Multi-UAV Specification and Control with a Single Pilot-in-the-Loop

Patricio Moreno*[1], Santiago Esteva[1], Ignacio Mas[3,4], and Juan I. Giribet[1,2,3]

[1]GPSIC - Facultad de Ingeniería, Universidad de Buenos Aires, Argentina
[2]Instituto Argentino de Matemática "Alberto Calderón" (IAM)
[3]Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina
[4]Instituto Tecnológico de Buenos Aires, Argentina

Abstract

This work presents a multi-unmanned aerial vehicle formation implementing a trajectory-following controller based on the cluster-space robot coordination method. The controller is augmented with a feed-forward input from a control station operator. This teleoperation input is generated by means of a remote control, as a simple way of modifying the trajectory or taking over control of the formation during flight. The cluster-space formulation presents a simple specification of the system's motion and, in this work, the operator benefits from this capability to easily evade obstacles by means of controlling the cluster parameters in real time. The proposed augmented controller is tested in a simulated environment first, and then deployed for outdoor field experiments. Results are shown in different scenarios using a cluster of three autonomous unmanned aerial vehicles.

1 Introduction

Unmanned autonomous systems, in general, is a topic of interest that has been growing steadily for some time. This raises from the diversity of applications in which these systems can be used. Examples of this are search and rescue missions [1], inspection of hazardous environments, goods delivery or object transportation, military and surveillance purposes, among others. Unmanned aerial vehicles (UAVs) are of special interest because of advances in technology, that have reduced cost and boosted the capabilities of all UAVs, particularly multicopters. This has raised the interest in formation control of multi-agent systems within academic and industry communities.

The theory used to design the control laws for these architectures feeds from different fields, such as game theory [2], biology [3, 4] or classic manipulator kinematic chains [5], [6]. Techniques derived from these studies include potential fields [7], behavioral primitives [8], swarm-like structures [9, 10], and leader-follower configurations [11].

All these techniques control a multi-agent system to operate in a cooperative fashion. Working with multiple unmanned aerial vehicles, the particular formation used in this work involves spatial constraints and impose physical limitations, such as communications range. Oh et al. [12] gave a detailed review on formation control and Yanmaz et al. [13] analyzed the communication network aspects of a formation.

In [5] Mas et. al. presented a cluster-space formulation for the coordinated control of a group of robots. The goal of the cluster-space approach is to promote the simple specification and monitoring of the motion of a multirobot mobile system, exploring a specific approach for formation control applications. This method considers the multirobot system as a single entity, or cluster, and desired motions are specified with respect to cluster attributes, such as position, orientation, and geometry. These attributes are the state variables that form the cluster space of the system. The method is flexible in the sense that these variables can be selected in different ways, favoring specific tasks or alternative implementations such as centralized or distributed control architectures [14]. Previous works showed results, both simulated and in real scenarios, for unmanned ground vehicles (UGVs)[5, 14] and autonomous surface vessels (ASVs) [15, 16], among others.

In this work we introduce a cluster space controller, where a feed-forward component is added to modify the trajectory in-flight. This formulation allows to naturally modify the position and geometric properties of the cluster in a way that enables a simple formation tele-operation by a single human pilot, using an intuitive remote control interface to command the motion of the formation. An alternative to this approach would be the specification of the trajectory of each vehicle or the relative position of each vehicle with respect to a neighbour.

To illustrate the benefits of such an architecture, in a task where multiple UAVs cooperatively transport a load, as in [17], if a tele-operated group of vehicles needs to pass through a narrow passage while keeping the load distribution constant, it may be of interest to momentarily modify the distance between vehicles without changing their spacial relative configuration. A single specification change such as "change the formation size" that can be commanded by an operator keeps the operation simple, regardless of the under-

*Email address: pamoreno@fi.uba.ar

lying complexity of the individual vehicles' motions.

Another benefit of a pilot-in-the-loop control arises when a multi-agent system is used for automated inspection. For example, electric power distribution lines may be located in areas of difficult access and unmanned vehicles can be used for inspection tasks [18]. The tele-operator may need an additional detailed view of a portion of a tower or cable, modifying a pre-loaded trajectory in-flight. Oil pipeline inspection [19] or civil engineering projects such as bridges or skyscrapers may also benefit from this approach.

This work is organized as follows. Section 2 shows the formation definition while the controller is described in section 3. The results of computer simulation and our experimental testbed are shown in section 4. Finally, section 5 draws the conclusions.

2 Cluster-space formulation

Cluster-space control [20] represents the state of a system as an articulated kinematic mechanism. The cluster is defined using variables which fully represent the pose and geometric structure of the formation. First, a cluster frame $\{C\}$ to represent the formation pose is defined. Then, each robot's pose, $\mathbf{r}_i \in \mathbb{R}^{m_i \times 1}$, is referenced to the $\{C\}$ frame. It is usually desired to define $\{C\}$ in a physically meaningful way, such as at the formation barycenter and oriented towards a particular vehicle. Additional cluster variables capture the formation shape and orientation, fully specifying the total number of degrees of freedom of the group. The formation motion is commonly defined using the cluster-space variables. Because of this, a formal set of kinematic transformations relating cluster-space variables and robot-space variables is needed. A cluster-space controller computes the compensation actions needed for the cluster and, using the defined kinematic transformations, converts the cluster compensation actions into robot compensation actions.

