

Implementation of UAV Coordination Based on a Hierarchical Multi-UAV Simulation Platform

Kun Xiao¹, Lan Ma², Shaochang Tan³, Yirui Cong², and Xiangke Wang²

¹ Beijing Institute of Aerospace Systems Engineering, Beijing 100076, China,

² College of Intelligence Science and Technology, National University of Defense Technology, Changsha 410073, China

³ Flying College, Beihang University, Beijing 100191, China
xkwang@nudt.edu.cn

Abstract. In this paper, a hierarchical multiple unmanned aerial vehicle(UAV) simulation platform, called XTDrone, is designed for UAV swarms, which is completely open-source⁴. There are six layers in XTDrone: communication, simulator, low-level control, high-level control, coordination, and human interaction layers. XTDrone has three advantages. Firstly, the simulation speed can be adjusted to match the computer performance, based on the lockstep mode. Thus, the simulations can be conducted on a work station or on a personal laptop, for different purposes. Secondly, a simplified simulator is also developed which enables quick algorithm designing so that the approximated behavior of UAV swarms can be observed in advance. Thirdly, XTDrone is based on Robot Operating System(ROS), Gazebo, and PX4, and hence the codes in simulations can be easily transplanted to embedded systems. Note that XTDrone can support various types of multi-UAV missions, and we provide two important demos in this paper: one is a ground-station-based multi-UAV cooperative search, and the other is a distributed UAV formation flight, including consensus-based formation control, task assignment, and obstacle avoidance.

Keywords: multi-UAV, hierarchical, simulation platform, UAV coordination, distributed

1 Introduction

Unmanned aerial vehicles (UAVs) develop rapidly due to their large potential in both civilian and military uses, such as disaster rescue, reconnaissance and surveillance. As missions becomes increasingly complex, the importance of multi-UAV cooperation grows. Therefore, a large number of researchers focuses on multi-UAV cooperation or UAV swarms in recent years [1,2,3,4,5].

Algorithm design and validation can waste time and energy without a reliable simulation platform, and thus much attention was drawn for simulation

⁴ Source code at https://gitee.com/robin_shaun/XTDrone
or <https://github.com/robin-shaun/XTDrone>,
and user manual at https://www.yuque.com/xtdrone/manual_en.

platform developing [6,7,8,9,10]. However, few platforms are open source and user friendly. Aerostack⁵ developed from [8], is the most popular multi-UAV simulation platform according to our investigation. However, compared to single UAV simulation platforms, such GAAS⁶ and RotorS⁷, the numbers of stars and forks in github are much smaller. There is still an urgent need for a user-friendly multi-UAV simulation platform for not only researchers but also engineers. Based on this motivation, A hierarchical and modular multi-UAV simulation platform called XTDrone is developed.

Considering usability, development efficiency and open source community, ROS⁸, Gazebo⁹, PX4¹⁰ and QGroundControl¹¹ are chosen as four bases of XTDrone. Python is the main development language and some outstanding C++ open source projects are also integrated. The platform is divided into six layers: communication, simulator, low-level control, high-level control, coordination and human interaction layers. In each layer, there are different kinds of modules, such as task allocation, Simultaneous localization and mapping (SLAM), object detection, trajectory generation, position controller and so on. All of these modules can be replaced conveniently because the input and output messages are well-defined. Therefore, platform developers can realize and test their own algorithm conveniently. Because ROS and PX4 is originally designed for embedding system, developers can deploy their algorithm to real UAVs conveniently after testing and debugging on the simulation platform.

High computational consumption is an important problem for multi-UAV simulation. Powerful workstations are not only expensive but also inconvenient compared to laptops. XTDrone runs in the lockstep mode, meaning that different numerical solvers maintain synchronized time, which makes it possible to run the simulation faster or slower than real time, and also to pause it in order to step through code. A powerful computer runs faster and less powerful one runs slower. Therefore, this feature makes computers with different performance possible to be used for simulation. Furthermore, a simplified simulator is provided, so developers can firstly test and debug their algorithm on the simplified simulator with a large-scale swarm. And then when simulating in Gazebo, they can choose to use a powerful workstation for a large-scale swarm, or reduce the number of UAVs.

