A Cost-Efficient Platform Design for Distributed UAV Swarm Research

Zhongxuan Cai
Institute for Quantum Information & State Key Laboratory of High Performance Computing, College of Computer Science and Technology, National University of Defense Technology, Changsha, 410073, China
caizhongxuan@nudt.edu.cn

Xuefeng Chang
Institute for Quantum Information & State Key Laboratory of High Performance Computing, College of Computer Science and Technology, National University of Defense Technology, Changsha, 410073, China
changxuefeng15@nudt.edu.cn

Minglong Li*
Institute for Quantum Information & State Key Laboratory of High Performance Computing, College of Computer Science and Technology, National University of Defense Technology, Changsha, 410073, China
liminglong10@nudt.edu.cn

ABSTRACT

Unmanned Aerial Vehicles (UAVs) have been attracting more and more attention in research and education. Specifically, Swarm intelligence is a promising future technology of UAVs and the frontier of multi-agent system research. It has the characteristics of low individual cost, strong system flexibility and robustness, and has great potential in many tasks. However, due to the constraints of research conditions and cost, most of the current researches on large-scale swarm UAVs are carried out in the simulation environment. Building a low-cost open-source software and hardware platform for swarm UAVs is an important basis for promoting researches on swarm UAVs and multi-agent systems. In this paper, we propose a design of a UAV platform with common cost-efficient hardware and a rich open-source software ecosystem, and provide a software solution for swarm robots based on the open-source robot operating system ROS. These software packages support the rapid programming development of swarm behaviors and different communication topology. Experiments have been conducted for typical UAV tasks like flocking and formation, indicating the effectiveness of the proposed platform.

CCS CONCEPTS

• **Computer systems organization** → Embedded and cyber-physical systems; • **Applied computing** → Computers in other domains.

KEYWORDS

Unmanned aerial vehicles, UAV platform for research, UAV hardware and software

ACM Reference Format:
Zhongxuan Cai, Xuefeng Chang, and Minglong Li*. 2021. A Cost-Efficient Platform Design for Distributed UAV Swarm Research. In *2021 3rd International Conference on Advanced Information Science and System (AISS 2021) (AISS 2021), November 26–28, 2021, Sanya, China.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3503047.3503070

1 INTRODUCTION

Unmanned aerial vehicles (UAVs) have been rapidly improved and applied in many areas, such as precision agriculture, remote sensing, etc. [1]. Apart from the actual usage in practice, UAVs are also becoming more and more important in research and education [2]. Multi UAV systems and UAV swarms are also being developed and tend to be a valuable research area [3]. However, even if the UAV hardware and software have developed a lot to reduce cost, when researching on multi UAV systems or swarm UAVs, the hardware cost and software programing work are still tedious. Therefore many works on UAV research can only be verified in the simulation world [3], which limits the progress of relevant research.

There are many commercial UAV products, especially multi-rotor UAVs are not as costly as before [2]. The major challenge to building cost-efficient UAV platforms for UAV swarms includes two aspects. On the one hand, current products tend to be designed for special purposes, e.g. drone photography [4]. Their software is usually specifically designed, difficult to modify and closed source. Although some products provide software development kits (SDKs), their design mainly focuses on programming to control single UAVs, regardless of the requirement of swarm UAVs, e.g. computation and communication. On the other hand, even a group of suitable and programmable hardware platforms are given, the cost for software development is still high, e.g. developing basic movement, communication, and adapting to different software platforms [5].

In this paper, we propose a cost-efficient UAV platform for distributed UAV experimental research. The platform uses low-cost off-the-shelf common commercial products, including a ROS-enabled UAV product, e.g. Parrot Bebop 2, and the onboard computational resources, e.g. the famous Raspberry Pi 3b. They can be easily connected through wired interfaces like micro USB. The onboard Raspberry Pi 3b supports Ubuntu Mate and ROS, which is the de facto standard of robot development framework in the research community, with a very large and rich software ecosystem. Thereby we separate the swarm relevant software module and the single UAV relevant modules as two layers, and can develop a common solution for distributed UAV swarms on the top layer of the onboard Raspberry Pi.