Consider an n-robot system, a cluster, where each of the robots has the same m degrees of freedom (although this is not necessary[1]). Let $\mathbf{r} \in \mathbb{R}^{mn \times 1}$ be a state vector comprised of the n robot poses, and $\mathbf{c} \in \mathbb{R}^{mn \times 1}$ a state vector corresponding to the cluster variables. These states are related through the following forward and inverse position kinematics transforms:

$$\begin{aligned}
\mathbf{c} &= \text{FORWARD\_KINEMATICS}(\mathbf{r}) && (1)\\
&= [\text{fwd}_1(r_1, \dots, r_{mn}), \dots, \text{fwd}_{mn}(r_1, \dots, r_{mn})]^\intercal,\\
\mathbf{r} &= \text{INVERSE\_KINEMATICS}(\mathbf{c}) && (2)\\
&= [\text{inv}_1(c_1, \dots, c_{mn}), \dots, \text{inv}_{mn}(c_1, \dots, c_{mn})]^\intercal,
\end{aligned}$$

where $\text{fwd}_k(r_1, \dots, r_{mn})$ is the forward position kinematic equation that relates the kth cluster parameter with the robot poses, and $\text{inv}_k(c_1, \dots, c_{mn})$ is the inverse position kinematic equation that related the kth robot state parameter with the cluster parameters.

Now, let $\mathbf{J}(\mathbf{r})$ be the jacobian matrix obtained from Equation 1, and $\mathbf{J}^{-1}(\mathbf{c})$, the jacobian matrix obtained from Equation 2, the mapping between the velocities are $\dot{\mathbf{c}} = \mathbf{J}(\mathbf{r})\dot{\mathbf{r}}$ and $\dot{\mathbf{r}} = \mathbf{J}^{-1}(\mathbf{c})\dot{\mathbf{c}}$, respectively.

Using generic kinematic transformations it is possible to envision a diagram of a system being controlled using the cluster-space formulation. Such an architecture is shown in Figure 1.

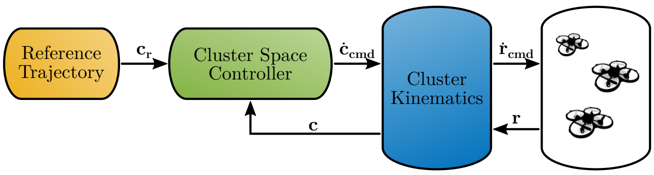


Figure 1: Control architecture for the cluster-space control method.

2.1 *Three-UAV Cluster Space definition*

The three-robot cluster state variables can be defined with the cluster reference frame located at the barycenter of the robots and the remaining variables describe a triangle with side lengths p and q and the necessary angles to articulate it and rotate it. Figure 2 shows all the parameters for the cluster of 3 UAVs. The equations for the forward position kinematics that define the cluster space are the following:

$$x_c = \frac{x_1 + x_2 + x_3}{3}, \tag{3}$$

$$y_c = \frac{y_1 + y_2 + y_3}{3}, \tag{4}$$

$$z_c = \frac{z_1 + z_2 + z_3}{3}, \tag{5}$$

$$\theta_c = -\arctan\left(\frac{2x_1 - x_2 - x_3}{2y_1 - y_2 - y_3}\right), \tag{6}$$

$$\rho_c = -\arctan\left(\frac{z_1 - z_c}{\sqrt{(x_1 - x_c)^2 + (y_1 - y_c)^2}}\right), \tag{7}$$

$$\gamma_c = -\arctan\left(\frac{z_2 - z_3}{|x_2 - x_3|}\right), \tag{8}$$

$$p = \tfrac{1}{2}\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}, \tag{9}$$

$$q = \tfrac{1}{2}\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2 + (z_1 - z_3)^2}, \tag{10}$$

$$\beta = \arctan\left(\frac{(x_3 - x_1)\sin\alpha - (y_1 - y_3)\cos\alpha}{(x_3 - x_1)\cos\alpha + (y_1 - y_3)\sin\alpha}\right), \tag{11}$$

where $\alpha = \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right)$. Also, each UAV heading angle is a cluster parameter by itself, defined as the heading offset with respect to the cluster yaw angle. They have been omitted in the formulation above for simplicity.

Considering the heading angle of the UAVs, there are 12

[1] Considering an n-robot system where each robot has $m_i, i = 1, \dots, n$ degrees of freedom, then the state vector $\mathbf{r}$ has $\sum_{i=1}^{n} m_i$ components.

cluster state variables for a formation of 3 stabilized UAVs, each with 4 degrees of freedom.

