphi) \tag{10}$$

$$\ddot{z} = g - \frac{-U_1}{m}(\cos\varphi \cos\theta) \tag{11}$$

Obviously that the translational subsystem is underactuated as it is related on both the rotational state variables and the translational ones. Further we define the state vectors x_1 to x_{12} which are projected to degrees of freedom of the quadcopter in the following state space form:

$$X = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 & x_{10} & x_{11} & x_{12} \\ \varphi & \dot{\varphi} & \theta & \dot{\theta} & \psi & \dot{\psi} & z & \dot{z} & x & \dot{x} & y & \dot{y} \end{bmatrix}^T$$

$$\dot{x_1} = \dot{\varphi} = x_2$$

$$\dot{x_2} = \ddot{\varphi} = x_4x_6a_1 - x_4\omega_r a_2 + b_1U_2$$

$$\dot{x_3} = \dot{\theta} = x_4$$

$$\dot{x_4} = \ddot{\theta} = x_2x_6a_3 + x_2\omega_r a_4 + b_2U_3$$

$$\dot{x_5} = \dot{\psi} = x_6$$

$$\dot{x_6} = \ddot{\psi} = x_2x_4a_5 + b_3U_4$$

$$\dot{x_7} = \dot{z} = x_8$$

$$\dot{x_8} = \ddot{z} = g - \frac{U_1}{m}(\cos x_1 \cos x_3) \tag{12}$$

$$\dot{x_9} = \dot{x} = x_{10}$$

$$\dot{x_{10}} = \ddot{x} = \frac{-U_1}{m}(\sin x_1 \sin x_5 + \cos x_1 \sin x_3 \cos x_5)$$

$$\dot{x_{11}} = \dot{y} = x_{12}$$

$$\dot{x_{12}} = \ddot{y} = \frac{U_1}{m}(\sin x_1 \cos x_5 - \cos x_1 \sin x_3 \sin x_5)$$

For stabilization of quadcopter using PID control, we use yaw, pitch, and roll angles data collected from the IMU sensor.

3. CLASSICAL PID CONTROL SYSTEM

The PID is a control system with a closed loop that is broadly used in commercial control systems and various other applications demanding continuously modulated control [10-20]. A PID control method continuously estimates an error value e(t) as the difference between the reference set (SP) value and a measured process value (PV) and takes a correction based on proportional, integral, and derivative terms.

$$e(t) = SP - PV(t) \tag{13}$$

$$u(t) = K_p e_{(t)} + K_i \int_0^T e_{(\tau)}\, d\tau + K_d \frac{de(t)}{dt} \tag{14}$$

In the above equation, K_p- proportional, K_i - integral and K_d, - derivative coefficients with positive values.

3.1. Roll angle correction formula

$$R_{er,roll} = x_1 - S_{roll} \tag{15}$$

$R_{er,\,roll}$ - roll angle error

x_1 - roll angle measured by IMU

S_{roll} - set point of roll angle

$$R_{err,roll} = K_{i,roll} * R_{er,roll} \tag{16}$$

$R_{err,\,roll}$ - accumulated error of roll angle

$K_{i,roll}$ - integral coefficient of roll angle

$$U_{roll} = K_{p,roll} * R_{er,roll} + R_{err,roll} + K_{i,roll} * (R_{er,roll} + R_{d,err,roll}) \tag{17}$$

$R_{d.err,roll}$ - previous error of roll angle

U_{roll} - correction value of roll angle

3.2. Pitch angle correction formula

$$R_{er,pitch} = x_3 - S_{pitc} \tag{18}$$

$R_{er,\,pitch}$ – pitch angle error

x_3 – pitch angle measured by IMU

S_{pitch} – set point of pitch angle

$$R_{ear,pitch} = K_{i,pitch} * R_{er,pitch} \tag{19}$$

$R_{ear,\,pitch}$ – accumulated error of pitch angle

$K_{i,\,pitch}$ – integral coefficient of pitch angle

$$U_{pitch} = K_{p,pitch} * R_{er,pitch} + R_{err,pitch} + K_{i,pitch} * (R_{er,pitch} + R_{d,err,pitch}) \tag{20}$$

