International Journal of Precision Engineering and Manufacturing (2021) 22:1147–1156
https://doi.org/10.1007/s12541-021-00523-z

Online ISSN 2005-4602
Print ISSN 2234-7593

REVIEW

Robust Landing Control of a Quadcopter on a Slanted Surface

Jiwook Choi[1] · Donghun Cheon[1] · Jangmyung Lee[1]

Received: 1 December 2020 / Revised: 15 February 2021 / Accepted: 31 March 2021 / Published online: 19 April 2021

Abstract
A robust landing control algorithm is proposed for a quadcopter, as well as for a landing platform to land on an inclined or problematic surface. To use the quadcopter for outdoor application, it is necessary to design a landing platform that can withstand environmental obstacles such as wind and weight load during landing. Conventional retractor landing platforms are not suitable for achieving a stable landing on inclined surfaces or obstacles. Therefore, in this paper, 2-link structured landing legs are applied to stably land on an inclined surface or obstacle with a suitable control algorithm. To achieve stable landing on a slanted surface, a cooperative control algorithm of the quadcopter and the landing platform has been proposed. The proposed robust landing system comprises two controllers, i.e., a high-speed proportional derivative control for the landing platform and a neural network-based proportional–integral–derivative control for controlling the quadcopter in real time. A quadcopter with a robust landing platform has been implemented, and the performance of the robust landing control algorithm has been demonstrated with the system.

Keywords Quadcopter · PID control · Robust landing · Neural network · Landing platform · Adaptive landing

1 Introduction

Drone refers to an unmanned aerial vehicle system that automatically or semi-automatically flies according to a pre-programmed route on the ground without a pilot boarding directly [1]. The conventional drone market focuses on hobby drones, which is established and developed by Chinese company DJI. In recent years, however, the drone industry has expanded its scope of use, targeting not only hobbyists but also applications in military and agricultural contexts [2]. Drones are divided into various types, such as rotating and fixed wing drones, tilt-rotor type, and flapping wing type, depending on the manufacturing method [3, 4].

Especially when comparing the rotary wing and the fixed wing types, the rotary-wing drone has the advantage of being able to perform vertical rise/fall movement, as well as hovering, which makes it function in less limited space. When operating a rotary-wing drone, environmental factors such as wind can have a significant impact. In a strong wind, a drone may be pulled in the wind direction, thus losing posture control and flying in an unwanted direction. Additionally, when a quadcopter lands, if the environment at the landing point is unstable, the quadcopter may malfunction and sustain damage. On the basis of these environmental factors, drones need a powerful controller for normal operation and to land safely in complex situations.

Conventional quadcopters mainly use proportional–integral–derivative (PID) controllers, which are relatively simple and easy to design [5–7]. However, the PID control has some disadvantages. First, it is difficult to define the gain values of P, I, and D in an environment that changes rapidly. Second, it is difficult to design the controller because the drone is nonlinear, time delayed, and overall a complex and ambiguous system. To overcome these disadvantages, a fuzzy PID controller was developed to set gain values automatically [8, 9]. However, the addition of one input variable to the fuzzy system causes a disadvantage in that the number of control rules increases significantly, making it difficult to design.

An existing quadcopter landing system attaches a retractor to a gear to reduce air resistance when flying and to control the gear when landing [10]. In this method, whether it is possible to land or not has been determined according to

✉ Jangmyung Lee
jmlee@pusan.ac.kr

Jiwook Choi
jiwook7379@pusan.ac.kr

[1] Department of Electronics Engineering, SPENALO National Robotics Research Center, Pusan National University, 63-2 Jangjeon-dong, Geumjeong-gu, Busan 46241, Korea

the situation on the ground. Although the landing is determined to be possible, an unstable landing can be performed by a reaction that occurs according to load on the quadcopter. There are many difficulties related to landing in an environment where an obstacle or an inclination exists on the ground [11]. To solve this problem, studies focusing on adaptive landing gear have been actively conducted. Disadvantages of this landing gear include that it is structurally complex and expensive, in terms of design, to apply these aspects to small aircraft such as quadcopters [12, 13].

In this paper, a landing platform with four landing legs has been designed to achieve a stable landing on a slanted surface. In addition, a cooperative control algorithm between the landing platform and quadcopter is proposed for the stable landing against disturbances. The cooperative control algorithm comprises a robust controller and adaptive landing algorithm. The robust controller is a neural network (NN)-based algorithm that determines PID gains in real time with their error values [14, 15]. The gradient descent method was adopted to adjust the weights of the NN [16, 17]. In this paper, a robust controller has been proposed for a quadcopter to perform hovering in wind or heavy load situations, and an experiment is conducted to perform landing in a complex environment using the adaptive landing algorithm to compare performance with the conventional PID controller. The efficiency of the algorithm is demonstrated using the acquired posture data.

Following Sects. 1 and 2 describes the quadcopter and landing leg modeling. Section 3 describes the proposed algorithm. Section 4 includes experiments and experimental results. Section 5 concludes this research with the summary of the results.

2 System Modeling

2.1 Quadcopter Modeling

The quadcopter motion can be defined by using rotation matrix, R, as shown in Fig. 1. R can be defined as follows [18]:

$$R = R_z(\psi)R_y(\theta)R_x(\phi) \tag{1}$$

where $R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$,

$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$ and $R_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$.

