

The Development of a Visual Tracking System for a Drone to Follow an Omnidirectional Mobile Robot

Jie-Tong Zou * and Xiang-Yin Dai

Department of Aeronautical Engineering, National Formosa University, Yunlin County 632301, Taiwan;
40530149@gm.nfu.edu.tw

* Correspondence: scott@nfu.edu.tw; Tel.: +886-5-6315556

Abstract: This research aims to develop a visual tracking system for a UAV which guides a drone to track a mobile robot and accurately land on it when it stops moving. Two LEDs with different colors were installed on the bottom of the drone. The visual tracking system on the mobile robot can detect the heading angle and the distance between the drone and mobile robot. The heading angle and flight velocity in the pitch and roll direction of the drone were modified by PID control, so that the flying speed and angle are more accurate, and the drone can land quickly. The PID tuning parameters were also adjusted according to the height of the drone. The embedded system on the mobile robot, which is equipped with Linux Ubuntu and processes images with OpenCV, can send the control command (SDK 2.0) to the Tello EDU drone through WIFI with UDP Protocol. The drone can auto-track the mobile robot. After the mobile robot stops, the drone can land on the top of the mobile robot. From the experimental results, the drone can take off from the top of the mobile robot, visually track the mobile robot, and finally land on the top of the mobile robot accurately.

Keywords: visual tracking system; embedded system; drone; omnidirectional mobile robot



Citation: Zou, J.-T.; Dai, X.-Y. The Development of a Visual Tracking System for a Drone to Follow an Omnidirectional Mobile Robot. *Drones* **2022**, *6*, 113. <https://doi.org/10.3390/drones6050113>

Academic Editors: Daobo Wang and Zain Anwar Ali

Received: 2 April 2022

Accepted: 25 April 2022

Published: 29 April 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years the drone industry has seen a boom, and drones are widely applied because they are cheap, light, and safe. A drone positions itself with Lidar, GPS, or an optical flow sensor so that it can fly with autonomy and stability. During the 1980s, research on robotic image recognition began with the rapid development of computer hardware. Then, in the 1990s, with faster computers and advanced cameras, drones were equipped with image recognition. For example, helicopters equipped with vision-based tracking technology are already in use, as studied in [1]. Since GPS signals cannot be received in indoor environments, the positioning method is mostly based on image-based optical flow positioning. In [2], the Lucas–Kanade (LK) algorithm for optical flow localization combines it with the drone tracking of a particular color to realize the drone localization and automatic flight indoors. In [3–5], the image recognition algorithm is designed to recognize ground markers using the camera mounted on the bottom of the drone to achieve automatic and precise landing of the drone. In [6], the developed algorithm was able to detect and track an object with a certain shape on AR. Drone quadcopter, which could follow the line, was able to predict the turn and also to make a turn on the corners. In [7], to solve the problem with complex structures and low resource utilization of a traditional target tracking UAV system based on visual guidance, this paper discusses an implementation method of a quadrotor UAV target tracking system based on OpenMV. In [8], a small drone is taken as an applicable platform, and relevant theoretical research and comprehensive experiments are carried out around monocular vision feature point extractions and matching, UAV target tracking strategies, etc. Finally, a target tracking system will be built in the future to realize real-time face tracking.

This research aims to enable a drone to track a mobile robot only with optical flow sensor and image processes, without the help of GPS, and then to land on the robot when it stops moving.

2. Architecture of Visual Tracking System

2.1. Drone

The drone used in the experiment is Tello EDU, as shown in Figure 1a, which positions itself with an optical flow sensor and can be controlled by SDK commands [9]. Red and blue LED lights are installed on the bottom of the drone, as in Figure 1b, so that the camera on the mobile robot can detect where the drone is.

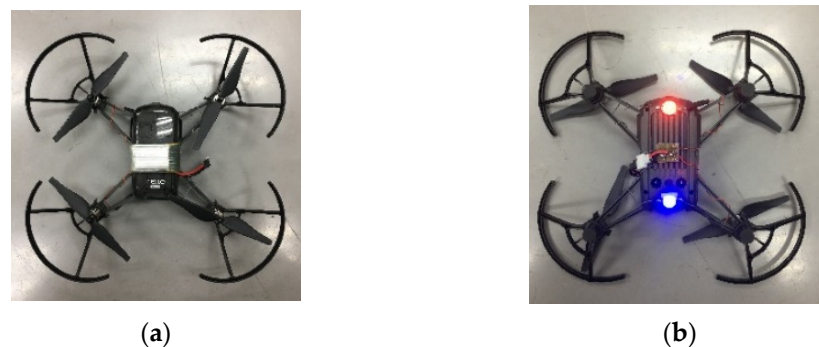


Figure 1. Tello EDU and two LEDs with different colors were installed on the bottom of the drone. (a) Top view; (b) Bottom view.

2.2. Omnidirectional Mobile Robot

Many wheeled mobile robots are equipped with two differential driving wheels. Since these robots possess two degrees-of-freedom (DOFs), they can rotate about any point, but cannot perform holonomic motion including sideways motion [10]. To increase the mobility of this mobile robot, three omnidirectional wheels driven by three DC servo motors are assembled on the robot platform (see Figure 2). The omnidirectional mobile robot can move in an arbitrary direction without changing the direction of the wheels.