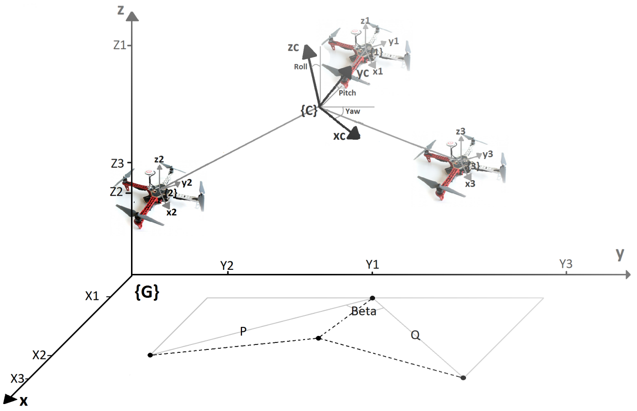


Figure 2: Cluster parameters definition for a formation of three UAVs.

3 Cluster-Space Controller

As shown in Figure 1, a classic PID controller was added for trajectory tracking. This controller receives the cluster state errors (or the cluster state reference and cluster state pose and computes the error) and generates a cluster state velocity control signal, using different proportional, integral and derivative gains for each cluster state variable. The PID output control signal is then multiplied by the inverse jacobian matrix to generate the compensation signal to be applied to each UAV.

To add the remote control operation, the addition of a feed-forward controller is proposed. This controller adds an external signal to the system. The external signal is a velocity command that can be readily sent to the cluster formation. It also modifies the trajectory to take into account the commanded velocity and integrates its value over time to modify the cluster space reference trajectory accordingly. Figure 3 shows a complete block diagram of the implemented controller.

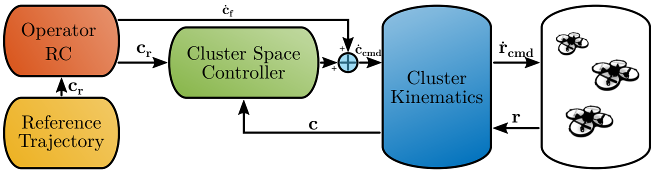


Figure 3: Implemented control scheme for the cluster-space control.

4 Results

The proposed system was first validated using a simulation environment. This environment consists of a computer running the XUbuntu 16.04 operating system with ROS Kinetic, the Robot Operating System, and Gazebo 7, a robot simulator. The multicopters are simulated using the firmware of the PX4 autopilot—version 1.5.1—, their Gazebo plugins, and a model of the IRIS drone from 3D-Robotics.

The experimental testbed consists of three commercially available UAVs built with DJI F450 frames, the Pixhawk 1 autopilot (FCU) from 3DR with the PX4 flight stack, and one Raspberry Pi 3B (RPi) for a pair of UAVs, the remaining one uses a 915 MHz link. Figure 4 shows a picture of one the UAVs with an RPi on top of the FCU. The communications network was build using a WiFi router, connecting all computer to it.

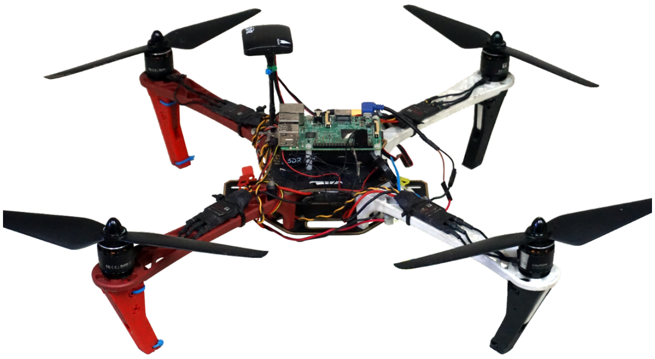


Figure 4: UAV with onboard computer and wifi link used for field experiments

The interface with the autopilot was through a mavros ROS node. The trajectory generator, the cluster kinematic equations and the controller were developed as ROS nodes, in the python programming language. The operator remote control was a gaming joystick with 14 buttons and 2 analog sticks.

The experiment consisted on a triangular-shaped formation having a predefined trajectory that would make one or more of the UAVs collide with objects placed in the environment. For this situation, two scenarios were proposed to overcome the conflict:

1. the formation changes it shape, becoming a line as it passes between the object, and

2. the formation scales down its size, maintaining its triangular shape as it moves between the obstacles.

In neither case, a collision avoidance maneuver is included *a priori* in the trajectory to evade the obstacles, nor an inter-vehicle collision avoidance; obstacle negotiation solely depends on the operators' commands executed on run-time.

Considering a position estimation error from the pixhawk's estimator of about $1.5\,\mathrm{m}$, with a standard deviation of $0.8\,\mathrm{m}$, a similar error is expected for the cluster's centroid, while it could be greater for the distance-based parameters.

If, for a test, the reference trajectory is followed with an error within the expected parameters, the result of the test is considered to be successful.

4.1 Simulation results

In both simulation scenarios the cluster has the same initial position: $z_c = 5\,\text{m}, p = 7.1\,\text{m}, q = 7.1\,\text{m}, \beta = 60°$ and all other parameters with a zero value. The obstacles, of $20\,\text{m}$ height, are placed at $(-4\,\text{m}, 6\,\text{m}, 0\,\text{m})$ and $(-4\,\text{m}, 6\,\text{m}, 0\,\text{m})$.

