Multi-UAV Systems for Scalability in Last-mile Logisitics with Formation Control

Dr. Chaouki T. Abdallah[1], Abhinandan Krishnan[2], Navami Srinivas Prabhu[3], Prerana Kolipaka[4], and Yash Srivastava[5]

[1,2,3,4,5]Georgia Institute of Technology, Atlanta, GA

***Abstract*—This paper primarily focuses on the application of a swarm of multi-rotor UAV systems to scale up efficiently in last mile delivery applications. The paper discusses a series of control regimes of a swarm of multi-rotor UAV systems to rendezvous at a location, assemble in the desired formation and transport the payload collaboratively to the desired location. This work uses ArduPilot gazebo plugin to perform the simulation and verify the control architecture.**

***Index Terms*—Networked control, Rendezvous problem, Formation control, ArduPilot, Quadrotors, Multi-robot systems.**

I. Introduction

Rapid technological advancements in autonomous unmanned aerial vehicles (UAVs) or drones over the last few years have led to an increased interest in their use, particularly for last-mile delivery applications. The growth in e-commerce and the expansion in urbanization have made drones an attractive solution, as they can provide unmanned delivery services while also addressing last-phase supply chain problems. With the dire need to move towards sustainable solutions, drones are the most promising alternative to on-road delivery and transportation services. Drones could lower the use of fossil-based fuels and thus contribute to reducing environmental emissions. In fact, they could also reduce traffic congestion, hence road accidents, and provide substantial coverage, especially in rural areas, where the road network is otherwise relatively inaccessible through cars or motorcycles. The use of automation for last-mile delivery, though challenging, could radically decrease labor and capital costs, making it more economically viable, and potentially disrupt the package delivery industry. Moreover, the increased demand for contactless delivery over the last three years due to COVID-19 has accelerated the deployment of drone-based services not only in the retail, food, and the merchandise sector but also in the health sector for the transportation of particular medicines, goods, and biological tests to hospitals and laboratories.

Such last-mile delivery applications employing drones necessitate the use of formation control to coordinate their movement. This would allow a set of drones/UAVs to align in the desired shape, namely a triangle, square, and so on, or even in the most unorthodox configurations, and can thus perform specific tasks. Depending on the existence of a control center in the formation, the formation control techniques can be categorized into centralized and decentralized formation control. The centralized approach includes a control center, say a ground station, that communicates with all the drones. As a result, once the control center crashes, the formations may break down, making them less robust. Unlike the centralized method, the decentralized strategy provides equal status to all drones in the formation and does not require a control center for information exchange or decision-making. Hence, it is ideal for bringing about the formation of a swarm of drones. It is also easier to accomplish and is robust as well as scalable. As every drone has the same capabilities, ideally, it is more autonomous and flexible, especially in the case of emergencies. The most common methods to achieve this include a leader-follower strategy, behavioral-based control, and a virtual structure-based approach. The leader-follower technique consists of a designated leader or the reference that follows a pre-determined path while the followers maintain their positions and velocities relative to that of the leader. There is no direct feedback from the followers to the leader or vice versa in this method. In the behavior-based strategy, each drone studies the information from the adjacent drones to design rules for cohesion, separation, and so on. It relies on the coordination mechanism between the behaviors of the neighboring drones. The primary advantage is that it allows communication amongst the drones, but it is challenging to ensure that all attributes of the formation are achieved through this method. On the contrary, the virtual structure approach assumes a geometric center of the formation based on the orientation and speed of the drones in the formation. The configuration is rigid and is guided by the dynamics of the virtual leader drone. If one of the drones misaligns, the overall swarm can modify its trajectory to assist that drone. However, it is rather difficult to execute in real-time scenarios [1] - [3].

Various methods have been proposed in literature to achieve formation control for a flock of drones. [4] discusses formation control using output consensus feedback control for a group of non-linear agents. In [5], drone formation is studied in a fixed and controlled laboratory setup which, although provides remarkable results, is expensive. [6] uses the concept of swarm optimization, which is centralized in nature and can result in processing difficulties on a single node along with the chances of a single point of failure. The use of image processing for formation control and navigation is illustrated in [7], but there are underlying challenges with the application of computer vision.