Using the thrusts from the four rotors in Fig. 1, it can be expressed as 6 degrees of freedom of the

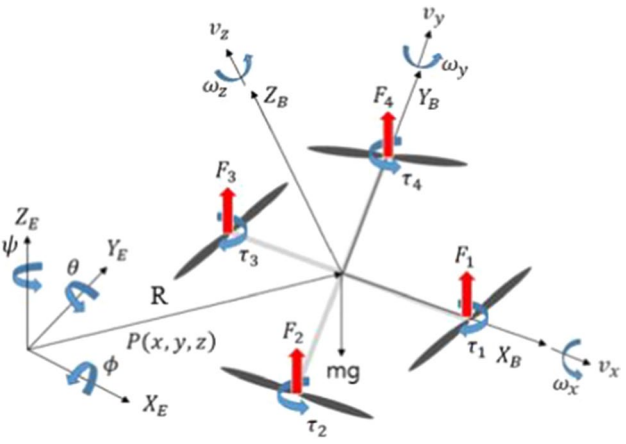


Fig. 1 The coordinate system of the quadcopter

translational motion $p = [\,x\ y\ z\,]^T$ and the rotational motion $\Omega = [\,\phi\ \theta\ \psi\,]^T$. Linear velocity $\dot{p}$ and angular velocity ω of the quadcopter can be described as follows:

$$\dot{p} = Rv_b \tag{2a}$$

$$\omega = C\omega_b \tag{2b}$$

where $v = [\,\dot{x}\ \dot{y}\ \dot{z}\,]^T$, and $\omega = [\,\dot{\phi}\ \dot{\theta}\ \dot{\psi}\,]^T$ represent the linear velocity and angular velocity in the global coordinate system. ω_b represent the linear velocity and angular velocity of the quadcopter with respect to the object coordinate system.

C is a matrix representing the relationship between the velocity component of the Euler angle of the inertial coordinate system and the angular velocity vector of the body frame coordinate system, which is represented as follows:

$$C = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \tag{3}$$

By differentiating Eqs. (2a) and (2b),

$$\ddot{p} = R\dot{v}_b + \dot{R}v_b \tag{4a}$$

$$\dot{\omega} = C\dot{\omega}_b + \dot{C}\omega_b \tag{4b}$$

where

$$\dot{C} = \left[\frac{\partial C}{\partial\phi}\dot{\phi} + \frac{\partial C}{\partial\phi}\dot{\theta} + \frac{\partial C}{\partial\phi}\dot{\psi}\right] = \begin{bmatrix} 0 & 0 & -\dot{\theta}\cos\theta \\ 0 & -\dot{\phi}\sin\phi & \dot{\phi}\cos\phi\cos\theta - \dot{\theta}\sin\phi\sin\theta \\ 0 & -\dot{\phi}\cos\phi & -\dot{\phi}\sin\phi\cos\theta - \dot{\theta}\cos\phi\sin\theta \end{bmatrix},$$

Equation (4a) can be reorganized as follows:

$$\ddot{p} = R(\dot{v}_b + \omega_b \times v_b) \tag{5}$$

The quadcopter's translational equation of motion can be expressed through Eq. (5). At this time, the force $f = [f_x, f_y, f_z]$ for the translational motion is actually affected by gravity, so it can be expressed as follows:

$$f = mR^T\ddot{p} = m(\dot{v}_b + \omega_b \times v_b) + mR^T g_0 \tag{6}$$

If Eq. (6) is expressed as the equation for the acceleration of the quadcopter, it can be expressed as follows:

$$\dot{v}_b = \omega_b \times v_b - R^T g_0 + \frac{1}{m} f \tag{7}$$

Here, $g_0 = [\,0\ \ 0\ \ -g\,]^T$ indicates the acceleration of gravity.

Using the law of conservation of moment, the following equation can be derived.

$$M = I\dot{\omega}_b + \omega_b \times (I\omega_b) \tag{8}$$

where, the moment of inertia is $I_{xx} = I_{yy}$ because the quadcopter is designed in a line symmetry [19, 20].

In the global coordinate system, Eqs. (2b) and (8) can be summarized as follows:

$$M = I(\dot{C}\dot{\Omega} + C\ddot{\Omega}) + (C\dot{\Omega}) \times (IC\dot{\Omega}) \tag{9}$$

Assuming that there are no centripetal force and Coriolis effects, Eqs. (7) and (9) can be simplified as follows:

$$\dot{V}_b = \begin{bmatrix} g\sin\phi \\ -g\cos\theta\sin\phi \\ -g\cos\theta\cos\phi \end{bmatrix} + \frac{1}{m}\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} \tag{10}$$

$$\ddot{\Omega} = C^{-1}I^{-1}M \tag{11}$$

Through the above equations, the final dynamic equation of the quadcopter can be derived as follows:

$$m\ddot{x} = f_t(\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)$$

$$m\ddot{y} = f_t(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)$$

$$m\ddot{z} = f_t\cos\phi\sin\theta - mg$$

$$I_{xx}\ddot{\phi} = \tau_\phi$$

$$I_{yy}\ddot{\theta} = \tau_\theta \tag{12}$$

where $f_t = f_z$ represents the thrust of the quadcopter.

