Multiple formations control using distinct virtual structures

Vinícius Pacheco Bacheti, Alexandre Santos Brandão, Pedro Castillo Garcia, Rogelio Lozano, Mário Sarcinelli-Filho

Multiple formations control using distinct virtual structures

Vinícius Pacheco Bacheti[†], Alexandre Santos Brandão[‡], Pedro Castillo[*], Rogelio Lozano[*] and Mário Sarcinelli-Filho[†]

***Abstract*— This paper addresses the problem of controlling two distinct formations of robots interacting with one another. In order to do that, two different approaches for a line structure are defined and a controller based on virtual structures is used, as well as a low level controller in order to compensate for the dynamics of each robot. Tying both controllers is a multilayer structure responsible for the overall control. Simulations and experiments show the controllers working with different levels of complexity and help validate the proposed method.**

I. Introduction

Drones have increasingly been used in more and more applications. Those range from disaster management [1], construction [2], agriculture [3], to even how they could be used to help humanitarian crises, such as the COVID-19 pandemic [4], and many, many more. Among these applications, one that has seen increased attention has been the entertainment sector, where things such as light shows [5], [6] have become more and more common. Some advantages the drone light shows have over the classical fireworks show are less noise, that can have detrimental effects on local wildlife and pets, less pollution and less danger, as it involves no explosions. However, these swarms of hundreds of drones bring their own complexity and disadvantages, such as a limited battery life and a limited viewing angle [7].

One could also consider smaller shows, with only a handful of drones, that could be tailored to be presented to much smaller audiences, such as in schools, circuses, parties, demonstrations of laboratories, etc. In such a context, the use of formations may help to create interesting movement patterns while remaining in control of the overall movement ranges. Formations come in different flavors such as consensus based formation [8], [9], leader-follower [10], [11], [12], as well as virtual structures [13], [14]. This last paradigm was chosen for this work, in part due to its high stability [15], as well as due to previous experience by the authors.

The concept of the virtual structure approach is that the shape of the formation is treated as rigid body, and then by modulating the desired shape and position of this rigid body as movement by the robots that compose it, it is possible to achieve an organized movement that behaves as one demands. However, given a number of robots, one could define either a complex single shape, or several simpler shapes that together form a complex structure. These simple shapes could be triangles [16], [17], or, in the case of the present work, lines. However, even simple structures such as lines can have more than one definition with their own set of advantages and disadvantages as this work shows.

To sum up, this work addresses the use and interface of distinct line structures to control a small team of drones in the context of entertainment for small groups of people. To discuss these topics, the paper is hereinafter split in five sections, beginning with Section II, which shows in detail the used formations. In the sequel, Section III describes the controllers used to guide the formations on the completion of the task, and Sections IV and V analyze the results of the simulations and experiment performed. Finally, Section VI closes the manuscript, presenting the final conclusions and ideas for future work.

II. Formation description

This section describes the formations used here. The first subsection deals with the same formation of [13], which is a line-shaped virtual structure, whose point of interest (POI) for the controller is at one of the extremities. In the second subsection, the POI of the formation is moved to its center, and all the relevant equations are shown. Finally, in the last subsection, the formations as they are used in the present work are defined.

A. Line virtual structure with POI at the end of the formation

As previously mentioned, this kind of formation was first described in [13]. There, however, it was used for an heterogeneous formation composed of one UGV and one UAV. In the present work, the formation is a homogeneous formation of two UAVs, as can be seen in Fig. 1.

The figure shows the positions of the robots, and the one of the formation as well. For the former, $\mathbf{x} = \begin{bmatrix}\boldsymbol{\xi}_1 & \boldsymbol{\xi}_2\end{bmatrix}^T$, where $\boldsymbol{\xi}_1 = \begin{bmatrix}x_1 & y_1 & z_1\end{bmatrix}^T$ and $\boldsymbol{\xi}_2 = \begin{bmatrix}x_2 & y_2 & z_2\end{bmatrix}^T$ represents each robot position along the space. Meanwhile, the latter is defined as $\mathbf{q} = \begin{bmatrix}\mathbf{p} & \mathbf{s}\end{bmatrix}^T$, where $\mathbf{p} = \begin{bmatrix}x_f & y_f & z_f\end{bmatrix}^T$ represents the position of the POI along the three axes and $\mathbf{s} = \begin{bmatrix}\rho_f & \alpha_f & \beta_f\end{bmatrix}^T$ represents the shape of the line in the space. Note that $\boldsymbol{\xi}_1 = \mathbf{p}$, which means that the POI of the formation coincides with the first robot and is, furthermore, located in one of the extremities of the line. The shape variables of the formation ρ_f, α_f and β_f, in turn, represent the length of the line, the angle between the $x-$axis and the

[†] V. P. Bacheti and M. Sarcinelli-Filho are with the Graduate Program on Electrical Engineering, Federal University of Espírito Santo, Vitória - ES, Brazil, where Mr. Bacheti is pursuing a Doctorate degree, and Dr. Sarcinelli-Filho is a professor vinicius.bacheti@gmail.com, mario.sarcinelli@ufes.br

[‡] A. S. Brandão is with the Department of Electrical Engineering, Federal University of Viçosa, Viçosa - MG, Brazil alexandre.brandao@ufv.br

[*] P. Castillo and R. Lozano are with Heudiasyc Laboratory CNRS UMR 7253, Universite de Technologie de Compiegne, Compiegne France castillo@hds.utc.fr, rlozano@hds.utc.fr

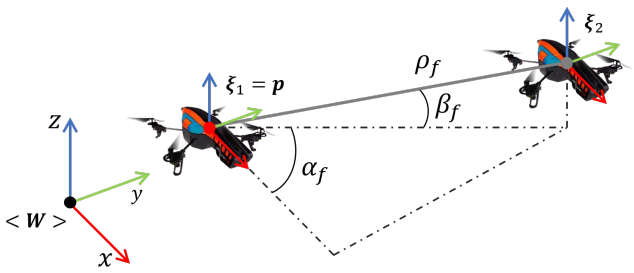

Fig. 1: Depiction of the virtual structure formation with POI in one of the formation extremities.

projection of the line in the $xy-$plane and the angle between the line and the $xy-$plane, respectively, as Figure 1 shows ($< \mathbf{W} >$ represents the inertial reference system).

