



# Leader–follower formation control of two quadrotor UAVs

Rifqi Rafifandi<sup>1,2</sup> · Dea Lana Asri<sup>1,2</sup> · Estiyanti Ekawati<sup>2,3</sup> · Eko Mursito Budi<sup>3</sup>

© Springer Nature Switzerland AG 2019

## Abstract

This paper reports the design and implementation of a leader–follower (L–F) formation control of two Parrot AR Drone 2.0 quadrotor UAVs (drones). In order to implement such control, three control elements were developed. These were the flight-dynamic characteristics, the position tracking and L–F formation control, and the data communication system between drones and ground-control station. The dynamic characteristics were identified as the first-order process with time delay models, which were asymmetrical to  $x_-$ ,  $x_+$ ,  $x_-$  and  $x_+$  directions. Accordingly, dedicated Proportional-Derivative (PD) controllers were designed for the respective models. A Pole-Placement strategy was applied to design the PD controllers in order to achieve a non-oscillatory position tracking. These controllers were combined with five operating modes to form an F–L formation control. The controls were implemented on two-drones and one-ground-control station. The ground-control station dictated a prescribed path for the leader drone. Subsequently, the leader position became follower set-point. During the implementation test, these drones completed their L–F formation. The follower drone tracked the  $y$ -axis coordinate of the leader's drone and maintained a safe distance to the respective  $x$ -axis coordinate. The accuracy of the controls was measured in terms of the root mean squares errors, and these were within 50–115 cm.

**Keywords** Swarm · Leader–follower · Control design · Quadrotor · UAV · Wireless network

## 1 Introduction

The interest in the control of multiple autonomous aerial robots (drones) working in a collaborative group, known as 'swarm,' has been increasing within the last decade. The trend is due to the potential applications in various fields, either for civilian or military purposes, and either for outdoor and indoor applications [1, 2, 8, 13, 15].

One of the approaches to the swarm control is the leader–follower (L–F) algorithm. This approach involves one drone leading one or more follower drone(s). The leader-drone is typically capable of tracking a path commanded by a ground-control station. The follower-drones track the leader position and maintain some safe distances between the drones, to avoid collisions.

The challenge to maintain the configuration while tracking a given path had inspired various studies. Reference [13] reviewed various approaches to the system modeling, control design and experimental flights of multiple drones. This references highlighted that in order to integrate the algorithm to experimental flight, one needs to consider additional factors. These include the dynamic characteristic of each drone, the positioning system, as well as the communication system among drones and with the ground-controller.

The example of such researches includes the comparison between the PID and the Sliding Mode strategies to control the circular F–L flights in a confined space [1]. This study utilized high definition cameras to capture the drone's coordinates and angles. These data were feedback to the ground-controller that coordinated the

✉ Estiyanti Ekawati, esti@instrument.itb.ac.id; Rifqi Rafifandi, rifqi.rafifandi@alumni.itb.ac.id; Dea Lana Asri, dealana@alumni.itb.ac.id; Eko Mursito Budi, mursito@tf.itb.ac.id | <sup>1</sup>Engineering Physics Program, Faculty of Industrial Technology, Institut Teknologi Bandung, Bandung, Indonesia. <sup>2</sup>Center for Instrumentation Technology and Automation, Institut Teknologi Bandung, Bandung, Indonesia. <sup>3</sup>Faculty of Industrial Technology, Institut Teknologi Bandung, Bandung, Indonesia.



flights. Another indoor implementation in [15] utilized the self-tuning PID to control an F–L configuration. A fiducial marker [5] was attached to the leader drone so that the follower can recognize and track the leader. In an outdoor setting and laboratory scale, the study in [2] implemented the self-deployment flight algorithm for three modified drones. This study added a Global Positioning System (GPS) to support the built-in Inertial Measurement Unit (IMU) sensor for the control feedback. The authors also customized an open source Robot Operating System (ROS) to serve the communication between drones and with the ground-control station.

In a similar spirit, this paper reports the integrated design and application of the L–F formation control to a duo Parrot AR Drone 2.0 quadrotors (referred to as drones from this point onwards). The system utilized the Proportional-Derivative (PD) control for each drone, as well as for the whole F–L control. The design was based on the identified dynamic characteristics of each drone. The control system utilized the Quick Response (QR) code-based VPS [12] to locate the position of each drone during the flight.

In the following sections, the architecture of the L–F formation control is explained. It is followed by the identification of the drone’s flight characteristics and the respective controller parameters design. Then the control system implementation for single-drone tracking performance and duo-drones is presented. Finally, the paper concludes with the measure of system performance.

## 2 System architecture

This research employed three main components to perform position tracking and L–F formation control. These are the drones, a ground-control station, and a Wi-Fi access point. Figure 1 shows the system architecture, the function of each component and the respective data communication.

This research used two identically specified drones, namely Drone A and Drone B (Fig. 2). The ground-control station was a computer, of which function was to control the flight of all drones based on the drones’ position. The drones sent the captured images of QR-codes to the ground-control. The ground-control extracted the drone’s coordinates from these images and calculated the required control actions. The controller was programmed using the software library in [14] which provided the flight control and image processing functionalities. The communication between the drones and the PC was carried through the Wi-Fi signals.