To evade the obstacles by switching from a triangle to a line, the varying parameters are p and β, while y_c vary just to go through the obstacles. This variation is shown in Figure 5.

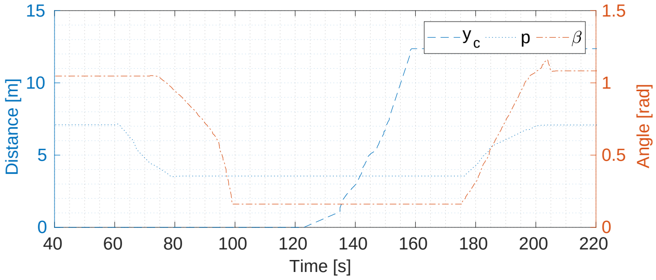


Figure 5: 3 UAV cluster varying parameters while evading obstacles by switching from a triangle to a line formation.

The obstacles positions and the cluster motion, on an XY plane, for the aforementioned cases are shown in Figure 6 and Figure 7. The first figure shows the trajectory while switching from a triangle to line, while the latter shows the formation while changing the triangle size.

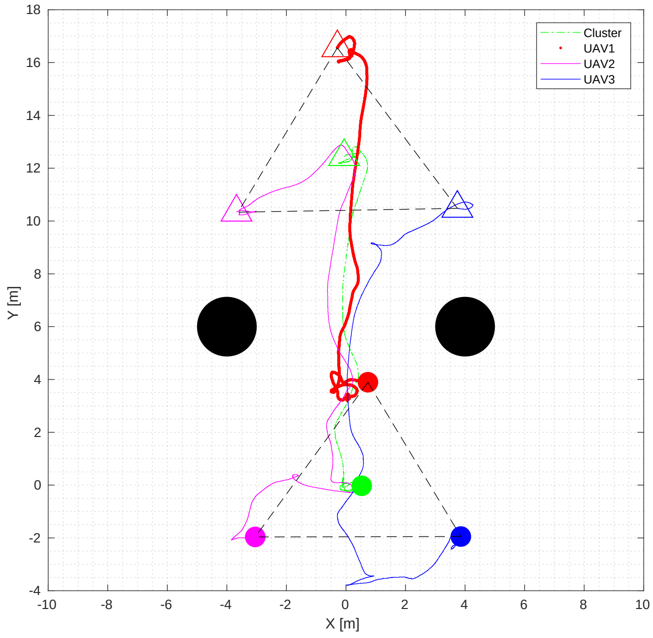


Figure 6: XY movement of the simulated 3 UAV cluster

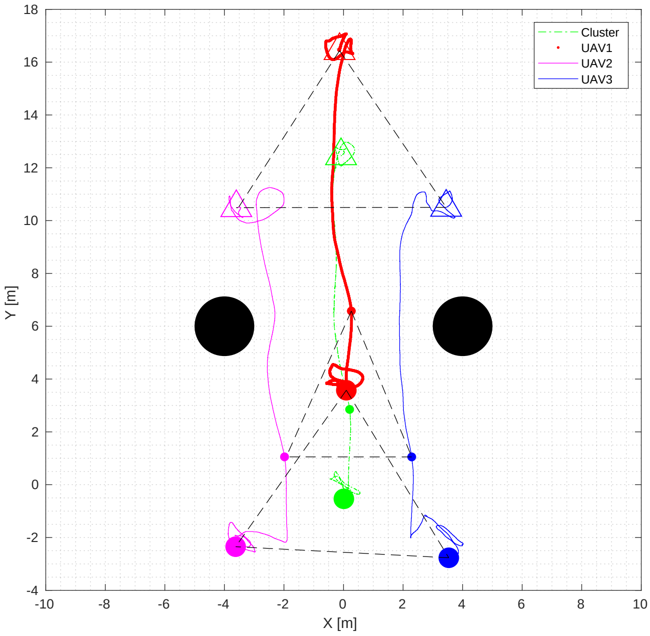


Figure 7: XY movement of the simulated cluster maneuvering through the obstacles reducing the area of coverage of the formation

Figure 8 shows the cluster state errors while maneuvering as a line formation. It can be seen that as soon as β approaches 0, the error of the roll parameter, γ_c, increases as there is a singularity when the agents are co-linear. Another error of importance can be seen for y_c near $t = 160\,\text{s}$, which is due to fast varying parameters and a relative slow system response.

Figure 13 shows the cluster state errors while maneuvering as a triangle formation. It can be seen that the formation goes between the obstacles, staying further away of the singularities. This property gives the controller a better performance.