These parameters are related by a function defined as

$$\mathbf{q} = f(\mathbf{x}), \tag{1}$$

called the direct transformation, responsible to map the robot variables to the formation variables. Such a function is essential because the tasks are usually in the formation space, which makes necessary to take the current positions of the robots given by the sensors and calculate the current state of the formation. This, in turn, is used to calculate the error for the controller (discussed in more details in Section III). Such a function is

$$\begin{aligned} x_f &= x_1, y_f = y_1, z_f = z_1, \\ \rho_f &= \sqrt{(x_2-x_1)^2+(y_2-y_1)^2+(z_2-z_1)^2}, \\ \alpha_f &= \arctan\frac{y_2-y_1}{x_2-x_1}, \\ \beta_f &= \arctan\frac{z_2-z_1}{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}. \end{aligned} \tag{2}$$

As discussed, the direct transformation maps the robots space to the formation space. Once the reference velocities for the formation to accomplish the desired task are defined, they should be mapped to each robot (see Section III). However, to perform such mapping it is necessary to define a transformation from the formation space to the robots space, called the inverse transformation, defined as

$$\mathbf{x} = f^{-1}(\mathbf{q}), \tag{3}$$

thus, from (2), it follows

$$\begin{aligned} x_1 &= x_f, y_1 = y_f, z_1 = z_f, \\ x_2 &= x_f + \rho_f \cos\alpha_f \cos\beta_f, \\ y_2 &= y_f + \rho_f \sin\alpha_f \cos\beta_f, \\ z_2 &= z_f + \rho_f \sin\beta_f. \end{aligned} \tag{4}$$

B. Line virtual structure with POI at the center of gravity

Notice that the previous section dealt with a formation with the POI at the extremity of the virtual structure. This came from a work where there was two different robots, one ground vehicle and one aerial vehicle, and the UAV landed on the UGV. It made sense to make the POI coincident with the UGV for the landing maneuver.

However, such a formation could be defined in a different way, such as with the POI in the center of the structure. This may be more natural in a formation with two equal robots, where both robots give equal contributions to the task. Fig 2 shows this case.

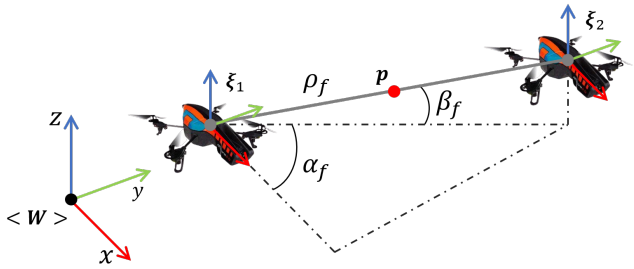

Fig. 2: Depiction of the virtual structure formation with POI in the center of gravity of the formation.

For such a formation, the direct transform is defined as

$$\begin{aligned} x_f &= \frac{x_1+x_2}{2}, y_f = \frac{y_1+y_2}{2}, z_f = \frac{z_1+z_2}{2}, \\ \rho_f &= \sqrt{(x_2-x_1)^2+(y_2-y_1)^2+(z_2-z_1)^2}, \\ \alpha_f &= \arctan\frac{y_2-y_1}{x_2-x_1}, \\ \beta_f &= \arctan\frac{z_2-z_1}{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}. \end{aligned} \tag{5}$$

From the equations above, one can notice that the shape variables transformations (ρ_f, α_f and β_f) remain the same. This comes from the fact that the POI could be anywhere on the line, or even outside it, without changing the shape of the line. What changes, however, is the position of the POI, which now depends on both robots. Such a change is reflected in the inverse transformation, defined as

$$\begin{aligned} x_1 &= x_f - \frac{\rho_f \cos\alpha_f \cos\beta_f}{2}, \\ y_1 &= y_f - \frac{\rho_f \sin\alpha_f \cos\beta_f}{2}, \\ z_1 &= z_f - \frac{\rho_f \sin\beta_f}{2}, \\ x_2 &= x_f + \frac{\rho_f \cos\alpha_f \cos\beta_f}{2}, \\ y_2 &= y_f + \frac{\rho_f \sin\alpha_f \cos\beta_f}{2}, \\ z_2 &= z_f + \frac{\rho_f \sin\beta_f}{2}. \end{aligned} \tag{6}$$

where it is evidently more complex than the previous case with the point in the extremity. Such a change also causes a difference in the functions mapping the velocities of the robots to the formation and vice versa.

C. Definition of the formations used in the present work

The last two subsections defined two different kind of formations in a generic sense. The present subsection, however,

defines the formations that are used in this paper as well as how they relate to each other.

First, a formation with POI at the center of the line is defined with two UAVs in its extremities

$$\mathbf{q}_1 = f(\mathbf{x}_1), \tag{7}$$

where $\mathbf{q}_1 = \begin{bmatrix}\mathbf{p}_1 & \mathbf{s}_1\end{bmatrix}^T$ represents the formation associated with the index 1 and $\mathbf{x}_1 = \begin{bmatrix}\boldsymbol{\xi}_1 & \boldsymbol{\xi}_2\end{bmatrix}^T$ represents the vector with the positions of the robots that compose the formation also associated with the index 1, namely, $\boldsymbol{\xi}_1$ and $\boldsymbol{\xi}_2$.

Then, a second formation with the POI in one of the extremities is defined. This takes the form of

$$\mathbf{q}_2 = f(\mathbf{x}_2), \tag{8}$$

where $\mathbf{q}_2 = \begin{bmatrix}\mathbf{p}_2 & \mathbf{s}_2\end{bmatrix}^T$ represents the formation associated with the index 2 and $\mathbf{x}_2 = \begin{bmatrix}\mathbf{p}_1 & \boldsymbol{\xi}_3\end{bmatrix}^T$ represents the vector with the positions of the agents that compose the formation also associated with the index 2, namely, $\mathbf{p}_1$ and $\boldsymbol{\xi}_3$. Notice, that in this second formation, one of the agents is not a robot, but it is the POI of the first formation.