XTDrone is firstly a single UAV simulation platform and then a multi-UAV one. Reference [11] has provided details about single UAV simulation, and this paper focuses on the multi-UAV simulation. The rest of the paper is structured as follows. Section 2 presents the architecture of the simulation platform. And then, two demos are shown to demonstrate how the platform works. Section 3

⁵ <https://github.com/Vision4UAV/Aerostack>

⁶ <https://github.com/generalized-intelligence/GAAS>

⁷ https://github.com/ethz-asl/rotors_simulator

⁸ <https://www.ros.org/>

⁹ <http://gazebosim.org/>

¹⁰ <https://px4.io/>

¹¹ <http://qgroundcontrol.com/>

presents a multi-UAV cooperative search mission, planned by the ground control station. Section 4 and Section 5 presents a distributed UAV formation, including consensus cooperative control, task assignment and obstacle avoidance. Section 4 describes algorithm designs and Section 5 presents its simulation implementation. Section 6 concludes the paper and indicates future work.

2 Architecture

Fig. 1 shows the architecture of XTDrone. Five layers communicate with each other through the communication layer. The architecture is inspired by [12], which is a a multi-layered and distributed architecture for real UAV swarm. To some extent, the architecture of XTDrone is a simulation version of that in [12]. In this section, the six layers are introduced respectively from down to top.

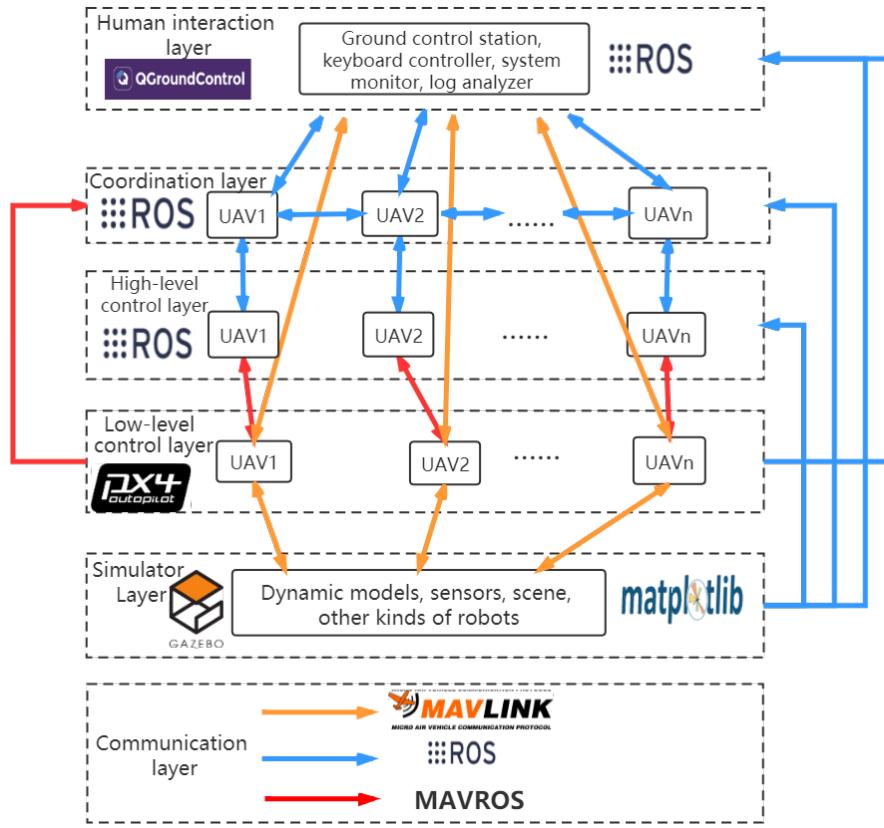


Fig. 1. Architecture of XTDrone

The communication layer is in charge of the messages transmission among all the UAVs and the ground control station, which is the base of all other layers. ROS, MAVLink¹² and MAVROS¹³, supply message-passing in the communication layer. To be user-friendly, XTDrone encapsulates complex protocols, and standardizes the topic names. Developers can conveniently subscribe and publish the needed topics in their designed modules.

The simulator layer supplies UAV dynamic models, sensors, scene and other kinds of robots, all of which is customizable, so that developers can modify the aerodynamics model, or add multiple cameras to the UAV according to their needs. The main simulator is Gazebo. Moreover, for quick algorithm developing in the early stage, a simplified simulator based on Matplotlib¹⁴ is provided. Reference [11] presents more details about this layer.