$R_{d.err,\,pitch}$ – previous error of pitch angle

$_{Pitch\,-}$ correction value of pitch angle

3.3. Yaw angle correction formula

$$R_{er,yaw} = x_5 - S_{yaw} \tag{21}$$

$R_{er,\,yaw}$ – yaw angle error

x_5 – yaw angle measured by IMU

S_{aw} – set point of pitch angle

$$R_{err,yaw} = K_{i,yaw} * R_{er,yaw} \tag{22}$$

$R_{ear,\,yaw}$ – accumulated error of yaw angle

$K_{i,\,yaw}$ – integral coefficient of yaw angle

$$U_{yaw} = K_{p,yaw} * R_{er,yaw} + R_{err,yaw} + K_{i,yaw} * (R_{er,yaw} + R_{d,err,yaw}) \tag{23}$$

$R_{d.\,err,\,yaw}$ – previous error of yaw angle

U_{yaw} - correction value of yaw angle

The electronic speed controller operates in the range of 800-2000 PWM. However, we limited the PWM output value to a maximum of 1700 and a minimum of 1300. The throttle value of PWM is set to w_0=1500. The following formulas estimate the rotation speed of each motor:

$$w_1 = PWM_1 = \omega + U_{pitch} + U_{roll} - U_{yaw} \tag{24}$$

$$w_2 = PWM_2 = \omega + U_{pitch} - U_{roll} + U_{yaw} \tag{25}$$

$$w_3 = PWM_3 = \omega - U_{pitch} - U_{roll} - U_{yaw} \tag{26}$$

$$w_4 = PWM_4 = \omega - U_{pitch} + U_{roll} + U_{yaw} \tag{27}$$

4. DESIGN OF PROPOSED NEURAL NETWORK MODEL

The main idea to use a neural network for controlling the quadcopter was based on the demand to deal with large indeterminate parameter estimates and wind disturbances [9]. Using neural networks can be one possible method to design such dynamics. The Universal Approximation Theorem tells us that Neural Networks has a kind of universality; their architecture gives them to design highly nonlinear functions [10]. Furthermore, neural networks can produce general models that can operate on data other than the observed data. However, it is not clear whether the proposed NN-based model can be used to control the system or whether the trained dynamics represent the system more accurately than the trained data.

4.1. Proposed Neural Network Model

In this work, three-layer NNs are expressed, an input layer, hidden layer, and output layer. The structure of the NN is shown in Figure 2, in which the first layer is the input layer that takes the current state - input of the model. Before training the input layer was assigned with random weights $\theta_{i,j}^{(1)}$ and added bias b^1. The second layer has ten hidden neurons with random weights $\theta_{i,j}^{(2)}$ and added bias, b^2. The sigmoid activation function is used in our model. Each of N activation units computes the inner product of the weights. The output layer has four neurons which represent the speed of each motor.

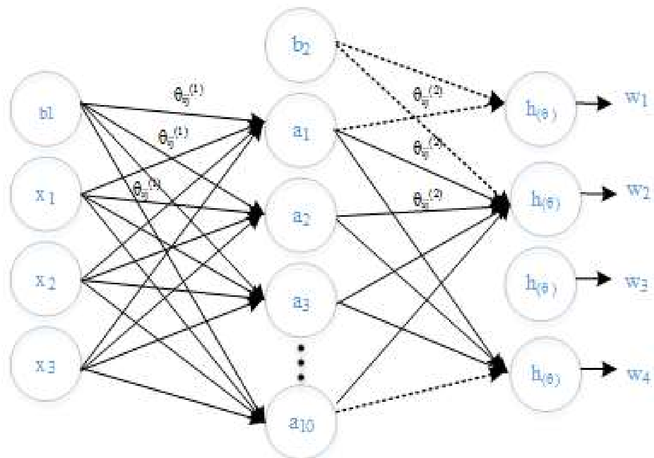


Figure 2. Neural network model.

The matrix representation of the model is shown below.

$$\begin{bmatrix} b^1 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \rightarrow \begin{bmatrix} b^{(2)} \\ a_1^{(2)} \\ a_2^{(2)} \\ . \\ . \\ a_{10}^{(2)} \end{bmatrix} \rightarrow \begin{bmatrix} h_\theta(x)_1 \\ h_\theta(x)_2 \\ h_\theta(x)_3 \\ h_\theta(x)_4 \end{bmatrix}$$

Hidden layer neurons can be estimated by the following equation (1-7).

$$a_i^2 = g(\theta^T x) \tag{28}$$

$$z = \theta^T x = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \tag{29}$$

$$a_i^2 = g(z) = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-\theta^T x}} \tag{30}$$