This fact makes it clearer why both types of formation were defined in the present work. By using them both, and the direct transformation defined in (1) it is possible to affirm

$$\mathbf{p}_1 = \mathbf{p}_2,$$

which means that the formations are always connected to each other. Fig. 3 illustrates the concepts explained above.

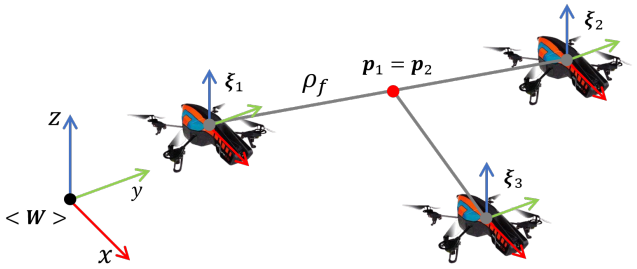

Fig. 3: Depiction of the complete formation, composed of two sub-formations.

III. Controller

This section discusses the controller used to guide the formation in accomplishing the task. The first subsection describes the formation controller, and how it is only a kinematic solution, while the second subsection discusses the dynamic compensation for the robots, and explain the control structure that ties both parts together, namely the multilayer structure.

A. Formation controller

In order for the formation to accomplish tasks successfully, whether it be a positioning, trajectory tracking or path following task, it needs a controller to guide it. The controller chosen in this work is the same as the one present in [13], where it is used for both positioning and trajectory tracking tasks. However, it has been proven to work in path following also in other works [18], [19]. This controller takes the form

$$\dot{\mathbf{q}}_{1_r} = \dot{\mathbf{q}}_{1_d} + \mathbf{L}_1 \tanh\left(\mathbf{L}_2 \tilde{\mathbf{q}}_1\right) \tag{9}$$

for the first formation and

$$\dot{\mathbf{q}}_{2_r} = \dot{\mathbf{q}}_{2_d} + \mathbf{L}_3 \tanh\left(\mathbf{L}_4 \tilde{\mathbf{q}}_2\right) \tag{10}$$

for the second formation. The variables $\mathbf{L}_1$ through $\mathbf{L}_4$ are diagonal positive definite gain matrices, the vector $\mathbf{q}_{i_d}$ represents the desired values for the position and shape of each formation, $\dot{\mathbf{q}}_{i_d}$ is its time derivative representing the desired velocity, while $\tilde{\mathbf{q}}_i$ represents the error associated with each formation, defined as

$$\tilde{\mathbf{q}}_i = \mathbf{q}_{i_d} - \mathbf{q}_i. \tag{11}$$

Finally, $\dot{\mathbf{q}}_{i_r}$ represents the vector of reference velocities for the formation to accomplish the desired task.

This, however, is a vector of references for a virtual structure which, as a imaginary construct, cannot move on its own. Ir order for it to move, its physical components, namely the robots, should move in such a way that the formation they belong to move in the desired manner. In order to accomplish this feat, a mapping from the reference velocities of the formation to the reference velocity of the robots should be done. To accomplish that, a differentiation with respect to time is performed in (3) for each of the formations, and afterwards a substitution for the reference velocities from the controller, such that

$$\dot{\mathbf{x}}_{i_r} = \mathbf{J}_i^{-1}(\mathbf{q})\dot{\mathbf{q}}_{i_r}, \tag{12}$$

where $\mathbf{J}_i$ is called the Jacobian matrix, and is defined as

$$\mathbf{J}_1^{-1}(\mathbf{q}) = \begin{bmatrix}\mathbf{I} & -\frac{\mathbf{J}_r}{2} \\ \mathbf{I} & \frac{\mathbf{J}_r}{2}\end{bmatrix}, \mathbf{J}_2^{-1}(\mathbf{q}) = \begin{bmatrix}\mathbf{I} & \mathbf{0} \\ \mathbf{I} & \mathbf{J}_r\end{bmatrix} \tag{13}$$

for the formation with the POI in the middle and the formation with the POI in the extremity, respectively, where

$$\mathbf{J}_r(\mathbf{q}) = \begin{bmatrix} c_{\alpha_f} c_{\beta_f} & -\rho_f s_{\alpha_f} c_{\beta_f} & -\rho_f c_{\alpha_f} s_{\beta_f} \\ s_{\alpha_f} c_{\beta_f} & \rho_f c_{\alpha_f} c_{\beta_f} & -\rho_f s_{\alpha_f} s_{\beta_f} \\ s_{\beta_f} & 0 & \rho_f c_{\beta_f} \end{bmatrix}. \tag{14}$$

Finally, c_x and s_x represent the functions $\cos(x)$ and $\sin(x)$, respectively.

For two formations, as is the case in this work, (12) is applied twice, yielding the two vectors $\dot{\mathbf{x}}_{1_r}$ and $\dot{\mathbf{x}}_{2_r}$. The first three positions of the former are the cartesian velocities references for the first robot, while the last three are the same for the second robot, which means $\dot{\mathbf{x}}_{1_r} = \begin{bmatrix}\boldsymbol{\nu}_{1_r} & \boldsymbol{\nu}_{2_r}\end{bmatrix}^T$. However, for the latter vector, the first three positions would be the velocity references for a virtual agent coinciding with the positions of both formations. Such a construct does not move on its own, so it makes no sense commanding it, as it moves according to the two first robots. This means that these first three values are not used, while the last three values of the second vector are the reference for the third, and last, robot, thus $\dot{\mathbf{x}}_{2_r} = \begin{bmatrix}\boldsymbol{\nu}_{p_r} & \boldsymbol{\nu}_{3_r}\end{bmatrix}^T$. Ultimately, $\boldsymbol{\nu}_{i_r}$ is the reference velocity for each robot composing the formation.

The controller presented is enough to guide the formation to accomplish its task, as long as the robots follow the reference, which, according to (12), is a velocity value. This means that the controller presented deals purely with the kinematics of the system. However, all robots have dynamics that may introduce noise in the system and, also, most drones do not take cartesian velocity commands directly. In order to deal with these problems, the next subsection introduces a new controller responsible for the dynamic compensation of each robot, as well as the structure responsible to tie both controllers together into a cohesive whole.

