



Development, Control Adjustment, and Gesture Recognition of a Quadrotor Helicopter

Yen-Lun Chen^{1*}, Yu-Ming Lu², and Chia-Yu Chang¹

¹National Kaohsiung Normal University

²Collective Elite Taiwan Branch

*Corresponding author: yenlun.chen@mail.nknu.edu.tw

Received: 7th December 2021

Accepted: 24st June 2022

OPEN ACCESS 

Abstract: Quadrotor helicopters (quadcopters) have become popular in recent years; because they are simple to operate and steady, they are used in aerial photography, competitive flying, and search-and-rescue missions. In addition, wireless sensors have enabled gesture recognition, and therefore, this study investigated the use of gesture recognition to control a quadcopter. A quadcopter was built using Arduino NANO and MPU-6050 modules, which provided insights into the flight principles of a quadrotor system; furthermore, a proportional integrative derivative controller was modified for steady flight. A Leap Motion device was used to understand the logic behind hand gestures and test the success rate of each gesture, and was then combined with a Parrot AR.Drone 2.0 to achieve gesture recognition controls and uncover problems with connections between hand gestures. These problems were addressed by changing the code in the gesture control program. The results of the completed gesture control experiment demonstrated high gesture recognition performance as well as the ability to meet the requirements for controlling a quadcopter.

Keywords: Gesture recognition; Gesture control; Leap Motion; Quadcopter.

Introduction

Compared with remote control aircraft, quadcopters stand out for having steady flight without the loss of sensitive controls. Remote control aircraft require much practice, whereas quadcopters are easy to operate. Furthermore, the technology used in the radio controllers for these types of aircraft is quite mature [1]; in addition to the stable transmission and reception of radio waves, they exhibit powerful performance at distances of and above 2 km. Conventional radio controllers control the various functions of an aircraft through different channels, such as the GPS, gyroscope, and level flight; however, multiple functions translate to a complex system of buttons, which results in larger, heavier, and more expensive controllers [2]. Due to the aforementioned considerations, a smart phone was used to replace the radio controller in the present study; it was connected to a wireless module onboard the quadcopter to enable remote control. This system was then used as the basis for

developing gesture control of the quadcopter. The robust development of motion sensors [3] has enabled the concrete representation of gestures in computer simulations [4]. For example, Leap Motion devices are excellent motion sensors that can capture images of human hands and convert them into precise models. This study adopted a Leap Motion device to complete its gesture-controlled quadcopter.

Quadcopters and PID Controllers

Quadcopters are a type of unmanned aerial vehicle (UAV) with a simple structure; they are small in size and light in weight and are propelled by four rotors of the same size, which are symmetrically distributed [5]. Quadcopters typically use an X or + structure [6]; two opposing rotors rotate clockwise, and the other opposing set of rotors rotates anticlockwise, thus achieving torque balance [7]. This stabilizes flight and allows the quadcopter to hover while providing it with considerable agility. Following the development of electronic

components, such as the GPS and a gyroscope for determining the aircraft body's orientation [8], as well as the convenience provided by smart products, UAVs have been widely used to perform tasks such as surveillance, observation, search and rescue, photography, and safety precautions [9]. Because of its simple, steady, and multifunction controllability, the quadrotor configuration is relatively common [10]. Because of the simple rigid structure of quadcopters, they can be dynamically fast but unstable, and therefore, they are viewed as having a system structure that lacks drivability and struggles to maintain balance of the fuselage. As such, a controller is required that can immediately and rapidly control the stability of a quadcopter as well as its altitude and position. For this purpose, the design of the controller algorithm is crucial and requires a great deal of modelling analysis to match the dynamic model of the quadcopter [11]. The most basic open-source system is PID control, which is commonly used in mandatory model configurations after acquiring a quadcopter. Setting the PID parameters allows the signal output model and requirements to be consistent for adjusting the altitude and position of the quadrotor system; however, limitations [12] exist in flexibility and angle changes that are time-consuming to configure and adjust. The other common method of control is the linear quadratic regulator (LQR), which may not outperform PID controllers in terms of feedback speed—in fact, LQR has a longer delay in signal conversion—but its errors in a steady state are relatively few [13]. Sliding-mode controllers (SMCs) are a type of nonlinear control technology that allows quadcopters to be used normally in terms of aerodynamic effects; when external interference causes a quadcopter to develop system instability, linear control technologies such as PID and LQR controllers will cause its performance to drop and the inability to maintain stability; by contrast, an SMC controller will perform a state simulation to correct the problems and overcome the instability [14]. Airplanes, while taking off and landing, can develop the wing-in-ground effect, also called the ground effect. This phenomenon also occurs in quadcopters; while flying along the ground or close to ceilings, a ground effect will occur [15], and the generated air currents will affect the fuselage and result in instability during descent. To

overcome this phenomenon, fuzzy logic control systems can make rapid and efficient improvements; compared with other control systems, a fuzzy logic control system can correct the landing orientation of a quadcopter more quickly to achieve a safe descent. Furthermore, fuzzy logic can be applied to other types of UAVs and helicopters [16]. Based on the aforementioned controllers and their development, innovative technology has led to advanced research on the optimization of UAVs.