The sigmoid function transforms any real number to the interval (0, 1).

$$\begin{gathered} a_1^{(2)} = g\left(\theta_{10}^{(1)} b^1 + \theta_{11}^{(1)} x_1 + \theta_{12}^{(1)} x_2 + \theta_{13}^{(1)} x_3\right) \\ a_2^{(2)} = g\left(\theta_{20}^{(1)} b^1 + \theta_{21}^{(1)} x_1 + \theta_{22}^{(1)} x_2 + \theta_{23}^{(1)} x_3\right) \\ a_n^{(2)} = g\left(\theta_{n0}^{(1)} b^1 + \theta_{n1}^{(1)} x_1 + \theta_{n2}^{(1)} x_2 + \theta_{n3}^{(1)} x_3\right) \end{gathered} \tag{31}$$

where, n = [1:10]

Output neurons calculated by the following formulas:

$$h_\theta(x)_1 = a_1^{(3)} = w_1 = g(\theta_{10}^{(2)} b^2 + \theta_{11}^{(2)} a_1^{(2)} + \theta_{12}^{(2)} a_2^{(2)} + \theta_{13}^{(2)} a_3^{(2)} + \dots + \theta_{110}^{(2)} a_{10}^{(2)})$$

$$h_\theta(x)_2 = a_2^{(3)} = w_2 = g(\theta_{20}^{(2)} b^2 + \theta_{21}^{(2)} a_1^{(2)} + \theta_{22}^{(2)} a_2^{(2)} + \theta_{23}^{(2)} a_3^{(2)} + \dots + \theta_{210}^{(2)} a_{10}^{(2)})$$

$$h_\theta(x)_3 = a_3^{(3)} = w_3 = g(\theta_{30}^{(2)} b^2 + \theta_{31}^{(2)} a_1^{(2)} + \theta_{32}^{(2)} a_2^{(2)} + \theta_{33}^{(2)} a_3^{(2)} + \dots + \theta_{310}^{(2)} a_{10}^{(2)})$$

$$h_\theta(x)_4 = a_4^{(3)} = w_4 = g(\theta_{40}^{(2)} b^2 + \theta_{41}^{(2)} a_1^{(2)} + \theta_{42}^{(2)} a_2^{(2)} + \theta_{43}^{(2)} a_3^{(2)} + \dots + \theta_{410}^{(2)} a_{10}^{(2)}) \quad (32)$$

Here:

x_n – inputs

$a_i^{(j)}$- activation unit i of layer j

$\theta_i^{(j)}$- the value of the weight in *i* to convert from layer *j* to layer *j + 1*

4.2. Cost Function

Our goal is to determine each layer's weight $\theta_{i,}^{(Jl)}$ parameters. Proper determination of these parameters requires that the difference between the prediction function $h_\theta(x)_i$ and the output value y^i be kept to a minimum. The Y^i is the rotation speed of the motor collected from PID control to train the neural network.

$$loss = h_\theta(x^{(i)}) - y^i \quad (33)$$

The neural network cost function $J(\theta)$ is defined by equation (34).

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[y_k^i \log\left(\left(h_\theta(x^{(i)})\right)_k\right) + (1 - y_k^{(i)}) \log\left(1 - \left(h_\theta(x^{(i)})\right)_k\right)\right] \quad (34)$$

Here, k is the number of output units.

The cost function determines the difference between the hypothesis function value and the output value.

The main idea of the concept of the learning process is to determine the minimum of the cost function $J(\theta)$ $(min_\theta J(\theta))$. We take a partial derivative from the evaluation function to determine the minimum. We use the "back propagation" algorithm to calculate this partial derivative. First, we calculate the difference $\delta^{(L)}$ between the output value $y^{(t)}$ and the assumed output value $a^{(L)}$ (9).

$$\delta^{(L)} = a^{(L)} - y^{(t)} \quad (35)$$

Subsequently, the difference between each layer is calculated by the following formula.

$$\delta^{(L-1)}, \delta^{(L-2)}, \dots, \delta^{(2)}$$

$$\delta^{(l)} = ((\theta^l)^T \delta^{(l+1)}) .* a^{(l)} .* (1 - a^{(l)}) \quad (36)$$

From this we can calculate the partial derivatives.

$$\frac{\partial}{\partial \theta_{i,j}^{(l)}} J(\theta) = a_j^{(l)} \delta_i^{(l+1)} \quad (37)$$

Updating weight $\theta_{i,j}^{(l)}$ parameters by the following formula.

$$\theta_{i,j}^{(l)} := \theta_{i,j}^{(l)} - \alpha \frac{\partial}{\partial \theta_{i,j}^{(l)}} J(\theta) \quad (38)$$

Here, α is the learning rate.