B. Dynamic compensation and multilayer control

The last subsection established how each robot should move in space. Now, in order to compensate for the dynamics of the drones, a controller taking it into account has to be used. In order to bridge the gap between the formation control and the dynamic compensation control, a multilayer control structure is used, which is presented in Fig. 4. Such structure has been used in many previous works by the authors such as in [18], [19], [17].

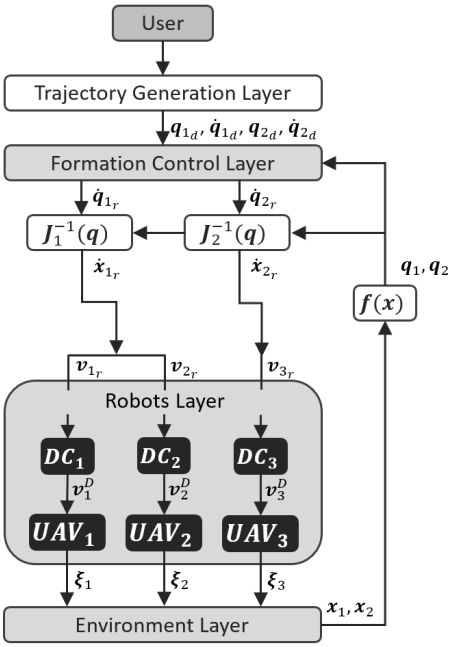

Fig. 4: System navigation.

As can be observed in Fig. 4, the formation controller is responsible to take the trajectory task and turn it into velocities references for each formation, $\dot{\mathbf{q}}_{i_r}$. Then, using the respective jacobian matrix, these formation velocity references are transformed into velocity references for each robot, $\dot{\mathbf{x}}_{1_r}$ and $\dot{\mathbf{x}}_{2_r}$, which are subdivided into $\boldsymbol{\nu}_{1_r}$, $\boldsymbol{\nu}_{2_r}$ and $\boldsymbol{\nu}_{3_r}$. Then, entering into the robots layer, each robot is responsible for its own dynamic compensation, which transforms the kinematic reference into a dynamic one, $\boldsymbol{\nu}_{i_r}^D$. Finally, these inputs are applied to each robot, and as they move, the sensors detect their new positions, which are then transformed into the formation space, through (2) and (5), and the whole process begins again.

Now, in order to present the dynamic compensation done in the present work, the first step is to present the model used. First, the generalized coordinates of the drone are defined as $\mathbf{x} = \begin{bmatrix}\boldsymbol{\xi}^T & \boldsymbol{\eta}^T\end{bmatrix}^T$, where $\boldsymbol{\xi} = \begin{bmatrix}x & y & z\end{bmatrix}^T$ is the position of the center of mass of the UAV in space, relative to the inertial frame, and $\boldsymbol{\eta} = \begin{bmatrix}\phi & \theta & \psi\end{bmatrix}^T$ is the Euler angle representation of the vehicles attitude. The model of each UAV is obtained through the Euler-Lagrange method, and defined as

$$m\ddot{x} = u(\sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi), \quad (15)$$
$$m\ddot{y} = u(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi), \quad (16)$$
$$m\ddot{z} = u\cos\phi\cos\theta - mg, \quad (17)$$
$$\ddot{\phi} = \tilde{\tau}_\phi, \quad (18)$$
$$\ddot{\theta} = \tilde{\tau}_\theta, \quad (19)$$
$$\ddot{\psi} = \tilde{\tau}_\psi. \quad (20)$$

where m is the mass of the vehicle, g is the gravity, u is the thrust generated by the motors and $\tilde{\tau}_\phi$, $\tilde{\tau}_\theta$ and $\tilde{\tau}_\psi$ are the rolling pitching and yawning moment, respectively, related to the input torques $\boldsymbol{\tau}$ by

$$\begin{bmatrix}\tilde{\tau}_\phi \\ \tilde{\tau}_\theta \\ \tilde{\tau}_\psi\end{bmatrix} = \mathbf{I}_{\boldsymbol{\eta}}^{-1}(\boldsymbol{\tau} - \mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})\dot{\boldsymbol{\eta}}), \quad (21)$$

where $\mathbf{I}_{\boldsymbol{\eta}}$ is the inertia matrix of the drone expressed in the generalized coordinates $\boldsymbol{\eta}$, and $\mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})$ is the Coriolis term.

The model presented implies that the input commands to the drones are the thrust force u and the torques τ_ϕ, τ_θ and τ_ψ. However, the formation controller presented has as output the desired velocities along the three axes of space. In order to find out the desired commands to the drone compensating for its dynamics a controller based on the one present in [20] is chosen. First, it is considered that the altitude is stabilized, and then the desired roll and pitch angles are chosen in a way to reach the desired x and y displacement.

In order to control the altitude, the thrust is calculated as

$$u = \frac{m(\nu_z + g)}{\cos\phi\cos\theta}, \quad (22)$$

where

$$\nu_z = \ddot{z}_d + k_z\dot{\tilde{z}}, \quad (23)$$

k_z is a positive constant, $\dot{\tilde{z}}$ is the velocity error, defined as $\dot{\tilde{z}} = \dot{z}_d - \dot{z}$ where $\dot{z}$ is the UAV current velocity in the $z-$axis and $\dot{z}_d$ is the desired velocity in the same axis, given by the third element of $\boldsymbol{\nu}_{i_r}$, and finally, $\ddot{z}_d$ is its temporal derivative, or the desired acceleration of the UAV.