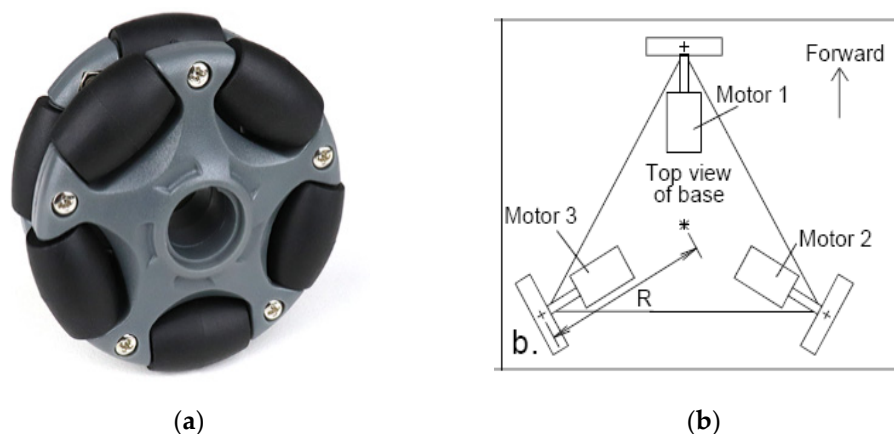


Figure 2. (a) Structure of omnidirectional wheel; (b) motor layout of robot platform.

The three-wheeled omnidirectional mobile robots are capable of achieving three DOF motions by driving three independent actuators [11,12], but they may have stability problems due to the triangular contact area with the ground, especially when traveling on a ramp with a high center of gravity, owing to the payload they carry.

Figure 2a is the structure of an omnidirectional wheel, and Figure 2b is the motor layout of the robot platform. The relationship of motor speed and robot moving speed is shown as:

$$\begin{aligned} V_1 &= \omega_1 r = V_x + \omega_p R \\ V_2 &= \omega_2 r = -0.5V_x + 0.867V_y + \omega_p R \\ V_3 &= \omega_3 r = -0.5V_x - 0.867V_y + \omega_p R \end{aligned} \quad (1)$$

where:

V_i = velocity of wheel i ;

ω_i = rotation speed of motor i ;

ω_p = rotation speed of robot;

r = radius of wheel;

R = distance from wheel to the center of the platform.

Hardware of the proposed system is shown in Figure 3. The mobile robot is adapted from an omnidirectional robot and reads remote control signals with Arduino Due. A web camera is installed on top of the mobile robot to detect the red and blue LEDs on the bottom of the drone, as in Figure 4a,b. A single board computer, Up Board, is installed inside the robot, as shown in Figure 4c. Up Board is equipped with Linux Ubuntu and processes images with OpenCV. It can calculate the heading angle and distance between the mobile robot and the drone, and then send control commands to the drone through UDP Protocol with a Wi-Fi module.

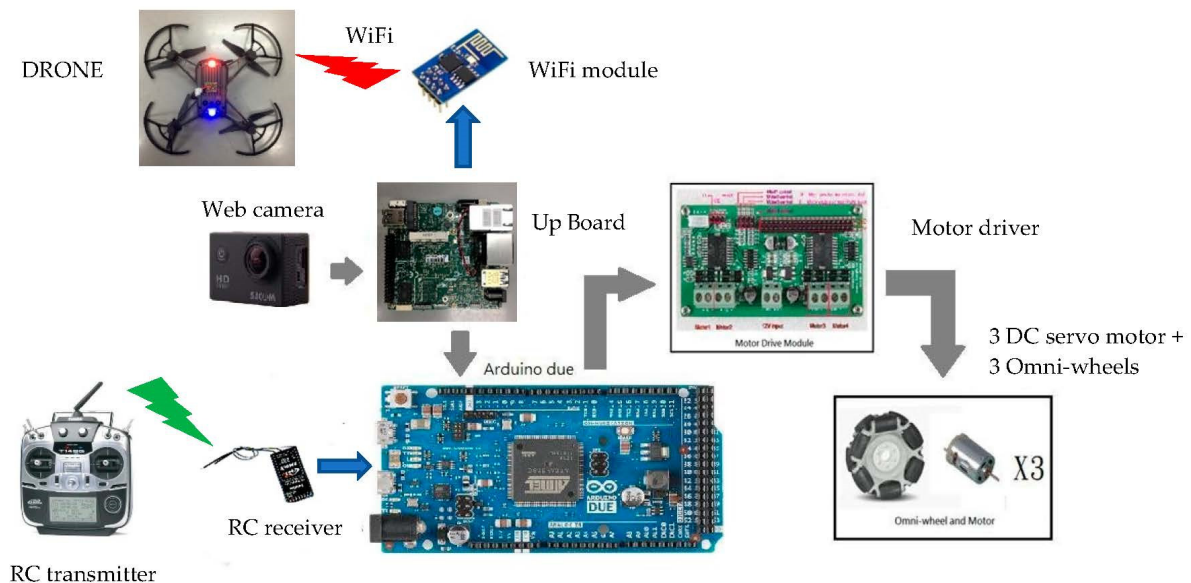


Figure 3. Hardware of the proposed system.