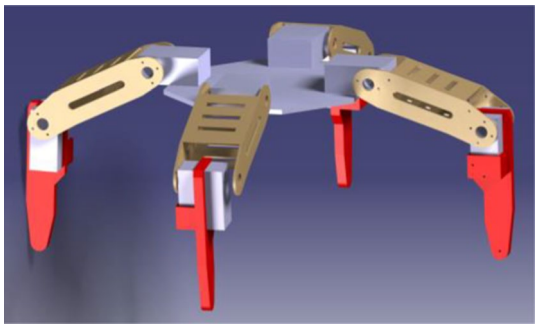


Fig. 2 The adaptive landing platform

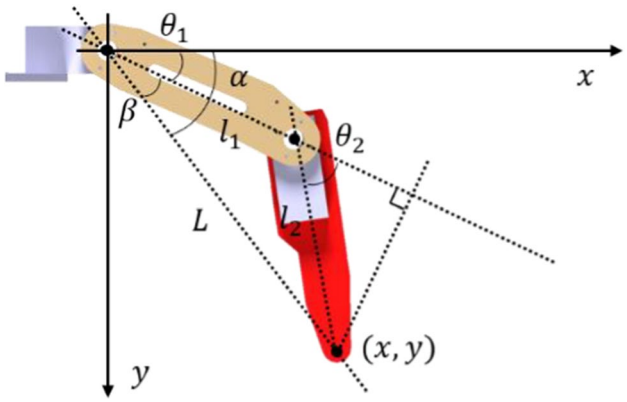


Fig. 3 Landing leg coordinate system

2.2 Kinematics of the Landing Legs

A landing platform was designed to guarantee the stable landing of the drone in uneven ground conditions. The landing platform with four legs was attached to the bottom of the drone. In this research, adaptive landing gear was designed to construct a landing platform. Two-degrees-of-freedom landing legs were used to achieve a stable landing, regardless of ground conditions.

Figure 2 illustrates the landing legs used in this research, which was designed using the Catia software package. The landing legs were designed with a 2-link structure by attaching a servo motor to each joint. It was designed to land stably on slopes or a complex object by controlling each joint according to the conditions of the landing location. To perform a stable landing, it is necessary to control the legs to maintain the balance of the quadcopter in complex ground conditions. To do this, the balance must be maintained through angle control using the servo motor of each joint. The target angle is calculated through the inverse kinematic analysis of the landing leg of the 2-link structure [21].

The forward kinematics of the 2-link leg in Fig. 3 is represented as follows:

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) \tag{13}$$

$$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) \tag{14}$$

where x and y are the axes of the robot in Cartesian coordinates, l_1 and l_2 are lengths of the first and the second links, and θ_1 and θ_2 are the angles of the revolute joints, θ_1 and θ_2 must be obtained for the landing leg location, (x, y), which can be obtained from Eqs. (13) and (14) as follows:

$$\cos\theta_2 = \frac{x^2 + y^2 - l_1^2 - l_2^2}{2l_1 l_2} \tag{15}$$

$$\theta_2 = \pm a\tan 2(\sin\theta_2, \cos\theta_2) \tag{16}$$

$$\theta_1 = a\tan 2(y, x) - a\tan\left(\frac{l_2 \sin\theta_2}{l_1 + l_2 \cos\theta_2}\right) \tag{17}$$

3 Robust Landing Algorithm

3.1 Cooperative Control

For the quadcopter to safely land on a slope, it is necessary to perform balance control of the landing leg for each axis. When the landing legs are controlled and balanced, it will be difficult to land the quadcopter because of the change in the center of gravity caused by the movement of the landing legs. To address this situation, a quadcopter cooperative control algorithm was developed to cope with the movement caused by the landing legs.

Figure 4 shows an overall block diagram of the cooperative control, which is divided into quadcopter and landing-leg control sections. The two controllers form a complementary relationship. Landing control is aiming to minimize roll and pitch errors that occur when the quadcopter lands on an inclined surface, which can be done by obtaining the inverse kinematic of four legs. The calculated error is horizontally controlled through the proportional derivative control. The motor of a leg in controlled to reduce the altitude error e_l of the quadcopter, with respect to the center of gravity generated, which is fed back to perform altitude control of the quadcopter. Quadcopter control is designed with an PID (NN-PID) controller, which quickly responds to posture errors caused by disturbances occurring in real time. This can effectively control posture errors caused by environmental factors such as disturbance and wind caused by the motion of the landing legs.

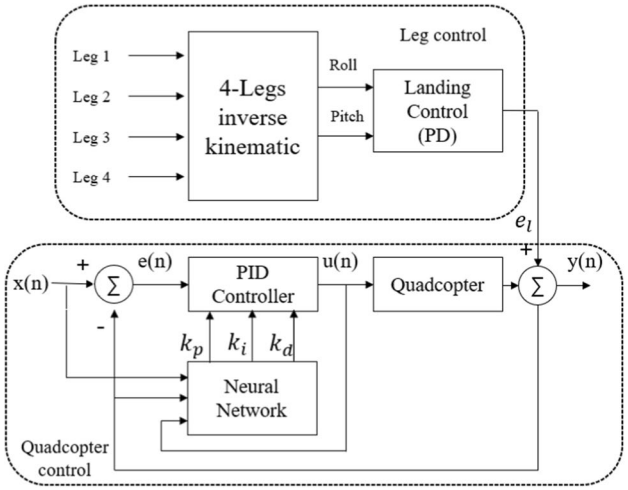


Fig. 4 Cooperative control block diagram

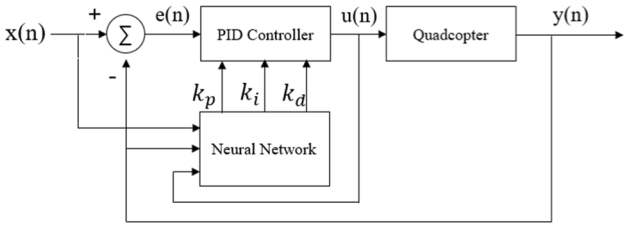


Fig. 5 The NN-PID control framework

3.2 NN-PID for Quadcopter Control

There are several ways in which to control a quadcopter, but the most commonly used among these is a PID controller. In this research, an NN-PID controller was designed for adaptive use in various situations.