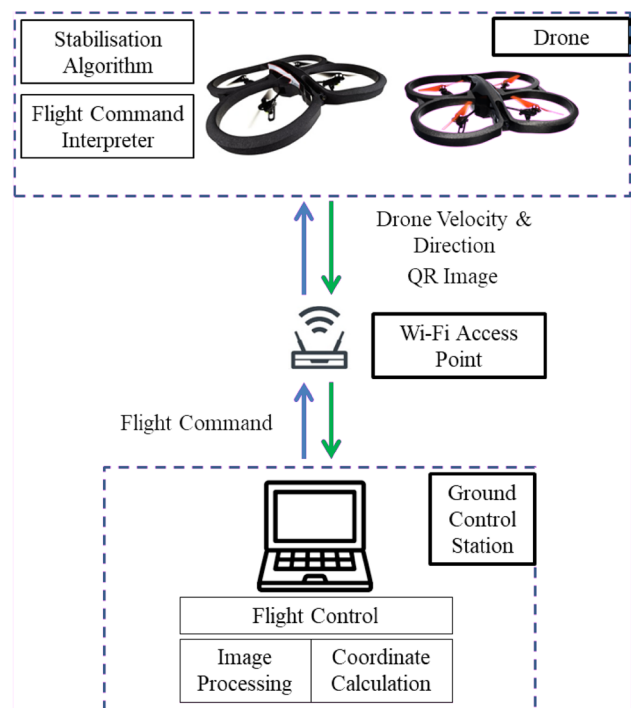


Fig. 1 System architecture

## 3 System identification and control

### 3.1 The identification process

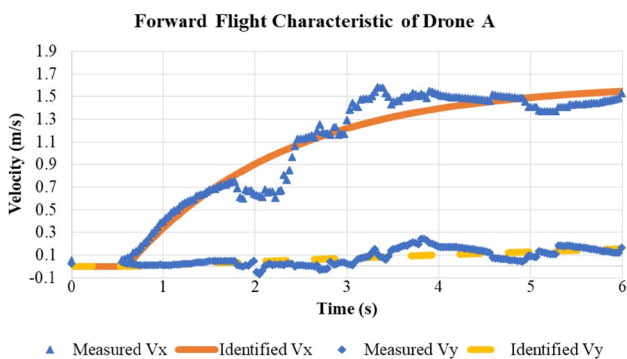
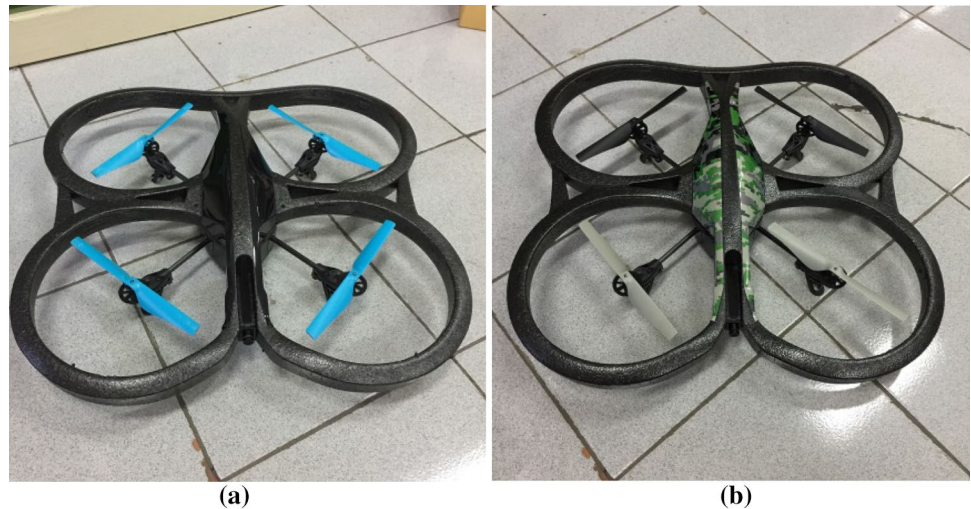
In order to control the drone’s flight, this research chose to identify the respective dynamic characteristics and to represent them as mathematical models based on the procedure in [4]. The procedure assumed that a first-order process with time delay (FOPTD) functions could approximate the dynamic velocity response [9]. Equation (1) represents a FOPTD system transfer function, in which  $K$  represents a gain,  $\tau$  represents a time constant, and  $t_d$  represents a time delay [9].

$$G(s) = \frac{K}{\tau s + 1} e^{-t_d s} \tag{1}$$

In order to identify these parameters, the drones were commanded to fly along a particular direction with a constant velocity. The transient velocity responses of the drones to the commands (set-points) were recorded, and the approximate FOPTD functions were determined.

The velocity along  $x$ - and  $y$ -directions were denoted by  $v_x$  and  $v_y$  and were in the range of  $-1.0 \leq v_x \leq 1.0$ ,  $-1.0 \leq v_y \leq 1.0$  respectively. These ranges were the ratio of the target velocity to the maximum available velocity. The drones also flew at a relatively constant altitude, about 70 cm above the ground. Therefore, the

**Fig. 2** Drones used in this research **a** Drone A and **b** Drone B



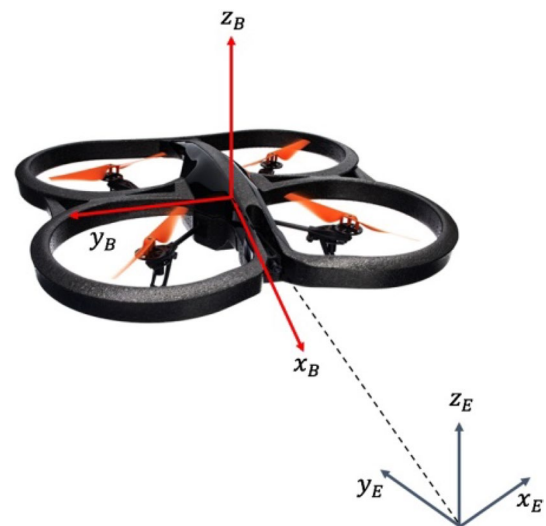
**Fig. 3** Velocity responses of Drone A and the identified FOPTD functions

identification process focused on the dynamic characteristics of the flight to forward ( $x_+$ ), backward ( $x_-$ ), rightward ( $y_+$ ), and leftward ( $y_-$ ) directions.

Figure 3 shows the sample of the velocity responses  $v_x$  and  $v_y$  during the forward flight of drone A. These responses indeed resemble FOPTD functions. The markers represent the velocity data provided by the drone’s IMU. The solid lines represent the approximate FOPTD functions, which were acquired by minimizing the sum of squared errors (SSE) between the measurement data and the FOPTD functions.

The figure also shows that during the flight, the drone tends to stray sideways. This secondary movement was also identified to facilitate simultaneous position control in  $x$ - and  $y$ -directions.

Since each drone sent velocity data based on its body frame of reference, these data should be transformed into the earth frame of reference. Figure 4 illustrates the relationship of a drone’s body frame relative to the earth frame. However, at low velocities, all rotational components were negligible. Hence the transformation matrix



**Fig. 4** The mapping of a drone’s body frame relative to the earth frame

became an identity matrix. Therefore, the transformed parameters are equal to their identified counterparts.

Table 1 shows the FOPTD parameters for drones A and B for primary directions. It shows that while the specification of drone A and drone B were identical, the respective dynamic movement characteristics were significantly different at all directions. In order to accommodate this condition, specific controller parameters were designed for each direction of flights.