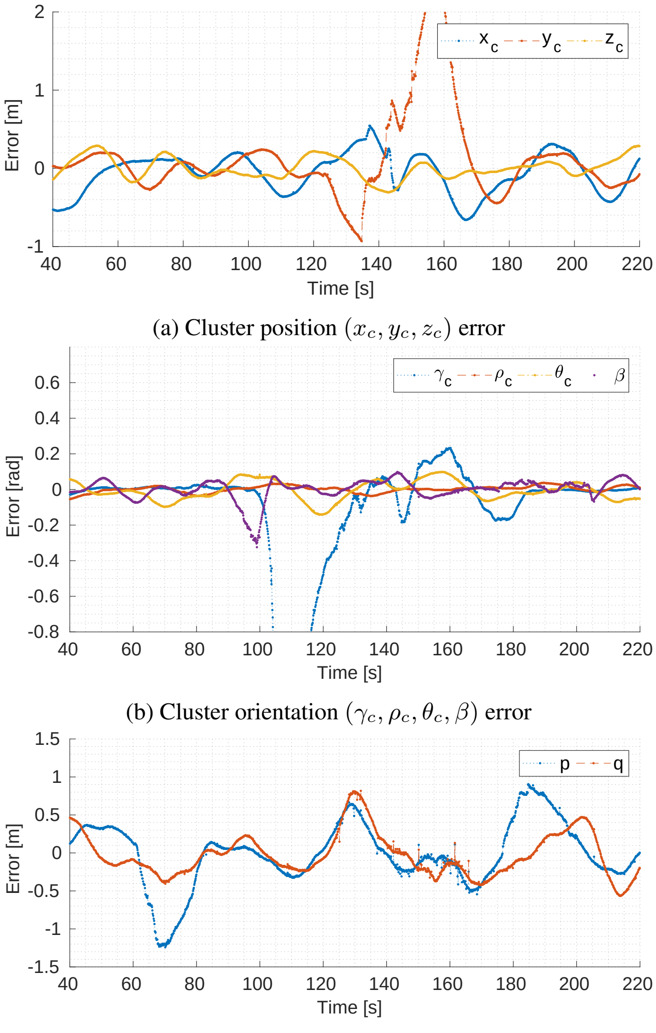
4.2 Experimental results

In these scenarios the cluster has the initial position: $z_c = 3\,\text{m}, p = 7.1\,\text{m}, q = 7.1\,\text{m}, \beta = 60°$ and all other parameters with a zero value. The obstacles were at $(-6\,\text{m}, -7\,\text{m}, 0\,\text{m})$ and $(2\,\text{m}, -7\,\text{m}, 0\,\text{m})$.

As in the simulation, for the first scenario the varying parameters are p and β, while y_c vary just to go through the obstacles. This variation is shown in Figure 9.

The obstacles positions and the cluster motion, on an XY plane, for both scenarios are shown in Figure 10 and Figure 11. The first figure shows the trajectory while switching from a triangle to line, while the latter shows the formation while changing the triangle size.

Figure 12 shows the cluster state errors while maneuvering as a line formation. It can be seen that β again approaches 0 and γ_c error increases as in the simulation case.



(a) Cluster position (x_c, y_c, z_c) error

(b) Cluster orientation $(\gamma_c, \rho_c, \theta_c, \beta)$ error

(c) Cluster vehicle distances (p, q) error

Figure 8: Cluster errors of a simulation using a joystick to control the formation (line shape obstacle avoidance)

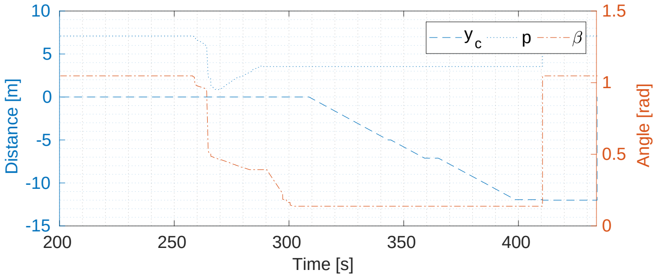


Figure 9: 3 UAV Cluster varying parameters while evading obstacles as a line formation during a field experiment.

The cluster state errors while maneuvering as a triangle formation, shown in Figure 14, present analogous results to those of the simulation.

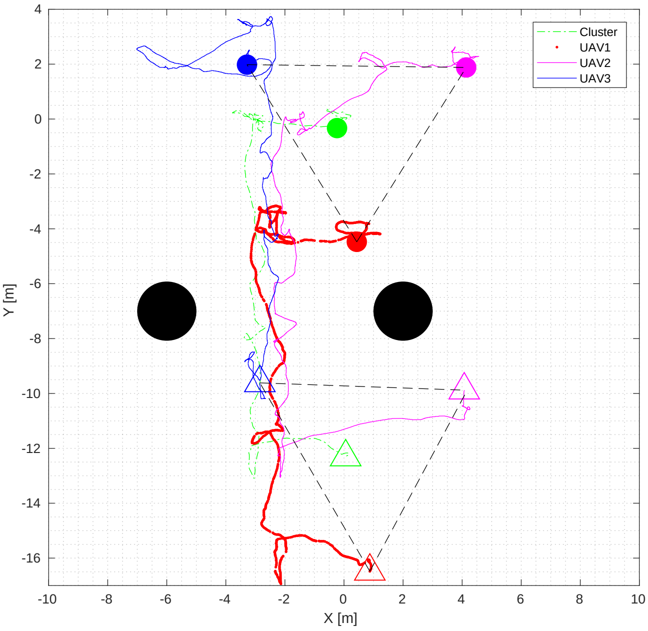


Figure 10: 2D movement of the real cluster evading the obstacles by switching the formation shape into a line

