IMAV2018-23
http://www.imavs.org/pdf/imav.2018.23

Simulation & Flight Test of Ducted-fan UAVs for Formation Flight

Taegyun Kim, Hyemin Mo, Seungkeun Kim*, and Jinyoung Suk
Department of Aerospace Engineering, Chungnam National University, 99 Daehak-ro, Yuseong-gu, Daejeon 305-764, Korea

ABSTRACT

This paper proposes to drive a group of ducted-fan UAVs in a formation, using Leader-follower strategy for surveillance and scouting. To each UAV the target position is assigned based on the leader UAVs position. The UAVs computes their position commands using the relative position data between the UAVs in order to keep a formation. The results via ardupilot SITL verifies that the UAVs are able to keep a predetermined formation for mission accomplishment. In the flight test, we conducted using the ducted-fan UAV manufactured in-house. The UAV has the stator which is fixed under main rotor to counteract anti-torque. A ducted-fan UAV is equipped with Pixhawk for flight control and a Raspberry Pi3 for UDP communication.The data communication between the ducted-fan UAVs is performed using Dronekit-python.

1 INTRODUCTION

In recent years UAVs have become popular because they are suitable for many applications such as surveillance and scouting. In addition, with the development of communication technology, multiple UAVs can simultaneously perform missions. This operation has a number of advantages: A group of UAVs can acquire more data and occupy a wider area during surveillance and scouting missions than single UAV [1]. In addtion, the survival rate is higher than when single UAV is operated in the battlefield. Even if rotary UAVs have been widely studied and operated due to its vertical flight capability, they sometimes experience accidents from high-speed rotor blades.
To overcome this risk, the ducted-fan type UAV is introduced, which has a duct to cover the main rotor. Moreover, a well-designed duct can improve thrust efficiency compared with an open-rotor system. This paper proposes to drive a group of ducted-fan UAVs in a formation, using leader-follower strategy for surveillance and scouting [2]. To each UAV the target position is assigned based on the leader UAVs position. The results via ardupilot SITL verifies that the UAVs are able to keep a predetermined formation for mission accomplishment. In the flight test, we conducted using the ducted-fan UAV manufactured in-house [3]. A ducted-fan UAV is equipped with Pixhawk for flight control and a Raspberry Pi3 for UDP communication [4]. The data communication between the ducted-fan UAVs is performed using Dronekit-python.
This paper is organized as : Section 2 describes the formation flight system. Section 3 discusses the formation flight simulation. Section 4 consist of the flight test configuration and flight test results.

*Corresponding author : skim78@cnu.ac.kr

2 FORMATION FLIGHT SYSTEM

2.1 *Ducted-fan UAV's dynamics and controller design*

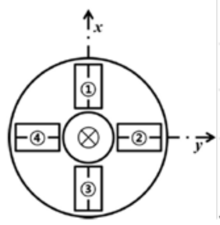
Ducted-fan UAV has an advantage of increasing the thrust efficiency compared to other UAV by equipping a duct. However, unlike the multi-rotor UAV, there is a disadvantage in that it requires to cancel the anti-torque generated by the main rotor. To solve this problem, the ducted-fan UAV compensates for the anti-torque generated by mounting the stator under the rotor [5]. Four vanes located at the end of the duct provide roll, pitch, and yaw control action. The names and effects of each vanes are shown in Figure 1. Equation 1 is ducted-fan UAV's 6 DOF dynamic model equations. Equation 2, 3 shows the ducted-fan UAV's total force and total moment equation.