Now, introducing (22) and (23) into (15)-(17), one gets

$$\ddot{x} = (\nu_z + g)(\frac{\tan\phi}{\cos\theta}\sin\psi + \tan\theta\cos\psi), \quad (24)$$
$$\ddot{y} = (\nu_z + g)(\tan\theta\sin\psi - \frac{\tan\phi}{\cos\theta}\cos\psi). \quad (25)$$
$$\ddot{z} = \ddot{z}_d + k_z\dot{\tilde{z}}. \quad (26)$$

Then, by choosing the desired roll and pitch values as

$$\phi_d = \arctan \frac{(\nu_x \sin\psi - \nu_y \cos\psi)\cos\theta}{\nu_z + g}, \quad (27)$$

$$\theta_d = \arctan \frac{(\nu_x \cos\psi + \nu_y \sin\psi)}{\nu_z + g}, \quad (28)$$

where

$$\nu_x = \ddot{x}_d + k_x \dot{\tilde{x}} \quad (29)$$

and

$$\nu_y = \ddot{y}_d + k_y \dot{\tilde{y}}, \quad (30)$$

and substituting them into (24) and (25), they become

$$\ddot{x} = \ddot{x}_d + k_x \dot{\tilde{x}}, \quad (31)$$

$$\ddot{y} = \ddot{y}_d + k_y \dot{\tilde{y}}. \quad (32)$$

Finally, all that remains, is to achieve the desired attitude values. To do that a simple controller is used, defined as

$$\tilde{\tau}_\phi = \dot{\phi}_d + k_\phi \tilde{\phi}, \quad (33)$$

$$\tilde{\tau}_\theta = \dot{\theta}_d + k_\theta \tilde{\theta}. \quad (34)$$

where k_ϕ and k_θ are positive constants, $\tilde{\phi}$ and $\tilde{\theta}$ are the roll and pitch attitude error, defined as $\tilde{\phi} = \phi_d - \phi$ and $\tilde{\theta} = \theta_d - \theta$, respectively, and ϕ_d and θ_d are the desired attitudes.

As for the yaw control, it is wholly independent of everything else, and is not contemplated in the formation control. Therefore, any yaw value can be chosen, remaining constant during the whole flight, besides of tracking the desired trajectory. The input is then set as

$$\tilde{\tau}_\psi = \dot{\psi}_d + k_\psi \tilde{\psi}, \quad (35)$$

where k_ψ is a positive constant, $\tilde{\psi}$ is the yaw orientation errors $\tilde{\psi} = \psi_d - \psi$, where ψ_d is the desired orientation.

IV. Simulations

This section describes the simulations used to validate the proposed solution. The first part shows a simulation performed on the software *Matlab*, focusing only on the formation controller, i.e., the kinematic part of the system with perfect velocity tracking. The second part shows a simulation performed on the system *Fl-Air*, developed by the Heudyasic Laboratory, and contemplates the whole system, together with the dynamic of the UAVs.

A. Formation controller validation

As mentioned, this first part aims at validating the formation controller for the proposed system composed of two linked formations. In order to do that, a simulation was written on *Matlab* and a task aiming at exciting all the different formation parameters in order to create complex movements was chosen. This task is described by

$$\mathbf{q}_{1_d} = \begin{bmatrix} \frac{\sin\omega t}{4} + 1 \\ 0 \\ \frac{\cos\omega t}{4} + 0.75 \\ 0.5 \\ \omega t \\ \frac{\pi}{4}\sin\omega t \end{bmatrix}, \mathbf{q}_{2_d} = \begin{bmatrix} \frac{\sin\omega t}{4} + 1 \\ 0 \\ \frac{\cos\omega t}{4} + 0.75 \\ 0.25 \\ \omega t - \frac{\pi}{2} \\ -\frac{\pi}{4}\sin\omega t \end{bmatrix}. \quad (36)$$

Notice that the desired positions for the POIs of both formations are the same, as they are essentially the same point. As for the angular frequency ω, it was chosen as $\omega = \frac{2\pi}{T}$, where the period is $T = 20$ seconds.

The navigation of all of the agents is shown in Fig. 5, alongside the trajectory each one performed. Furthermore, both line structures are depicted in the figure, the first one in magenta and the second one in yellow.

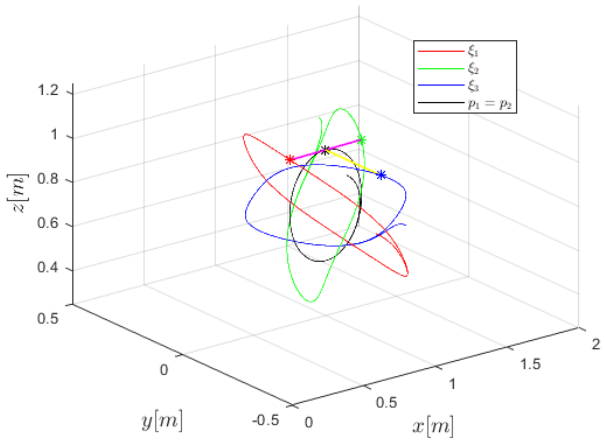

Fig. 5: System navigation in the *Matlab* simulation.

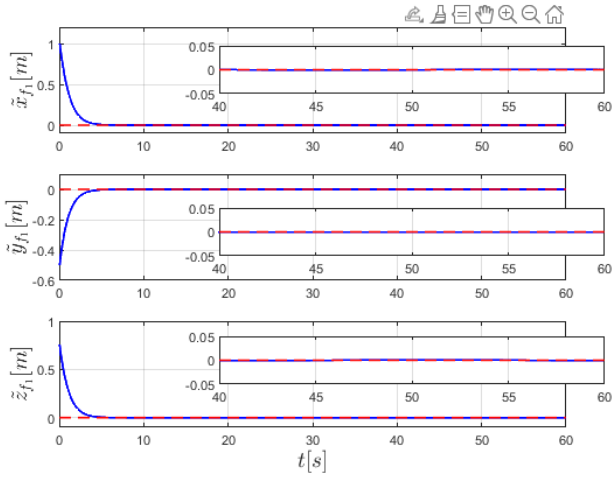

Fig. 6: Formation 1 position errors for the simulation using Matlab.

Figures 6, 7 and 8 plot the errors of each formation parameter and show the results. A small box zooming in is shown in order to better demonstrate the magnitude of the errors after the trajectory is reached. It can be seen from all of them that the formation errors are very small, even though very complex movements were followed by the formation. Also, for the sake of space, the figure showing the position errors for the second formation was omitted, as it was identical to the first, due to the fact that both their desired and actual values are the same.