Low-level control layer is totally based on PX4 software in the loop (SITL), which contains state estimation and low-level control such as position control, attitude control and flight modes. XTDrone focuses more on high-level control layer and coordination layer, and developers can just use this layer without extra modification. To develop algorithm in this layer, developers are expected to be familiar with PX4.

High-level control layer contains perception and motion planning. It's a key layer for intelligence of a single UAV. Object detection and tracking, SLAM, obstacle avoidance, etc. are all in this layer. Some demos have been realized in XTDrone, and developers can modify them or rewrite them according to their needs. For a UAV swarm, this layer receives tasks allocated from coordination layer, and then each UAV achieves the task.

The coordination layer is responsible for the tasks related to negotiation (e.g., task allocation) among UAVs for mission coordination. It divides the total mission into different small missions, and then sends them to high-level control layer. This layer is very flexible, and may contain different modules according to different missions and coordination strategies. For example, in the formation demo (Section 4 and 5), the task assignment and the consensus controller is in this layer.

The human interaction layer is a set of interfaces for the developer to control and monitor the UAV swarm. For UAVs, a typical human machine interface is ground control station, and specifically, XTDrone utilizes QGroundControl, shown as Fig. 2(a). QGroundControl communicates with PX4 SITL through MAVLink, so developers can monitor and adjust parameters and targets of low-level control layer. Besides the low-level layer, coordination layer and high-level layer are controlled and monitored through a set of tools based on ROS, such as a customizable keyboard controller, shown as Fig. 2(b), rviz¹⁵ and rqt¹⁶. Moreover, logging is essential for analyzing the algorithm and debugging the

¹² <https://mavlink.io/en/>

¹³ <https://github.com/mavlink/mavros>

¹⁴ <https://matplotlib.org/>

¹⁵ <https://github.com/ros-visualization/rviz>

¹⁶ <https://github.com/ros-visualization/rqt>

code. For ROS, ROSBag is a useful tool to record logs, and for PX4, flight logs, containing almost every key message, are recorded as ULog files¹⁷. And there are some user-friendly software¹⁸ to analysis ULog files.

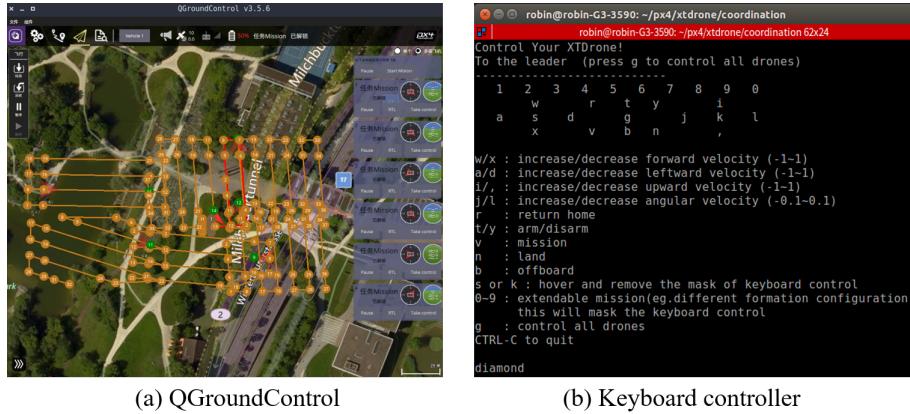


Fig. 2. Ground control station and keyboard controller

3 Cooperative Search Simulation

In this section, a multi-UAV cooperative search is implemented. We plan the trajectories for six UAVs through QGroundControl, the low-level controller of each UAV tracks the desired trajectory and the high-level controller of each UAV detects the target objects.

The mission goal is to search a human in a large region. A tiny-YoloV3 network [13] is trained beforehand, and deployed in the high-level controller layer¹⁹. For searching in a wide region, a simple but efficient way is to divide the region into six non-overlapping regions. Each UAV searches one region and detects the target objects. By setting and uploading way points in QGroundControl, shown as Fig. 2(a), the low-level layer of each UAV tracks the desired flight trajectory, and meanwhile, tiny-YoloV3 works, shown as 3. Finally, the NO.5 UAV detects the human and then the mission is completed.