$$\begin{aligned}\dot{u} &= vr - wq + (F_x)/m \\ \dot{v} &= wp - ur + (F_y)/m \\ \dot{w} &= uq - vp + (F_z)/m \\ \dot{p} &= \{qr(I_{yy} - I_{zz}) + M_x\}/I_{xx} \\ \dot{q} &= \{pr(I_{zz} - I_{xx}) + M_y\}/I_{yy} \\ \dot{r} &= \{pq(I_{xx} - I_{yy}) + M_z\}/I_{zz}\end{aligned} \tag{1}$$

where

$$F_{total} = F_{fuse} + F_{prop} + F_{duct} + F_{flap} + F_{grav} \tag{2}$$

$$M_{total} = M_{fuse} + M_{prop} + M_{duct} + M_{flap} + M_{gyro} \tag{3}$$



Sense	Flaps	Effect
$+\delta_{ail}$	①, ③	$+M_x$
$+\delta_{ele}$	②, ④	$+M_y$
$+\delta_{rud}$	①, ②, ③, ④	$+M_z$

Figure 1: control flaps sign conventions.

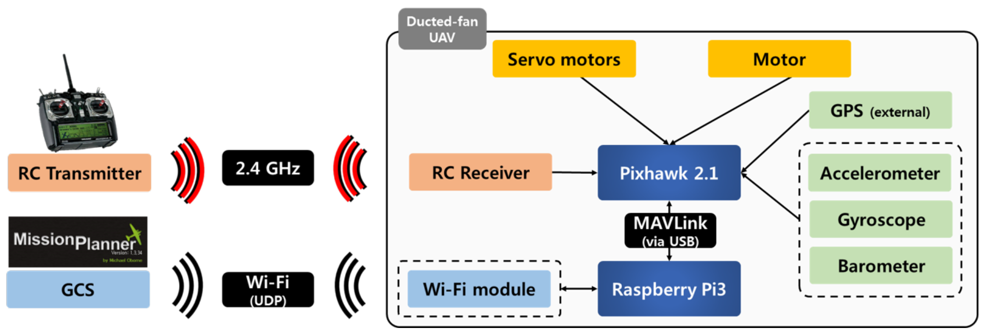


Figure 2: Ducted-fan UAV's components.

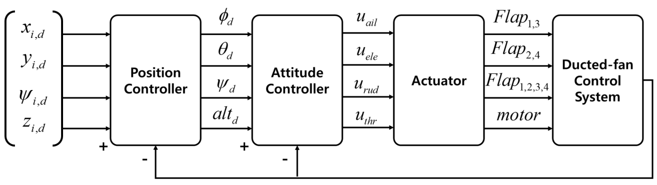


Figure 3: Ducted-fan UAV's control system diagram.

The baseline controller of the ducted-fan UAV is based on PID controller which is provided by arducopter. This attitude controller is designed as an angular position and rate cascaded P-PID inner-outer loop structure. Figure 3 shows the control block diagram of the ducted-fan UAV. The inner loop rate command can be defined as

$$\begin{aligned} e_p &= K_\phi(\phi_d - \phi) - p \\ e_q &= K_\theta(\theta_d - \theta) - q \\ e_r &= K_\psi(\psi_d - \psi) - r \end{aligned} \tag{4}$$

where ϕ_d, θ_d and ψ_d are desired commands, e_p, e_q and e_r are the inner loop rate error. K_ϕ, K_θ and K_ψ are the proportional gains for the Euler angle error for outer loop. Virtual control inputs which act as aileron, elevator and rudder can be calculated as

$$\begin{aligned} u_{ail} &= K_p^p \cdot e_p + K_i^p \int e_p d\tau + K_d^p \frac{de_p}{d\tau} \\ u_{ele} &= K_p^q \cdot e_q + K_i^q \int e_q d\tau + K_d^q \frac{de_q}{d\tau} \\ u_{rud} &= K_p^r \cdot e_r + K_i^r \int e_r d\tau + K_d^r \frac{de_r}{d\tau} \end{aligned} \tag{5}$$

where $K^p_{p,i,d}, K^q_{p,i,d}, K^r_{p,i,d}$ represents proportional, integral and derivative gains of body angular rate (p,q,r) controller. The ducted-fan UAV's position controller is designed as P controller. The position error in the inertial frame can be transformed to the body frame position error depending on the current heading angle (ψ) as follows.

$$\begin{aligned} e_{x\_body} &= e_{x\_in} \cdot \cos\psi + e_{y\_in} \cdot \sin\psi \\ e_{y\_body} &= e_{y\_in} \cdot \cos\psi - e_{x\_in} \cdot \sin\psi \end{aligned} \tag{6}$$

where (e_{x_body}, e_{y_body}) represents the position error in the body coordinate system and (e_{x_in}, e_{y_in}) represents the error in the inertial coordinate system.