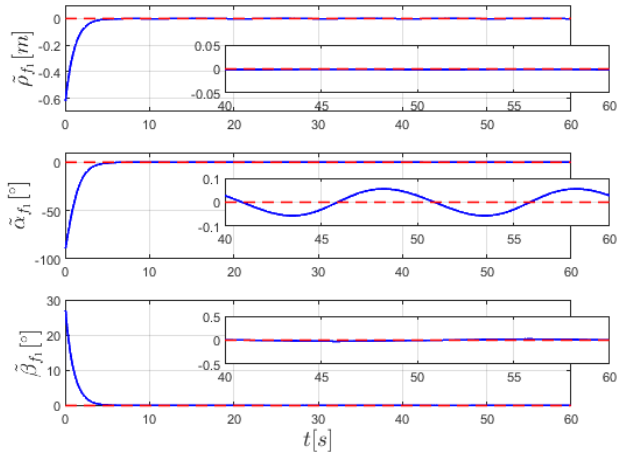

Fig. 7: Formation 1 shape errors for the simulation using Matlab.

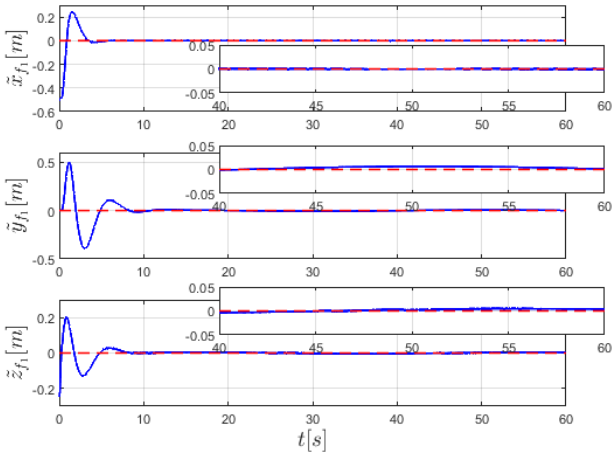

Fig. 9: Formation 1 position errors for the simulation using Flair.

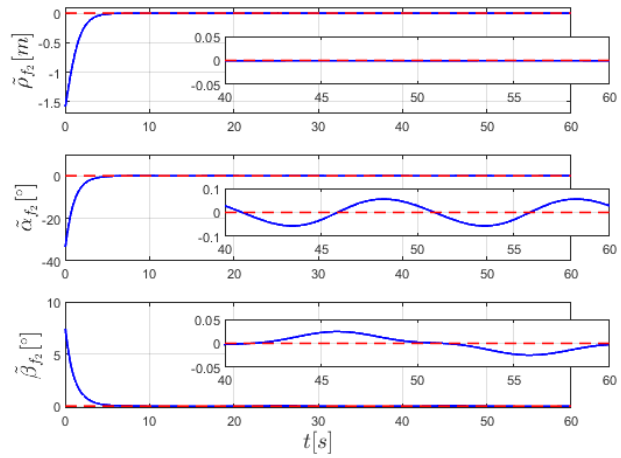

Fig. 8: Formation 2 shape errors for the simulation using Matlab.

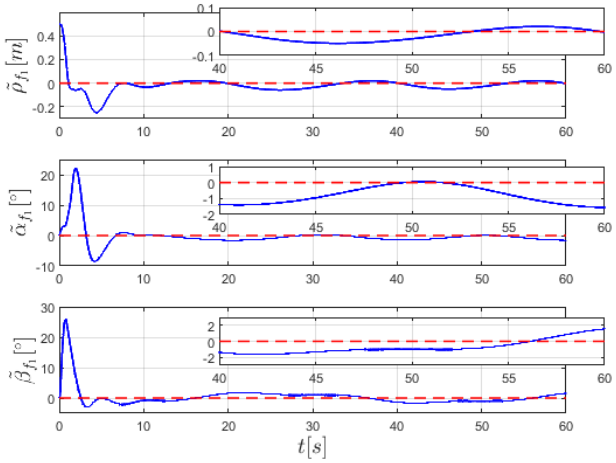

Fig. 10: Formation 1 shape errors for the simulation using Flair.

B. *Complete system simulation*

After validating that the formation controller works for the proposed formations in the proposed tasks, this subsection shows the results for the simulations done in the Flair platform, developed at the Heudyasic laboratory, where all the drones are much closer to their real counterparts, which means they are implemented with the dynamics of the system, and so the controller must contemplate these effects.

The task is described by the same formulas as the previous simulations, with two important distinctions. The first one is, due to a non zero size of the UAVs, it was deemed to be necessary to increase the values of the desired ρ_{f_1} and ρ_{f_2} from 0.5 and 0.25 to 1.5 and 0.75, respectively. This was so the drones were not flying too close to one another. As for the second distinction, due to the increased length of the formation, the arcs described by each UAV also increased in size, and to counter this effect, the period of the trajectories was increased to $T = 40$ seconds.

The result of the simulation is given in Figures 9, 10 and 11. Here, it is possible to observe a very similar behavior to the previous simulation, where the error tends to small values. However, it is possible to see, mainly in the errors associated with the shape of the formations, an increase in the magnitude of the error, as well as a change of behavior. As an example, the error associated with the length of the second formation ($\tilde{\rho}_{f_2}$), is offset from the origin, as well as clearly oscillatory. This most likely stems from the dynamic component that is now present in this simulation.

V. Experiment

This section describes the experiments that were run using the system developed in this work. These experiments were run at the Heudyasic laboratory, in a room with the motion capture system *OptiTrack*. All the drones used to run the experiment were of the model *ARDrone 2.0*, made by *Parrot*. Once again, the task was very similar to the one in the

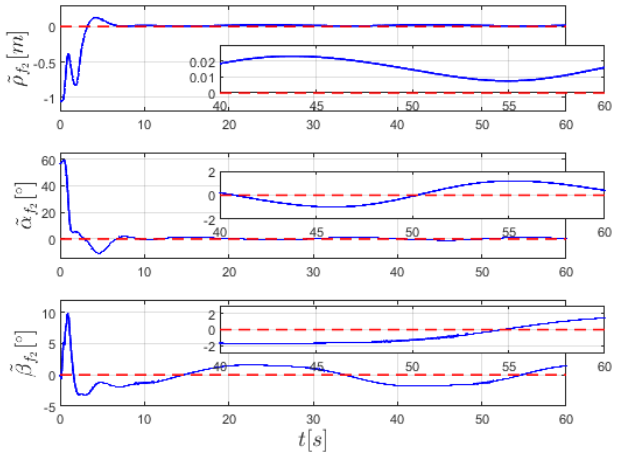

Fig. 11: Formation 2 shape errors for the simulation using Flair.