This project used different networked control strategies to form the desired formations. First, a using multi-dimensional consensus over positions in the 3-D space, all the agents rendezvous to the centroid of the system, with appropriate checks in place to ensure they maintain some distance. Then using a combination of the leader-follower model and rigid-formation control, the drones navigate between different waypoints. This project uses the Ardupilot-Gazebo plugin [12] to ensure that the simulations are consistent with the real world in terms of physics, communication latency, and drone flight behavior.

The rest of the report is organized as follows. Section II discusses the mathematical formulation for rendezvous and leader-follower based rigid formation control. Section III delves into the implementation of the algorithms in the simulation environment, and the results obtained through simulation for different scenarios are presented and analyzed. Section IV concludes the work and outlines future considerations for the project.

II. Control Strategy

Ardupilot provides open-source quadrotor firmware, using which we can abstract away the low level control and use the quad rotors global positions and velocities to formulate our control regime. We have divided our approach into 3 different stages, viz. Rendezvous problem, Rigid Formation and Leader follower problem. The adjacency matrix is updated with time. This is done to ensure that when an agent moves out of the communication range Δ, the connection is lost and the adjacency matrix is updated. The inter-agent distances are measured and the adjacency matrix is updated accordingly given in (1).

$$A_{ij} \equiv \begin{Bmatrix} 1, & \text{if } \|x_i - x_j\| < \Delta \\ 0, & \text{if } \|x_i - x_j\| \geq \Delta \end{Bmatrix}, \tag{1}$$

A. Rendezvous Problem

For forming a Rigid formation, it is desirable to have a complete graph i.e every agent is within the communication range of every other agent. To ensure this, Rendezvous algorithm given by (2) is used to make agents meet at the centroid of its initial positions. When all the agents are within communication range the control strategy is switched to formation control.

$$\dot{x_i}(t) = \sum_{j \in N_i} A_{ij}(t) \cdot (x_j(t) - x_i(t)) \tag{2}$$

B. Formation Control

After the agents come close enough to form a complete graph, the control sequence is switched to formation control given by (3) which is used to drive the agents into a desired formation. The formation used for this application is rigid, i.e., the edge lengths are fixed. The edge lengths are defined using the weight matrix, where w_{ij} defines the distance between agents i and j. As we have ensured that the agents form a complete graph, the rank of the Rigidity matrix is is $3*N-6$ i.e full rank where N is the number of agents [9]. This ensures that the rigidity is maintained between the agents.

$$\dot{x_i}(t) = \sum_{j \in N_i} A_{ij}(t) \cdot (w_{ij} - \|x_i(t) - x_j(t)\|) \cdot (x_j(t) - x_i(t)) \tag{3}$$