The major contribution of this paper includes:

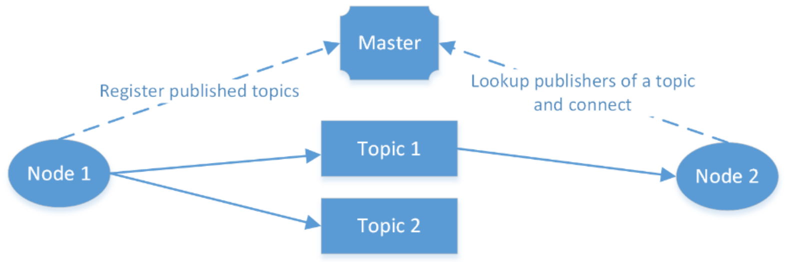
Master
Register published topics
Lookup publishers of a topic and connect
Node 1
Topic 1
Topic 2
Node 2

Figure 1: The ROS communication mechanism based on topics

1. We propose a cost-efficient platform design for distributed UAV swarm research, where a two-layer architecture is constructed for commercial UAV products and the onboard computer.

2. Despite the off-the-shelf SDKs provided by the UAV products, common requirements like different communication mechanisms for distributed computing can be realized on board in a unified framework.

3. We demonstrate the design with commercial products Parrot bebop 2 UAV and Raspberry Pi 3b. Experiments of flocking and formation with different communication topologies are conducted to verify the effectiveness of the proposed platform design.

2 BACKGROUND AND RELATED WORK

In recent years, UAVs with aerial photography as the representative application have been developed more and more mature. At the same time, in order to support the relevant research, many manufacturers provide SDKs to enhance the programmability of UAVs. For example, computers and UAVs can be connected through wireless hotspots or wired connections, so as to send the UAV control command or receive the image, pose and other sensor information returned by the UAV. With this information, researchers can accomplish a wide variety of studies, like target-tracking [6], obstacle avoidance [7], communication relay [8], etc. With the programmability, the commercial off-the-shelf UAVs have also been widely used in education [2]. Meanwhile, cooperating multiple robots to accomplish more intelligent tasks has been attractive in research [9].

When it comes to swarm UAVs, the work is much less mature. Some work focus on the coordination methods of agents [3], others focus on the communication mechanism and swarm development of the software [5, 10]. Due to the cost of actual UAV platforms, many works are only designed and verified in simulation, where many practical problems are omitted or intrinsically simplified. For example, the communication of UAVs and the visualization of their data in simulation may be function calls, inter-process communication, instead of wireless communication over real channels, which can be much more unreliable. The system time of each UAV is also intrinsically synchronized as their processes actually run on the same host. And the launch of a large number of UAVs can be easily done by writing some scripts.

However, in actual UAV swarm experiments, things are quite different. All the above problems require some extra software components to solve, which further leads to the problem of efficiently developing and coordinating these components. Robot Operating System [11, 12] is the de facto robot software developing middleware, which provides a flexible topic-based communication mechanism, as shown in Fig. 1. It mainly works on Linux operating systems, especially Ubuntu. With ROS, different robot function modules can interact by publishing to or subscribing from a topic, regardless of the underlying communication details. With this modular characteristic, ROS has developed a rich software ecosystem, consisting of a wide variety of software packages for perception, decision, control, etc. However, since ROS works on a centralized master-slave architecture, it tends to be insufficient for UAV swarms when communication is not reliable.

This paper considers the problem of the overall design of platforms for UAV swarm research, including the cost-efficient hardware and software components. We will point out and propose effective methods for the above-mentioned problems outside of simulation, according to our practical experiences in flocking and formation tasks, and in both centralized and decentralized manner.

3 THE PLATFORM DESIGN OF DISTRIBUTED UAV SWARM RESEARCH

In this section, we first describe the platform architecture, including the hardware and software in the swarm, under centralized and decentralized scenarios. Next, we analyze and discuss some issues encountered in practical experiments other than simulation, as well as the effective methods in our swarm flocking and formation applications.