The present study mainly used PID controllers, which comprise proportional, integral, and derivative actions. System errors are minimized through the proportional control, and then integral controls are used to eliminate the smallest error; finally, the derivative control is used to accelerate the entire process [17]. These industrial controllers are widely used in engineering and can be applied to numerous engineering conditions. To obtain reasonable dynamic performances and ensure the safety and sustainable use of the equipment, the PID controller's parameters must be adjusted [18]. In other words, because of the complexities of the electromechanical dynamics in the system, quadcopters require very steady control [19]. To achieve precise control, the quadcopter must be stopped in the event of tilting or rotation to adjust the controls. At this time, adjustment of the PID parameters is highly useful. A return to typical PID controls can be achieved through an Arduino platform; compared with other types of flight platforms, the overall cost of an Arduino platform is lower for achieving the same performance. Using radio controls, a PID controller can control requirements without failure when faced with complex operations [20]. This is the main reason a PID controller was selected for the present study. Moreover, it could conveniently be controlled through Bluetooth from a smartphone app.

This paper has four sections: Section 1 presents the background and motivation behind this study on quadcopters as well as a summary of the relevant literature; Section 2 illustrates the development of a self-built quadrotor helicopter (quadcopter) and proportional integrative derivative (PID) controllers. Section 3 expounds on Leap Motion devices and experiments of gesture recognition. Section 4 concludes this paper.

Construction of the Quadcopter and Testing of the PID Controls

This section first introduces the principles behind quadcopters and the electronic components that were used to form the flight control board, such as the Arduino Nano and MPU-6050. This is followed by a description of the soldering and assembly of the quadcopter. The final subsection presents the programming and PID control

Yen-Lun Chen received B.S. and M.S. degrees from Department of Electrical Engineering at National Taiwan University, and Ph.D. degree from Department of Electrical and Computer Engineering at the Ohio State University. Her research interests include machine learning, pattern recognition, computer vision, and multimedia signal processing.

Yu-Ming Lu received Master degree from National Kaohsiung Normal University in 2019, and Bachelor degree from National Kaohsiung Normal University in 2017. His research interests include gesture recognition and computer vision.

Chia-Yu Chang received Master degree from National Kaohsiung Normal University in 2021, and Bachelor degree from Chinese Culture University in 2018. His research interests include gesture recognition.

experiment.

Principles and Components

A quadcopter has four rotors, which extend in four directions and are positioned symmetrically. The rotors generate torque in different directions, which cancel each other out to achieve a stable and horizontal flight attitude without spinning. Given this equilibrium, when the rotational speed of all four rotors is increased or decreased simultaneously, the increase or decrease in thrust will result in ascending or descending movements. Increasing or decreasing the thrust of two neighboring rotors can create forward, backward, leftward, and rightward flight maneuvers; by contrast, increasing or decreasing the thrust of two diagonal rotors can achieve clockwise or anticlockwise rotation. These changes in actions provide quadcopters with their agility, which enables diverse applications such as competitive flying, aerial photography, and terrain exploration.

Based on this understanding of the principles, we began to construct the flight control board, which mainly comprised an Arduino Nano board and an MPU-6050 six-axis motion-tracking device. The Arduino Nano was paired with an ATmega328 microcontroller, which has comprehensive functions and a small body; the specifications are presented in Table 1. Stored programs can easily be executed through the Arduino compiler. Furthermore, because the Arduino Nano can be combined with multiple Arduino circuit boards or other modules, it can easily complete programs and control tasks multiple functions, which makes it suitable for the development of flight control boards.

Table 1. Specifications of the ARDUINO NANO Board.

<i>Microcontroller</i>	<i>ATmega328</i>
Operating voltage (logic level)	5 V
Input voltage (recommended)	7-12 V
Output voltage (limit)	6-20 V
Digital I/O leads	14 (six of which provide the PWM output)
Simulation input leads	8
DC of each I/O lead	40 Ma
Size	0.73" x 1.70"
Length	45 mm
Width	18 mm

The MPU-6050 six-axis motion-tracking device is the body of the flight control board; its specifications are presented in Table 2. It functions as both an accelerometer and a gyroscope. An accelerometer measures acceleration and can detect the acceleration of an object. It can also detect gravitational acceleration due

to the Earth’s gravity. When not in motion, an accelerometer can also be used as an inclinometer through measuring the angle of inclination using the projection of gravity on its three axes; because gravity is constant on Earth, the gravitational acceleration will not change over time, and therefore, an accelerometer has considerable accuracy in long-term data detection. Gyroscopes are used to measure the angular velocity of an object’s horizontal rotation and can accurately and sensitively calculate data in a short time. However, a gyroscope cannot detect the center of gravity. The MPU-6050 combines both functions into one module, making it highly suitable as a sensor for aircraft rotation and tilt.