simulations, with a few small modifications in order to better use the limits of the room and to preserve the safety of the equipment, and is defined as

$$\mathbf{q}_{1_d} = \left[\frac{\sin\omega t}{4} \quad 0 \quad \frac{\cos\omega t+8}{4} \quad 1.5 \quad \omega t \quad \frac{\pi}{4}\sin\omega t\right], \quad (37)$$

$$\mathbf{q}_{2_d} = \left[\frac{\sin\omega t}{4} \quad 0 \quad \frac{\cos\omega t+8}{4} \quad 1.5 \quad \omega t - \frac{\pi}{2} \quad -\frac{\pi}{4}\sin\omega t\right]. \quad (38)$$

where $\omega = \frac{2\pi}{T}$, and the period is $T = 40$ seconds.

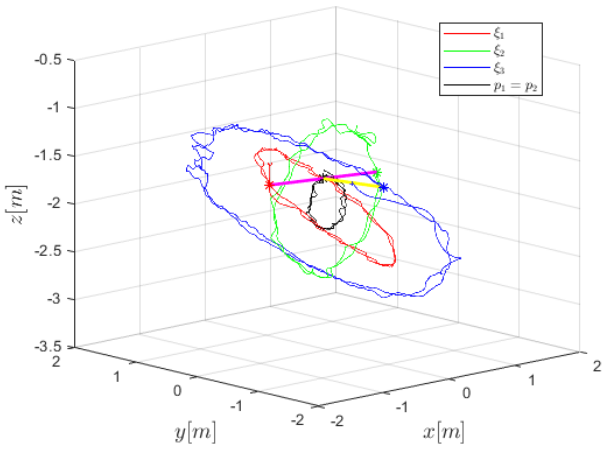

Fig. 12: System navigation in the experiment.

As in the simulation, Figure 12 shows the navigation of the agents as well as the formation, along with the line structures in magenta and yellow. Meanwhile, Figures 13, 14 and 15 show the errors during the whole experiment for each of the formation parameters. A video showing the experiment can be found at `https://youtu.be/uoF2SHTrpQ4`. It is important to note that, due to the Flair platform using a different convention for the values of z (z-axis pointing down), the data recovered from it shows a mirrored effect from real life, which can be seen in the video.

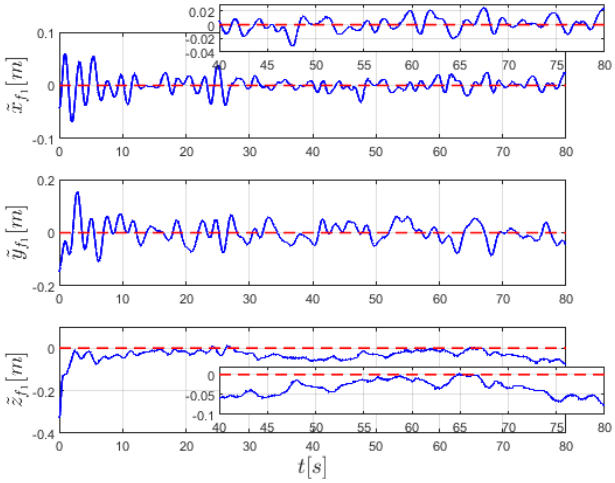

Fig. 13: Formation 1 position errors for the experiment.

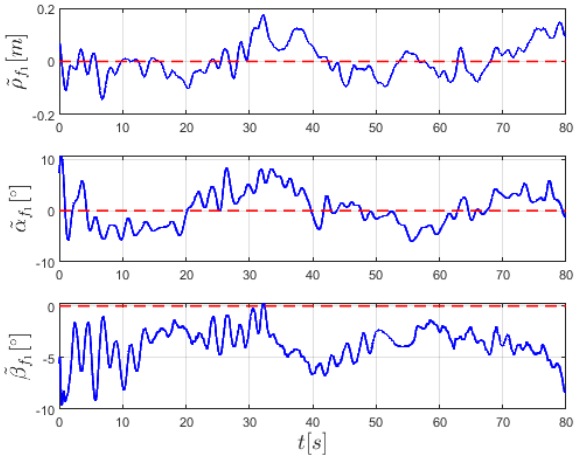

Fig. 14: Formation 1 shape errors for the experiment.

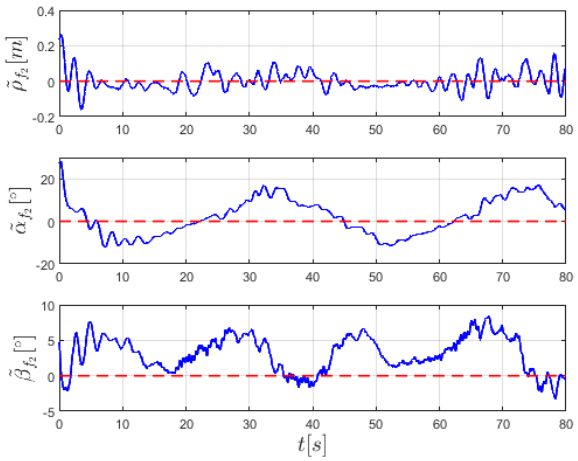

Fig. 15: Formation 2 shape errors for the simulation using Flair.

From the video and the Fig. 12, it is possible to ob-

serve that the formations, and in turn the drones, have accomplished the task of following the desired values of the trajectories. However, it is easy to observe that the errors are higher now, which is to be expected when switching from simulation to real life experiments. It is possible to see that the position errors of the formation, presented in Fig. 13, were kept stabilized, with the errors, once stable movement has been reached, in the $x-$axis not higher than 3 centimeters, in the $y-$axis not higher than 9 centimeters, and in the $z-$axis around 5 centimeters or less. On the other hand, the shape errors, specially the length of the formations ρ_i, are somewhat larger with peaks close to 20 centimeters for the aforementioned ρ_i and 10 degrees for α_i and β_i.