¹⁷ https://dev.px4.io/master/en/log/ulog_file_format.html

¹⁸ https://docs.px4.io/master/en/log/flight_log_analysis.html

¹⁹ https://www.yuque.com/xtdrone/manual_en/target_detection_tracking

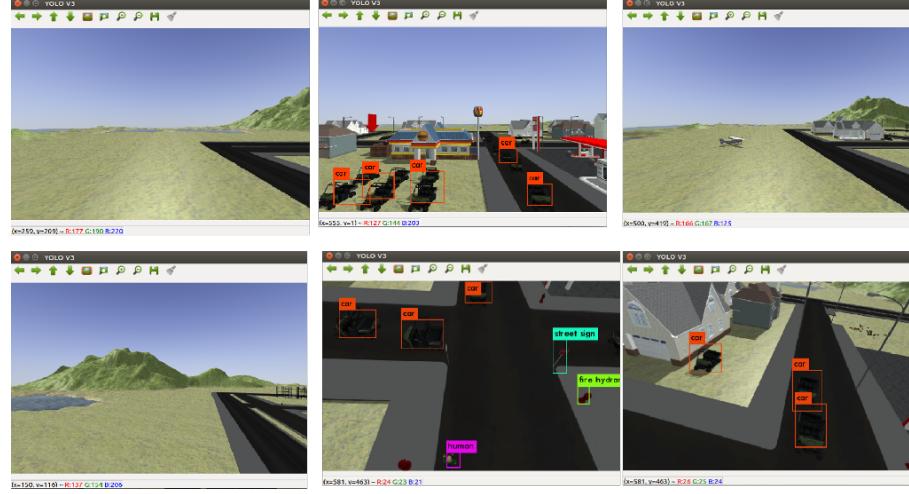


Fig. 3. Human searched by six UAVs

4 Distributed UAV Formation Design

In this section, a design of distributed UAV formation control is represented. A consensus controller, a task assignment strategy and an obstacle avoidance algorithm are designed. The formation is a leader-following formation. Firstly, the leader assigns task to all the followers. And then for each follower, the output velocities of the consensus controller and the obstacle avoidance module are composed to desired velocity in the high-level control layer. And then the desired velocity is sent to the low-level control layer. Finally, the desired formation is achieved.

4.1 Consensus-based Formation

Considering a group of N UAVs, each can be modeled as an first-order integrator model on the basis of its kinematic model, then it can be described as:

$$\dot{\xi}_i = u_i, \quad i = 1, 2, \dots, N \quad (1)$$

where the input $u_i \in \mathbb{R}^3$ is the velocity of UAV i . $\xi_i \in \mathbb{R}^3$ is the position of UAV i . The target state is that all the UAVs achieve a given formation pattern. The formation controller is given as follows:

$$u_i = - \sum_{j=1}^N w_{ij} \{ [(\xi_i - \delta_i) - (\xi_j - \delta_j)] \} \quad (2)$$

where w_{ij} is the (i, j) element of the adjacency matrix $W = [w_{ij}] \in \mathbb{R}^{N \times N}$, which represents that UAV j communicates with UAV i if and only if $w_{ij} > 0$.

δ_i denotes the formation offset of UAV i and it is determined by the desired formation configuration.

We consider that system 2 achieves consensus when $\| \xi_i - \delta_i \) - \(\xi_j - \delta_j \| \rightarrow 0$. Because the consensus of the system and the stability of the formation are equivalent, our system could achieve formation stability in this condition [14].

In [15], Yu Ding built an interaction topology based on their proposed “Veteran Rule”. We use this to determine the communication topology of UAVs, e.g. the adjacency matrix $W = [w_{ij}]$. Assume that the number of UAVs is 6, and the current position is “T” shown in Fig.4, then we get the communication topology as Fig.5 and the adjacency matrix \mathcal{L} as (3)

$$\mathcal{L} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \quad (3)$$

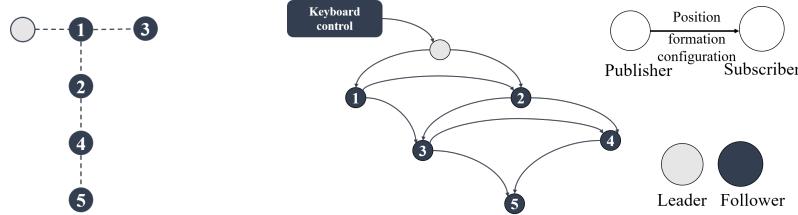


Fig. 4. Current position of 6 UAVs

Fig. 5. Communication topology of 6 UAVs

4.2 Task assignment

We consider formation reconfiguration as a kind of task assignment which sets different UAVs into different formation positions for the shortest total distance while changing formation. We adopt Kuhn-Munkres (K-M) algorithm for task assignment. As the current formation offset and the target formation offset could be obtained by each UAV, we set them as a bipartite graph. Each weight of graph is the distance (in practice turned to negative) between the current formation offset and the target formation offset. The assignment solved by KM algorithm obtains the best total performance with the shortest total distance[16]. For instance, for the formation reconfiguration from ‘T’ to ‘diamond’, the settings and results of KM algorithm are shown in Fig.6