We updated weight parameters until the cost function reached the minimum value.

5. EXPERIMENTAL RESULTS

The test environment is shown below. The quadcopter is placed on the metal sphere that moves in three degrees of freedom. The quadcopter is equipped with frame F450, has four brushless motors with 1000KV, and uses 30A ESC. The total mass of the quadcopter is 900 grams. To collect data for training, we flew and recorded the states (x_1-roll, x_2-pitch, x_3-yaw angles) as input data and four propeller rotation speeds (w_1, w_2, w_3, w_4) as output data. The experimental environment is shown in Figure 3. We collected 7000 data to develop a neural network model. The collected data is split into three parts: 70% for training, 20% for evaluation, and 10% for tests.

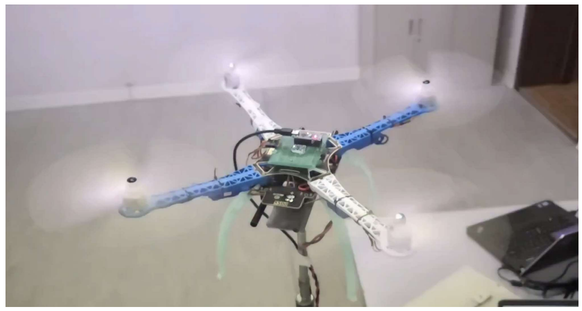


Figure 3. Test environment of the quadcopter.

The neural network model was trained using MATLAB. Some network parameters are set as follows: learning rate (0.001), without regularization factor, iteration (9M), a number of layers, and a number of neurons in the hidden layer. Before training the neural network, we follow a few data pre-processing steps.

We do not provide position as inputs; instead, we provide angles (degree units) as inputs to the neural network. The rotational speed of the rotor or PWM value is given in intervals between 800 to 2000. We scale (13) the observed outputs (rotational speed - PWM value) such that each of them has a value in intervals between 0 to 1.

$$y_i = \left(\frac{w_i - 700}{2000}\right) \tag{39}$$

This ensures that the neural network gives equal weightage to the collected value.

We gave an external influence on the drone and recorded the angles of the drone, as shown in Fig. 4.

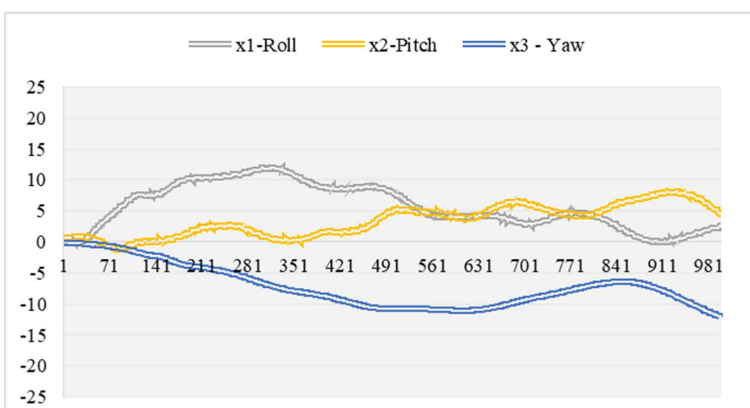
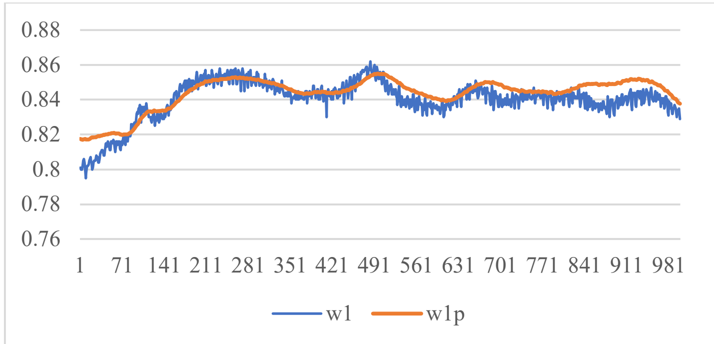
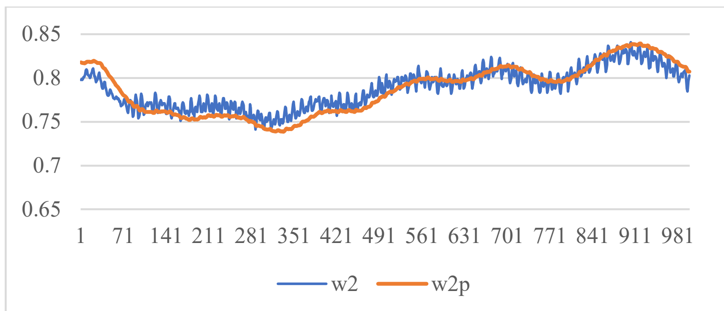