### 3.2 The control design

The control design aimed to ensure that the position of the drones did not overshoot the given set-points during a flight. The Proportional-Derivative (PD) controller structure [9] designed by the Root Locus method was selected for

**Table 1** Drones' FOPTD parameters

Direction	Drone A			Drone B		
	$K$	$\tau$	$t_d$	$K$	$\tau$	$t_d$
Forward ( $x_+$ )	1.792	1.816	0.633	1.358	1.100	0.708
Backward ( $x_-$ )	2.389	2.534	0.574	2.121	0.776	0.583
Leftward ( $y_+$ )	1.322	1.238	0.616	1.446	3.390	0.784
Rightward ( $y_-$ )	1.432	1.438	0.644	0.832	0.493	0.671

that purpose. In (2), The PD controllers were represented by the transfer function  $G_c(s)$ , in which  $K_p$  was proportional gain and  $T_d$  was derivative time.

$$G_c(s) = K_p(1 + T_d s) \tag{2}$$

Based on the concept that a position is the integral of its velocity, the transfer function  $G(s)$  in (1) was multiplied by a factor of  $1/s$ . The time delay element ( $e^{-t_d s}$ ) was also converted into the pole-zero form using the first-order Padé approximation [9]. The completed transfer function  $G_p(s)$  became as follows:

$$G_p(s) = \frac{1}{s} G(s) \approx \frac{K}{s(\tau s + 1)} \frac{1 - \frac{t_d s}{2}}{1 + \frac{t_d s}{2}} \tag{3}$$

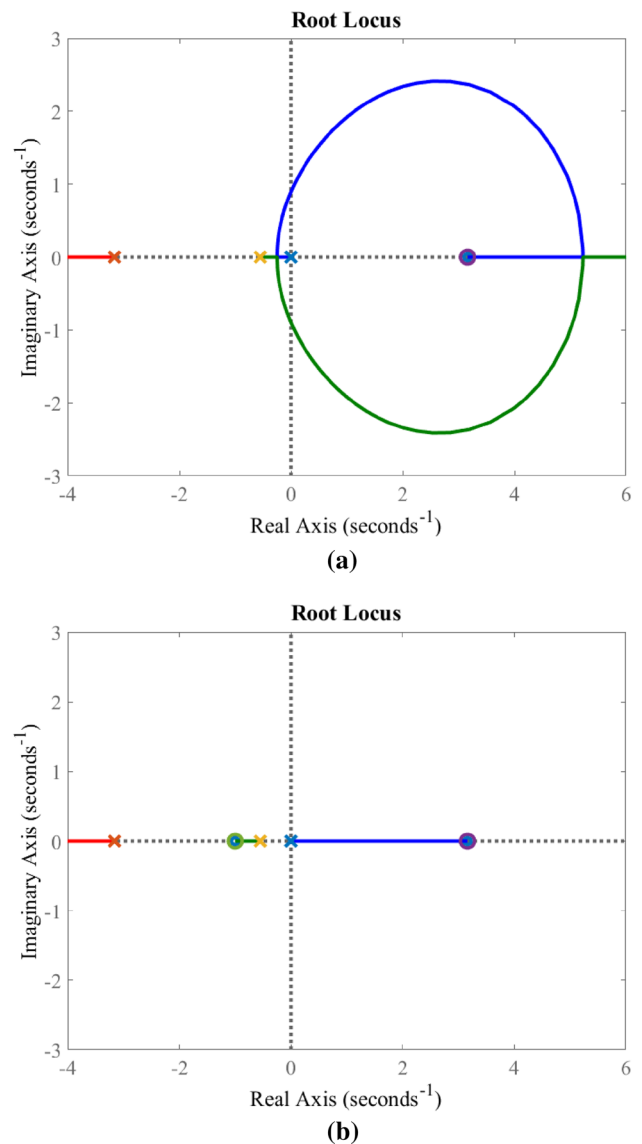
Figure 5a shows the root locus plot of  $G_p(s)$  representing the forward flight of Drone A. The plot shows that the system could be pushed to instability even by a small proportional gain. An addition of a zero at the left half plane, hence by the use of PD controller, would significantly rectify this problem. Hence, the zero of the PD controller ( $s = -1/T_d$ ) was placed such that the closed loop locus plot of  $G_c G_p(s)$  was laid entirely in the real axis. This plot ensured the non-oscillatory responses for all values of system gains. The value of  $K_p$  was chosen such that the closed-loop system gain equals one. Figure 5b shows the completed root locus plot of  $G_c G_p(s)$ .

All PD control parameters of both drones were determined using the same procedure. Table 2 shows the results for primary directions. As expected, different dynamic characteristics of the drones lead to different controller parameters.

### 3.3 Position tracking

The drone ability of the controlled drones to track a given path was tested through experiments. The results were later extended for the L-F formation control.

Figure 6 shows the closed-loop block diagram for a single drone position tracking control, in this case, Drone A. This figure shows that the set-points and the controlled variables were the coordinates of the drone in the forward ( $x_+$ ) and the rightward ( $y_+$ ) directions based on the earth



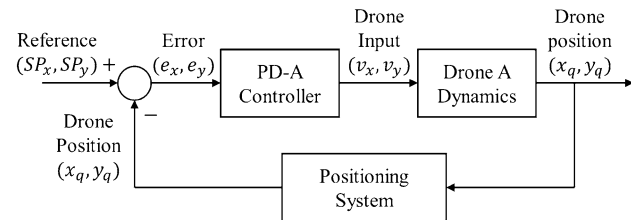
**Fig. 5** Root locus plot for the forward flight of Drone A. **a** Without controller and **b** with the controller

frame. The ground-control main program recognized the set-points as cases (operating mode). Every time the drone reached a set-point, a new set-point to pursue was generated. The cycle continued until the drone arrived at the final set-point.



**Table 2** PD controller parameters

Direction	Drone A		Drone B	
	$K_p$	$T_d$	$K_p$	$T_d$
Forward	0.5581	1	0.7364	0.5
Backward	0.4186	1	0.4716	1
Leftward	0.757	1	1.2013	1
Rightward	0.6981	1	0.6915	1



**Fig. 6** Block diagram of a single-drone position tracking control loop

The visual positioning system set-up in [12] provided the coordinates of the controlled variables. The ground-controller received these variables and sent them to the PD controller.