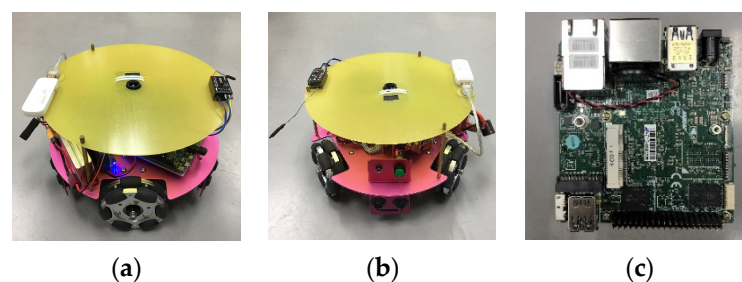


Figure 4. (a,b) Omnidirectional mobile robot installed with camera; (c) the embedded system (Up Board).

3. Positioning Drone through Image Processing

Usually, objects in images have distinct colors (hues) and luminosities, so that these features can be used to separate different areas of the image. In the RGB representation the hue and the luminosity are expressed as a linear combination of the R, G, B channels, whereas they correspond to single channels of the HSV image (the Hue and the Value channels).

The code for image processing is programmed with Python and cited from the open-source library OpenCV. First, images with RGB color space are converted to those with HSV color space so that a color can be easily presented with numbers, as the conversion algorithm in Equations (2)–(4). Suppose R, G, and B are the value of red, green, and blue in a color, and the value is a real number between 0 and 1. Suppose max is the biggest number among R, G, and B, and suppose min is the smallest number among R, G, and B.

$$h = \begin{cases} 0^\circ, & \text{if max} = \text{min} \\ 60^\circ \times \frac{G-B}{\text{max}-\text{min}} + 0^\circ, & \text{if max} = R \text{ and } G \geq B \\ 60^\circ \times \frac{G-B}{\text{max}-\text{min}} + 360^\circ, & \text{if max} = R \text{ and } G < B. \\ 60^\circ \times \frac{B-R}{\text{max}-\text{min}} + 120^\circ, & \text{if max} = G \\ 60^\circ \times \frac{R-G}{\text{max}-\text{min}} + 240^\circ, & \text{if max} = B \end{cases} \quad (2)$$

$$s = \begin{cases} 0, & \text{if max} = 0 \\ \frac{\text{max}-\text{min}}{\text{max}} = 1 - \frac{\text{min}}{\text{max}}, & \text{otherwise} \end{cases} \quad (3)$$

$$v = \text{max} \quad (4)$$

At different altitudes, the LEDs on the drone would manifest different levels of brightness and chroma, so the threshold value cannot be fixed, as in study [13]. A self-tuning threshold is applied to make linear adjustments according to the height of the drone. Images are converted to white and black binary images through image thresholding to filter out red and blue LEDs, as in Figure 5a,b. Images are also put through medium filter, erosion, and dilation to filter noise, as in Figure 5c, even if there is still noise, as in Figure 6a–c. For this situation, we designed a robust enough image recognition algorithm. By utilizing the distance proximity of red and blue LEDs, the red and blue LEDs on the bottom of the drone are accurately identified under noise interference. The flow chart of the algorithm is shown in Figure 7. First, the binary images of the red and blue LEDs are dilated so that the areas of the red and blue LEDs overlap, and then the two images are added (see Figure 8a). After dilation, the binary images of the red and blue LEDs are then AND operated to produce the overlapping part of the red and blue LEDs after dilation (see Figure 8b). Calculating the contours and image moments of Figure 8a,b, the center point of the contours is calculated from the image moments (see Figure 8c,d). Let all contours in the added image after dilation be M_i . Let the contour of the overlapping part after AND operation be S , and the center of this contour be S_{center} . If there is noise in the environment, there will be multiple M_i . So we check whether S_{center} is inside M_i to confirm which M_i is the contour formed by adding up the binary images of red and blue LEDs after dilatation. This contour is called M_{LED} . Based on the previously derived M_{LED} , find which two center points are inside the M_{LED} , then the correct center point of the red and blue LEDs can be determined (see Figure 8e).

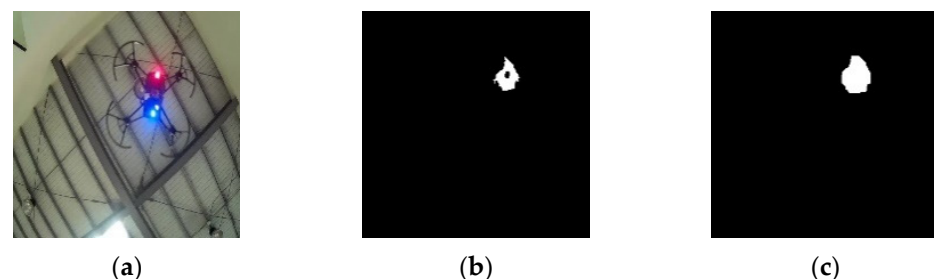


Figure 5. (a) Original image; (b) red LED after thresholding; (c) red LED after median filter, erosion, and dilation.

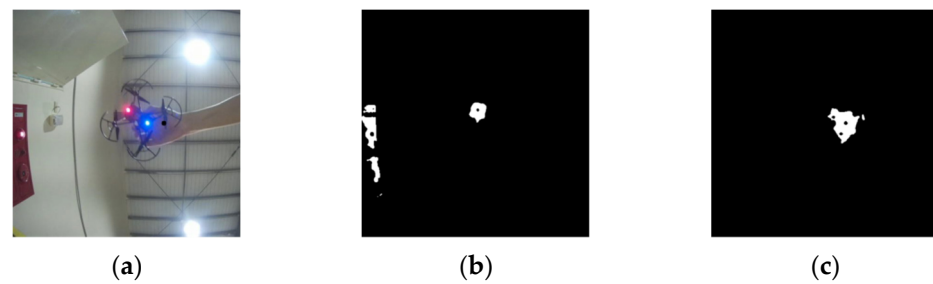


Figure 6. (a) Original image; (b,c) images of red and blue LEDs after thresholding, median filter, erosion, and dilation but with noise.