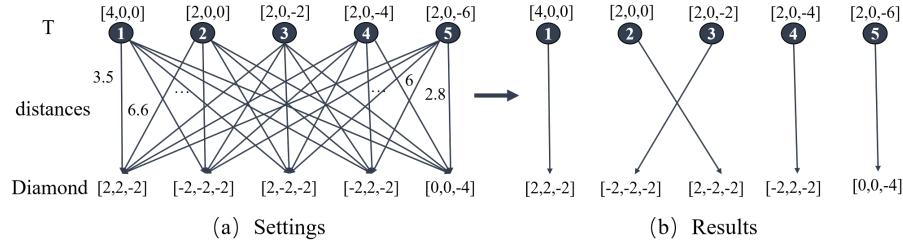


Fig. 6. Settings and results of KM algorithm

4.3 Obstacle avoidance

Fig. 7 shows the strategy of obstacle avoidance. A UAV will avoid when another UAV is in the range of b meters. \mathbf{r} is the distance vector and points to the obstacle UAV. \mathbf{a} is the avoidance vector of UAV1 and \mathbf{a}' is the avoidance vector of UAV3. \mathbf{a} and \mathbf{a}' are in the opposite direction, and perpendicular to \mathbf{r} . Because there are infinite vectors perpendicular to \mathbf{r} , a simple method, shown as Algorithm 1 is designed to identify a unique solution. Two auxiliary vectors \mathbf{n}_1 and \mathbf{n}_2 are used to avoid the cross product result close to zero vector. kp is a proportion factor. If there are more than one UAVs in the range of b meters, avoidance vector \mathbf{a} will be modified several times. The final \mathbf{a} will be added to the desired velocity, and then the desired velocity will be sent to the low-level control layer.

Another question is how to obtain the relative position of other UAVs. Because of the distributed communication, a UAV cannot obtain all other UAVs' absolute positions. There are many researches focused on the cooperative relative localization algorithm [17,18,19]. In this demo, for simplification, the relative position ground truth is sent to each UAV. Developers can just add a relative localization module to replace the ground truth with the module output.

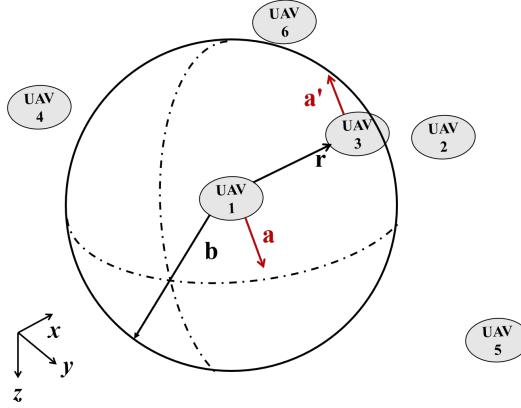


Fig. 7. Strategy of obstacle avoidance

Algorithm 1 Obstacle avoidance

```

1: loop
2:   Initialization:  $\mathbf{a}_0 = [0, 0, 0]$ ;  $\mathbf{n}_1 = [1, 0, 0]$ ;  $\mathbf{n}_2 = [0, 1, 0]$ ;  $i = 1$ 
3:   while  $i <$  number of UAVs in the range of  $b$  meters do
4:     if  $\mathbf{r}_i \cdot \mathbf{n}_1 < \mathbf{r}_i \cdot \mathbf{n}_2$  then
5:        $\mathbf{a}_0 = \mathbf{a}_0 + kp * (1 - |\mathbf{r}_i|/b) * (\mathbf{r}_i \times \mathbf{n}_1 / |\mathbf{r}_i \times \mathbf{n}_1|)$ ;
6:     else
7:        $\mathbf{a}_0 = \mathbf{a}_0 + kp * (1 - |\mathbf{r}_i|/b) * (\mathbf{r}_i \times \mathbf{n}_2 / |\mathbf{r}_i \times \mathbf{n}_2|)$ ;
8:     end if
9:      $i = i + 1$ ;
10:   end while
11: end loop

```

5 Distributed UAV Formation Simulation

XTDrone contains two python classes, one for the leader and another for the follower. Developers can inherit these two classes to modify the communication and control scheme. By starting multiple ROS nodes, multiple UAV controllers are simulated.