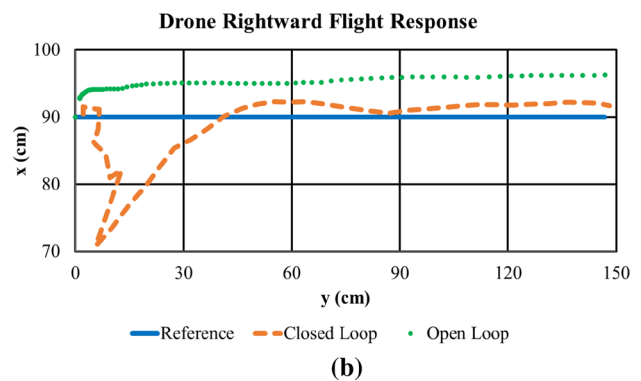
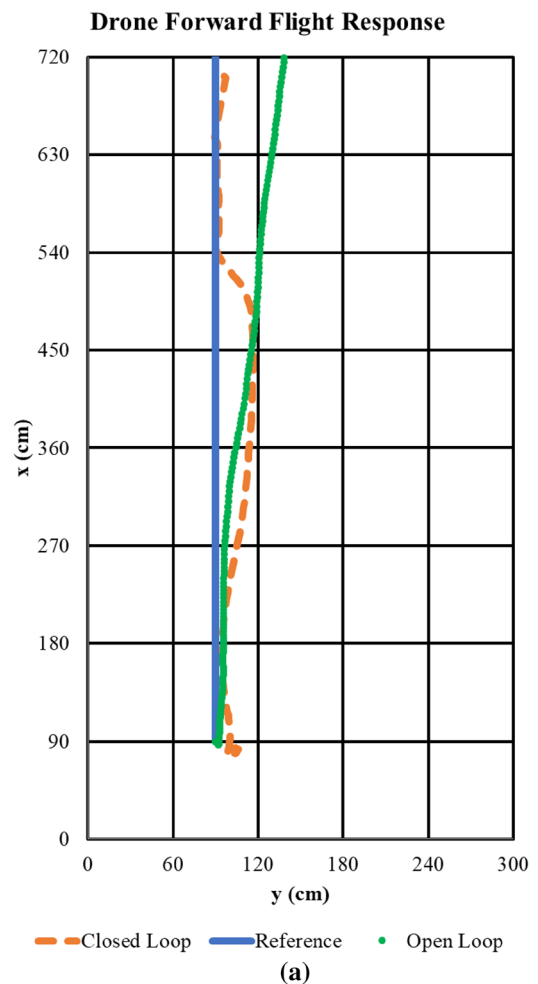
When the controller detected the error in both directions,  $e_x$  and  $e_y$  respectively, it produced the control commands  $v_x$  and  $v_y$ . These commands were in the form of a discrete-time PD controller function as follows:

$$\begin{bmatrix} v_x(k) \\ v_y(k) \end{bmatrix} = \begin{bmatrix} K_p e_x(k) + T_d \frac{e_x(k) - e_x(k-1)}{T_s} \\ K_p e_y(k) + T_d \frac{e_y(k) - e_y(k-1)}{T_s} \end{bmatrix} \quad (4)$$

In (4),  $T_s$  was the sampling time of the closed loop discrete system, which was 30 ms. The manipulated inputs were held at the constant values between the sampling sequences. The drones responded to these inputs, and the flight was controlled simultaneously to both directions.

Figure 7 shows the ability of a single drone to track a straight path. This figure shows the comparison between the uncontrolled and the controlled responses of Drone A, in response to the command to fly in forward ( $x_+$  direction) and rightward ( $y_+$  direction) separately.

In Fig. 7a, b, the solid lines represent the reference track, the dotted lines represent the uncontrolled responses, and the dashed lines represent the controlled ones. These figures show that the controller was able to rectify the stray flight and drive the drone to the correct path. Therefore, the test was continued to the tracking performance of multiple directions.



**Fig. 7** Comparison of the uncontrolled and controlled flight of Drone A. **a** Forward flight and **b** right-ward flight

The visual positioning system in [12] was utilized to provide the position feedback to the controller. This positioning system arranged the QR-code markers in a rectangular formation of 4 columns and 15 rows, with 60-cm spacing between the rows and 90-cm spacing between the columns.

The drones captured the QR images through the respective built-in vertical camera and sent them to the ground-control station. The latter extracted the coordinate data from the images, and recognize them either as the intermediate points or as the set-points. In case of the drone did not capture any QR image, the ground-controller determined the coordinates by adding the integral of the velocity data of the drone to the most recent coordinates extracted from QR images.

In order to ensure that the drone captured the images as frequently as possible, several intermediate set-points were provided for a given track. The set-points were located at every 120 cm in the x-direction and every 90 cm in the y-direction.

Figure 8 shows the flowchart of the position tracking algorithm. The set-points were keyed into the main ground-controller program. The distance between the drones to the next set-point was defined as the Euclidean distance of  $e_x$  and  $e_y$ . If the drone entered the 30 cm of this distance to the nearest set-point, then the next set-point was activated. If the drone reached the final set-point, it automatically lands. Note that during the position tracking experiments, the drones flew autonomously.

There were two measurements conducted respectively for forward, for rightward, and for diagonal tracking. One measurement was conducted for letter-S tracking. The track completion times were significantly varied, as shown in Table 3. Figure 9a–d show those that had faster completion times.

Figure 10a, b show the dynamics in the forward and sideward directions during the letter-S tracking. These figures show that the drone reached the set-points of either direction. The flights were slow because the maximum value of control signals  $v_x$  and  $v_y$  were limited to 0.5. This limitation was intentional, to enable the drone to accurately capture the image of the QR marker during its course.

Figure 10 also shows that the drone’s tracking accuracy was higher in the forward direction in comparison to the sideward direction. One reason for this condition was that the coverage of the built-in drone’s vertical camera was not the same in either direction. The camera had higher resolution in the forward direction of the body frame (640 pixels) than the sideward direction (480 pixels). Therefore, the camera captured a larger area of the marker while flying forward and reported more accurate position to the controller. Accordingly, the controller was able to command the drone to stay on its track.

### 3.4 Leader–follower formation

Having established that a single drone can track several types of paths, this research proceeded to develop and