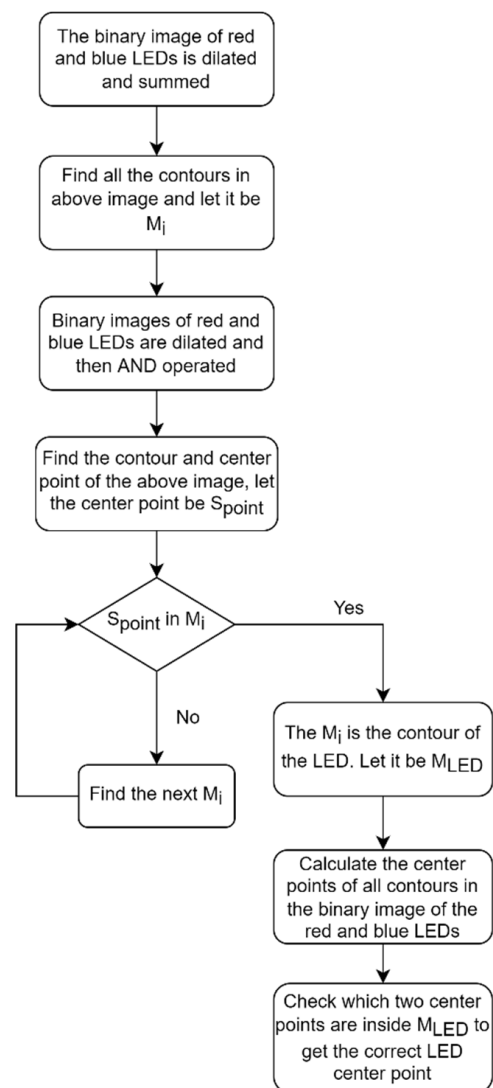


Figure 7. Flow chart of the algorithm to recognize red and blue LEDs.

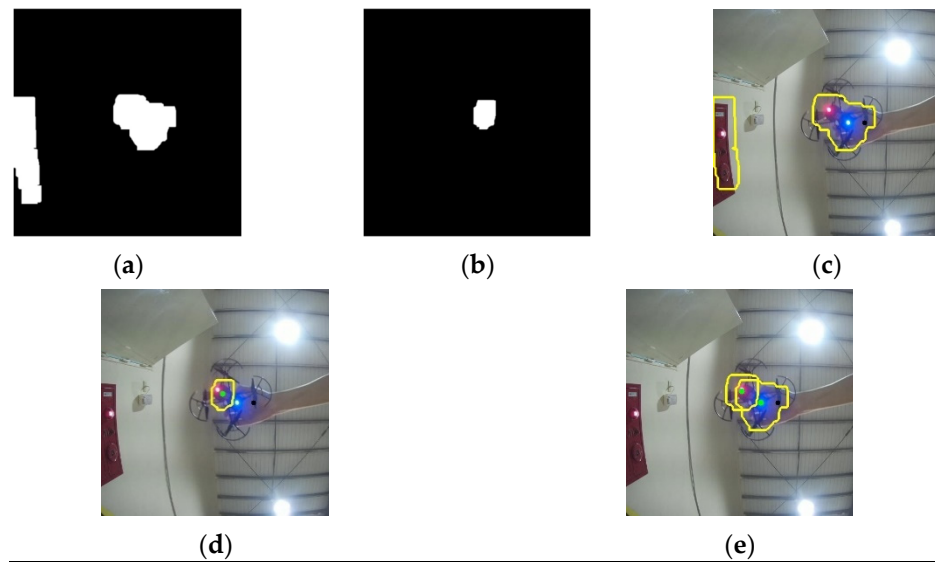


Figure 8. (a) The binary images of the red and blue LEDs are dilated and added together; (b) the binary images of the red and blue LEDs are dilated and operated to obtain the overlap; (c) all contours in Figure 8a; (d) contours and center points in Figure 8b (marked in green); (e) the correct contours and center points of the red and blue LEDs.

There are two coordinate systems in Figure 9: (X_r, Y_r) is the coordinate system of the camera on the mobile robot, and (X_d, Y_d) is the moving coordinate system of the drone. After confirming the contour of red LEDs and blue LEDs, we can figure out the “Image Moments” of red and blue LEDs, which can be used to calculate the coordinate of the central point, denoted as (x_1, y_1) and (x_2, y_2) . With it, the distance between the two LEDs, denoted as d_1 , as in Figure 9, can be calculated (refer to Equation (5)), and that distance d_1 can be used to judge the height of the drone. In addition, the central point of the line between the two LEDs represents the central point of the drone, denoted as (x_c, y_c) (refer to Equation (6)). By calculating the distance between the central point (x_c, y_c) of the drone and the central point (O) of the camera, denoted as d (refer to Equation (7)), the process variable of PID control can be worked out. With the line through two LEDs as the axis, the rotating angle of two LEDs, which is also the heading angle of the drone, denoted as φ , can be calculated. The angle between d and X_r -axis, denoted as θ , can serve as the components for the pitch and roll of the drone.

$$d_1 = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (5)$$

$$x_c = \frac{x_1 + x_2}{2}, y_c = \frac{y_1 + y_2}{2} \quad (6)$$

$$d = \sqrt{(x_c - 320)^2 + (y_c - 240)^2} \quad (7)$$

where:

(X_r, Y_r) : the coordinate system of the camera on mobile robot;

O : the origin of the coordinate system (X_r, Y_r) , and O is also the center point (320,240) of the camera with 640×480 resolution;

(X_d, Y_d) : the moving coordinate system of the drone;

(x_1, y_1) : the central point of the red LED;

(x_2, y_2) : the central point of the blue LED;

d_1 : the distance between the two LEDs;

(x_c, y_c) : the central point of the drone;

d : the distance between the central point (x_c, y_c) of the drone and that of camera (O);

φ : the heading angle of the drone;

θ : the angle between d and X_r -axis.