3.1 The Swarm Hardware and System Setting

The overall swarm is consists of several UAVs and a ground station, i.e. a laptop for control and visualization of the whole experimental procedure. Since a laptop is basic research equipment for decades, the major concern for swarm UAV research would be some cost-efficient UAV platforms. To facilitate the future flexible and efficient software development for various research targets, the UAV is desired to support ROS and have an onboard computer that runs ROS and Linux, so as to make use of the rich software ecosystem of the ROS community and the rich system management tools provided by Linux community. In our practice, we combine the Parrot Bebop 2 UAV and a Raspberry Pi 3b (with battery) to form a single UAV unit in the swarm. They can be directly connected through a micro USB to USB cable. The Raspberry Pi requires a micro SD card as storage and supports installing Ubuntu Mate system, a system for ARM CPUs used by many devices including Raspberry Pi. Then ROS and the official package for Bebop 2 can be installed onboard. By far

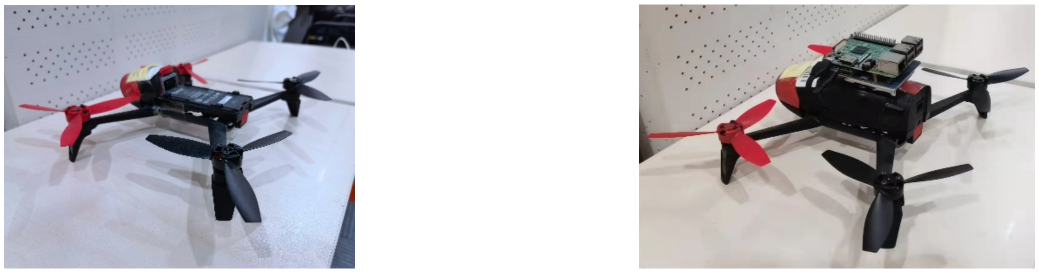

Figure 2: The UAV swarm unit made up of Bebop 2 and Raspberry Pi.

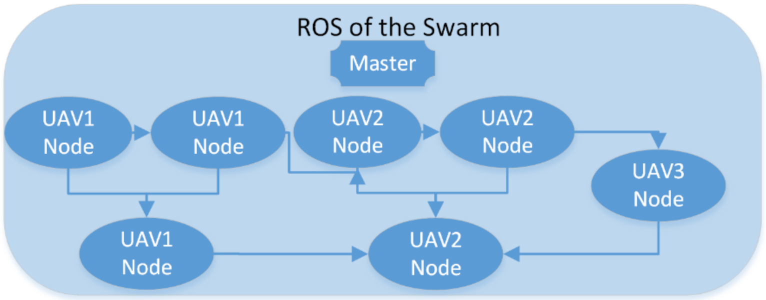

Figure 3: The architecture of the centralized ROS-based design

any program on the Raspberry Pi may receive sensor information from the UAV by subscribing to certain topics, and send control commands to the UAV by publishing to certain topics.

3.2 The Centralized ROS based Design for Rapid Development

After setting up each UAV and the ground station, the next problem is how to organize their software, so as to facilitate their cooperation to form swarm behavior. One direct way is to use ROS directly, i.e. all units in the swarm (the Raspberry Pi and the ground station) are set to one ROS environment, with the ground station as the ROS master. Under this architecture, any nodes (programs) on each board of UAV and the ground station can directly communicate through the topics in ROS. Therefore the cooperation of each UAV to generate swarm behavior and the visualization user interface on the ground station can be directly implemented with the ROS communication mechanism. The advantage of this architecture is the convenience for rapid development with the help of the ROS ecosystem and a fully connected system view. Therefore it is an efficient way in UAV swarm practice to support early quick verification and small-scale experiments.

3.3 The Distributed ROS-based Design for Large Scales and Swarms

However, the architecture in the previous subsection is far from sufficiently supporting further UAV swarm studies, because of the centralized communication topology. In large-scale tasks and large-number UAV swarms, the communication is not reliable. Since ROS requires a centralized master and TCP transmission, therefore the fully ROS-based design would be easy to fail. The programs on UAVs may not be able to communicate or even discover each other with limited communication bandwidth. For example, in the practical experiments of UAV swarm flocking [3], when the number of UAVs is 10, a fully ROS-based design will encounter serious communication problems. Even doing remote login from the ground station to a Raspberry Pi can have very high latency or failure, not to mention the complex swarm coordination and the ground station visualization.