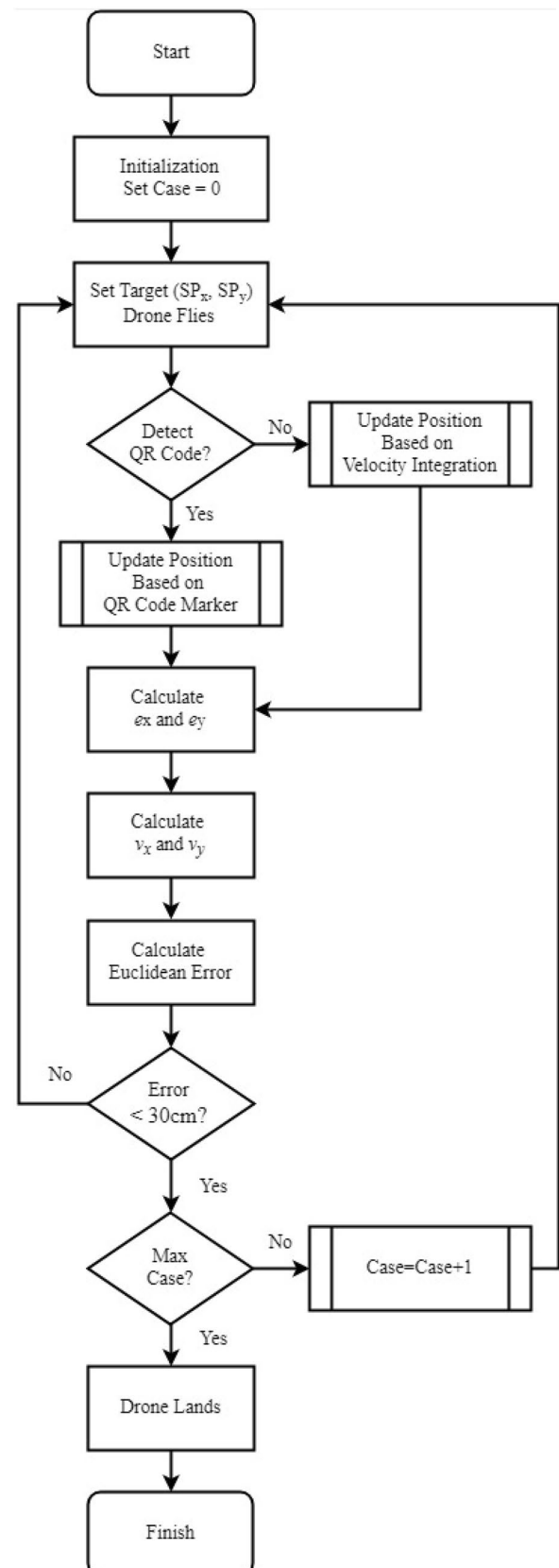


Fig. 8 Flowchart of the position tracking algorithm

**Table 3** Drone track completion time

Movement	Measurement	Track completion time (s)
Forward	1	43.722
	2	59.325
Rightward	1	47.447
	2	25.353
Diagonal	1	67.275
	2	47.715
S-Letter	1	143.474

implement the L–F formation control for two drones. There were many studies related to the coordination between drones in L–F formation. For instance, the implementation of various communication topologies to facilitate the distributed formation control [7, 8], the consensus of motion using the feedback formation control [6] and the model predictive control [3].

There were also studies in avoiding collision within the configuration, as well as with the environment. These include the combination of collision avoidance strategies with the fault tolerant control [10] and with the dynamic evolutionary algorithm and a reinforcement learning algorithm [11]. These researches involve intensive communication and decision making among drones, and therefore many were in the simulation stage. In order to integrate the control strategies to the experimental flights, this research chose to focus on some of the goals in [10] and to formulate the associated strategies in concise rules.

Therefore, the aims of this L–F formation control were:

- To ensure that the leader tracked the given path, arrived at the final set-point and landed.
- To ensure that the follower tracked the leader's position, maintained a safe distance to the leader, and landed when the leader completed the course.

The following rules were also applied to avoid collisions due to the communication problems between drones:

- If the follower lost its contact with the leader, the follower would land.
- If the leader lost its contact with the follower, the leader continued its flight to reach the final setpoint.

Five cases (operating modes) listed in Table 4 accommodate the above aims and rules. In this L–F formation, Drone A was the leader, and Drone B was the follower. Accordingly, the respective PD-controllers (PD-A and PD-B) were extended with relevant cases, so that altogether they

formed an L–F formation control. The ground-control station hosted these F–L controllers. These controllers received the flight cases, detected the errors ( $e_{xlr}$ ,  $e_{yfr}$ ,  $e_{xfl}$ ,  $e_{yfl}$ ), and commanded the required control inputs ( $v_{xlr}$ ,  $v_{yfr}$ ,  $v_{xfl}$ ,  $v_{yfl}$ ) to the respective drones. The visual positioning system setup identical to the position tracking control provided the drone coordinate. Figures 11 and 12 show the algorithms developed for L–F control for the leader and the follower respectively.

Figure 11 shows the possible operations during the leader flight to a final set-point. The leader received the first set-point and the first case = FLY. Due to this case, the flight began, and the positioning system sent the associated coordinates to both leader and follower.

The leader also received the follower's status and relative distance. The 300 cm distance between the leader drone and the follower drone was defined, in order to avoid collisions and to avoid the disturbance of air turbulence produced by the other drone. The follower is considered NEAR if the actual distance was equal or closer than 400 cm, and considered FAR otherwise.

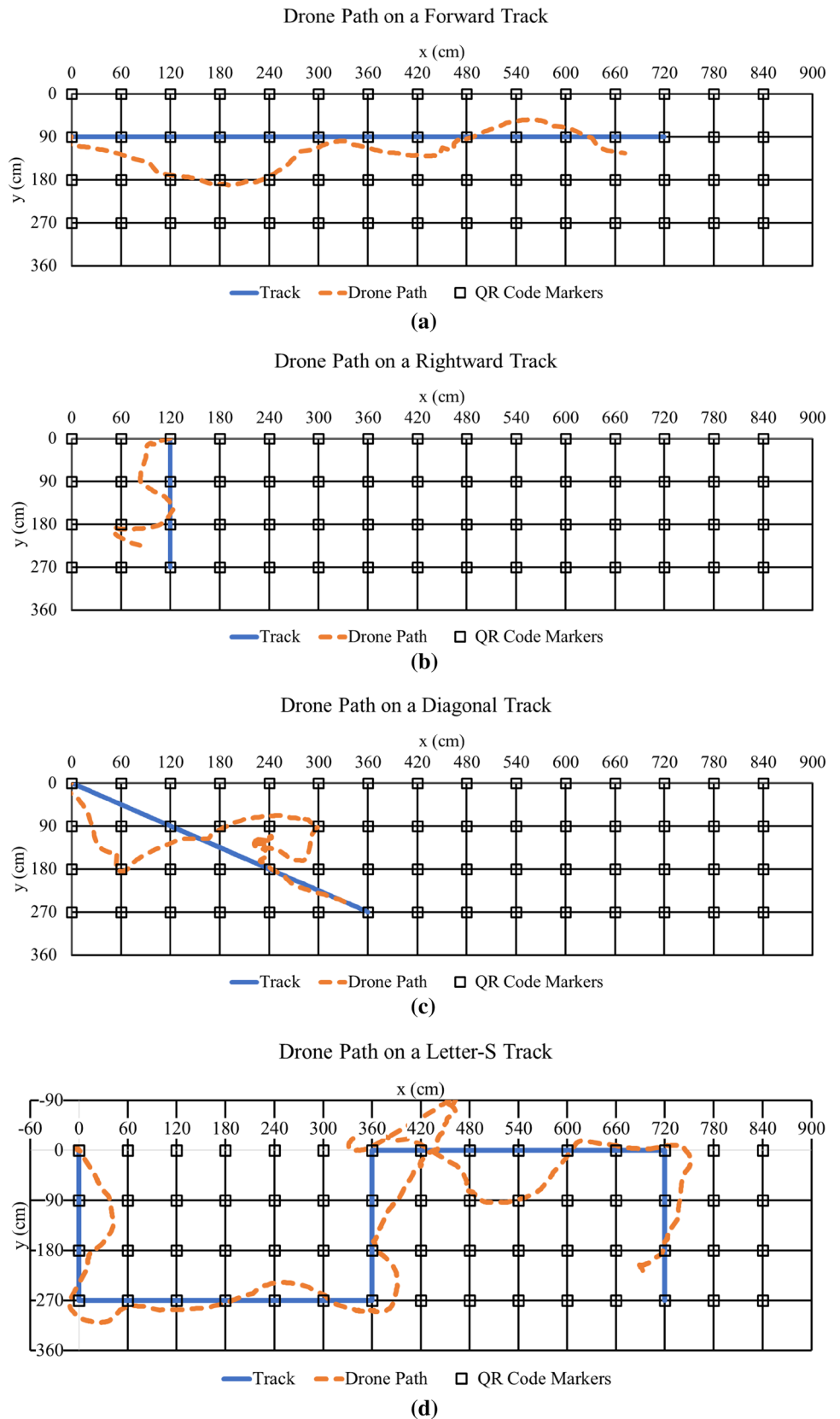
If the follower was NEAR, then the leader continued the flight along with the follower to the next set-point. If the follower was FAR, then the leader's case changed to HOVER. The leader hovered and waited for the follower's status changed to NEAR before resuming the flight. Otherwise, if the leader did not receive the follower's status until the allocated waiting time was up, then the case change to TIME OUT and then the leader continued the flight in SOLO, without the follower. The contact between the leader and follower was assumed lost. The algorithm continued until the leader arrived and landed at the final set-point.