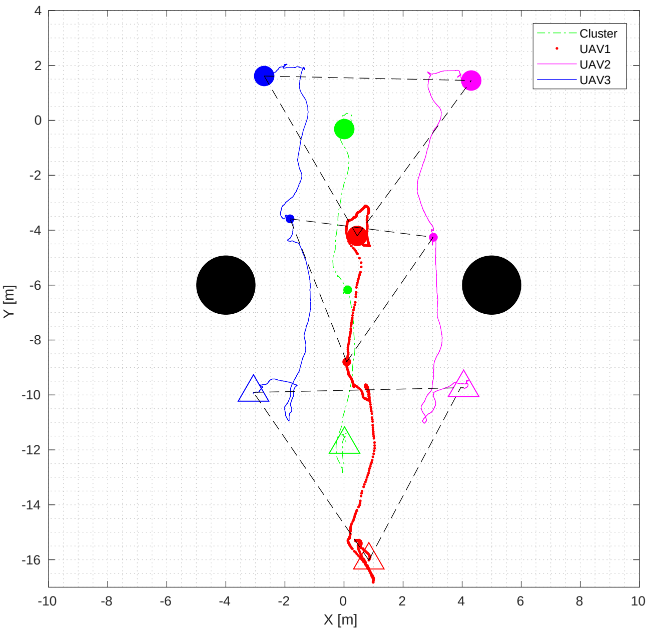
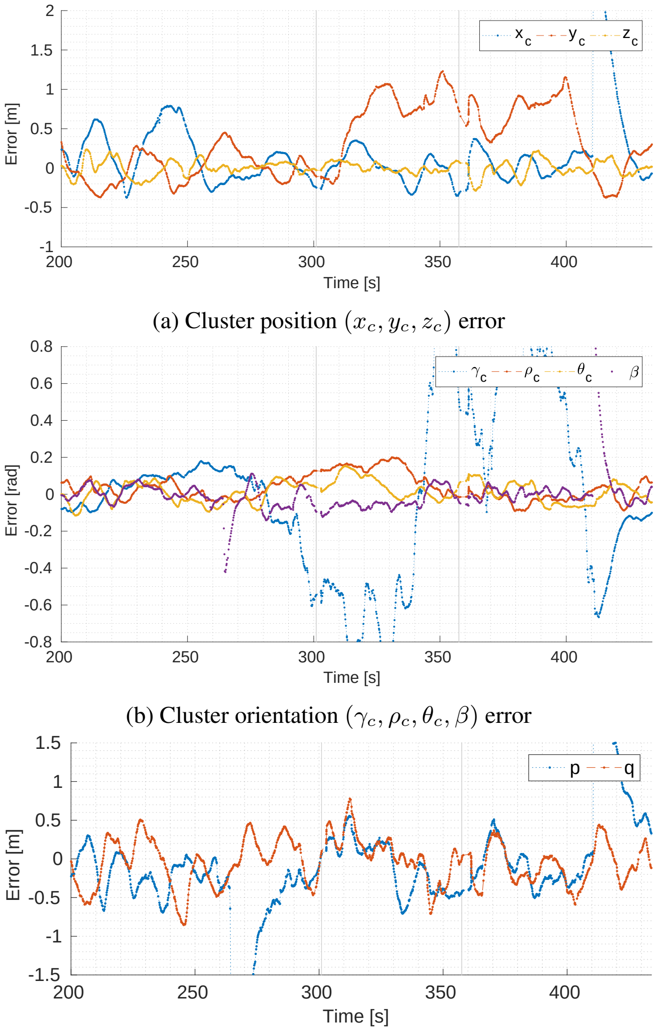


Figure 11: XY movement of the cluster maneuvering through the obstacles reducing the area of coverage of the formation (field experiment)

5 Conclusion

This work presented a cluster space controller with pilot-in-the-loop capability to allow for run-time actuation at the formation level, which provides the ability to modify a predefined trajectory to execute maneuvers such as collision avoid-



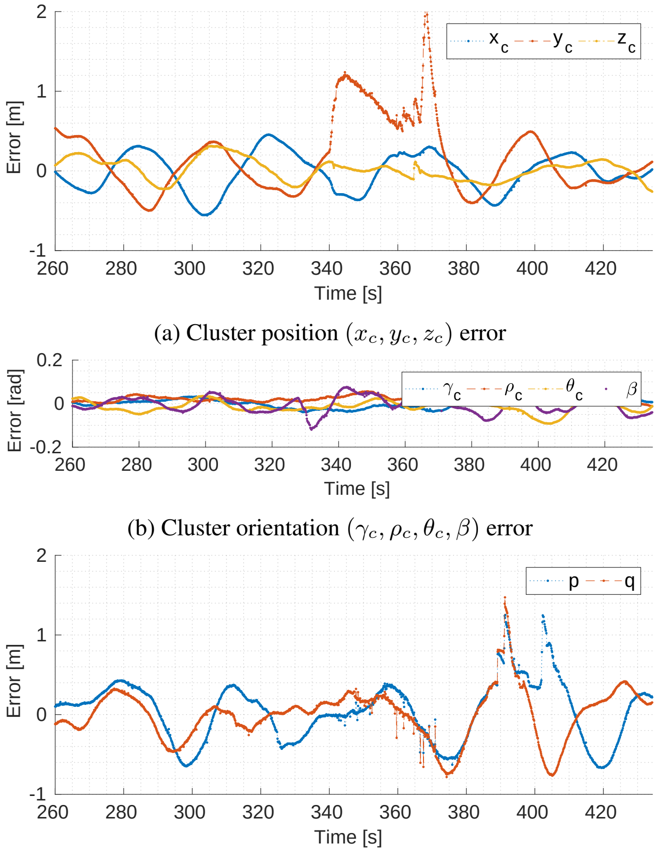
(a) Cluster position (x_c, y_c, z_c) error

