Dynamic Modelling and Control of Flying Parallel Robots

Damien Six, Sébastien Briot, Abdelhamid Chriette, Philippe Martinet

Available online at www.sciencedirect.com

ScienceDirect

Control Engineering Practice 00 (2021) 1–16

CEP

Dynamic Modelling and Control of Flying Parallel Robots

Damien Six*[a], Sébastien Briot[b], Abdelhamid Chriette[a], Philippe Martinet[c]

** Corresponding author.*

[a]*École Centrale de Nantes at the Laboratoire des Sciences du Numérique de Nantes (LS2N), UMR CNRS 6004, 44300 Nantes, France*
[b]*Centre National de la Recherche Scientifique (CNRS) at the Laboratoire des Sciences du Numérique de Nantes (LS2N), UMR CNRS 6004, 44300 Nantes, France*
[c]*Inria Sophia Antipolis, 06902 Valbonne, France*

Abstract

This paper deals with the full dynamic modeling and control of a flying architecture called a *flying parallel robot* (FPR). This architecture, which can be seen as a parallel robot whose actuators have been replaced by drones, offers novel possibilities for robotic and aerial manipulation. Over the last decade, many prototypes have been developed in the field of aerial manipulation, like attaching a gripper or a manipulator to the drone. Nevertheless, the proposed approaches suffer from several drawbacks such as a limitation of payload, autonomy constraints and also a manipulability impacted by the quadrotor underactuation, if standard underactuated quadrotors are used. To overcome these limitations, the FPR concept has several advantages: all possible DoF of the end-effector can be controlled; by sharing the efforts over several drones, and by using no additional embedded motors, the payload capability is enhanced. In this paper, the generic dynamic model is established, whatever are the legs topologies and number of drones used. It is shows that a decoupling property of the dynamic model can be established, which can be then exploited for the design of a cascade controller handling the underactuation of the FPR. The proposed modelling approach and control strategy have been applied in order to perform real experiments with a proof-of-concept prototype belonging to the category of flying parallel robots made of three drones and three legs.

Keywords: Unmanned aerial vehicles, parallel robot, Newton's law, Dynamic model, cascade controller.

1. Introduction

Unmanned aerial vehicles (UAVs) have become increasingly used in commercial applications. In particular, UAVs are exploited in entertainment and surveillance applications, and work is currently underway towards adapting their use to construction, inspection, and logistics industries (Ollero et al., 2018). However, the possibility for air vehicles to grasp an object in order to manipulate or transport it can considerably expand the kind of missions and possible applications that can be carried out by these vehicles: for instance, object manipulation, inspection and maintenance of industrial settings, structure construction, or also other outdoor applications. The new field of aerial manipulation, which is at the boundary between robotic manipulators and UAVs, has attracted the attention of researchers over the past decade (Jimenez-Cano et al., 2013; Kondak et al., 2014; Heredia et al., 2014; Zhou et al., 2016; Suarez et al., 2020).

Aerial manipulation is usually obtained with the integration of one or more robotic arms on an unmanned aerial vehicles (UAVs), typically multirotors or autonomous helicopters, in such a way that the manipulator is capable to perform certain operations in workspaces at high altitude or in areas out of the reach for ground robots.

The most basic approach to design this type of robot is simply to attach a gripper or an arm under the aerial vehicle, usually a standard quadrotor UAV (Lindsey et al., 2011; Pounds et al., 2011; Ghadiok et al., 2011; Backus et al., 2014). This approach is not satisfactory because the manipulating capacities are limited due to the under-actuation of this type of vehicle, as well as the limited on-board load and autonomy capacities. In fact, the arm and the associated actuators weight are added to the vehicle. A second approach consists of attaching a manipulating arm to the vehicle (Huber et al., 2013; Kondak et al., 2013; Yang et al., 2014; Danko and Oh, 2014; Ruggiero et al., 2015; Cataldi et al., 2016; Suarez et al., 2017) which offers better manipulating capabilities but accentuates the problem of on-board load and autonomy. The drawbacks of such solutions are a limited payload and autonomy due to the additional embedded motors and also a manipulability impacted by the quadrotor underactuation. Alternatives using fully actuated multirotors have also been explored Ryll et al. (2015); Rajappa et al. (2015); Nikou et al. (2015); Lee et al. (2018), but they require the design of specific flying devices that will not be available off-the-shelf

and have generally a reduced autonomy and an increased system complexity.

To overcome the limits in payload, some studies investigated co-manipulation between several UAVs (Nguyen et al., 2015, 2018; Loianno et al., 2018; Castillo-Zamora et al., 2019; Rossi et al., 2019). Multiple UAVs working in unison have the advantage of improving the payload capacity, achieving more controllable degrees of freedom (DOFs), and exerting better dynamic control over the suspended payload. An increase in the load capacity and maneuverability of aerial payloads could have applications in firefighting, inspection and repair of transmission lines, and a number of other fields. Beyond new applications, multi-UAV collaboration is economically desirable because of its inherent modularity and reconfigurability. It allows the division of resources (Anzai et al., 2017; Zhao et al., 2018) and, according to (Bernard et al., 2011), "the costs for two small helicopters are often less than for one with double load capacity".