Figure 4. Drone angles are changed by external influence

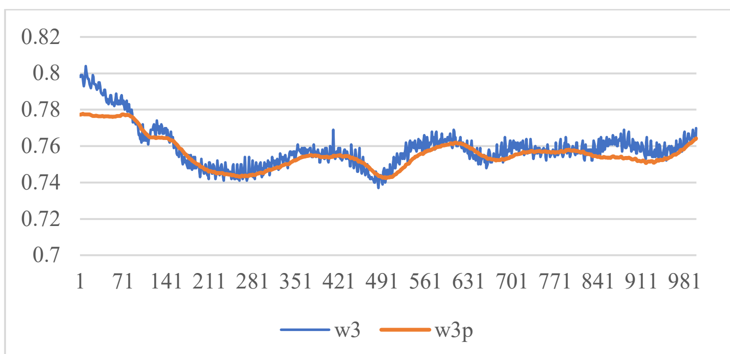
Figures 5a, 5b, 5c, and 5d show the rotational speed of each motor due to the change in angle. Rotational speed dependence on the change in angle is drawn in blue color, and the rotational speed of the motor is learned through the neural network model shown by the orange color.



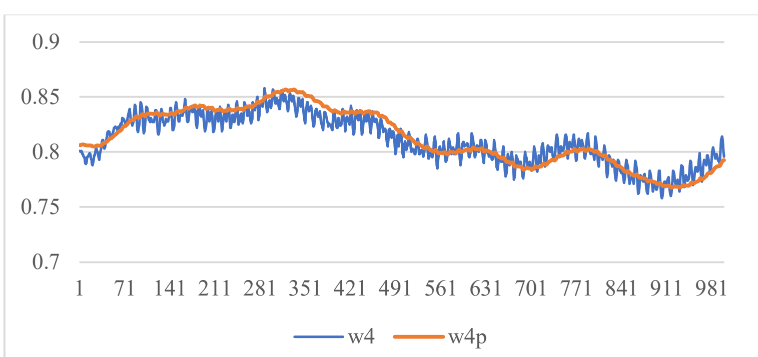
(a)



(b)



(c)



(d)

Figure 5. PID motors rotation speeds vs. NN learned rotation speeds

(a) -first, (b)-second, (c)-third, (d)- fourth motor rotation speed w1 compared with learned rotation speed w1p

In the second test, we changed the number of nodes in the hidden layer. Depending on the number of nodes in the hidden layer, the dynamic design of the drone will change. We change the nodes to 3, 10, and 20 to determine the value of the cost function. Figure 6 shows changes in cost function value depending on nodes number.