Table 2. Specifications of the MPU-6050.

<i>MPU-6050</i>	
Power supply	3–5 V (internal low dropout voltage regulator)
Method of communication	Standard I2C protocol
Sensor type	Three-axis accelerometer, three-axis gyroscope

Soldering and Assembly of the Flight Control Board

The flight control board comprised an Arduino Nano and a MPU-6050, which were connected by a circuit board along with the pins required for assembling the module, as depicted in Figure 1. First, the MPU-6050 required eight Dupont-style pin header housings, and the Arduino Nano required 30 Dupont-style pin header housings; a breadboard was used to secure the pin locations (Figure 2). Due to the necessity of a gyroscope, the pins had to be soldered straight to ensure the balance of the aircraft when horizontal. The eight pins were inserted into the breadboard, and then the MPU-6050 was placed on top. We began soldering when we were sure there was no tilting and that the soldering iron temperature was sufficiently high. The key was to preheat the metal sheets at the connection points; this way, after the solder melted, it covered the pins and contacted the metal more easily. This is a summary of the basic soldering process, which was repeated in subsequent fabrication of other pins. The completed flight control board is depicted in Figure 3.

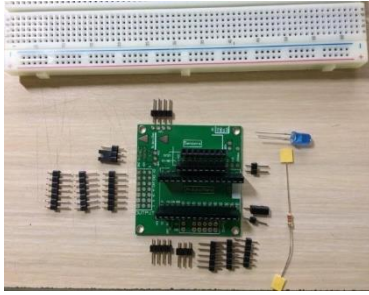


Figure 1. Flight control board, single row pin headers, Dupont housings, resistor, and LED arrangement.

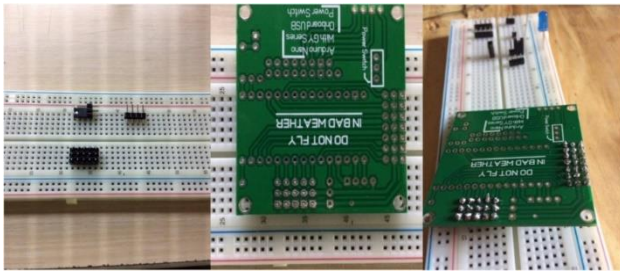


Figure 2. Schematic of flight control board soldering.



Figure 3. Completed flight control board.

Once we had completed the flight control board, we began to assemble the quadcopter. First, we confirmed the location of the quadcopter's four rotors because they had to be connected by circuitry to the drive board, which is the key module for controlling their speed and configuration. Therefore, the circuitry connections had to be verified to ensure the success of the experiments. The power supply of the quadcopter was a 3.7 V lithium battery, but because the flight control board requires 5 V, a boost module was employed to increase the voltage output to 5 V; a multimeter was used to confirm that the voltage, after being connected to the boost module, was sufficient. Lastly, we ensured that the upload and download connectors of the Bluetooth module were not crossed with those of the Arduino Nano. This was to ensure that the quadcopter could successfully send and receive signals. Accidental upload-to-upload and download-to-download connections would prevent data transfers between the two components, not to mention the successful connection of the quadcopter or successful flight control. Once the connections had been verified, all the components were affixed using double-sided tape to an acrylic board, which was cut in advance for the

completed quadcopter.

Program Settings and PID Control Experiment

Once we had completed the quadcopter, we began the software settings. First, no program was saved on the Arduino Nano, and therefore, it had no functions. The necessary program was temporarily saved on the Arduino Nano using a compiler. Before importing the program, the function settings had to first be verified. Once the overall settings were completed, the compiled program was imported to the Arduino Nano, thus completing the software installation for the quadcopter.

Next, the successful operations of each module of the quadcopter were verified using the MultiWii flight control program MultiWiiConf. This program was also used to adjust the sensitivity and balance of the PID controls. The adjustments were made by directly observing the flight changes of the quadcopter and making adjustments based on the necessary flight performances.

Next, we began experimenting with adjusting the PID controls of the quadcopter. We began by adjusting the P value because it is the proportional control that directly affects the output rotation speed of the quadcopter and can be visibly observed and adjusted. First, the P values of the three axes—roll, pitch, and yaw—were reverted to zero, then the roll and pitch P values were increased in increments of 0.5 while we observed the quadcopter's attitude. When the P parameters were 6.5, the quadcopter was able to take off steadily and maintain balance in the air. When the P parameters were smaller than 6.5 and decreased gradually, the rotors were unable to rotate due to the insufficient voltage output, and the quadcopter failed to take off. When the P parameters were greater than 6.5 and increased gradually, the excessive oscillation amplitude resulted in an unstable voltage output from the motor; although the quadcopter was able to take off, it also demonstrated obvious shaking, and the fuselage was unable to maintain balance for steady flight.