Fig. 8 shows the formation of 9 UAVs implemented in the simplified simulator and Fig. 9 shows the formation implemented in Gazebo. Three configurations from left to right are cube, pyramid and triangle. The videos of the formation reconfiguration can be seen at <https://www.yuque.com/xtdrone/demo/uavFormation>.

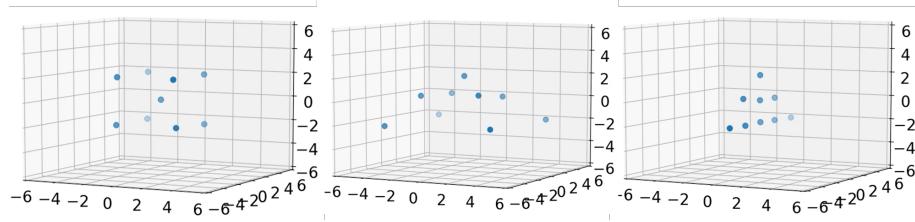


Fig. 8. Formation of 9 UAVs implemented in the simplified simulator

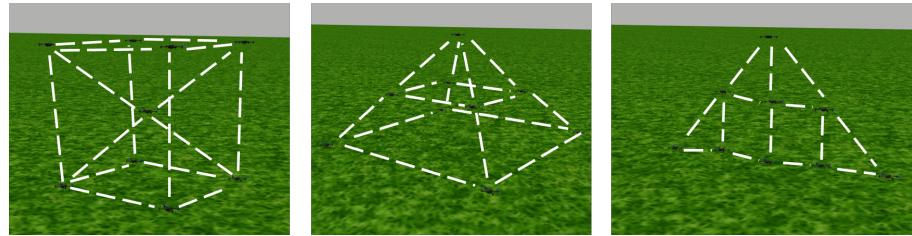


Fig. 9. Formation of 9 UAVs implemented in Gazebo

To validate the consensus control algorithm, a flight log is recorded during configuration from origin to ‘cube’, and the position response curves are shown in Fig. 10. Although some overshooting can be seen on the response curves, all the UAVs reaches stability.

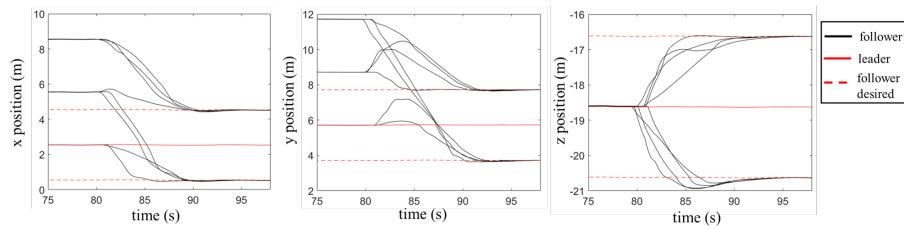


Fig. 10. Position response curves

6 Conclusion and Future Work

A hierarchical multi-UAV simulation platform is developed. Six layers: communication, simulator, low-level control, high-level control, coordination and human interaction layers are designed. Two demos are presented to demonstrate how the platform works. The first one is a cooperative search mission. The low-level controller of each UAV tracks the desired trajectory planned by ground control station and the high-level controller of each UAV detects the target objects. The other one is a distributed UAV formation including consensus control, task assignment and obstacle avoidance. The simulation validation contains 9 UAVs simulation in the simplified simulator and in Gazebo, which validates the consensus control algorithm.

The platform is all open source, and is being developed continually. Now it only supports multi-rotor UAVs and the support to fixed-wing UAVs and vertical take-off and landing UAVs is coming. Besides, another simulator, AirSim, is being integrated into the platform. Furthermore, a multi-UAV competition is planned to be held based on the simulation platform.

Acknowledgement

This work was supported by the National Natural Science Foundation of China under Grants 61973309 and 61801494.