2.2 *Development of Ducted-fan UAV*

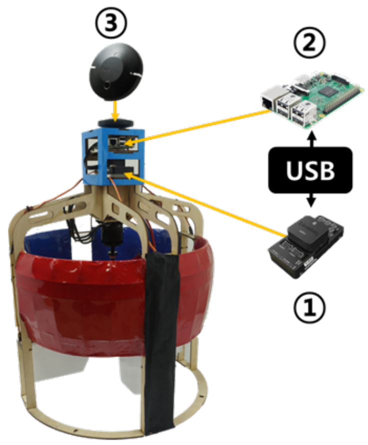


Figure 4: Ducted-fan UAV's equipment.

Figure 4 shows the equipments mounted on the ducted-fan UAV. The ducted-fan UAVs are controlled by an open source flight controller board called ① Pixhawk2.1[6]. The Pixhawk2.1 is equipped with gyroscope, 3D accelerometers and barometer, and it can be connected to ③ an external GPS module. The Pixhawk2.1's software is able to obtain the UAV's attitude and position from these sensors. It enables automatic flight through the obtained attitude data and position data.
In general, when controlling the UAV via Pixhawk2.1, the pilot controls via radio controller or receives commands from a Ground Control System (GCS). However, it is able to communicate with other companion computer via a protocol called MAVLink [7]. In this paper, we use MAVLink by using USB to send commands from a companion computer(② Raspberry Pi3) which is mounted on ducted-fan UAV. In this way, the UAV performs automatic flight as computed by the companion computer. Communication with the GCS is done via UDP communication using the built in Wi-Fi module raspberry pi3. Figure 2 shows the total configuration and architecture connections for the ducted-fan UAV.

2.3 *Formation Flight Architecture Design*

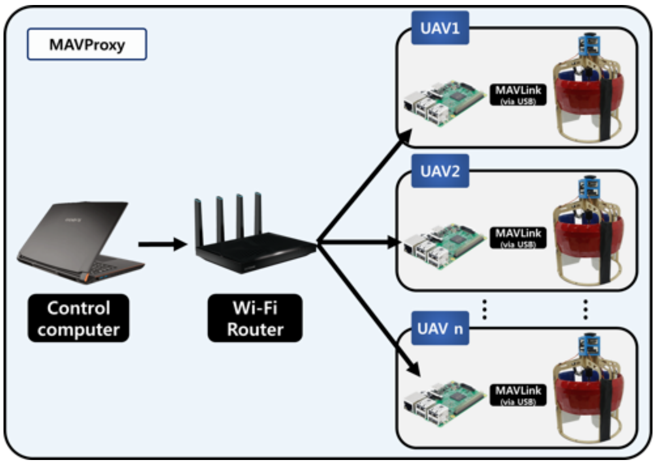
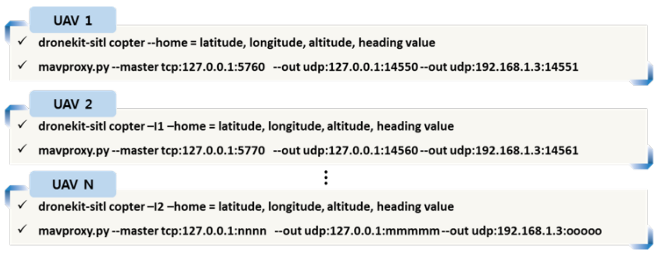


Figure 5: Formation Flight System.