III. Implementation and Results

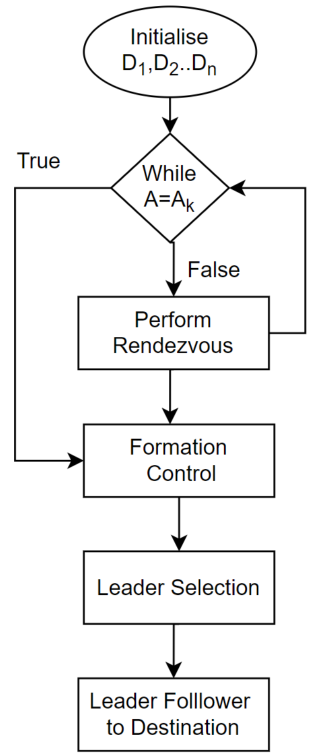

Fig. 1. Process Flowchart

For our application, we use a swarm of 4 drones D_1, D_2, D_3, D_4. These drones carry packages from the pick-up location to the destination. All these drones are initially present at the base station where they are randomly placed. When a delivery task is assigned to these drones, they first come together (rendezvous) so that all the drones are in a fully connected network. The adjacency matrix is then updated to one of a complete graph. Once rendezvous is achieved, the drones form a line formation. The drones then travel to the pick-up location maintaining a line formation and uses leader-follower control law for movement, see Fig. 4. The drone whose position is closest to the destination is assigned as the leader of the swarm. Once they reach the pick-up location, to align with the package, the agents form a square formation and lower its altitude so that the payload can be latched. The agents then move to the destination where the package needs to be dropped in a leader-follower fashion, maintaining the formation. The process flowchart is explained in Fig. 1. As the agents are localised using GPS coordinates, to account for the spherical shape of the Earth haversine distance given by (4) is used to calculate the inter agent distances and the distance error between goal and destinations

[8]. For practical considerations, the velocities obtained from the control equations are bounded using a maximum threshold to ensure the velocity inputs to quad-rotors are not saturated.

$$d = 2r\ arcsin\sqrt{sin^2\left(\delta_1\right) + cos(\theta_1)cos(\theta_2)sin^2\left(\delta_2\right)} \quad (4)$$

$$\delta_1 = \frac{\theta_2 - \theta_1}{2}, \delta_2 = \frac{\lambda_2 - \lambda_1}{2} \quad (5)$$

where, r is the radius of the chosen sphere. θ and λ are the latitude and longitude respectively.

A. *Leader Follower Algorithm*

Once the desired formation is achieved, a leader is picked among the agents and the rest of the agents act as followers. The leader has information about the direction and position of the destination. This leader agent drives all the follower agents to the desired destination.

The dronekit python library used for creating the simulation uses the NORTH-EAST-DOWN (NED) frame of reference. NED Frame consists of 3 orthogonal axes, the N_x , N_y and N_z. The N_x points to the true north, the N_y points to the East and N_z points towards the interior of the Earth. As per NED frame of reference, the heading angle is given by,

$$\theta = \tan^{-1}\frac{\Delta x}{\Delta y} \quad (6)$$

atan2 function is used for calculating θ in the interval [-π, π].

The followers follow the same control law as the rigid formation control law to make sure the formation is maintained at all times. The control law followed by the leader is given (7). For the leader to reach the destination a pre defined velocity of V is given. A gain k is used to scale up or down the position values to velocity range.

$$\begin{aligned} V_x &= V\cos\theta \\ V_y &= V\sin\theta \\ V_z &= -k\Delta z \end{aligned} \quad (7)$$