Figure 5 illustrates the NN-PID controller for the quadcopter. NN-PID controller continuously adjusts the parameter (k_p ,k_i,k_d) of PID to tune the gain value in real time. The time domain PID controller input $u(n)$ can be expressed as follows:

$$u(n) = u(n-1) + k_p e(n) - e(n-1) + k_i e(n) + k_d e(n) - 2e(n-1) + e(n-2) \tag{18}$$

where $e(n) = y(n) - u(n)$ represents the error between the input and the output.

Neural network design for 3 inputs as shown in Fig. 6.

For some inputs from the network input layer are expressed as follows:

$$O_j^{(1)} = x(j), \quad (j = 1, 2, \ldots, m) \tag{19}$$

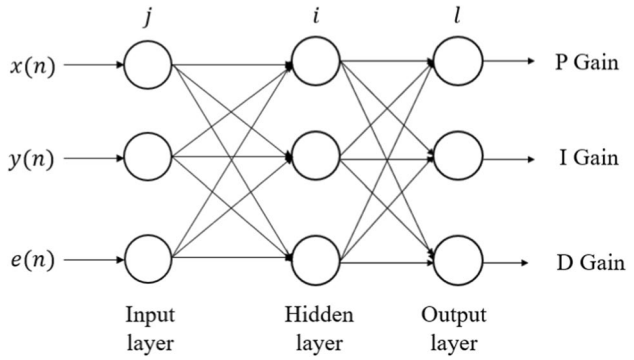


Fig. 6 The neural network structure

In the above equation, the subscript refers to the input layer of the network.

The hidden layer can be expressed as follows:

$$net_j^{(2)}(n) = \sum_{j=1}^{m} w_{ij}^{(2)} o_j^{(1)}, \quad (j = 1, 2, \ldots, m) \tag{20}$$

$$o_i^{(2)} = f(net_i^{(2)}(n)), \quad (i = 1, 2, \ldots, q) \tag{21}$$

The subscript (2) stands for the hidden layer and w_{ji} is weight connecting neurons from the input layer to the hidden neurons layer.

The activation function of the hidden layer that explains the relationship between the input and output of a neuron is a sigmoid function and is as follows:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{22}$$

The input and output of the network output layer is given. by the following equation.

The input and output of the network output layer is given by the following equation.

$$net_l^{(3)}(n) = \sum_{i=1}^{q} w_{li}^{(3)} o_i^{(2)}(n) \tag{23}$$

Superscript (3) shows the output layer, k represents the weight between the hidden layer and the output layer.

$$o_l^{(3)} = g(net_l^{(3)}(n)), \quad (l = 1, 2, 3) \tag{24}$$

$$o_l^{(3)} = k_p, \quad o_2^{(3)} = k_i, \quad o_3^{(3)} = k_d, \tag{25}$$

Output o_l is employed as the gains of the PID controller. The control values corresponding to the outputs k_p, k_i and k_d are given a multiplier so that they are within the defined range before entering. In this algorithm, the system's output error function is defined as:

$$E(n) = \frac{1}{2}(u(n) - y(n))^2 \tag{26}$$

The change for the update weight on the output can be explained in the following equation.

$$\Delta w_{li}^{(3)}(n) = -\eta \frac{\partial E(n)}{\partial w_{li}^{(3)}} + \alpha w_{li}^{(3)}(n-1) \tag{27}$$

where $O_3 = k_d$ is a learning rate and α is momentum factor.

Using the partial differential equation, the control signal of the output can be summarized as follows:

$$\frac{\partial u(n)}{\partial O_k^{(3)}} = \begin{Bmatrix} e(n) - e(n-1) & k = 1 \\ e(n) & k = 2 \\ e(n) - 2e(n-1) + e(n-2) & k = 3 \end{Bmatrix} \tag{28}$$

where $e(n)$ is the error. The weights are updated by the back-propagation algorithm [22, 23].

Learning algorithm of update weight on Output layer can be obtained.

$$\Delta w_{li}^{(3)}(n) = \eta \delta^{(3)} o_i^{(2)}(n) + a w_{li}^{(3)}(n-1) \tag{29}$$

$$\delta_l^{(3)} = e(n) \frac{\partial \hat{y}(n)}{\partial u(k)} \frac{\partial u(n)}{\partial o_l^{(3)}(n)} f'(net_l^{(3)}(n)) \tag{30}$$

where the derivative of the activation function can be expressed as follows:

$$f'(x) = f(x)(1 - f(x)) \tag{31}$$

3.3 Adaptive Landing Algorithm

An adaptive landing algorithm was designed to achieve a stable landing by adapting to the ground situation using the landing legs when the quadcopter performed a landing in a complex ground situation. The force sensor attached to the lower part of the landing leg can determine whether each axis of the leg touches the ground; in this way, balance using the inertial measurement unit sensor attached to the quadcopter is achieved to perform the landing.

It controls the legs located on a high slope to maintain balance and controls each joint of the legs by analyzing the inverse kinematics of the 2-link structure.

In Fig. 7, to maintain the balance of the quadcopter, end point A of the body inclined by ϕ must be moved to B.