Figure 12 shows the possible operations during the follower flight. At the beginning of the flight, the follower's case is HOVER. It hovered until the leader's status and coordinates were received. Based on the leader's status and position, the follower updated its set-point and then flew towards the point. If the follower did not receive the leader's status until an allocated waiting time was up, then the case change to TIME OUT. The follower aborted the flight and landed.

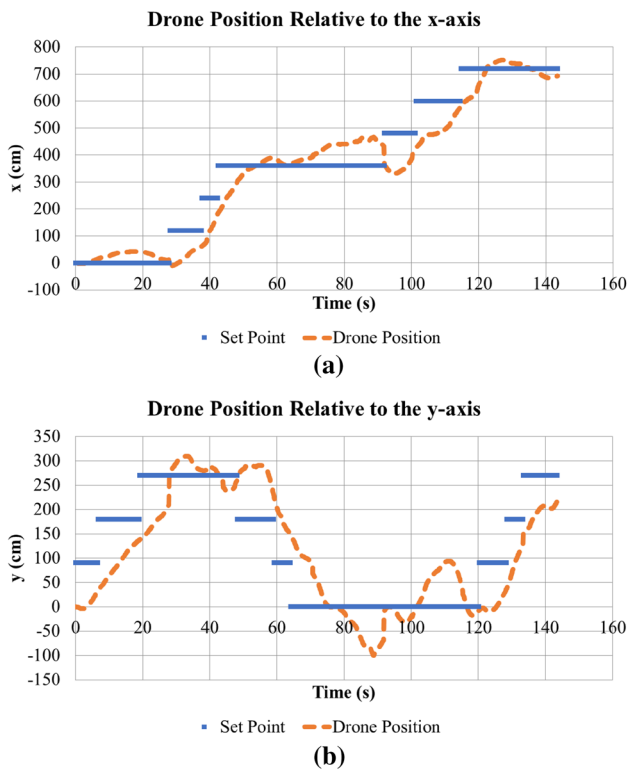
These algorithms were programmed using the software library in [14]. The ground-control station operated these programs. The controllers received the drone's coordinates from the visual positioning system and sent the associated commands to the drones in the form of velocity inputs. The communication between the ground-control station and the drones was through a TCP/IP protocol. Figure 13 shows the block diagram of this L–F formation control and Fig. 14 shows the associated communication scheme.

This L–F formation control was implemented in a live experiment using Drone A as the leader and Drone B as

**Fig. 9** Controlled flight of Drone A on various tracks. **a** Forward, **b** rightward, **c** diagonal and **d** letter-S







**Fig. 10** Drone A position relative to its set-point. **a** x-direction and **b** y-direction

the follower. The first experiment facilitated the measurement of the leader’s ability to track a given path (Fig. 15), and the follower’s ability to track the position of the leader (Fig. 16). The second experiment measured the ability of both drones to maintain their configuration (Fig. 17). These experiments focused on minimizing the coordinate errors, rather than the flight duration.

The first experiment was to command the leader to fly in the forward direction. Throughout the flight, the leader drone arrived at three intermediate set-points before reaching the final set-points. At 60 s, the drone arrived at the final set-point and hovered to wait for the follower drone. The observation of the flight was

completed at 100 s. The figure shows that the leader responded well to the set-points. The final position was within 30 cm radius. Although the leader did stray from the final point due to a temporary communication problem, its position tracking control recovered the position. The root means squared error (RMSE) of this experiment was 111.31 cm.

The leader’s position was sent to the ground-control station and became the follower set-point with an offset of 300 cm. Figure 16 shows how the follower (dashed line) tracked the set-point (solid line). It is shown that there is a delay time of 8.45 s for the follower to track the set-point. To provide a direct comparison, the follower position without the delay component (dotted line) is also shown. This comparison shows that the follower tracks the leader reasonably well. By taking away the errors due to the delay, the RMSE between the set-point and the follower path was 57.45 cm.

Figure 17 shows the variation of distances between the leader and the follower during an L–F flight. In this experiment, the follower was assigned to maintain 300 cm distance to the leader in the x-direction and to track the leader’s position in the y-direction. In the figure, solid lines represent the reference distances, and the dashed lines represent the actual distances.

The distance in the x-direction indicated the combined efforts of reaching the final set-point by employing the cases defined in Table 4 and the PD controllers of both drones. The distance in y-direction highlighted the drone’s tendencies to stray sideways, and the associated PD controls to drive them back to the intended course. In this experiment, the RMSE of variations along the x- and the y-direction were 89.65 cm and 105.62 cm respectively.

The RMSE at all experiments were within 50–115 cm. The result showed that although the L–F control worked, there was still room for improvement in flight accuracy and duration. The flight accuracy relied on the performance of the visual positioning system. In order to improve its performance, the use of simple fiducial marker, higher resolution camera, as well as the use of indoor GPS would be considered in future works.