After we adjusted the P values of the PID controller, the quadcopter was able to take off and land normally. However, its direction could not be controlled as the I values of the integer controls had not yet been set, and thus, the machine was unable to perform error corrections to control the direction. Therefore, we began adjusting the three axial I values while simultaneously adjusting the yaw P value. First, the three axial I values were reset to zero. When the roll and pitch I values were 0.033, the yaw I value was 0.066, and the yaw P value was 8.91, the quadcopter was able to control its direction steadily and rotate in the air simultaneously. While observing the quadcopter throughout the adjustments,

when the parameters were higher, the quadcopter could only fly and rotate at a low altitude, and although the directions could be controlled, the changes were not sufficiently visible; when the parameters were lower, the direction of the quadcopter could not be controlled, and while flying at a higher altitude, the quadcopter was highly unstable, suddenly changing altitude and spinning.

Up to this point in the experiments with adjustments to the PID controllers, the quadcopter was able to take off normally and change directions. However, steady controlled flight had not yet been achieved regarding subtle operations, which required the adjustment of the Derivative (D) values of the PID controls for calibrating the sensitivity of the quadcopter to human controls. First, the three axial D values were reset to zero before being increased while the flight differences of the quadcopter were observed. The roll and pitch D values of 12 with a yaw D value of 7.2 resulted in the steadiest control. When the D values were lower, the oversensitive controls and excessively rapid output response time caused the quadcopter to charge or the fuselage to flip over, resulting in a crash. Conversely, higher D values resulted in delayed responses, which caused the quadcopter to be overly balanced and unable to change direction.

Control of the Quadcopter Using Gestures

Leap Motion technology provides high precision and a broad detection range in hand tracking, opening up the possibility of replacing mouse and keyboard controls by capturing hand images. Leap Motion also provides powerful support and can be developed in C#, C++, Java, and Python environments.

Static and Dynamic Gestures

The gestures used in this study were distinguished into static gestures and dynamic gestures. Static gestures refer to static hand postures. The static gestures used in this study were observed to have their own pitch, roll, and yaw data that happened to correspond with the quadcopter's three axes, demonstrating that hand axial changes can be used to control the quadcopter's axial changes. For example, if the user's hand tilts forward, the quadcopter will also tilt and move forward. This feature could be used to change the tilt angle of the quadcopter, which was sorted into four directions: forward, backward, left, and right. This was mainly accomplished through pitch and roll changes. Because of these axial changes, the quadcopter tilts in these four directions, resulting in movement forward, backward, to the left, or to the right.

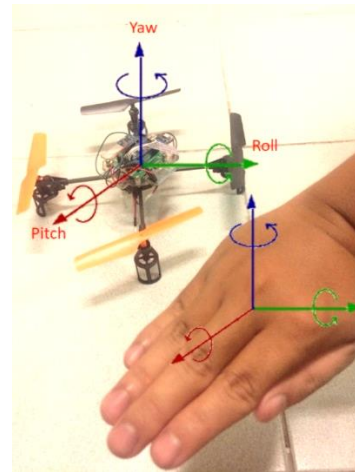


Figure 4. Controlling the quadcopter's direction through static gesture.

The static gestures based on different tilts of the hand are presented in Figure 4: when the palm of the hand rotates approximately 30° clockwise, relative to the roll axis, the hand will exhibit a downward tilt, and the quadcopter will fly forward simultaneously. Conversely, if the palm rotates anticlockwise, the hand will tilt up, and the quadcopter will fly backwards. When the palm rotates approximately 30° clockwise, relative to the pitch axis, the palm will tilt to the right, and the quadcopter will thus fly to the right. Conversely, if the palm rotates anticlockwise, the hand will tilt to the left, and the quadcopter will fly to the left. These are the uses of the static gestures. Through observations of simultaneous changes in the user's hand and the quadcopter, the quadcopter's direction of flight could be intuitively controlled.

Dynamic gestures refer to hand movements, and dynamic gesture recognition uses these dynamic hand movements as the program's basis of determination. Dynamic gestures were further distinguished into trajectory gestures and vector gestures. As suggested by the name "Circle" is listed as a gesture in the LeapSDK inbuilt data, a trajectory gesture occurs a circular hand movement trajectory is detected; the program will make a positive or negative determination based on whether the trajectory is clockwise or anticlockwise. Vector gestures are horizontal (left and right) or vertical (up and down) movements of the hand. The program detects changes to the hand position; for example, if the user's hand moves approximately 10 cm from its original location to the right, the program will determine that the hand movement is to the right. Based on this principle, the program can distinguish four vector gestures.