When the body length is d, the movement distance of the x- and y-axes can be obtained as follows:

$$x_\phi = 2d \sin^2\left(\frac{\phi}{2}\right) \tag{32}$$

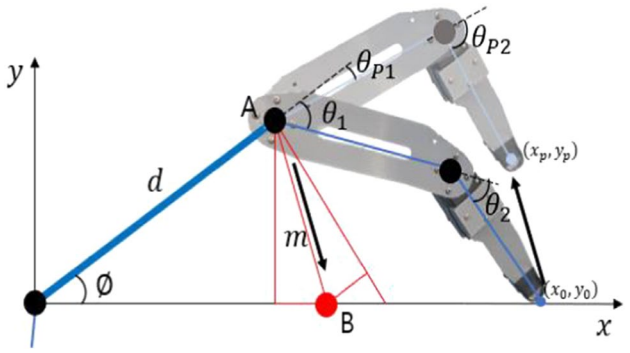


Fig. 7 Landing leg coordinate system

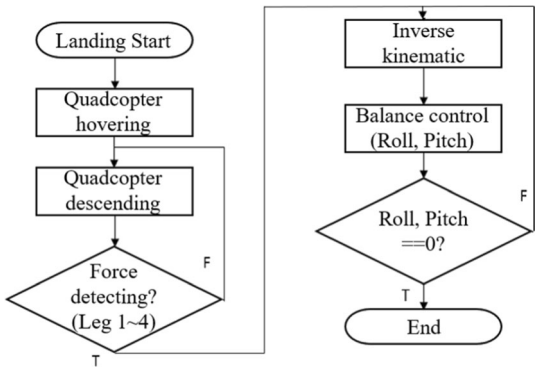


Fig. 8 Landing algorithm flowchart

$$y_\phi = d \sin \phi \tag{33}$$

The coordinates (x_p,y_p) obtained by moving the position of the end effector of the landing legs based on x and y, calculated in Eqs. (32) and (33), are as follows:

$$x_p = x_0 - x_\phi,\ y_p = y_0 + y_\phi \tag{34}$$

Using inverse kinematic Eqs. (16) and (17), θ_{p1} and θ_{p2} were found for the position of the leg at x_p, y_p

$$\theta_{p1} = a\tan 2(y_p, x_p) - a\tan\left(\frac{l_2 \sin\theta_{p2}}{l_1 + l_2 \cos\theta_{p2}}\right) \tag{35}$$

$$\theta_{p2} = \pm a\tan 2(\sin\theta_{p2}, \cos\theta_{p2}) \tag{36}$$

A landing scenario on an obstacle inclined at 20° can be described as three steps: (a) the landing platform has been reached the obstacle, (b) the legs are controlled to keep the drone horizontal to the ground, and (c) the drone is kept horizontal against disturbances cooperating with the legs.

Notice that landing of the quadcopter senses the ground through the force sensor attached to the lower part of the leg and performs balance control when the four legs reach the ground. When the force sensor does not reach the ground, it descends smoothly. At this time, the quadcopter descends in a stationary flight pattern.

Figure 8 shows the flowchart for landing algorithm. When the quadcopter attempts to land following flight, it will initially hover. Following on, the quadcopter will slowly descend and determine whether to make contact with the ground through the force sensor attached to the landing leg. If all the landing legs are in contact, the inverse kinematic is calculated, balance control is performed through the servo motor attached to the joint of each leg, and landing control is completed.

4 Landing Experiments

4.1 Quadcopter Control Simulations

To apply the proposed algorithm to the quadcopter, a thrust test was conducted on the rotor. Before conducting the test, a brushless direct current (BLDC) motor was modeled using Matlab Simulink, and a PID controller and NN-PID controller were designed and simulated. Target revolutions per minute (rpm) were set to 3,000. Then, the two controllers (PID and NN-PID) were simulated to test which achieved the target first.

The rotor thrust test was carried out using a RC benchmark frame. Rotor was used T-motor's MN3508 380KV with a 16-inch propeller. Figure 9 shows the experimental environment configuration when the thrust test was performed.

When the rotor's thrust test was performed using the PID and NN-PID controllers designed in Simulink, the

Fig. 9 The rpm torque test bench

NN-PID controller was superior in speed at reaching the target rpm as shown in the rpm change graph in Fig. 10.

Figure 10 shows rotor's RPM, torque graph and PID gain value change. The NN-PID generated high torque at the start to rapidly reach the target speed. To reach the target RPM, each parameter of PID was changed in real time as shown in Fig. 10(c).

4.2 Landing Experiment

The specifications of the quadcopter used in the experiment are shown in Table 1. The frame of the aircraft was made of 650 mm carbon fiber, and the landing platform was tested by attaching the landing leg proposed in this paper after mounting a 14 inch propeller.

The frame of the landing platform was made of filament material and used through a 3D printer.

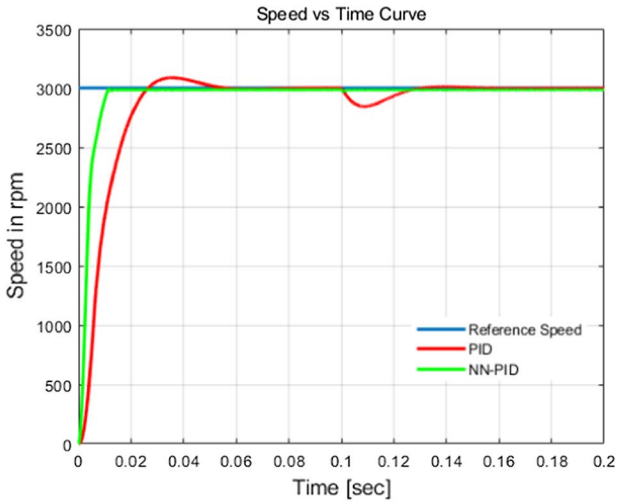
The visual appearance of the quadcopter comprising the specifications in Table 1 is shown in Fig. 11. Each joint of the landing legs comprises a servo motor, and a force sensor is attached to the end effector part of the legs. The sensor is used to determine whether the quadcopter's legs touch the ground.