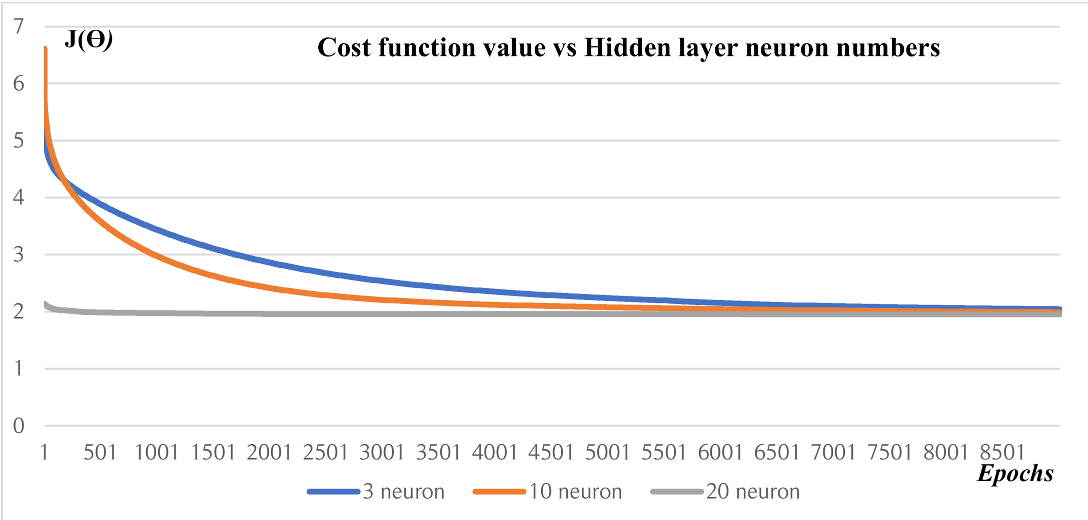


Figure 6. Dependence of Cost function value and node numbers.

6. CONCLUSION

The experimental results presented in the research work have shown that a neural controller can be used for the stabilization of a quadcopter. Implementing such control in the concrete flying object should enhance the object's stability, as the neural controller better mimics with control when noisy excitations are given into its inputs. In the PID control, three coefficients are used for each angle adjustment, and nine coefficients are needed to control the quadcopter. The NN with three neurons in the hidden layer requires 28 coefficients to control the quadcopter. With the increased number of hidden neurons, the number of the coefficient increased drastically. The choice of a number of hidden layers and hidden layer neurons depends on controller memory and the speed of the processor. In the next research step, we will analyze another neural network mixed with a reinforcement learning model.

REFERENCES

[1] M. Lower and W. Tarnawski, “Quadrotor Navigation Using the PID and Neural Network Controller,” Wrocław University of Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland

[2] A. Zulu*, S. John, "A Review of Control Algorithms for Autonomous Quadrotors," Open Journal of Applied Sciences, 2014, 4, 547-556

[3] H. Boudjedir, O.Bouhali, and N. Rizoug, "Neural Network Control Based on Adaptive Observer for Quadrotor Helicopter," International Journal of Information Technology, Control and Automation (IJITCA) Vol.2, No.3, July 2012

[4] H. Boudjedir, O.Bouhali, and N. Rizoug, "Adaptive neural network control based on neural observer for quadrotor unmanned aerial vehicle," Advanced Robotics, 2014, Vol.28, No.17, 1151–1164, http://dx.doi.org/10.1080/01691864.2014.913498

[5] T. Dierks and S. Jagannathan, "Neural Network Control of Quadrotor UAV Formations," 2009 American Control Conference Hyatt Regency Riverfront, St. Louis, MO, USA June 10-12, 2009

[6] Holger Voos, "Nonlinear and Neural Network-based Control of a Small Four-Rotor Aerial Robot," 2007 IEEE/ASME international conference on advanced intelligent mechatronics.

DOI: 10.1109/AIM.2007.4412470

[7] Yan-Fei Teng, Bin Hu, Zhi-Wei Liu, Jian Huang, and Zhi-Hong Guan, "Adaptive Neural Network Control for Quadrotor Unmanned Aerial Vehicles," 2017 11th Asian Control Conference (ASCC) Gold Coast Convention Centre, Australia December 17-20

[8] X Yu, Z Lv, Y Wu, XM Sun, "Neural network modeling and backstepping control for quadrotor," Chinese Automation Congress (CAC), 2018

DOI: 10.1109/CAC.2018.8623432

[9] C. Nicol1, C.J.B. Macnab, A. Ramirez-Serrano, "Robust Neural Network Control Of A Quadrotor Helicopter" Canadian Conference on Electrical and Computer Engineering, 2008. DOI: 10.1109/CCECE.2008.4564736

[10] S. Bansal; Anayo K. Akametalu; Frank J. Jiang; F. Laine; Claire J. Tomlin, "Learning Quadrotor Dynamics Using Neural Network for Flight Control," IEEE 55th Conference on Decision and Control (CDC), 2016.