When applied to the quadcopter, static gestures control the relatively intuitive directional movements, whereas dynamic gestures control the movements of the quadcopter itself, such as take-off, landing, rotating, and vertical ascent and descent (when flying). As depicted in Figure 5, upward or downward gesture vectors represent

the upward or downward motion of the quadcopter while flying, whereas the hand moving left or right represents the quadcopter spinning clockwise or anticlockwise. For take-off and landing, to avoid the gestures being too simple and easily confused with other gestures, trajectory gestures were used as the basis of determination. A clockwise circular trajectory causes the quadcopter to take off, whereas an anticlockwise circular trajectory causes it to land.

Gesture Control Program Concepts and Flow

In the Leap Motion gesture recognition experiments, gestures were not distinguished solely by their trajectories or directions of movement. The main sorting principle was the program’s determination logic. Each gesture has its down logic gate, and once the logic gate is passed, the gesture is distinguished by a positive or negative value. If the Leap Motion device detects a gesture trajectory that controls the quadcopter’s take-off or landing, the program will receive the signal and pass it through the logic gate; then, after reading the obtained value, the program sends

a signal to the quadcopter to execute.

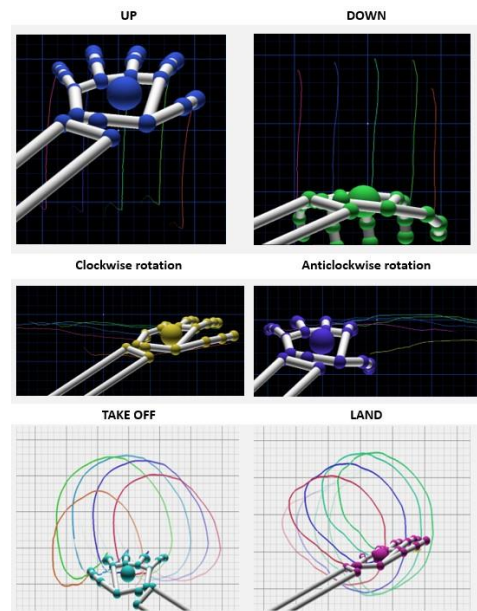


Figure 5. Controlling the quadcopter’s direction through dynamic gestures.

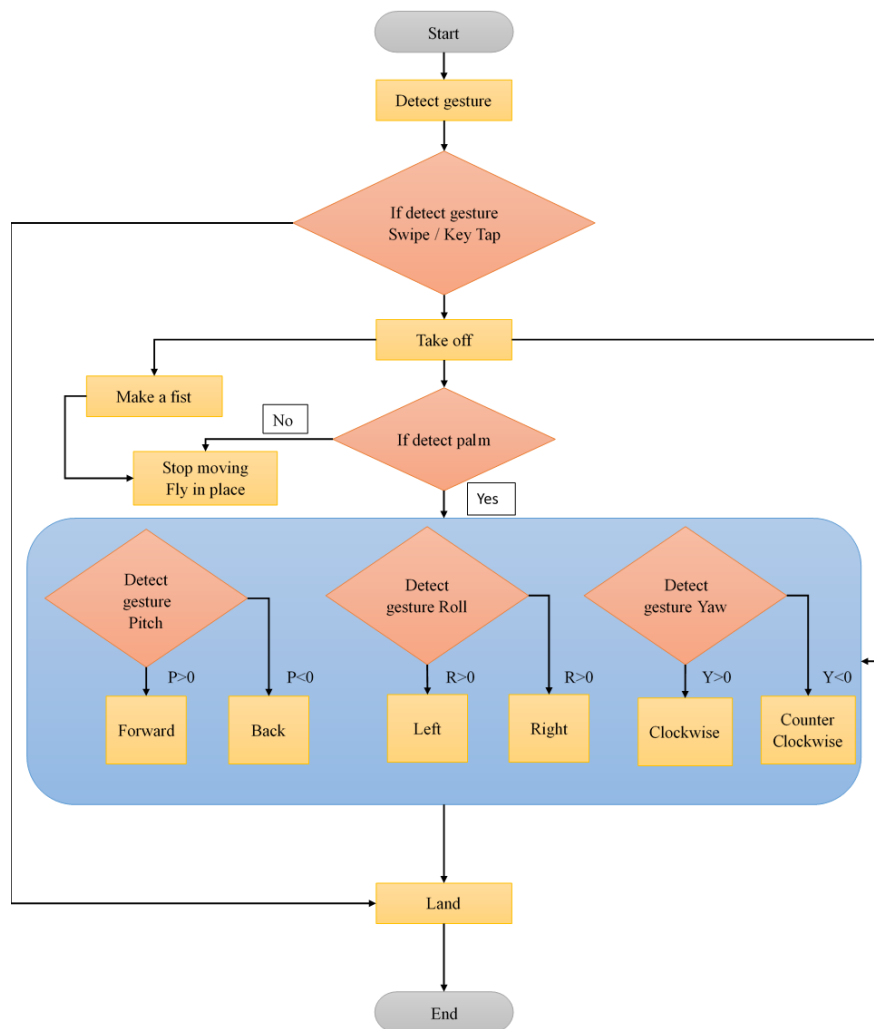


Figure 6. Gesture control program flowchart.