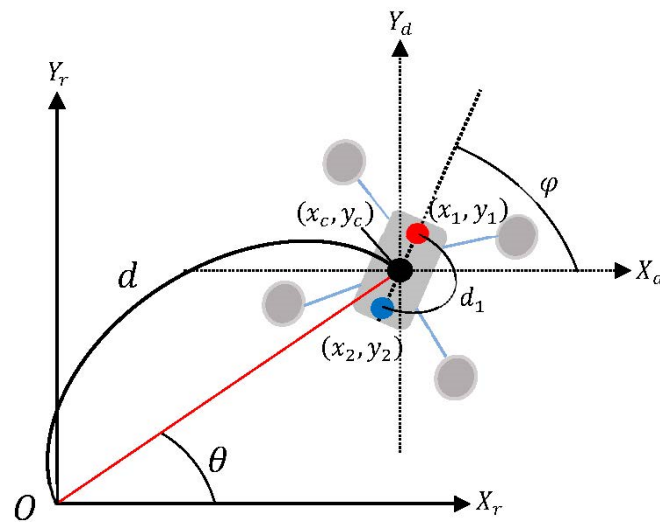


Figure 9. Diagram for calculating distance and angle of drone.

4. Guidance Law

Through image processing, we can obtain d , d_1 , φ , and θ , and a guidance law can be developed. The guidance law can direct a drone to track a mobile robot. Tello EDU can follow SDK 2.0 commands and perform various kinds of simple and quick actions. For example, the “rc” command in SDK 2.0 command is used, as in Figure 10, to control the four moving directions of the drone: pitch, roll, yaw, and throttle (height).

rc a b c d	Set remote controller control via four channels. “a” = left/right (-100-100) “b” = forward/backward (-100-100) “c” = up/down (-100-100) “d” = yaw (-100-100)
------------	--

Figure 10. The description of rc command with SDK 2.0 of Tello EDU [9].

4.1. PID Control

PID (proportional-integral-differential) control is the most frequently used industrial control algorithm because it is effective, widely applicable, and simple to operate. It can also be applied on drones [14]. A popular method for tuning PID controllers is the Ziegler-Nichols method [15]. This method starts by zeroing the integral gain (K_I) and differential gain (K_D) and then raising the proportional gain (K_P) until the system is unstable. The value of K_P at the point of instability is called K'_P ; the oscillation period is T_C . The P , I , and D gains are set as Equations (8)–(10).

$$K_P = 0.6K'_P \quad (8)$$

$$K_I = \frac{2}{T_C} \quad (9)$$

$$K_D = \frac{T_C}{8} \quad (10)$$

This research adjusts the flight velocity in pitch and roll direction with a PID control algorithm. With $u_p(t)$ as the output, the distance between the drone and mobile robot (d) is the process variable, the setpoint (target value) is zero, and process variable minus setpoint is the error value, represented as $e_p(t)$. PID control includes three kinds: proportional,

integral, and derivative. Proportional control considers current error, and the error value will be multiplied by positive constant K_{p_p} . When the value of K_{p_p} increases, the response speed of the system will become faster. However, when the value of K_{p_p} becomes too large, fluctuation of the process variable will happen. Integral control considers that the past error and the sum of the past error value multiplied by positive constant K_{i_p} can be used to eliminate the steady state error. Derivative control considers future error, calculating the first order derivative of error, which will be multiplied by positive constant K_{d_p} , thereby predicting the possibilities of error changes and overcoming the delay of the controlled subject, as in Equation (11). When a drone is tracking a mobile robot, the drone is required to react quickly, so higher K_{p_p} and K_{d_p} values are needed. Firstly, the Ziegler–Nichols method was used to tune the PID controller. After some tuning from experimental results, I set $K_{p_p} = 0.14$, $K_{i_p} = 0.15$, $K_{d_p} = 0.13$. However, this will make a drone extremely sensitive to changes in the moving speed of the mobile robot and cause it to overreact to a very low error value, which will lead to a not so smooth landing. To solve this problem, the K_{p_p} and K_{d_p} values are adjusted according to the height of the drone. If the height of the drone is lower than 60 cm, set $K_{p_p} = 0.075$ and $K_{d_p} = 0.05$, so that the drone does not overreact. Apart from this problem, it takes a while for the drone to fly above the mobile robot when the mobile robot stops moving, because the guidance law cannot predict the flying direction of the drone very accurately. The heading angle of the drone is modified by PID control, as in Equation (12), so that the flying angle is more accurate, and the drone can land quickly.

$$u_p(t) = K_{p_p} e_p(t) + K_{i_p} \int_0^t e_p(\tau) d\tau + K_{d_p} \frac{de_p(t)}{dt} \quad (11)$$

$$u_h(t) = K_{p_h} e_h(t) + K_{i_h} \int_0^t e_h(\tau) d\tau + K_{d_h} \frac{de_h(t)}{dt} \quad (12)$$

4.2. Logic of Guidance Law

As shown in Figure 11, the angle between d and X_r -axis (θ) is used to determine which zone (partition 1) in the (X_r , Y_r) coordinate system the drone is located in. The heading angle of the drone (φ) is used to determine which zone (partition 2) in the (X_d , Y_d) coordinate system the nose of the drone is facing. For example, in Figure 11, θ is in partition 1a ($0 < \theta < 90^\circ$), and φ is in partition 2b ($45^\circ < \varphi < 135^\circ$).