Finally, the errors in the shape of the formation seem to be periodic, which could indicate that they happen when the rate of changes of each of the functions that make up the overall trajectory are working against one another. This could be mitigated in a few ways; finding a better set of gains that could make a better bound for the errors without destabilizing the system, finding trajectories for each of the formations parameters where their movements complement each other in a more natural way, or using techniques that assign a higher priority to a selected task. For instance, a null space based controller could be used to give a higher priority for the task of following the shape for the second formation.

VI. Conclusion

This work proposed an application where two distinct virtual structures interacted in order to create a presentation for entertainment purposes. This was accomplished by defining both formations and their spatial and differential relationship with the robots, as well as the application of two controllers, a kinematic and a dynamic one, which were managed through a multilayer interface.

From the simulations and experiment presented, it is possible to conclude that the controllers were broadly successful in guiding the formations in the accomplishment of the tasks. The experiments presented a higher amount of errors, specially in the shape parameters of the formations, and the causes as well as possible solutions have been discussed.

As future work, it is planned to further investigate the solutions proposed, such as gain adjustment, better trajectory rules definition and a null space based controller for managing the different task priorities.

Acknowledgments

The authors thank CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico, a Brazilian agency that supports scientific and technological development – and FAPES - Fundação de Amparo à Pesquisa e Inovação do Espírito Santo, an agency of the State of Espírito Santo, Brazil, that supports scientific and technological development – for financing this work and for the scholarship granted to the first author. They also thank UFES - Universidade Federal do Espírito Santo and UTC - Université de Technologie de Compiègne for the support to the development of the work. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

[1] S. M. S. M. Daud, M. Y. P. M. Yusof, C. C. Heo, L. S. Khoo, M. K. C. Singh, M. S. Mahmood, and H. Nawawi, "Applications of drone in disaster management: A scoping review," *Science & Justice*, vol. 62, no. 1, pp. 30–42, 2022.

[2] H.-W. Choi, H.-J. Kim, S.-K. Kim, and W. S. Na, "An overview of drone applications in the construction industry," *Drones*, vol. 7, no. 8, p. 515, 2023.

[3] R. Kalamkar, M. Ahire, P. Ghadge, S. Dhenge, M. Anarase *et al.*, "Drone and its applications in agriculture," *International Journal of Current Microbiology and Applied Sciences*, vol. 9, no. 6, pp. 3022–3026p, 2020.

[4] Á. Restás, "Drone applications fighting covid-19 pandemic—towards good practices," *Drones*, vol. 6, no. 1, p. 15, 2022.

[5] D. Nar and R. Kotecha, "Optimal waypoint assignment for designing drone light show formations," *Results in Control and Optimization*, vol. 9, p. 100174, 2022.

[6] O. Zerlenga, I. Rosina, V. Cirillo *et al.*, "Once upon a time there were fireworks. the new nocturnal drones light shows." *img Journal*, vol. 4, pp. 402–425, 2021.

[7] K.-C. Weng, S.-T. Lin, C.-C. Hu, R.-T. Soong, and M.-T. Chi, "Multi-view approach for drone light show," *The Visual Computer*, vol. 39, no. 11, pp. 5797–5808, 2023.

[8] W. Chen, J. Liu, and H. Guo, "Achieving robust and efficient consensus for large-scale drone swarm," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 15 867–15 879, 2020.

[9] T. Namerikawa, Y. Kuriki, and A. Khalifa, "Consensus-based cooperative formation control for multiquadcopter system with unidirectional network connections," *Journal of Dynamic Systems, Measurement, and Control*, vol. 140, no. 4, p. 044502, 2018.

[10] Y. LIANG, D. Qi, and Z. Yanjie, "Adaptive leader–follower formation control for swarms of unmanned aerial vehicles with motion constraints and unknown disturbances," *Chinese Journal of Aeronautics*, vol. 33, no. 11, pp. 2972–2988, 2020.

[11] Z. A. Ali, A. Israr, E. H. Alkhammash, and M. Hadjouni, "A leader-follower formation control of multi-uavs via an adaptive hybrid controller," *Complexity*, vol. 2021, pp. 1–16, 2021.

[12] V. P. Bacheti, A. S. Brandão, and M. Sarcinelli-Filho, "Leader-follower ugv-uav formation as control paradigm for package delivery," in *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2022, pp. 772–778.

[13] M. F. S. Rabelo, A. S. Brandão, and M. Sarcinelli-Filho, "Landing a uav on static or moving platforms using a formation controller," *IEEE Systems Journal*, vol. 15, no. 1, pp. 37–45, 2020.

[14] V. S. M. Rosa and E. M. Belo, "Virtual structure formation flight control based on nonlinear mpc," in *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2021, pp. 1383–1390.

[15] H. T. Do, H. T. Hua, M. T. Nguyen, C. V. Nguyen, H. T. Nguyen, H. T. Nguyen, and N. T. Nguyen, "Formation control algorithms for multiple-uavs: a comprehensive survey," *EAI Endorsed Transactions on Industrial Networks and Intelligent Systems*, vol. 8, no. 27, pp. e3–e3, 2021.

[16] A. S. Brandão and M. Sarcinelli-Filho, "On the guidance of multiple uav using a centralized formation control scheme and delaunay triangulation," *Journal of Intelligent & Robotic Systems*, vol. 84, pp. 397–413, 2016.

[17] A. Brandão, P. Castillo, and R. Lozano, "On the guidance of an uav formation applying multi-layer control scheme," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 425–431.

[18] V. P. Bacheti, A. S. Brandão, and M. Sarcinelli-Filho, "Path-following with a ugv-uav formation considering that the uav lands on the ugv," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2020, pp. 488–497.

[19] ——, "A path-following controller for a uav-ugv formation performing the final step of last-mile-delivery," *IEEE Access*, vol. 9, pp. 142 218–142 231, 2021.

[20] A. S. Brandão, M. Sarcinelli Filho, and R. Carelli, "High-level underactuated nonlinear control for rotorcraft machines," in *2013 IEEE International Conference on Mechatronics (ICM)*. IEEE, 2013, pp. 279–285.