(b) Cluster orientation $(\gamma_c, \rho_c, \theta_c, \beta)$ error

(c) Cluster vehicle distances (p, q) error

Figure 12: Cluster errors for an outdoor experiment using a joystick to control the formation (line shape obstacle avoidance)

ance. The proposed architecture was applied to a formation of three UAVs. By means of computer simulations and outdoor experiments the controller was shown to work and to be adequate for the case study applications. It was also shown that the multi-UAV formation could be intuitively operated using a single remote control, meaning that the operator can command the cluster as a whole in an abstracted manner that does not require to focus on the motions of the individual vehicles. Although the specification shows adequate results, this approach could be improved by adding an inter-vehicle collision avoidance mechanism, such as restrictions to cluster parameters or collision avoidance at the vehicle level.



(a) Cluster position (x_c, y_c, z_c) error

(b) Cluster orientation $(\gamma_c, \rho_c, \theta_c, \beta)$ error

(c) Cluster vehicle distances (p, q) error

Figure 13: Cluster errors for a simulation experiment using a joystick to control the formation (triangle shape obstacle avoidance)

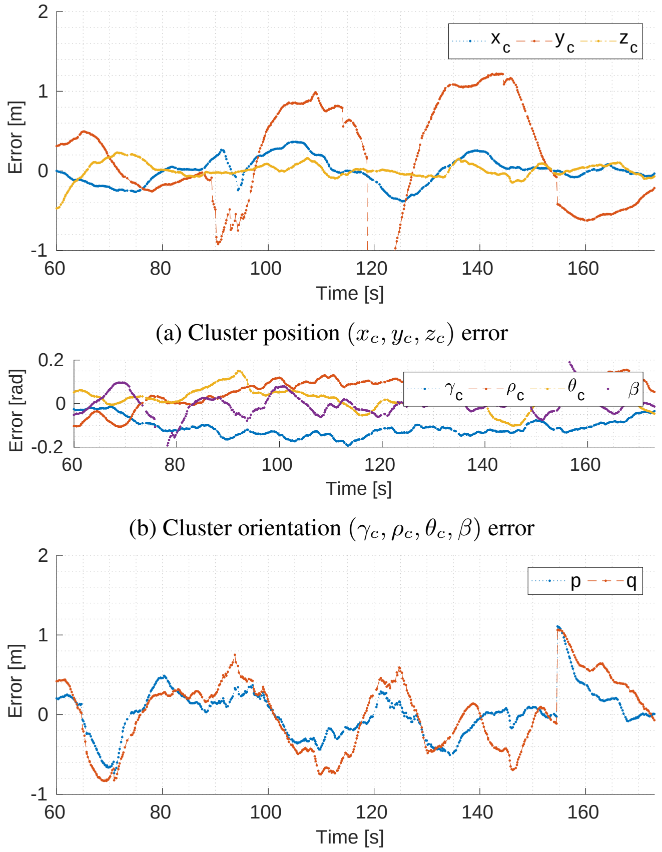
Acknowledgements

The authors would like thank the authorities of the School of Agriculture of the University of Buenos Aires for providing access to their campus to conduct the experiments. This work has been sponsored through the UBA-PDE-18- 2019 grant from Universidad de Buenos Aires, Argentina, Instituto Tecnolgico de Buenos Aires ITBACyT Interdiciplinario DT13/2018, and PICT 2016-2016 grant from Agencia Nacional de Promoción Científica y Tecnológica (Argentina). The experiments and research were done with a fellowship from the Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET) from Argentina.

References

[1] FJ Perez-Grau, R Ragel, F Caballero, A Viguria, and A Ollero. Semi-autonomous teleoperation of uavs in search and rescue scenarios. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1066–1074. IEEE, 2017.

[2] J. Barreiro-Gomez, I. Mas, C. Ocampo-Martinez,



(a) Cluster position (x_c, y_c, z_c) error

(b) Cluster orientation $(\gamma_c, \rho_c, \theta_c, \beta)$ error

(c) Cluster vehicle distances (p, q) error

Figure 14: Cluster errors for an outdoor experiment using a joystick to control the formation (triangle shape obstacle avoidance)

R. Sanchez-Pena, and N. Quijano. Distributed formation control of multiple unmanned aerial vehicles over time-varying graphs using population games. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 5245–5250, Dec 2016. doi: 10.1109/CDC.2016.7799072.