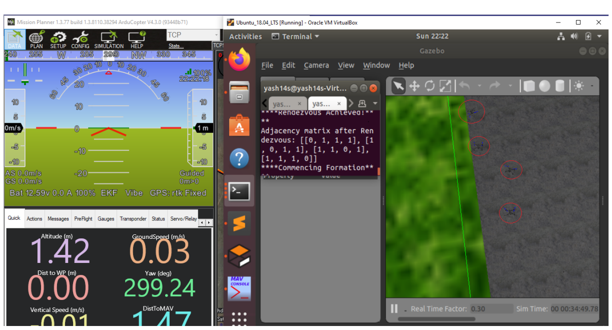

Fig. 2. Formation of drones on gazebo

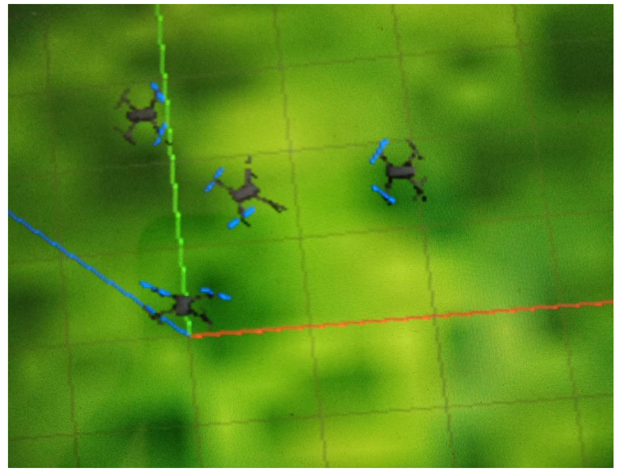

Fig. 3. Triangle formation of drones on gazebo

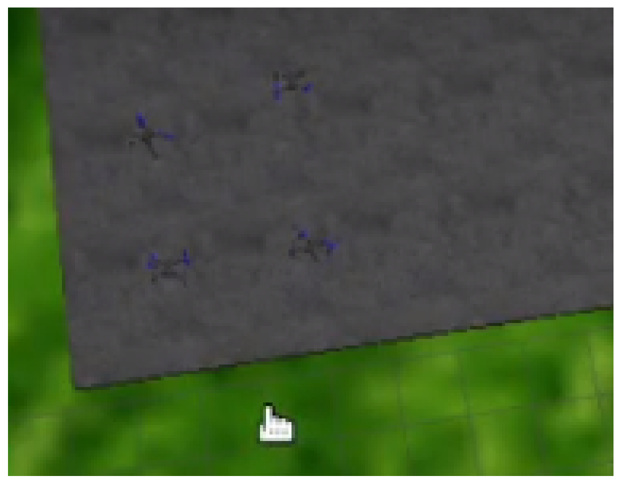

Fig. 4. Square formation of drones on gazebo

IV. Conclusion and Future Work

In this project, we present different formation control strategies for UAV swarms tailored for last-mile logistics applications. By using Gazebo for creating the simulation environment and Ardupilot firmware for the drones, we have created realistic simulations. The implementation is agnostic of the multi-rotor frame, and the same code can be deployed on real-world multi-rotor UAVs by only configuring the communication port. Future work includes adding low-cost sensors such as ultrasonic sensors etc. for improving collision avoidance capabilities of each UAV. Furthermore different applications such as search and rescue could be explored and concepts like maximum coverage can be applied. And, deployment on Ardupilot based UAVs (hardware) can be done.

References

[1] Wu, F., Chen, J. & Liang, Y. Leader-follower formation control for quadrotors. *IOP Conference Series: Materials Science And Engineering*. **187**, 012016 (2017)

[2] Ali, Z., Israr, A., Alkhammash, E. & Hadjouni, M. A leader-follower formation control of multi-UAVs via an adaptive hybrid controller. *Complexity*. **2021** (2021)

[3] Ouyang, Q., Wu, Z., Cong, Y. & Wang, Z. Formation control of unmanned aerial vehicle swarms: A comprehensive review. *Asian Journal Of Control*. (2022)

[4] Wang, W., Huang, J., Wen, C. & Fan, H. Distributed adaptive control for consensus tracking with application to formation control of nonholonomic mobile robots. *Automatica*. **50**, 1254-1263 (2014)

[5] Turpin, M., Michael, N. & Kumar, V. Trajectory design and control for aggressive formation flight with quadrotors. *Autonomous Robots*. **33**, 143-156 (2012)

[6] Spanogianopoulos, S., Zhang, Q. & Spurgeon, S. Fast formation of swarm of UAVs in congested urban environment. *IFAC-PapersOnLine*. **50**, 8031-8036 (2017)

[7] Upadhyay, J., Rawat, A. & Deb, D. Multiple Drone Navigation and Formation Using Selective Target Tracking-Based Computer Vision. *Electronics*. **10**, 2125 (2021)

[8] H. Mahmoud and N. Akkari, "Shortest Path Calculation: A Comparative Study for Location-Based Recommender System," 2016 World Symposium on Computer Applications & Research (WSCAR), 2016, pp. 1-5, doi: 10.1109/WSCAR.2016.16.

[9] Asimow, L. & Roth, B. The rigidity of graphs, II. *Journal Of Mathematical Analysis And Applications*. **68**, 171-190 (1979,3), https://doi.org/10.1016/0022-247x(79)90108-2

[10] https://dronekit-python.readthedocs.io/en/latest/examples/guided-set-speed-yaw-demo.html

[11] https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html

[12] https://github.com/khancyr/ardupilot_gazebo

[13] https://github.com/Intelligent-Quads/iq_tutorials/blob/master/ /swarming_ardupilot.md

[14] http://wiki.ros.org/melodic/Installation/Ubuntu