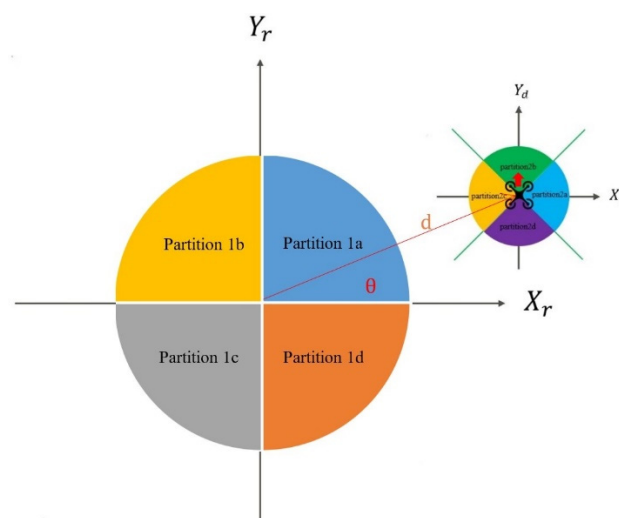


Figure 11. θ is used to determine which zone (partition 1) in the (X_r , Y_r) coordinate system the drone is located in. φ is used to determine which zone (partition 2) in the (X_d , Y_d) coordinate system the nose of the drone is facing.

The flow chart of the guidance law is shown in Figure 12. According to the angle (θ) and the heading angle φ , gained through image processing, the component for the pitch and roll movement of the drone can be figured out from the guidance law in Figure 12. The component for the pitch and roll multiplied by the output of PID control is the flight velocity in the pitch and roll direction.

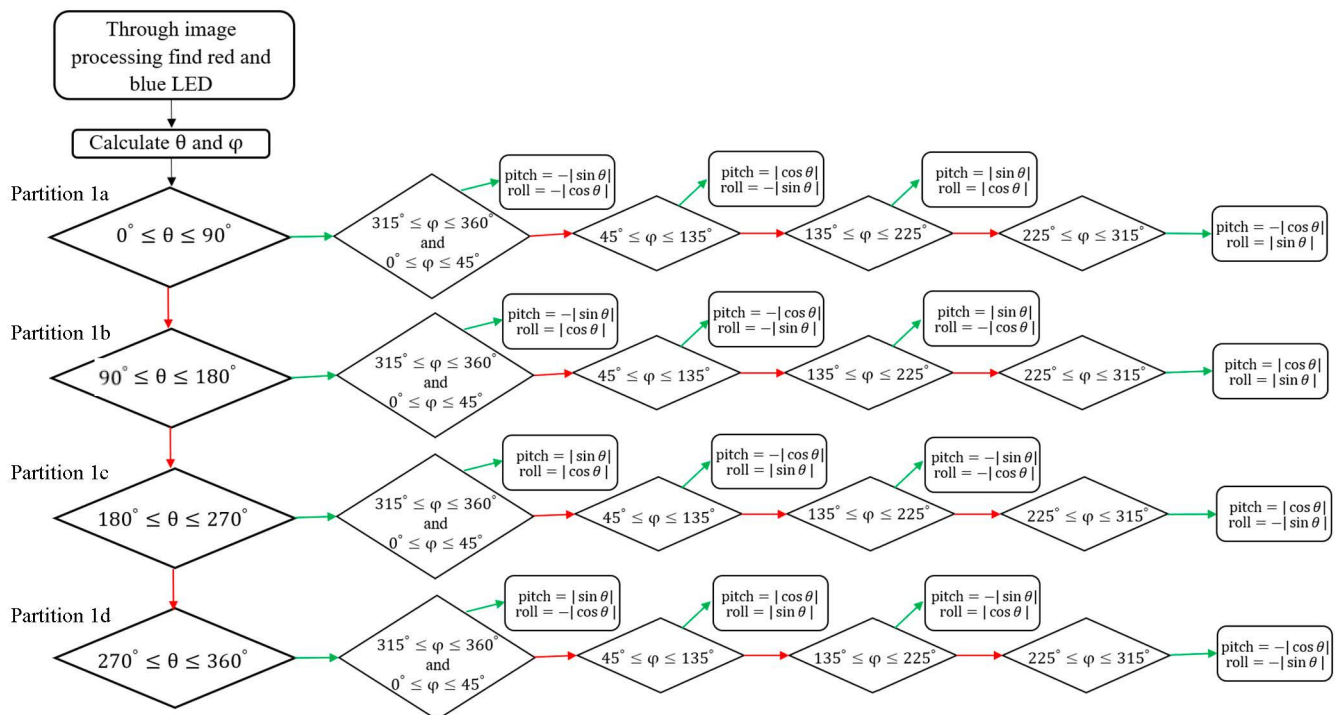


Figure 12. Flow chart of the guidance law.