The physical hardware and connections of the flight system are depicted in Figure 5. The system consists of a control computer, a Wi-Fi router, companion computers (Raspberry Pi3) and ducted-fan UAVs. The GCS calculates the formation flight algorithm on the control computer. After that, UDP communication environment is established through Wi-Fi router [8], and the calculated result is transmitted to the raspberry Pi3 mounted on the ducted-fan UAV. The Wi-Fi router connects the control computer to the UAV's companion computers which have different IP and port addresses. Further, UAV companion computers also serve to send those directives to the Pixhawk2.1 via the MAVLink. When the user inputs a formation flight mission on the control computer, the companion computer receives information about the mission and current UAV's flight plan. The flight plan is determined differently according to each UAV, and the corresponding flight plan is transmitted to each Pixhawk2.1. The UAV receives the command and sends information about its current GPS position and attitude status to the companion computer. This UDP communication environment could be constructed through Dronekit-python.

2.4 *Dronekit-python framework*



UAV 1
```
✓ dronekit-sitl copter --home = latitude, longitude, altitude, heading value
✓ mavproxy.py --master tcp:127.0.0.1:5760  --out udp:127.0.0.1:14550 --out udp:192.168.1.3:14551
```
UAV 2
```
✓ dronekit-sitl copter –I1 –home = latitude, longitude, altitude, heading value
✓ mavproxy.py --master tcp:127.0.0.1:5770  --out udp:127.0.0.1:14560 --out udp:192.168.1.3:14561
```
⋮

UAV N
```
✓ dronekit-sitl copter –I2 –home = latitude, longitude, altitude, heading value
✓ mavproxy.py --master tcp:127.0.0.1:nnnn  --out udp:127.0.0.1:mmmmm --out udp:192.168.1.3:ooooo
```

Figure 6: Connect dronekit-sitl and MAVProxy.

Dronekit-python is based on Python which allows developers to create apps that run on a companion computer and communicate with the ArduPilot using a low-latency link [9]. The API communicates with vehicles over 'MAVProxy' which is based on MAVLink [10]. It provides programmatic access to a connected vehicles telemetry and state information, and enables both mission management. In this paper, Dronekit-python runs in the control computer for connecting multi UAVs via UDP communication in order to update the UAV's GPS position data in real time. Figure 6 shows the commands used to connect Dronekit-sitl. It receives data from '– master' and transfers to '– out' IP address.

2.5 *'Leader-Follower' formation flight algorithm*

The Leader-Follower formation flight is an algorithm that calculates the relative coordinates of Follower UAV based on the position of the Leader UAV and maintains the flight through it. The advantage of this algorithm is that it is easier to construct than other formation flight algorithms because it only needs GPS relative position. Equation 7 represents the relative vector between the Leader UAV positons (X_L) and the Follower UAV positions (X_F) [11]. We constructed the formation to keep the GPS position and barometer altitude data of each UAV in accordance with Equation 1 to describe the relative position for Leader-UAV and Follower-UAV pair at Figure 7.

$$S_{L,F} = X_L - X_F = \begin{bmatrix} latitude_L - latitude_F \\ longitude_L - longitude_F \\ altitude_L - altitude_F \end{bmatrix} \quad (7)$$

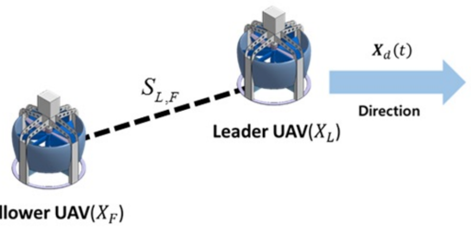


Figure 7: Formation Flight concept drawing.

3 FORMATION FLIGHT SIMULATION

3.1 *Simulation Environment*

In this part, the Leader-Follower formation flight algorithm written using Dronekit-python is verified via simulations before flight test. The flight simulation model uses the 'singlecopter' model provided by 'Ardupilot' [12]. This model is the same as the firmware used in actual flight. The GCS uses the commercial GCS 'Mission Planner' for UDP communication. The formation flight mission is carried out in the following order.