**Table 4** Commands for the leader–follower formation program

Case	Leader drone	Follower drone
FLY	Flies to the set-point as long as the follower drone is NEAR	Receives the position of the leader as a new set-point, then flies to the set-point
HOVER	Hovers at the current position, because the follower is FAR	
SOLO	Flies to the set-point without the follower, because the follower’s status is unknown	–
ABORT	–	Lands immediately because the leader’s status is unknown
END	Arrives at final set-point, the follower is NEAR. Therefore the leader lands	The follower lands

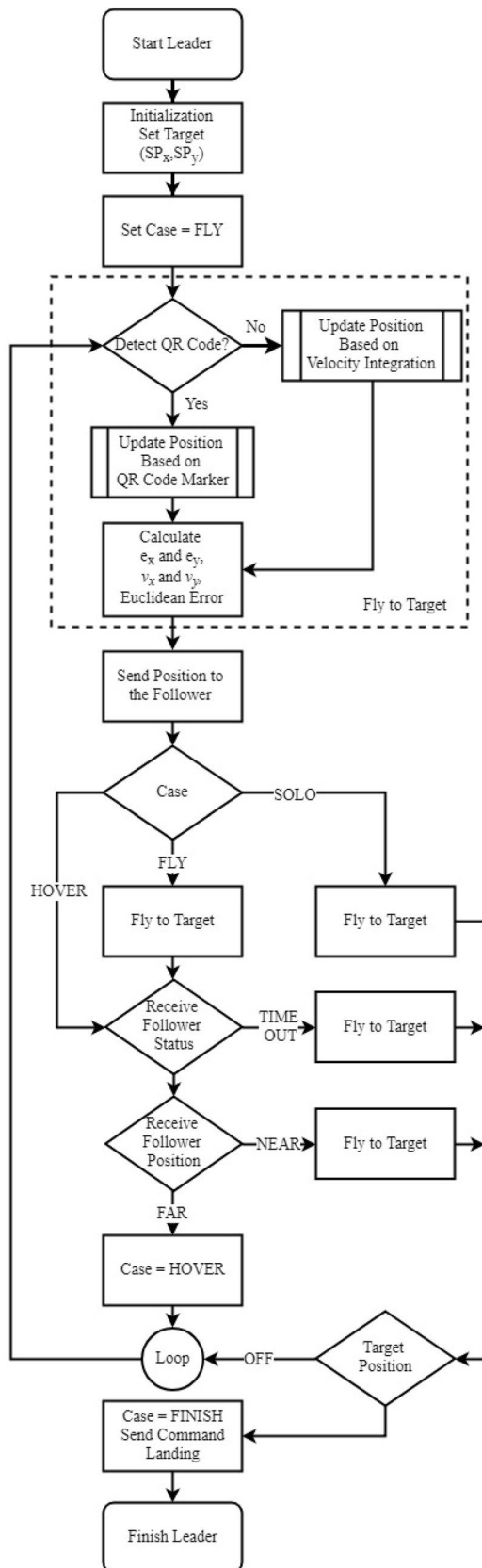


Fig. 11 Flowchart of the leader drone program

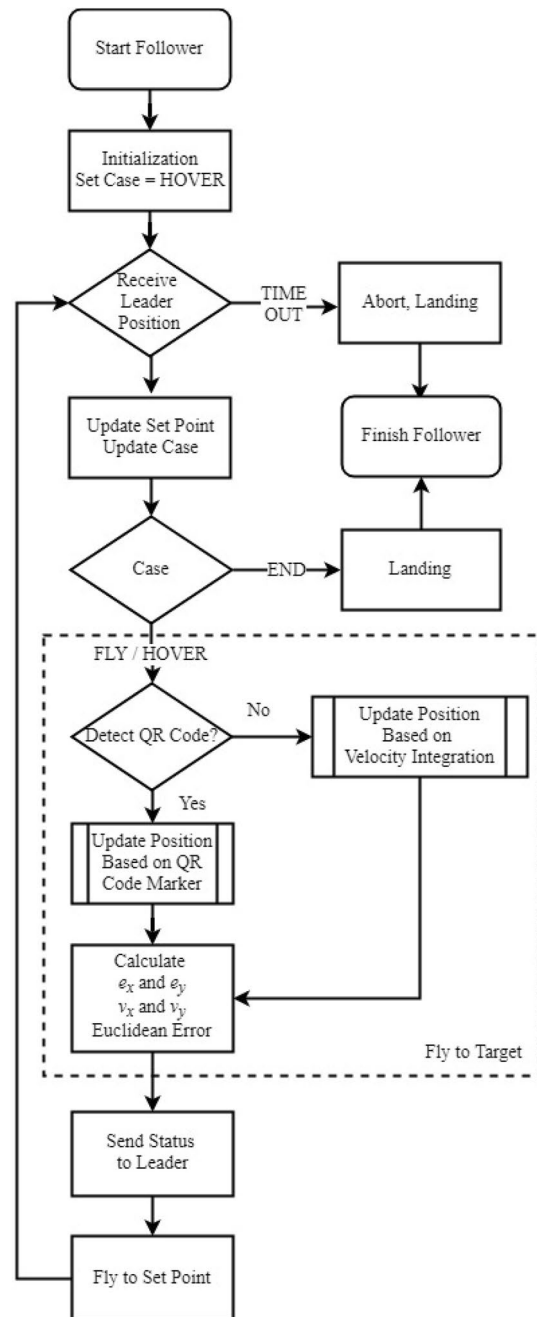
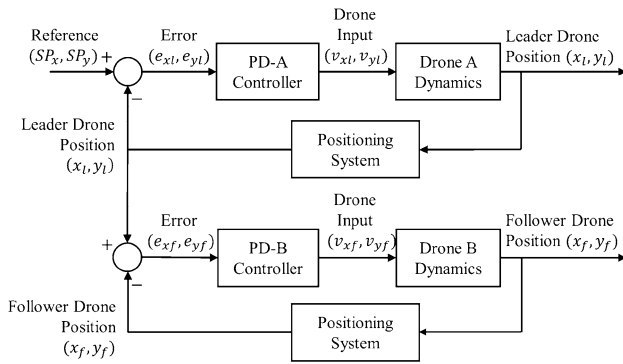
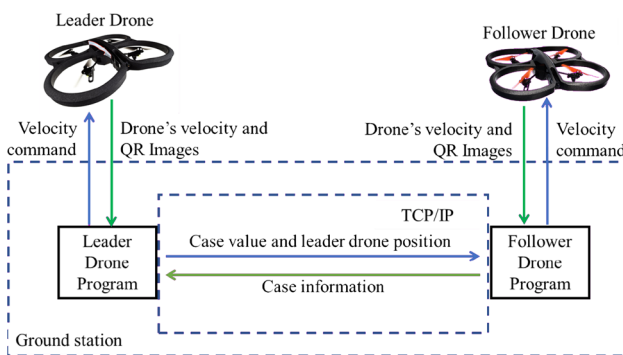


Fig. 12 Flowchart of the follower drone program

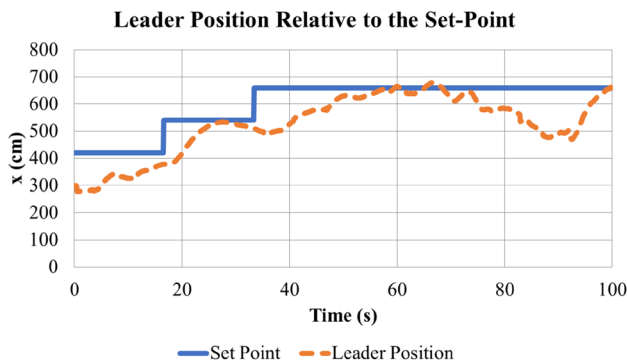
Another factor that affects the flight duration was the delay of the follower tracking the leader set-point. This delay was caused by the relatively slow communication between the drones and the ground-control station. Since the delay was significant, the improvement to the communication aspect would also be the focus of future work.