If the value is smaller than 0, the quadcopter will take off; values greater than 0 result in landing. After the program is initiated (START), it will start to detect whether the dynamic hand gesture is present. This logic gate determines whether the quadcopter can initiate flight and is the major premise for controlling the flight direction. After the successful detection of the gesture and take off, the program enters the next logic gate and detects whether the hand is open or closed in a fist. When the hand is open, the program will begin detecting the four logic gestures that control direction. Detected vertical hand movements can control the upward and downward movements of the quadcopter. Detected horizontal hand movements make the quadcopter rotate clockwise or anticlockwise. Detected hand movements along the pitch axis make the quadcopter move forward or backwards, whereas movements along the roll axis prompt the quadcopter to move to the left or right. In the overall process, gestures after take-off must all return to the initial logic gate. To stop the quadcopter, when the hand is successfully detected as making a gesture key tap in the air, a landing command is sent to the quadcopter, ending the program and terminating operations.

Gesture Control Experiments

A preliminary control practice was performed before the experiment. A Parrot AR.Drone 2.0 quadcopter was used in this part, and in the Node.js execution context this device is fully compatible and successfully connected to a computer through a Wi-Fi transceiver. The actual practices are depicted in Figure 7 with images of conditions while moving forward, backward, to the left, and to the right. The tilt of the hand can be observed to be nearly identical to tilt of the quadcopter; this demonstrates that controlling the movements of the quadcopter through static gestures was successful and intuitive. After several practices, a control problem was found: the Leap Motion device had a limited range of detection despite its wide lens range, and therefore, when the user's hand moved out of the lens range during the hand control process, the quadcopter continued to execute the last image recognition command, but the program did not end; when the hand returned to the detection range, because of the change in gesture, the quadcopter lost control and made an emergency landing. Our solution was to add a logic gate to the control procedure (Figure 6); that is, the hand must be detected as open to enter the direction control logic gate. If the program determines the hand is not open, it will send out a "STOP" command that tells the quadcopter to hover in the air until the hand is once again detected as open, which is when the direction can be controlled. If the hand

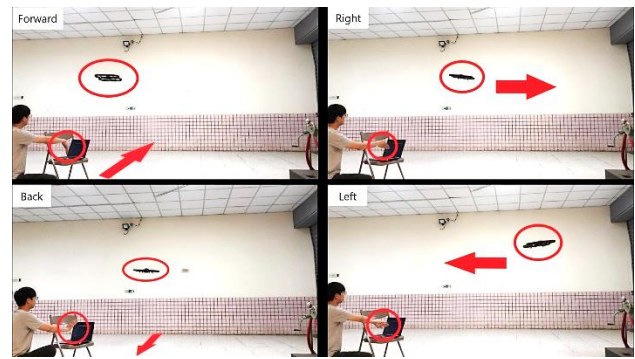


Figure 7. Gesture control practice with the Parrot AR.Drone 2.0.

moves out of the lens range, to avoid losing control of the quadcopter, the user can immediately make a fist in front of the Leap Motion device; the program will determine that the hand is not open and send a command to the quadcopter to hover in the air, successfully resolving this minor control problem. Although whether the hand is open is expressed in a descriptive manner, from a program perspective, this determination is based on whether a certain number of fingers are detected. When testing the Leap Motion device, we found that it could determine the number of fingers and therefore, we used this ability as the basis. When the number of fingers is greater than 0, the program will determine that YES, the hand is open, and continue to detect the hand gesture to control the direction of the quadcopter; when the number of fingers is 0, the program will determine that NO, the hand is not open, and the quadcopter will hover in the air until the number of fingers is determined to be greater than 0 before the flight direction is controlled again.