[3] V. Kumar, N. E. Leonard, and A. S. Morse. *Cooperative Control: A Post-Workshop Volume, 2003 Block Island Workshop on Cooperative Control*. New York: Springer-Verlag, 2005.

[4] H. M. La, R. Lim, and W. Sheng. Multirobot cooperative learning for predator avoidance. *IEEE Transactions on Control Systems Technology*, 23(1):52–63, Jan 2015. ISSN 1063-6536. doi: 10.1109/TCST.2014.2312392.

[5] I. Mas and C. Kitts. Dynamic control of mobile multirobot systems: the cluster space formulation. *IEEE Access*, 2:558–570, 2014.

[6] I. Mas and C. Kitts. Quaternions and dual quaternions: Singularity-free multirobot formation control. *Journal of Intelligent & Robotic Systems*, pages 1–18, 2016. ISSN 1573-0409. doi: 10.1007/s10846-016-0445-x.

[7] Peng Song and V. Kumar. A potential field based approach to multi-robot manipulation. *Robotics and Automation. Proceedings. ICRA. IEEE International Conference on*, 2:1217–1222, 2002. doi: 10.1109/ROBOT.2002.1014709.

[8] T. Balch and R.C. Arkin. Behavior-based formation control for multirobot teams. *Robotics and Automation, IEEE Transactions on*, 14(6):926–939, Dec 1998. ISSN 1042-296X. doi: 10.1109/70.736776.

[9] C. Belta and V. Kumar. Abstraction and control for groups of robots. *Robotics, IEEE Transactions on*, 20(5):865 – 875, oct. 2004. ISSN 1552-3098. doi: 10.1109/TRO.2004.829498.

[10] Martin Saska, Tomas Baca, Justin Thomas, Jan Chudoba, Libor Preucil, Tomas Krajnik, Jan Faigl, Giuseppe Loianno, and Vijay Kumar. 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, April 2017. ISSN 0929-5593, 1573-7527. doi: 10.1007/s10514-016-9567-z.

[11] Zhao Yunyun, Wang Xiangke, and Zhu Huayong. Multiple uavs configuration formation control via the dual quaternion method. In *Control Conference (CCC), 2015 34th Chinese*, pages 7207–7211. IEEE, 2015.

[12] Kwang-Kyo Oh, Myoung-Chul Park, and Hyo-Sung Ahn. A survey of multi-agent formation control. *Automatica*, 53:424 – 440, 2015. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2014.10.022.

[13] Even Yanmaz, Saeed Yahyanejad, Bernhard Rinner, Hermann Hellwagner, and Christian Bettstetter. Drone networks: Communications, coordination, and sensing. *Ad Hoc Networks*, 68:1 – 15, 2018. ISSN 1570-8705. doi: https://doi.org/10.1016/j.adhoc.2017.09.001. Advances in Wireless Communication and Networking for Cooperating Autonomous Systems.

[14] I. Mas and C. Kitts. Centralized and decentralized multi-robot control methods using the cluster space control framework. *Advanced Intelligent Mechatronics (AIM), 2010 IEEE/ASME International Conference on*, pages 115 –122, July 2010. doi: 10.1109/AIM.2010.5695768.

[15] P. Mahacek, C. A. Kitts, and I. Mas. Dynamic guarding of marine assets through cluster control of automated surface vessel fleets. *IEEE/ASME Transactions*

on Mechatronics, 17(1):65–75, Feb 2012. ISSN 1083-4435. doi: 10.1109/TMECH.2011.2174376.

[16] C. Kitts, P. Mahacek, T. Adamek, and I. Mas. Experiments in the control and application of automated surface vessel fleets. In *OCEANS'11 MTS/IEEE KONA*, pages 1–7, Sept 2011. doi: 10.23919/OCEANS.2011.6106976.

[17] Konstantin Kondak, Aníbal Ollero, Ivan Maza, Kai Krieger, Alin Albu-Schaeffer, Marc Schwarzbach, and Maximilian Laiacker. Unmanned aerial systems physically interacting with the environment: Load transportation, deployment, and aerial manipulation. *Handbook of Unmanned Aerial Vehicles*, pages 2755–2785, 2015.

[18] Gino J Lim, Seonjin Kim, Jaeyoung Cho, Yibin Gong, and Amin Khodaei. Multi-uav pre-positioning and routing for power network damage assessment. *IEEE Transactions on Smart Grid*, 9(4):3643–3651, 2016.

[19] Huang Xiaoqian, Hamad Karki, Amit Shukla, and Zhang Xiaoxiong. Variant pid controller design for autonomous visual tracking of oil and gas pipelines via an unmanned aerial vehicle. In *2017 17th International Conference on Control, Automation and Systems (ICCAS)*, pages 368–372. IEEE, 2017.

[20] I. Mas and C. Kitts. Dynamic control of mobile multirobot systems: The cluster space formulation. *Access, IEEE*, 2:558–570, 2014. ISSN 2169-3536. doi: 10.1109/ACCESS.2014.2325742.