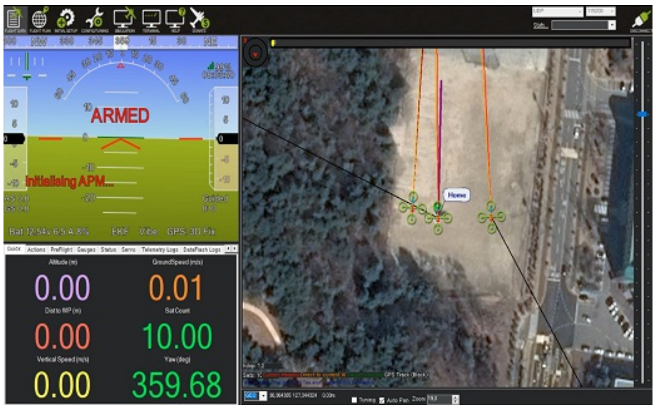


Figure 8: Formation Flight Simulation GCS Monitor.

• Two UAVs take off to 5m.
• 'Follower UAV' moves to position according to formation flight algorithm.
• It keeps formation flying to the way points.
• After arriving at the end point, 'Leader UAV' do Return to launch(RTL).
• 'Follower UAV' do land at the end point.

In this simulation, the ducted-fan UAV's attitude controller uses a PID controller same as using for flight test. It is assumed that the attitude control of the UAV is well controlled by PID controller's gain tunning. The simulation location was selected as the same filed as the flight test location. The formation interval between the 'Leader UAV' and the 'Follower UAV' follows the relative position vector of Equation 8.

$$S_{L,F} = X_L - X_F = \begin{bmatrix} -0.00007 \\ 0.00007 \\ -2 \end{bmatrix} \quad (8)$$

3.2 *Formation Flight Simulation Results*

Figure 9 is the formation flight simulation's 2D path graph. $\bullet$ is start point of each UAV, and $\bigcirc$ is destination of each UAV. Follower UAV forms a formation with Leader UAV in order. The Leader UAV returned to launch after it arrived the last point. At this time, the altitude is 15m which is Ardupilot's default setting values. Figure 10 shows the formation flight simulation's 3D path. Simulation results in a slight delay when the 'Follower UAV' follows the 'Leader UAV'. This is due to that the GPS position data of the 'Leader UAV' sent in real time and the 'Follower UAV position' is calculated afterwards. Prior to the ducted-fan UAVs formation flight test, we confirm that the current formation flight algorithm is feasible through this simulation.

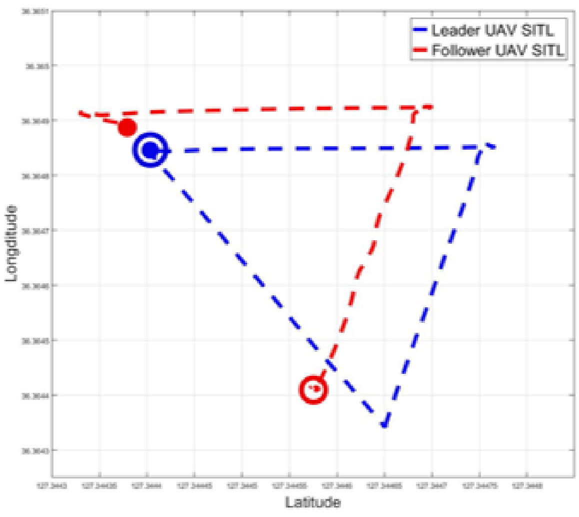


Figure 9: In the simulation 2D path.

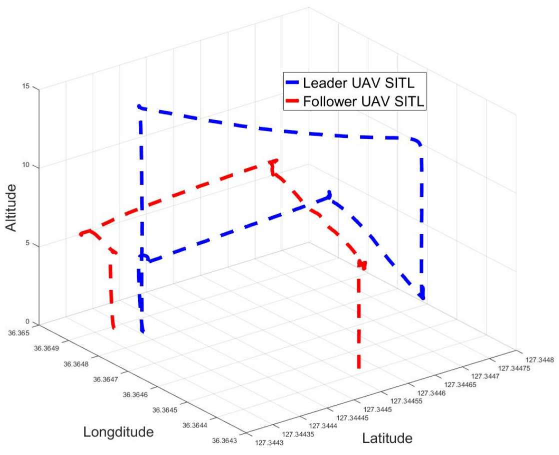


Figure 10: In the simulation 3D path.