References

1. Maza, I., Kondak, K., Bernard, M., Ollero, A.: Multi-uav cooperation and control for load transportation and deployment. In: Selected papers from the 2nd International Symposium on UAVs, Reno, Nevada, USA June 8–10, 2009, pp. 417–449. Springer (2009)
2. He, L., Bai, P., Liang, X., Zhang, J., Wang, W.: Feedback formation control of uav swarm with multiple implicit leaders. *Aerospace Science and Technology* **72**, 327–334 (2018)
3. Liu, Z., Wang, X., Li, J., Cong, Y., Zhao, S.: A distributed and modularised coordination framework for mission oriented fixed-wing uav swarms. In: 2019 Chinese Control And Decision Conference (CCDC), pp. 3687–3692. IEEE (2019)
4. Chung, S.J., Paranjape, A.A., Dames, P., Shen, S., Kumar, V.: A survey on aerial swarm robotics. *IEEE Transactions on Robotics* **34**(4), 837–855 (2018)
5. Madey, G.R., Blake, M.B., Poellabauer, C., Lu, H., McCune, R.R., Wei, Y.: Applying dddas principles to command, control and mission planning for uav swarms. *Procedia Computer Science* **9**, 1177–1186 (2012)
6. Wei, Y., Blake, M.B., Madey, G.R.: An operation-time simulation framework for uav swarm configuration and mission planning. *Procedia Computer Science* **18**, 1949–1958 (2013)
7. Meng, W., Hu, Y., Lin, J., Lin, F., Teo, R.: Ros+ unity: An efficient high-fidelity 3d multi-uav navigation and control simulator in gps-denied environments. In: IECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society, pp. 002,562–002,567. IEEE (2015)

8. Sanchez-Lopez, J.L., Pestana, J., De La Puente, P., Campoy, P.: A reliable open-source system architecture for the fast designing and prototyping of autonomous multi-uav systems: Simulation and experimentation. *Journal of Intelligent & Robotic Systems* **84**(1-4), 779–797 (2016)
9. Rodriguez-Fernandez, V., Menéndez, H.D., Camacho, D.: Design and development of a lightweight multi-uav simulator. In: 2015 IEEE 2nd International Conference on Cybernetics (CYBCONF), pp. 255–260. IEEE (2015)
10. Garcia, R., Barnes, L.: Multi-uav simulator utilizing x-plane. In: Selected papers from the 2nd International Symposium on UAVs, Reno, Nevada, USA June 8–10, 2009, pp. 393–406. Springer (2009)
11. Xiao, K., Tan, S., Wang, G., An, X., Wang, X., Wang, X.: Xtdrone: A customizable multi-rotor uavs simulation platform. arXiv preprint arXiv:2003.09700 (2020)
12. Liu, Z., Wang, X., Shen, L., Zhao, S., Cong, Y., Li, J., Yin, D., Jia, S., Xiang, X.: Mission oriented miniature fixed-wing uav swarms: A multi-layered and distributed architecture. arXiv preprint arXiv:1912.06285 (2019)
13. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
14. Ren, W., Beard, R.W.: Distributed consensus in multi-vehicle cooperative control. Springer (2008)
15. Ding, Y., Wang, X., Cong, Y., Li, H.: Scalability analysis of algebraic graph-based multi-uavs formation control. *IEEE Access* **7**, 129,719–129,733 (2019)
16. Zhu, H., Zhou, M., Alkins, R.: Group role assignment via a kuhn–munkres algorithm-based solution. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* **42**(3), 739–750 (2012)
17. Xu, H., Wang, L., Zhang, Y., Qiu, K., Shen, S.: Decentralized visual-inertial-uwb fusion for relative state estimation of aerial swarm. arXiv preprint arXiv:2003.05138 (2020)
18. Saska, M., Baca, T., Thomas, J., Chudoba, J., Preucil, L., Krajnik, T., Faigl, J., Loianno, G., Kumar, V.: System for deployment of groups of unmanned micro aerial vehicles in gps-denied environments using onboard visual relative localization. *Autonomous Robots* **41**(4), 919–944 (2017)
19. Guo, K., Qiu, Z., Meng, W., Xie, L., Teo, R.: Ultra-wideband based cooperative relative localization algorithm and experiments for multiple unmanned aerial vehicles in gps denied environments. *International Journal of Micro Air Vehicles* **9**(3), 169–186 (2017)