DOI: 10.1109/CDC.2016.7798978

[11] Yan-Fei Teng, Bin Hu, Zhi-Wei Liu, Jian Huang, Zhi-Hong Guan, "Neural Network Adaptive Inverse Model Control Method for Quadrotor UAV," 35th Chinese Control Conference (CCC), 2016

DOI: 10.1109/ChiCC.2016.7553921

[12] Cai Luo, Zhenpeng Du, and Leijian Yu, "Neural Network Control Design for an Unmanned Aerial Vehicle with a Suspended Payload" Electronics 2019, 8, 931; doi:10.3390/electronics8090931

[13] Hadi Ramzi a, ∗, Sima Afshinfar, "Neural network-based adaptive sliding mode control design for position and attitude control of a quadrotor UAV" Aerospace Science and Technology Volume 91, August 2019, Pages 12-27 https://doi.org/10.1016/j.ast.2019.04.055

[14] Javier Gómez-Avila; Carlos López-Franco; Alma Y. Alanis; Nancy Arana-Daniel, "Control of Quadrotor using a Neural Network based PID", IEEE Latin American Conference on Computational Intelligence (LA-CCI), 2018. DOI: 10.1109/LA-CCI.2018.8625222

[15] Xrumer Lin, Yulu Wang, Yunfei Liu, "Neural-network-based robust terminal sliding-mode control of quadrotor," Chinese Automatic Control Society and John Wiley & Sons Australia, 2020 https://doi.org/10.1002/asjc.2478

[16] Christopher Edmond Nicol, "A Robust Adaptive Neural Network Control for a Quadrotor Helicopter," Master thesis, University of Calgary, 2010

[17] Osman Çakira*, Tolga Yüksel, "Neural Network Control for Quadrotors," American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS), (2017) Volume 31, No 1, pp 191-200

[18] Qingzheng Xu · Zhisheng Wang · Ziyang Zhen, "Adaptive neural network finite-time control for quadrotor UAV with unknown input saturation," Nonlinear Dynamics volume 98, pages1973–1998 (2019)

[19] Yan-min Chen, Yong-ling He & Min-Feng Zhou, "Decentralized PID neural network control for a quadrotor helicopter subjected to wind disturbance," Journal of Central South University volume 22, pages168–179 (2015)

[20] Hadi RAMZI, "Adaptive neural network-based sliding mode attitude control for a quadrotor UAV," Journal of Central South University volume 25, pages2654–2663 (2018)

[21] Travis Dierks, "Output Feedback Control of a Quadrotor UAV Using Neural Networks," IEEE Transactions on Neural Networks, Vol. 21, No. 1, January 2010

[22] Weinan Gao, Jialu Fan1 and Yannong Li1, "Research on Neural Network PID Control Algorithm for a Quadrotor," Applied Mechanics and Materials Vols. 719-720 (2015) pp 346-351

[23] Ts. Tengis; A. Batmunkh, "Experimental approach to Pole Placement problem of State Feedback Control for Quadrotor Stabilization in Hovering Mode," International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol. 6, Issue 9, September 2017

[24] Ts. Tengis; A. Batmunkh, "State feedback control simulation of quadcopter model," 11th International Forum on Strategic Technology (IFOST), 2016

DOI: 10.1109/IFOST.2016.7884178

BIOGRAPHIES

Bat-Erdene Gotov, is a PhD candidate in the Electronic Department of MUST. Between 2012 and 2018 he was the researcher Institute in Physical Technology of the Mongolian Academy of Sciences. He is currently a lecturer at the Mongolian National University of medical sciences. His areas of interest are: Information and Communication Technology, artificial intelligence, Deep Learning, machine learning, and Data Processing.

Tengis Tserendondog received the Ph.D. degree in electronical engineering in 2018 from MUST. He currently works as an assistant professor in the Department of Electronics, Mongolian University of Science Technology. His current research interests include the control of robotic systems, machine learning and quadcopters.

Lodoiravsal Choimaa received his B.S. and M.S. degrees in Electronics Engineering from the National University of Mongolia, Mongolia in 1999 and 2001, respectively. He received his Ph.D. degree from the National University of Mongolia (by DAAD sandwich program at the Research Center Juelich, Germany) in 2006. Since 2016, he has been the head of the Machine Intelligence Laboratory, the National University of Mongolia. His research interests include image processing, robotics, and machine intelligence.

Batmunkh Amar is master in electrical engineering (1978), and doctor in computer science (1997) with the Mongolian Technical University. Between 1998 and 2013 he was a teacher, researcher, associate professor of the Department of Electronics in Computer Technique and Management school and between 2014 and 2020 he was a full professor of the Department of Electronics of the School of Information and Communication Technology at Mongolian University of Science and Technology. His areas of interest are: Signal Processing, Control systems, Machine Learning and Analogue Electronic circuits.