4 FORMATION FLIGHT TEST

4.1 *Formation Flight Test Setup*

This section builds a formation flight system environment for using the formation flight simulation code. Figure 11 shows the configuration required for the formation flight tests. This test performs the same tasks, location and dronekit-python code as the simulation. Two ducted-fan UAVs of the same specification are divided into the leader UAV and the follower UAV. The GCS computer runs the 'Mission Planner' and the Dronekit-python computer runs the dronekit code. The connection between the ducted fan UAV and the two computers uses UDP communication using the MAVProxy in the Wi-Fi environment.

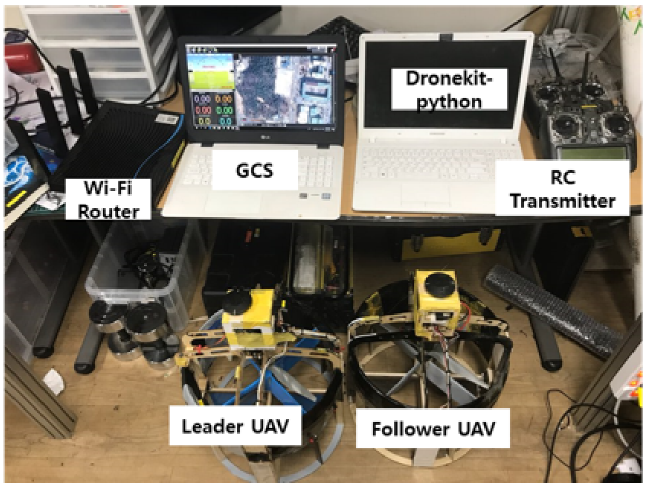


Figure 11: Formation Flight Test Setting.

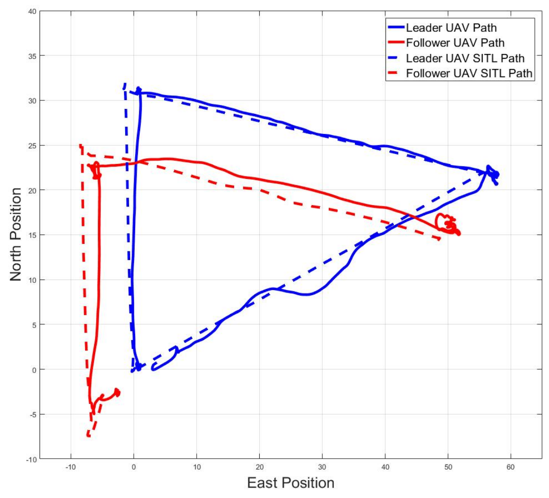


Figure 13: 2D path (Formation Flight Test).

4.2 *Formation Flight Test Results*

Figure 12: Formation Flight Test 1.

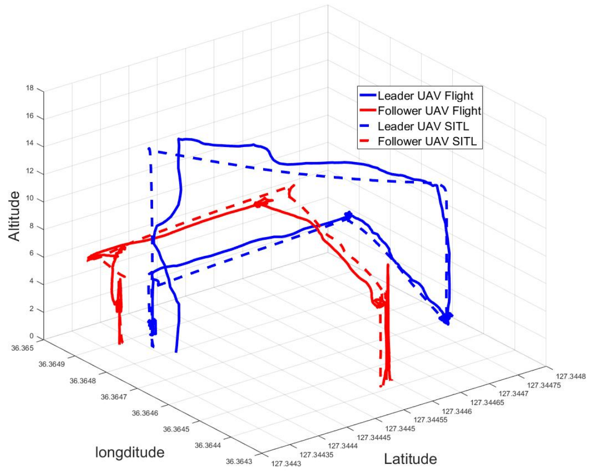


Figure 14: 3D path (Formation Flight Test).