In this subsection, we further adapt a hybrid ROS-based approach to our cost-efficient platform [13]. The basic idea is to run multiple individual ROS systems on each platform, i.e. the Raspberry Pi and the ground station. This design maintains the advantage of the rich ROS software ecosystem, including not only many off-the-shelf algorithm packages, but also a wide variety of drivers of sensors, actuators and UAV hardware. Since each ROS system runs on a single host, the network congestion and single point of failure in swarm experiments can be avoided. But to coordinate the swarm, there should be another mechanism for inter-platform communication. There are many communication mechanisms more robust than ROS when network resource is quite limited, e.g. a simple but efficient way is frequently broadcast by UDP. Compared with the fully ROS-based design, this architecture requires extra engineering on the communication bridge between different platforms. This part may need to be developed according to the actual requirements of UAV swarms, or maybe using some existing swarm development framework in the literature, like GSDF [5].

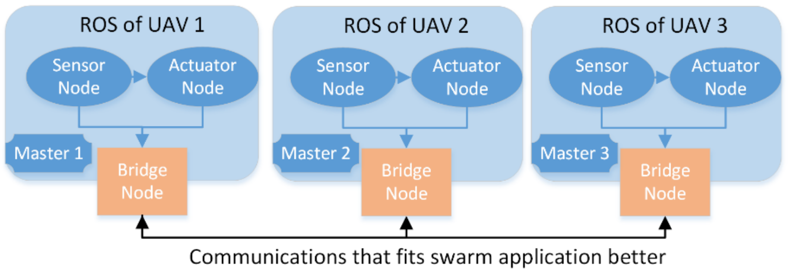

Figure 4: The architecture of the distributed ROS-based design

3.4 Practical Engineering Problems and Solutions

In practical UAV swarm experiments, there are problems apart from the task design. This subsection introduces some major problems we are encountered with the proposed problem in UAV swarm flocking and formation experiments.

These problems include:

1. Time synchronization. Although Raspberry PI 3b is a powerful and well-known computational device, one of its drawbacks is that its system time will be reset after each reboot. This may cause problems in swarm cooperation and visualization, since in many cases ROS relies on the message timestamp to properly work. Luckily, the operating system we use is Ubuntu Mate, where many off-the-shelf tools in the Linux community can be used. We use the *chrony* and *ntp* package to synchronize all drones with the ground stations.

2. Swarm application startup. When the number of UAVs is small, we can remote control, e.g. *ssh* each board to launch the required nodes on each UAV. However, when the number of UAVs is large, doing so is time-consuming and tedious. ROS provides a remote launch mechanism, by which ROS nodes on different hosts can be launched in one single host by one configuration file. But this requires a fully connected network and consumes bandwidth resources. In our empirical study, the best way is to automatically launch nodes individually on each UAV, by adding commands in the startup shell of the Ubuntu Mate operating system.

3. UAV USB connection startup. By default the Parrot Bebop 2 does not enable its USB connection, therefore the Raspberry Pi cannot establish the connection. The official instruction is to continuously press the power button several times. This method is also problematic when the UAV number is large. It is not only time-consuming, but there may also frequently occur flawed press that fails to connect some UAVs. This connection problem can be solved once for all. After the USB is connected, we can telnet to the UAV firmware and add the press-button scripts to its startup scripts. In this way, we can both eliminate manual pressing and ensure that the connection is always successful.

4. Positioning. The Parrot Bebop 2 supports both GPS and visual-inertial odometer. When the expected distance between UAVs is not large, e.g. 5 meters in our practice, the noise of GPS is unacceptable. The odometer works very well, but since the origin of the odometer for each UAV is itself, the actual position of each UAV should be further corrected by adding its initial position.

If the same platform were built as our proposal, the above problems can be efficiently solved by our proposal. If not, these problems should also be carefully considered when selecting software and hardware types to build the swarm UAV platform.

4 EXPERIMENTS

With our platform, we verify the proposed whole system in UAV swarm flocking and formation. Flocking is a phenomenon in many natural systems, e.g. birds, ants, Immune cells and so on. Each unit can only interact with its limited neighbors but the whole swarm emerges. The method we use is in [3], where each UAV follows a virtual agent under the following control:

$$\mathbf{u}_{\mathrm{vi}} = f^{g}_{\mathrm{vi}}(\mathbf{q}_v) + f^{d}_{\mathrm{vi}}(\mathbf{q}_v, \mathbf{p}_v) + f^{\gamma}_{\mathrm{vi}}(\mathbf{q}_{\mathrm{vi}}, \mathbf{p}_{\mathrm{vi}}, \mathbf{q}_r, \mathbf{p}_r)$$

The detailed function of the three parts is shown in Table 1, where **p** denotes position and **q** denotes velocity.