Once the gesture controls of the quadcopter had been finalized, the final experiments could begin. After confirming that the Leap Motion device was connected to the computer and was running normally, we connected the Parrot AR.Drone 2.0 to the computer by Wi-Fi and began running the program, thus initiating the experiment. The experiment process was recorded starting from the take-off gesture, and after take-off, we began controlling the quadcopter's direction—to the left, to the right, forward, backward, up, and down, followed by clockwise and anticlockwise rotation, and ending with landing. A successful landing marked the completion of one experiment, and a total of 50 tests were performed. If a gesture had to be executed twice or more to make the quadcopter move, it was logged as a failure. Furthermore, there was an interval of 5 seconds between each gesture; during this interval, if the user's hand was discovered to have deviated from the Leap Motion device's range of detection, the user would make a fist to make the quadcopter hover in place, and then move back into the range of detection before continuing the experiment. The results of all 50 tests are presented in Table 3, which

demonstrate that recognition failures for moving left and right were more evident than those of other gestures. The reason is that when the hand's movement along the pitch axis was greater than 60° from horizontal or near vertical, the Leap Motion device could not successfully identify the gesture. When the gesture was maintained under 60° and gently waved for approximately 1 second, the direction of the quadcopter was controlled according to the gesture. All failures in the overall experiment were concentrated in the first 10 tests, which was determined to be caused by the user's lack of familiarity with the gesture recognition controls; however, after 10 attempts, the familiarity led to increased success, and the number of times that each gesture was successfully recognized in one execution increased. The aforementioned gesture connection problem was also successfully resolved through determining the number of fingers. All in all, the gesture recognition controls met expectations and were able to control the quadcopter normally.

Table 3. Results of the gesture recognition control experiments.

Gestures	Success	Total	Success rate
Right	43	50	86%
Left	41	50	82%
Back	47	50	94%
Forward	48	50	96%
Couterclockwise	48	50	96%
Clockwise	47	50	94%
Down	48	50	96%
Up	47	50	94%
Land	46	50	92%
Take off	45	50	90%

Conclusion

In this paper, the fabrication of the quadcopter is presented, which also provided insights into the operation of some of the major electronic components and how the overall structure operated. These insights were helpful in the later PID controller adjustments, specifically the rapid replacement of parts during malfunctions caused by collisions and return to the experiment. The quadcopter manufactured in this study was formed with an Arduino Nano, which is cheaper than professional flight control boards; wireless control was achieved through a smartphone Bluetooth connection rather than a conventional radio controller, keeping the overall manufacturing costs down while meeting the control efficiency requirements. In the PID control experiments, adjusting the PID values of the MultiWii flight control program enabled the quadcopter to achieve steady flight, provided insights into its flight principles, and enabled an

analysis of the PID controller's effects on the four motor units, which yielded the PID control parameters suitable for the developed quadcopter. Gesture recognition was combined with the quadcopter for gesture control; we took advantage of the high performance of the Leap Motion device in hand imaging and recognition to understand the principles in gesture recognition and analyze dynamic and static gestures. These insights were applied to controlling the quadcopter to move in four basic directions as well as rotate, take off, and land. The results of the gesture recognition and the gesture control experiments verified that the Leap Motion device was highly precise in gesture recognition, and also that this type of wireless motion sensor can meet the requirements for controlling a quadcopter.

Acknowledgment


This work was supported in part by the Ministry of Science and Technology under contract MOST 108-2221-E-017-013-MY2.

References

- [1] M.-H. Hung, "Spotlight Editorial: Five Golden Years for Intelligent Automation Technology," *International Journal of Automation and Smart Technology*, vol. 1, no. 1, pp. 19-20, 2011.
<https://doi.org/10.5875/ausmt.v1i1.103>
- [2] M.-H. Hung, Y.-C. Lin, S.-H. Li, H.-C. Yang, and F.-T. Cheng, "Design and implementation of a data exchange platform for TFT-LCD virtual production control systems," *International Journal of Automation and Smart Technology*, vol. 2, no. 2, pp. 83-94, 2012.
<https://doi.org/10.5875/ausmt.v2i2.121>
- [3] M.-J. Tsai, H.-W. Lee, T.-N. Chau, and C.-H. Chao, "An Embedded System for Tracking Human Motion and Humanoid Interfaces," *International Journal of Automation and Smart Technology*, vol. 2, no. 4, pp. 287-293, 2012.
<https://doi.org/10.5875/ausmt.v2i4.144>
- [4] H.-C. Yang, C.-H. Cheng, T.-W. Su, L.-W. Kung, C.-M. Jan, W.-C. Wu, and M.-N. Wu, "Intelligent Sensing Unit for Estimation Roughness of Electrical Discharge Machining," *International Journal of Automation and Smart Technology*, vol. 7, no. 3, pp. 125-131, 2017.
<https://doi.org/10.5875/ausmt.v7i3.1431>
- [5] M. Rabah, A. Rohan, S. A. S. Mohamed and S.-H. Kim,