Formation flight test is performed with the same procedure as the simulation. Figure 12 shows the second formation flight path. Figure 13- 14 illustrate 2D and 3D trajectories comparing the second simulation with the actual flight path. The 'Leader UAV' and the 'Follower UAV' followed the commands for keeping their formation. When the 'Leader UAV' returned to launch, it landed at a position away from the target point when landing. This might be caused by GPS sensor error. As a result of the comparison between the two simulations and the flight test, it is confirmed that similar results were obtained when the flight test is performed within the Wi-Fi communication range.

Table 1 shows the mean distance error in the simulation and flight test for the formation flight command. There is an error of about 20%, which is interpreted to be caused by the GPS sensor error and the transmission delay in real time position data.

Table 1: Comparing distance

	CMD	SITL	FlIGHT
Mean distance (m)	10.0995	12.6979	13.8934
Error (%)	–	20.5	27.3

5 Conclusions and Future work

This paper, developed a Leader Follower ducted-fan UAV formation flight algorithm using dronekit-python. In addition, MAVProxy was used to construct formation flight simulation and flight test environment. After that, the simulation was performed and compared with the flight test. In the future, we will supplement the algorithm to reduce the GPS data delay, increase the response rate of the follower, and conduct formation flight tests with number of ducted-fan UAVs.

Acknowledgements

This work is supported by the research project (No. 10062497) funded by the Ministry of Trade, Industry and Energy of Korea, Republic of.

References

[1] Rafael G Braga, Roberto C da Silva, Alexandre CB Ramos, and Felix Mora-Camino. A hybrid approach for 3d formation control in a swarm of uavs using ros.

[2] Zainab Zaheer, Atiya Usmani, Ekram Khan, and Mohammed A Qadeer. Aerial surveillance system using uav. In *Wireless and Optical Communications Networks (WOCN), 2016 Thirteenth International Conference on*, pages 1–7. IEEE, 2016.

[3] Taegyun Kim, Yoonjeong Jang, Minkyu Kim, Junho Jeong, Jinyoung Suk, and Seungkeun Kim. Development and flight test of an electric powered small ducted-fan uav. *KSAS Spring Conference*, pages 210–211, 2017.

[4] Rafael Braga, Alexandre Carlos Brandao Ramos, Roberto Claudino Da Silva, and Félix Mora-Camino. A combined approach for 3d formation control in a multi-uav system using ros. In H. de Plinval J.-M. Moschetta, G. Hattenberger, editor, *International Micro Air Vehicle Conference and Flight Competition 2017*, pages 196–202, Toulouse, France, Sep 2017.

[5] Hong Zhao, Shou-Zhao Sheng, Jian-Bo Li, and Chen-Wu Sun. Modelling and attitude control of a miniature ducted fan uav. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 230(5):953–964, 2016.

[6] Pixhawk homepage. https://pixhawk.org/. Accessed: August, 2018.

[7] Mavlink homepage. https://mavlink.io/en/. Accessed: August, 2018.

[8] Brad Hyeong-Yun Lee, James R Morrison, and Rajnikant Sharma. Multi-uav control testbed for persistent uav presence: Ros gps waypoint tracking package and centralized task allocation capability. In *Unmanned Aircraft Systems (ICUAS), 2017 International Conference on*, pages 1742–1750. IEEE, 2017.

[9] Dronekit-python homepage. http://python.dronekit.io/. Accessed: August, 2018.

[10] Mavproxy homepage. https://ardupilot.github.io/MAVProxy/html. Accessed: August, 2018.

[11] Matthew Turpin, Nathan Michael, and Vijay Kumar. Trajectory design and control for aggressive formation flight with quadrotors. *Autonomous Robots*, 33(1-2):143–156, 2012.

[12] Ardupilot homepage. http://ardupilot.org/ardupilot/. Accessed: August, 2018.