The experiment comparing the PID controller and cooperative control algorithm proposed in this paper was conducted. First, when the quadcopter was in flight, it was analyzed using altitude data to control how strong it was in response to the weight load caused by the movement of the landing legs. Then, when the quadcopter landed, the landing process was performed in a situation where the slope of the ground was tilted 10°.

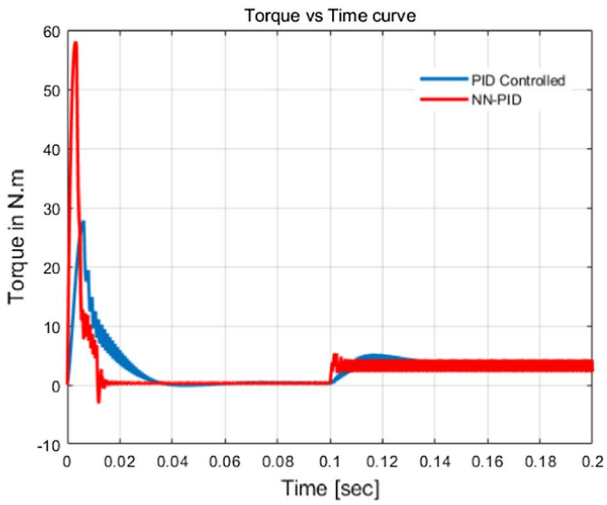
Figure 12 shows the quadcopter landing on a 10° slope. After landing, the quadcopter's roll and pitch altitude errors were less than 0.5°. Figures 13, 14 shows the altitude data acquired during the quadcopter's flight and landing.

In both experiments, the initial 5 s quadcopter altitude data presented a posture error that occurred during the take-off. Quadcopter with a PID controller landed between 30 and 35 s. Landings in experiments with algorithms were performed between 35 and 40 s. When analyzing the experimental data recorded during the flight of the quadcopter, the posture data using PID showed several changes; the flight data of the quadcopter when applying the algorithm showed fewer posture changes compared with those when using the PID controller.

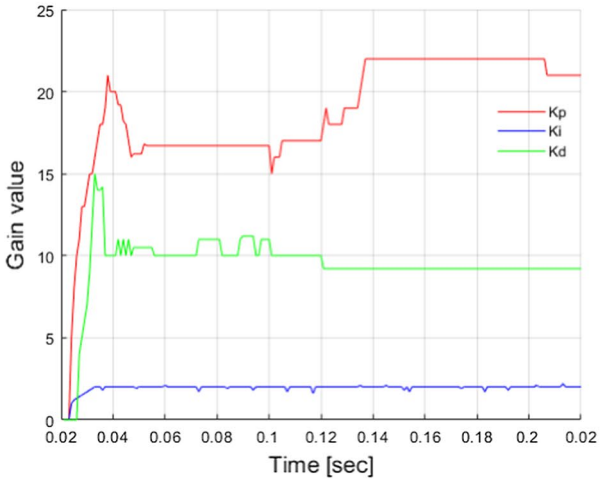
Figure 15 shows the degree of change in PID gain value in real time during the quadcopter experiment. The quadcopter with the NN-PID controller applied the gain value as shown in Fig. 15 in response to the weight load caused by the landing legs and wind disturbance. In this way, it was possible to perform more precise altitude control compared with that using the PID controller.



(a) Speed in RPM



(b) Torque



(c) NN-PID gains

Fig. 10 Rotor test results

Table 1 Quadcopter specifications

	Quadcopter specification
Flight controller	DJI A3 Flight controller (with GPS and IMU)
Motor	T-Motor MN 3508 380 kV BLDC motor
ESC	Hobbywing X-Rotor 40 A (COB Type)
Frame	650 mm Tarot carbon fiber frame
Prop	Dualsky 14×5.5 MR Prop
Battery	22.2 V 8000 mAh Li-Po

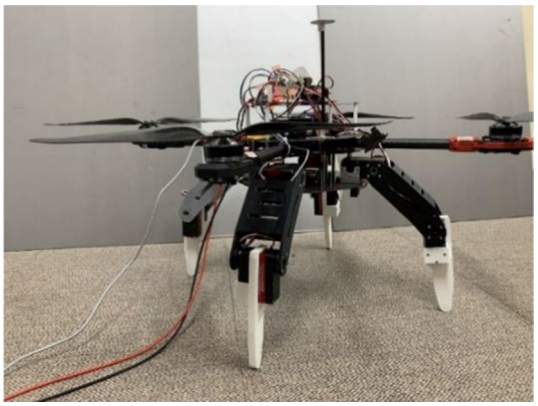


Fig. 11 Quadcopter used in the experiment

Fig. 12 The quadcopter landing experiment (10°slope)

Following on, an experiment was conducted to measure the maximum slope that could be landed on using the landing legs. A slope was created on a flat surface to serve as an experimental environment. The experiment confirmed that it was possible to perform a landing up to a maximum angle of 20° on the landable slope.

However, at a slope above this degree, the landing legs and the arm of the quadcopter collided, so that a stable landing could not be performed.