5. Experimental Results

5.1. Monitoring of Image Processing

This research processes images using a UP Board, which processes 30 images per second. The image processing can be monitored through Windows remote desktop connection, as in Figure 13.

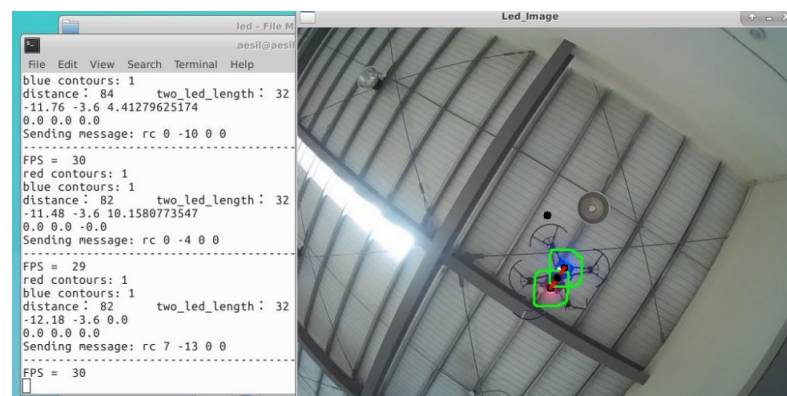


Figure 13. Monitor image processing, red and blue LEDs indicated by the green circle.

5.2. Experimental Results of Visual Tracking

With appropriate PID parameters, the mobile robot can move in any direction with radio control. The drone can accurately track the mobile robot and land on top of it when it stops moving. The experimental results of visual tracking are shown in Figure 14, and

the experimental video can be seen on YouTube (<https://www.youtube.com/watch?v=kRorTz26XSg>) (accessed on 3 January 2020).

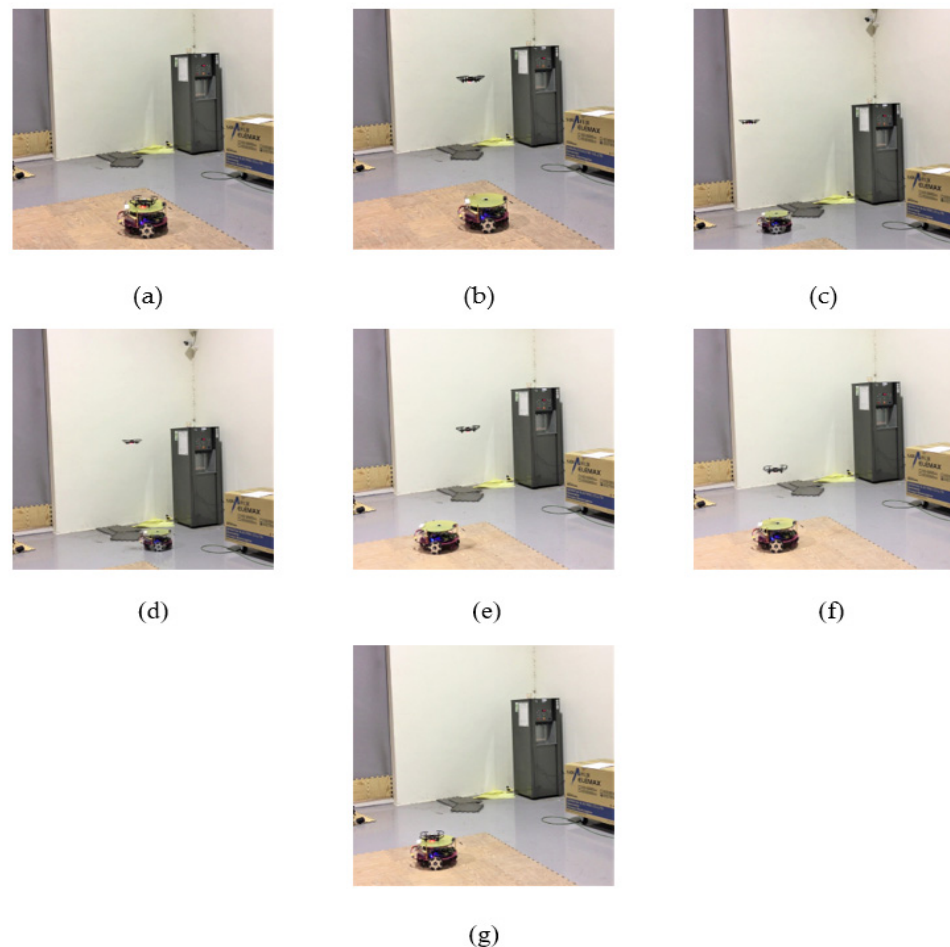


Figure 14. (a,b) Takeoff from the top of the mobile robot; (c,d) tracking the mobile robot; (e–g) when the mobile robot stops moving, the drone can land on the top of the mobile robot accurately (<https://www.youtube.com/watch?v=kRorTz26XSg>) (accessed on 3 January 2020).

1. Figure 14a,b: The drone takes off from the top of the mobile robot.
2. Figure 14c,d: The drone visually tracks the mobile robot.
3. Figure 14e–g: When the mobile robot stops moving, the drone can land on the top of the mobile robot accurately.