The flocking experimental results are shown in Fig. 5, where the UAV swarm using our design successfully performs the flocking behavior to generate a flock where different UAVs gradually keep the same desired distance with only local interaction.

Besides flocking, we also conduct formation experiments practically. Fig. 6 is the snapshot of the UAVs performing a formation switch from a horizontal line to a vertical line.

5 LIMITATION AND ETHICAL STATEMENT

This work is an empirical study on the cost-efficient UAV platform design for UAV swarm research. One major limitation of the system is the limited flying time restricted by the battery, which is about 15 to 20 minutes typically. Besides, due to the intrinsic high time and other costs of large-scale UAV physical experiments, the work is empirically verified on fixed applications, flocking and formation.

In practice, UAV swarm experiments may face more problems out of scope in this paper, and face much higher risks than single UAV or other robot experiments, including but not limited to personal injury, property loss and legal risk of the flight. This paper only proposes a low-cost platform design method and limited empirical physical experiment experience. Experimenters need to be fully prepared to deal with any possible emergencies or system failures, and bear their own risks when carrying out practical experiments.

Table 1: The Three Terms of the Flocking Algorithm

Term and equation	Function
$f_{vi}^{g} = \sum_{v_j \in N_{vi}} \phi_{\alpha}(\lVert \mathbf{q}_{vj} - \mathbf{q}_{vi} \rVert_{\sigma}) \mathbf{n}_{vivj}$	Keeping desired distance between neighboring UAVs
$f_{vi}^{d} = \sum_{v_j \in N_{vi}} a_{vivj}(\mathbf{q}_{v})(\mathbf{p}_{vj} - \mathbf{p}_{vi})$	Aligning the speed of neighboring UAVs
$f_{vi}^{\gamma} = -c_1(\mathbf{q}_{vi} - \mathbf{q}_r) - c_2(\mathbf{p}_{vi} - \mathbf{p}_r)$	The flocking target position

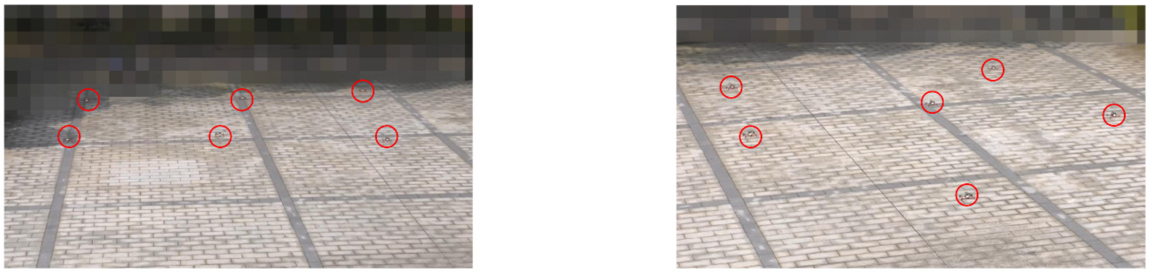
Figure 5: The flying snapshots before and after the flocking of our UAV swarm

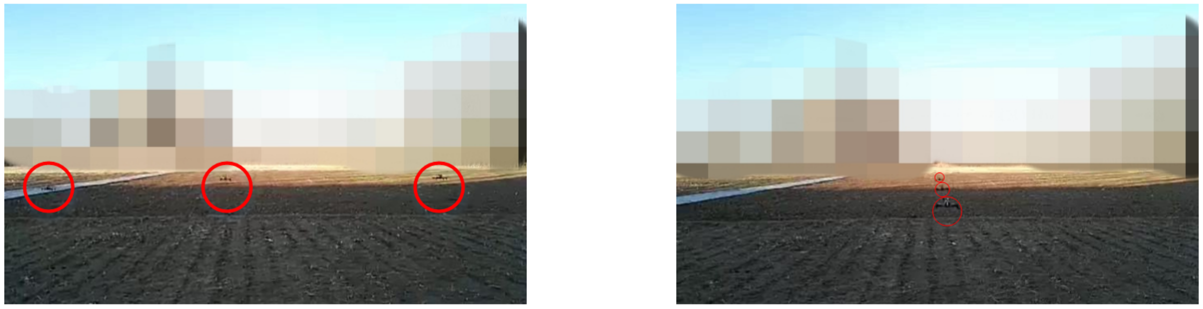
Figure 6: The horizontal and vertical line formation switch by our UAV swarm