**Fig. 13** Block diagram of the leader–follower formation control loop



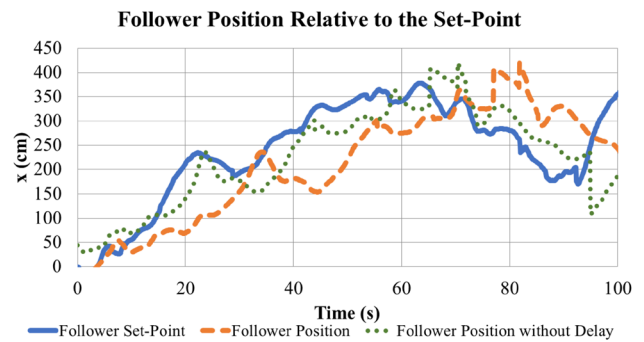
**Fig. 14** Communication scheme between the leader drone program and the follower drone program



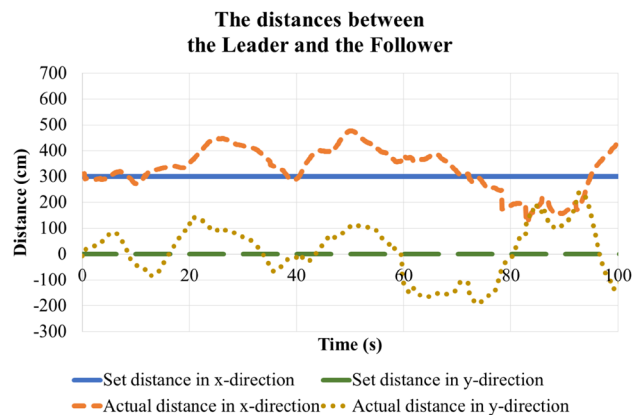
**Fig. 15** Leader position relative to the set-point

### 4 Conclusion

In this paper, we have reported the implementation of a leader–follower formation of two Parrot AR Drone 2.0



**Fig. 16** Follower position relative to the set-point



**Fig. 17** The distance between the leader and the follower drones during the flight

drones. The activities consisted of identification of the movement characteristic of each of the drones, performing position tracking with a single drone, and performing the leader–follower formation. All of these activities involving data communication between the ground-control station and the drones.

The velocity characteristics of the drone were identified as FOPTD systems. Accordingly, the PD controller parameters were defined. The performance test for a single-drone position tracking showed that the drones were able to track various types of paths, including diagonal and S-letter tracks. This result confirmed the readiness to conduct the L–F formation.

In the L–F formation test, the leader drone was programmed to fly at a straight path. The follower drone tracked the leader position, with a 300 cm distance in the x-direction and no gap in the y-direction. The quality of the position tracking and the L–F formation was within the RMSE of 50–115 cm.

## Compliance with ethical standards

**Conflict of interest** On behalf of all authors, the corresponding author states that there is no conflict of interest.

## References

1. Abas MF, Pebrianti D, Ali SAM, Iwakura D, Song Y, Nonami K, Fujiwara D (2013) Circular leader–follower formation control of quad-rotor aerial vehicles. *J Robot Mechatron* 25(1):60–71
2. Alvissalim MS, Zaman B, Hafizh A, Ma'sum MA, Jati G, Jatmiko W, Mursanto P (2012) Swarm quadrotor robots for telecommunication network coverage area expansion in disaster area. In: SICE annual conference, Akita, Japan
3. Chang CW, Shiau JK (2018) Quadrotor formation strategies based on distributed consensus and model predictive controls. *Appl Sci* 8(11):2246
4. Ekawati E, Widyotriatmo A, Askandari I (2014) Quadrotor position control based on model identification and proportional-derivative algorithm. In: 2nd international conference on technology, informatics, management, engineering and environment (TIME-E), Bandung, Indonesia
5. Garrido-Jurado S, Muñoz-Salinas R, Madrid-Cuevas FJ, Marín-Jiménez MJ (2014) Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognit* 47(6):2280–2292
6. He L, Bai P, Liang X, Zhang J, Wang W (2018) Feedback formation control of UAV swarm with multiple implicit leaders. *Aerosp Sci Technol* 72:327–334
7. Hou Z, Fantoni I (2015) Distributed leader–follower formation control for multiple quadrotors with weighted topology. In: 10th IEEE system of systems engineering conference (SoSE 2015), May 2015, San Antonio, TX, USA, pp 256–261
8. Hou Z (2016) Modeling and formation controller design for multi-quadrotor systems with leader–follower configuration. Dissertation, Université de Technologie de Compiègne, France
9. Leva A, Maggio M (2012) Model-based PI(D) autotuning. In: Villanova R, Visioli A (eds) PID control in the third millennium, advances in industrial control. Springer, London, pp 45–73
10. Liu ZX, Yu X, Yuan C, Zhang YM (2015) Leader–follower formation control of unmanned aerial vehicles with fault tolerant and collision avoidance capabilities. In: 2015 international conference on unmanned aircraft systems (ICUAS), Denver, Colorado, USA
11. Majd A, Ashraf A, Troubitsyna E, Daneshlab M (2018) Using optimization, learning, and drone reflexes to maximize the safety of swarms of drones. In: IEEE congress on evolutionary computation (IEEE CEC 2018), Rio de Janeiro, Brazil
12. Raffandi R, Asri DL, Ekawati E, Budi EM (2018) QR-code-based visual positioning system on UAV quadrotor. In: Engineering physics international conference (EPIC), Surabaya, Indonesia
13. Santana LV, Brandao AS, Sarcinelli-Filho M (2016) Navigation and cooperative control using the ar. drone quadrotor. *J Intell Robot Syst* 84(1–4):327–350
14. Shinpuku N (2017) CV drone (= OpenCV + AR.Drone), GitHub, Feb 11, 2017. <https://github.com/puku0x/cvdrone>. Accessed 31 Aug 2017
15. Vankadari MB, Das K, Kumar S (2017) Autonomous leader–follower architecture of A.R. Drones in GPS constrained environments. In: Proceedings of AIR'17, New Delhi, India

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.