Figure 16 shows the posture data and experimental photos of the maximum inclination angle for achieving landing using the landing legs, where (a) shows the front and side images when landing at 20° and (b) shows the altitude data

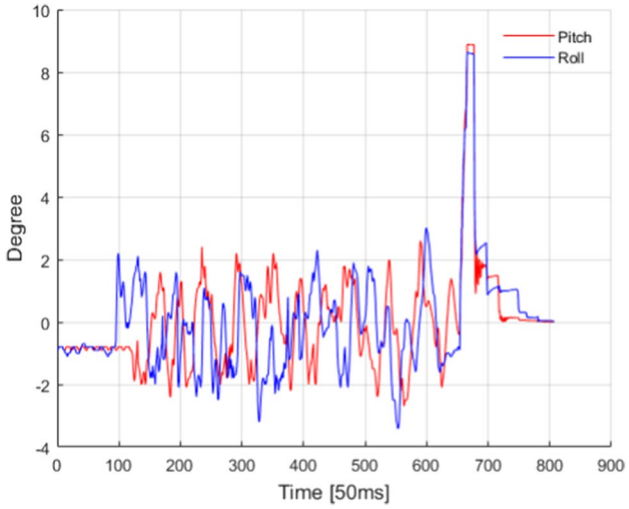


Fig. 13 Altitude data on a slope inclined at 10° (PID)

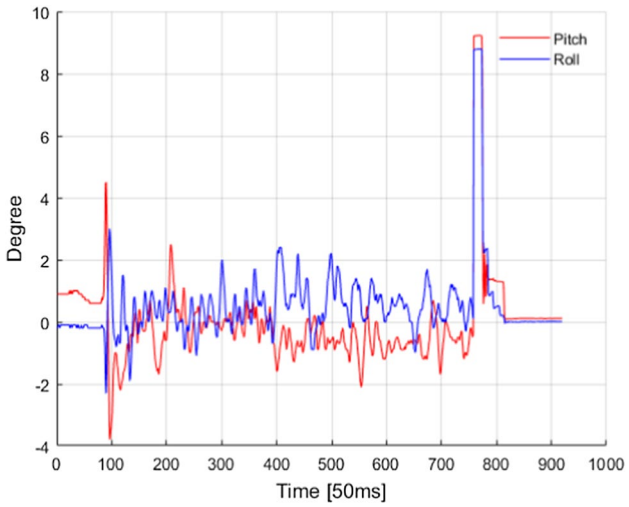


Fig. 14 Altitude data on a slope inclined at 10° (with the application of the algorithm)

of the quadcopter when controlled horizontally in the case of a 20° slope.

5 Conclusions

In this paper, a cooperative control algorithm for a quadcopter and a landing platform which has landing legs to land in various ground conditions has been proposed. The quadcopter was controlled with a robust controller based on an NN-PID to land on the ground safely against the load and wind disturbances during the landing operation. Also an adaptive landing algorithm has been implemented for the landing platform to perform landing stably on a slanted

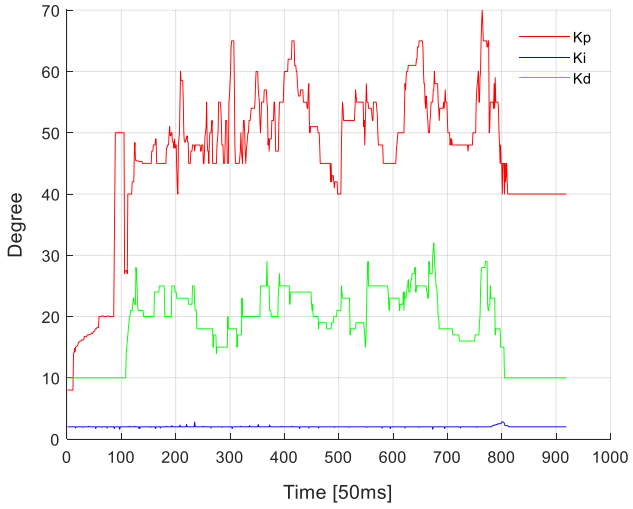
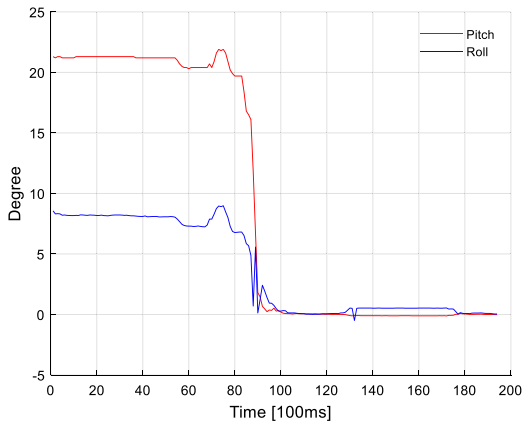


Fig. 15 The PID gain value (NN-PID)

(a)



(b)

Fig. 16 The quadcopter landing experiment (20° slope)

surface cooperating with the quadcopter. The experimental environment has been designed to demonstrate the performance of the robust landing algorithm, which is a slanted surface with a 10° slope. In addition, the maximum angle of inclination that can be landed using the proposed landing gear is 20°, and on slopes above 20°, landing was difficult due to structural issues. To demonstrate the superiority of the proposed robust landing algorithm, two experiments with different algorithms have been performed with the same experimental conditions: (1) the proposed NN-PID and (2) the conventional PID algorithms with optimal gain tuning.

Acknowledgements This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government(MSIT) (No. 2019R1A2C2088859). This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2020R1A6A3A13072460). This paper was supported by Korea Institute for Advancement of Technology(KIAT) grant funded by the Korea Government(MOTIE)(P0008473, HRD Program for Industrial Innovation).