6 CONCLUSION

This paper proposes the design of a cost-efficient platform design for UAV swarm research, which not only makes a proper combination of off-the-shelf common commercial products, but also leverages the advantage of the ROS software ecosystem and Linux community. Some crucial problems in practical experiments with swarm UAVs are pointed out, the corresponding methods or tools are analyzed and efficient solutions are proposed. Experiments with two application domains verify the effectiveness of the proposed design. Finally, the limitation and the potential risk of conducting experiments with UAV swarms are discussed. Future work is to consider more complex fixed-wing UAVs.

ACKNOWLEDGMENTS

This work was partially supported by the National Key Research and Development Program of China (No. 2017YF-B1301104), the National Natural Science Foundation of China (No. 91948303, 91648204, 61803375 and 62106278), and the National Science and Technology Major Project.

REFERENCES

[1] Coombes, M. , Chen, W. H. , & Liu, C. . Flight Testing Boustrophedon Coverage Path Planning for Fixed Wing UAVs in Wind.*International Conference on Robotics and Automation.* Department of Automotive and Aeronautical Engineering, Loughborough University, Loughborough, LE11 3TU, UK.

[2] Hong, J. H. , Shin, H. S. , & Tsourdos, A. . (2019). A design of a short course with cots uav system for higher education students. IFAC-PapersOnLine, 52(12), 466-471.

[3] Cai, Z. , Chang, X. , Wang, Y. , Yi, X. , & Yang, X. J. . (2017). Distributed control for flocking and group maneuvering of nonholonomic agents.*Computer Animation & Virtual Worlds,28*(3-4), -.

[4] Chen, W. Y. , & Hu, J. K. . Research on Drone's Aerial Photography Aided Learning System Based on Deep Learning. *IEEE International Conference on Consumer Electronics- Taiwan.* Chinese Culture University Department of Mass Communication Taiwan R.O.C.

[5] Chang X., Cai Z., Wang Y., Yi X., Xiao N. (2017) GSDF: A Generic Development Framework for Swarm Robotics. In: Huang Y., Wu H., Liu H., Yin Z. (eds) Intelligent Robotics and Applications. ICIRA 2017. Lecture Notes in Computer Science, vol 10462. Springer, Cham. https://doi.org/10.1007/978-3-319-65289-4_62

[6] F. Lin, C. Fu, Y. He, F. Guo and Q. Tang, "BiCF: Learning Bidirectional Incongruity-Aware Correlation Filter for Efficient UAV Object Tracking," 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 2365-2371, doi: 10.1109/ICRA40945.2020.9196530.

[7] M. Odelga, P. Stegagno and H. H. Bülthoff, "Obstacle detection, tracking and avoidance for a teleoperated UAV," 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 2984-2990, doi: 10.1109/ICRA.2016.7487464.

[8] W. Huang, Y. Wang and X. Yi, "A deep reinforcement learning approach to preserve connectivity for multi-robot systems," *2017 10th International Congress*

on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), 2017, pp. 1-7, doi: 10.1109/CISP-BMEI.2017.8302157.

[9] Geng, M., Xu, K., Zhou, X., Ding, B., Wang, H., Zhang, L. (2019).Learning to cooperate via an attention-based communication neural network in decentralized multi-robot exploration. Entropy, 21(3)

[10] C. Pinciroli and G. Beltrame, "Buzz: A Programming Language for Robot Swarms," in *IEEE Software*, vol. 33, no. 4, pp. 97-100, July-Aug. 2016, doi: 10.1109/MS.2016.95.

[11] Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.:Ros: an open-source robot operating system. In: International Conference on Robotics and Automation, workshop on open source software, vol. 3 (2009)

[12] Liu, Z. , Mao, X. , & Yang, S. . (2018). Autorobot: a multi-agent software framework for autonomous robots. *IEICE Transactions on Information and Systems, 101*(7), 1880-1893.

[13] T. H. Chung, M. R. Clement, M. A. Day, K. D. Jones, D. Davis and M. Jones, "Live-fly, large-scale field experimentation for large numbers of fixed-wing UAVs," 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 1255-1262, doi: 10.1109/ICRA.2016.7487257.