- “Autonomous moving target-tracking for a UAV quadcopter based on fuzzy-PI,” *IEEE Access*, vol. 7, pp. 38407–38419, 2019. <https://doi.org/10.1109/ACCESS.2019.2906345>
- [6] M. M. Ferdous, S. G. Anavatti, M. A. Garratt and M. Pratama, “Fuzzy clustering based nonlinear system identification and controller development of pixhawk based quadcopter,” in *2017 Ninth International Conference on Advanced Computational Intelligence (ICACI)*, 2017. <https://doi.org/10.1109/ICACI.2017.7974513>
- [7] M. M. Ferdous, S. G. Anavatti, M. Pratama and M. A. Garratt, “A Novel Self-Organizing Neuro-Fuzzy based Intelligent Control System for a AR. Drone Quadcopter,” in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2018. <https://doi.org/10.1109/SSCI.2018.8628815>
- [8] H. Jin, Q. Chen, Z. Chen, Y. Hu and J. Zhang, “Multi-LeapMotion sensor based demonstration for robotic refine tabletop object manipulation task,” *CAAI Transactions on Intelligence Technology*, vol. 1, pp. 104–113, 2016. <https://doi.org/10.1016/j.trit.2016.03.010>
- [9] M. F. Santos, L. M. Honório, E. B. Costa, E. J. Oliveira and J. P. P. G. Visconti, “Active fault-tolerant control applied to a hexacopter under propulsion system failures,” in *2015 19th International Conference on System Theory, Control and Computing (ICSTCC)*, 2015. <https://doi.org/10.1109/ICSTCC.2015.7321334>
- [10] P. H. Nguyen, K. W. Kim, Y. W. Lee and K. R. Park, “Remote marker-based tracking for UAV landing using visible-light camera sensor,” *Sensors*, vol. 17, pp. 1987, 2017. <https://doi.org/10.3390/s17091987>
- [11] P. Wang, Z. Man, Z. Cao, J. Zheng and Y. Zhao, “Dynamics modelling and linear control of quadcopter,” in *2016 International Conference on Advanced Mechatronic Systems (ICAMechS)*, 2016. <https://doi.org/10.1109/ICAMechS.2016.7813499>
- [12] C.-L. Chen, C.-C. Chang, C.-C. Chen, T.-S. Chang, X.-H. Zeng, J.-W. Liu, Z.-W. Wang, and W.-C. Lu, “Developing an Ornamental Fish Warehousing System Based on Big Video Data,” *International Journal of Automation and Smart Technology*, vol. 8, no. 2, pp. 79-83, 2018. <https://doi.org/10.5875/ausmt.v8i2.1693>
- [13] L. M. Argentim, W. C. Rezende, P. E. Santos and R. A. Aguiar, “PID, LQR and LQR-PID on a quadcopter platform,” in *2013 International Conference on Informatics, Electronics and Vision (ICIEV)*, 2013. <https://doi.org/10.1109/ICIEV.2013.6572698>
- [14] Y.-R. Tang and Y. Li, “Dynamic modeling for high-performance controller design of a UAV quadrotor,” in *2015 IEEE International Conference on Information and Automation*, 2015. <https://doi.org/10.1109/ICInfA.2015.7279823>
- [15] F. Ahmadinejad, J. Bahrami, M. B. Menhaj and S. S. Ghidary, “Autonomous Flight of Quadcopters in the Presence of Ground Effect,” in *2018 4th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS)*, 2018. <https://doi.org/10.1109/ICSPIS.2018.8700556>
- [16] M. Talha, F. Asghar, A. Rohan, M. Rabah and S. H. Kim, “Fuzzy logic-based robust and autonomous safe landing for UAV quadcopter,” *Arabian Journal for Science and Engineering*, vol. 44, pp. 2627–2639, 2019. <https://doi.org/10.1007/s13369-018-3330-z>
- [17] B. L. Stevens, F. L. Lewis and E. N. Johnson, *Aircraft control and simulation: dynamics, controls design, and autonomous systems*, John Wiley & Sons, 2015. <https://doi.org/10.1002/9781119174882>
- [18] T. T. Mac, C. Copot, T. T. Duc and R. De Keyser, “AR. Drone UAV control parameters tuning based on particle swarm optimization algorithm,” in *2016 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, 2016. <https://doi.org/10.1109/AQTR.2016.7501380>
- [19] R. S. M. Sadigh, “Optimizing PID Controller Coefficients Using Fractional Order Based on Intelligent Optimization Algorithms for Quadcopter,” in *2018 6th RSI International Conference on Robotics and Mechatronics (IcRoM)*, 2018. <https://doi.org/10.1109/ICRoM.2018.8657616>
- [20] M. F. Silva, A. C. Ribeiro, M. F. Santos, M. J. Carmo, L. M. Honório, E. J. Oliveira and V. F. Vidal, “Design of angular PID controllers for quadcopters built with low cost equipment,” in *2016 20th International Conference on System Theory, Control and Computing (ICSTCC)*, 2016. <https://doi.org/10.1109/ICSTCC.2016.7790668>

Publisher: Chinese Institute of Automation Engineers (CIAE)
ISSN: 2223-9766 (Online)

 **Copyright:** The Author(s). This is an open access article distributed under the terms of the [Creative Commons Attribution License \(CC BY 4.0\)](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are cited.