References

1. Eslam, N. M., Ouda, A. N., & Abdelhalim, A. Z. (2016). Mathematical representation, modeling and linearization for fixed wing UAV. *International Journal of Computer Applications, 147*(2), 24–31.
2. Jo, H. J., & Kim, J. Y. (2020). Industrial drone market analysis & forecasts. *The Korean Institute of Power Electronics, 25*(1), 45–48.
3. Raghuwaiya, K., & Chand, R. (2018). 3D motion planning of a fixed-wing unmanned aerial vehicle. In *Asia-Pacific world congress on computer science and engineering*, 241–245.
4. Nurrohaman, M.F., Titalim, B.A., Utama, T.H., & Adiprawita, W. (2019). Design and implementation of on-board hybrid generator for rotary wing drone. In *International conference on electrical engineering and informatics,* 310–314.
5. Szafranski, G., & Czyba, R. (2011). Different approaches of PID control UAV type quadrotor. In *Proceedings of the international micro air vehicles conference 2011 summer edition,* 70–75.
6. Hernández-Guzmán, V., Santibáñez, V., & Campa, R. (2009). PID control of robot manipulators equipped with brushless DC motors. *Robotica, 27*(2), 225.
7. Toha, S. F., & Tokhi, M. O. (2011). PID and inverse-model-based control of a twin rotor system. *Robotica, 29*(6), 929.
8. Khooban, M. H., Alfi, A., & Abadi, D. N. M. (2013). Teaching-learning-based optimal interval type-2 fuzzy PID controller design: A nonholonomic wheeled mobile robots. *Robotica, 31*(7), 1059.
9. Abeywardena, D., Amaratunga, L., Shakoor, S., & Munasinghe, R. (2009). A velocity feedback fuzzy logic controller for stable hovering of a quad rotor UAV, In *Fourth international conference on industrial and information systems,* 558–562.
10. Choi, J., Hwang, D., An, J., & Lee, J.-M. (2019). Object detection using CNN for automatic landing of drones. *Journal of the Institute of Electronics and Information Engineers, 56*(5), 82–90.

11. Komatsuzaki, Y., Doi, T., & Tadakuma, K. (2016). Sensor based controlled leg type automatic landing system for aerial vehicles. *IEEE Sensors, 2016*, 1–3.
12. Sarkisov, Y. S., Yashin, G. A., Tsykunov, E. V., & Tsetserukou, D. (2018). Dronegear: A novel robotic landing gear with embedded optical torque sensors for safe multicopter landing on an uneven surface. *IEEE Robotics and Automation Letters, 3*(3), 1912–1917.
13. Stolz, B., Brödermann, T., Castiello, E. Englberger, G., Erne, D., Gasseret, J., et al. (2018). An adaptive landing gear for extending the operational range of helicopters. In *International conference on intelligent robots and systems,* 1757–1763.
14. Wlas, M., Krzeminski, Z., Guzinski, J., Abu-Rub, H., & Toliyat, H. A. (2005). Artificial-neural-network-based sensorless nonlinear control of induction motors. *IEEE Transactions on Energy Conversion, 20*(3), 520–528.
15. Wang, T., Gao, H., & Qiu, J. (2015). A combined adaptive neural network and nonlinear model predictive control for multirate networked industrial process control. *IEEE Transactions on Neural Networks and Learning Systems, 27*(2), 416–425.
16. Hee Do, K., Byeong Seon, Y., Chae Rin, L., & Seok-Kyoon, K. (2018). Auto-Tuning altitude controller with steepest gradient descent algorithm for quadrotors. In *18th International conference on control, automation and systems,* 867–871.
17. Lin, F., & He, L. Research on the quadrotor of AHRS based on gradient descent algorithm. (2018). In *Eighth international conference on instrumentation and measurement, computer, communication and control,* 1831–1834.
18. Yu, D. H., Park, J. H., Ryu, J. H., & Chong, K. T. (2015). Attitude control of quad-rotor by improving the reliability of multi-sensor system. *Transactions of the Korean Society of Mechanical Engineers, 39*(5), 517–526.
19. Lee, K.U., Yun, Y.H., & Chang, W., Park, B.J., & Choi, H.Y. (2011). Modeling and controller design of quadrotor UAV. In *The Korean Institute of Electrical Engineers*, 1922–1923.
20. Park, M.K., Kim, K.B., & Yoon, K.J. (2012). Design and Fabrication of coaxial quadrotor for indoor flight. In *The Korean society for aeronautical and space sciences*, 1837–1840.
21. Muslimin, S., & Istardi, D. (2018). Inverse kinematics analysis for motion prediction of a hexapod robot. *International Conference on Applied Engineering, 2018*, 1–5.
22. Jialiang, G., & Zhimin, Li. (2011). Research on self-tuning PID control strategy based on BP neural network. *International Conference on Electronics and Optoelectronics, 2011*, 16–21.
23. Gyöngy, I. J., & Clarke, D. W. (2006). On the automatic tuning and adaptation of PID Controllers. *Control Engineering Practice, 14*(2), 149–163.

Jiwook Choi received his M.S. degree in Electrical Engineering from Pusan National University, Busan, in 2019. He is currently doing a Ph.D. course in Pusan National University, Busan. His research interests include intelligent control, robust control, and microprocessor.

Donghun Cheon received his B.S. degree in Electronic Engineering from Pukyong National University, Busan, in 2017. Now he is doing his M.S. course in Pusan National University, Busan. His research interests include intelligent control, robust control, and microprocessor.

Jangmyung Lee received his B.S. and M.S. degrees in Electronics Engineering from Seoul National University, Seoul, Korea, in 1980 and 1982, respectively, and his Ph.D. in Computer Engineering from the University of Southern California (USC), Los Angeles, in 1990. Since 1992, he has been a professor at the Intelligent Robot Laboratory, Pusan National University, Busan, Korea. His current research interests include intelligent robotic systems, ubiquitous ports, and intelligent sensors. Prof. Lee is a past president of the Korean Robotics Society, and a vice president of ICROS. He is also the head of the National Robotics Research Center, SPENALO.