6. Conclusions

This research developed a system that enables a drone to track a mobile robot via image processing and to land on top of the mobile robot when it stops moving. The web camera on the mobile robot can capture blue and red LEDs which can determine the heading angle and the distance between the drone and mobile robot. The heading angle and flight velocity in the pitch and roll direction of the drone are modified by PID control, so that the flying speed and angle are more accurate, and the drone can land quickly. Firstly, the Ziegler–Nichols method was used to manually tune the PID controller initially, and make fine PID tuning with experimental results. The PID tuning parameters were also adjusted according to the height of the drone.

The embedded system (Up Board) on the mobile robot, which is equipped with Linux Ubuntu and processes images with OpenCV, can send the control command (SDK 2.0) to the Tello EDU drone through WIFI with UDP Protocol. The guidance law can direct the

drone to track the mobile robot. Finally, the drone can land on the top of the mobile robot when it stops moving.

The proposed system can also guide drones to land on a wireless charger via image processing and can be applied to the auto-tracking of certain mobile objects. In the future, the accuracy with which a drone recognizes a certain object in a complicated background should be heightened, so that the image recognition technology can be more reliable.

Author Contributions: Conceptualization, J.-T.Z.; methodology, J.-T.Z. and X.-Y.D.; software, X.-Y.D.; validation, X.-Y.D.; formal analysis, J.-T.Z. and X.-Y.D.; investigation, X.-Y.D.; resources, J.-T.Z.; data curation, X.-Y.D.; writing—original draft preparation, X.-Y.D.; writing—review and editing, J.-T.Z.; visualization, X.-Y.D.; supervision, J.-T.Z.; project administration, J.-T.Z.; funding acquisition, J.-T.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Formosa University.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Choi, J.H.; Lee, W.-S.; Bang, H. Helicopter Guidance for Vision-based Tracking and Landing on a Moving Ground Target. In Proceedings of the 11th International Conference on Control, Automation and Systems, Gyeonggi-do, Korea, 26–29 October 2011; pp. 867–872.
2. Granillo, O.D.M.; Beltrán, Z.Z. Real-Time Drone (UAV) Trajectory Generation and Tracking by Optical Flow. In Proceedings of the 2018 International Conference on Mechatronics, Electronics and Automotive Engineering, Cuernavaca, Mexico, 26–29 November 2018; pp. 38–43.
3. Shim, T.; Bang, H. Autonomous Landing of UAV Using Vision Based Approach and PID Controller Based Outer Loop. In Proceedings of the 18th International Conference on Control, Automation and Systems, PyeongChang, Korea, 17–20 October 2018; pp. 876–879.
4. Tanaka, H.; Matsumoto, Y. Autonomous Drone Guidance and Landing System Using AR/high-accuracy Hybrid Markers. In Proceedings of the 2019 IEEE 8th Global Conference on Consumer Electronics, Osaka, Japan, 15–18 October 2019; pp. 598–599.
5. Liu, R.; Yi, J.; Zhang, Y.; Zhou, B.; Zheng, W.; Wu, H.; Cao, S.; Mu, J. Vision-guided autonomous landing of multirotor UAV on fixed landing marker. In Proceedings of the 2020 IEEE International Conference on Artificial Intelligence and Computer Applications, Dalian, China, 27–29 June 2020; pp. 455–458.
6. Boudjit, K.; Larbes, C. Detection and implementation autonomous target tracking with a Quadrotor AR.Drone. In Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics, Colmar, France, 21–23 July 2015.
7. Shao, Y.; Tang, X.; Chu, H.; Mei, Y.; Chang, Z.; Zhang, X. Research on Target Tracking System of Quadrotor UAV Based on Monocular Vision. In Proceedings of the 2019 Chinese Automation Congress, Hangzhou, China, 22–24 November 2019; pp. 4772–4775.
8. Sun, X.; Zhang, W. Implementation of Target Tracking System Based on Small Drone. In Proceedings of the 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference, Chengdu, China, 20–22 December 2019; pp. 1863–1866.
9. RYZE. *Tello SDK 2.0 User Guide*, 1st ed.; RYZE: Shenzhen, China, 2018; p. 5.
10. Song, J.-B.; Byun, K.-S. Design and Control of an Omnidirectional Mobile Robot with Steerable Omnidirectional Wheels. In *Mobile Robots; Moving Intelligence*; Zaltbommel, The Netherlands, 2006; p. 576.
11. Carlisle, B. An Omnidirectional Mobile Robot. In *Development in Robotics*; Kempston: Bedford, UK, 1983; pp. 79–87.
12. Pin, F.; Killough, S. A New Family of Omnidirectional and Holonomic Wheeled Platforms for Mobile Robot. *IEEE Trans. Robot. Autom.* **1999**, *15*, 978–989. [[CrossRef](#)]
13. Liu, Y.; Jiang, N.; Wang, J.; Zhao, Y. Vision-based Moving Target Detection and Tracking Using a Quadrotor UAV. In Proceedings of the 11th World Congress on Intelligent Control and Automation, Shenyang, China, 29 June–4 July 2014; pp. 2358–2368.
14. Salih, A.L.; Moghavvemi, M.; Mohamed, H.A.F.; Geaid, K.S. Flight PID Controller Design for a UAV Quadrotor. *Sci. Res. Essays* **2010**, *5*, 3660–3667.
15. Ziegler, J.G.; Nichols, N.B. Optimum settings for automatic controllers. *Trans. ASME* **1942**, *64*, 759–768. [[CrossRef](#)]