With the advances in UAV technology, recent work has been undertaken to study the use of multiple vehicle actuation of a payload using cables, with the goal of replacing manned helicopters in existing fields and extend the hitherto inaccessible benefits of aerial manipulation into new sectors (Michael et al., 2011; Sreenath and Kumar, 2013; Erskine et al., 2019; Jamshidifar and Khajepour, 2020). However, cables cannot apply any pushing forces to the environment, limiting the manipulation area under the drones. Such flying devices can also be seen as cable-driven parallel robots where the actuators have been replaced by quadrotors.

Inspired by this analogy, a new architecture of flying parallel robot (FPR) using rigid links instead of cables has been proposed in (Six et al., 2018). The use of rigid links requires a new computation of the dynamic model to be able to implement an appropriate control law. The study in (Six et al., 2018) has however been limited to only one specific architecture and no experiments were performed.

The main advantages of FPR architecture are described below:

- A control of all possible DoFs of the end-effector;
- An enhanced payload capability, by sharing the efforts between several drones;
- No additional embedded motors, the robot is controlled only by the quadrotors;
- The end-effector can be placed away from the drone, limiting the perturbations due to rotor wash;
- A large choice of leg topologies, issued from the study of parallel robots, can lead to several properties of interest.

In this paper, the work initiated in (Six et al., 2018) is extended to the generic case of a flying parallel robot composed of n drones. The main contributions of the paper are:

- The proposition of a dynamic model for a generic FPR, i.e. whatever are the legs topologies, and whatever is the number of drones used. While the work (Six et al., 2018) investigates the dynamic model of a single particular case study, the results are generalized by showing that this generic model can always be separated into two subsystems, one describing the dynamics of the passive architecture of the flying robot and the other describing the dynamics of the orientation coordinates of the drones (Section 3).
- The design of a cascade controller handling the underactuation of the FPR and exploiting the properties of the dynamic model separated into two subsystems (Section 4). Here, the main difference with respect to (Six et al., 2018) is the definition of an orientation control law based on quaternions instead of Euler angles, thus avoiding any type of representation singularity.
- The application of the previous modelling approaches and control strategies for performing real experiments of trajectory tracking with a proof-of-concept prototype of flying parallel robots made of three drones, developed at LS2N (Section 5).

The rest of the paper is organized as follows. Section 2 describes the geometric and kinematics modelling of a generic FPR and highlights all the parameterization used in the paper. Section 3 describes the dynamic model of FPRs by computing the dynamics of the passive architecture of the flying robot and the dynamics of the orientation coordinates of the drones. The control law design is presented in Section 4 and the experimental results are presented in Section 5. Finally, conclusions are drawn in Section 6.

2. Geometric and kinematics modelling of generic flying parallel robots

In this section, the kinematic parameters of a flying parallel robot are defined and its kinematic model is introduced.

2.1. Parametrization

In the scope of this study, a flying parallel robot is a multibody mechanism composed of (Fig. 1):

- A rigid platform named $\mathcal{B}_p$, which is considered as the robot end-effector.
- n legs. A leg i is denoted by $\mathcal{L}_i$, with $i = 1, \ldots, n$, and is composed of n_i rigid bodies, denoted $\mathcal{B}_{i,j}$, $j = 1, \ldots, n_i$. The bodies are linked by passive joints, such that they form a kinematic chain. For each leg, the body $\mathcal{B}_{i,n_i}$ is named the *last body* of the leg.
- n flying drones. Each drone i, $i = 1, \ldots, n$, is denoted by $\mathcal{D}_i$. In what follows, all drones will be considered to be quadcopters, but any other kind of drones could be used (e.g. hexarotors).

Each leg $\mathcal{L}_i$ links the platform to a drone as follows. The leg $\mathcal{L}_i$ is attached:

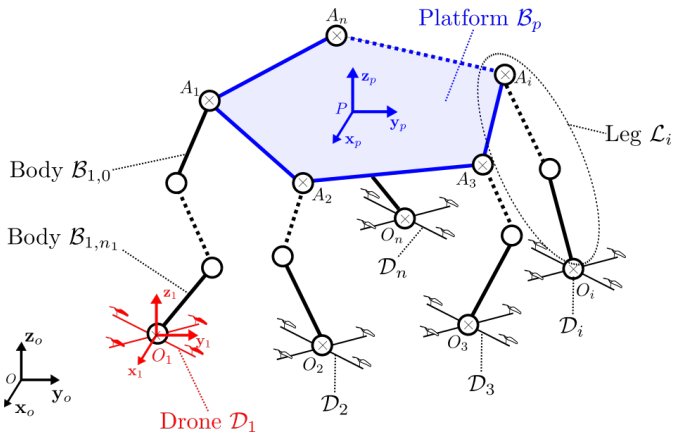

Figure 1. General scheme of a flying parallel robot

- to the platform $\mathcal{B}_p$ at point A_i by the body $\mathcal{B}_{i,0}$ with a passive joint, that could be of any type.
- to the drone $\mathcal{D}_i$ center of mass (CoM) at O_i through the body $\mathcal{B}_{i,n_i}$ with a passive *spherical* joint. This joint allows a free rotation of the quadrotor at the leg tip. As a consequence, no torque applied by the drones at their CoM is transferred to the leg. This choice of design may be technically challenging but is mandatory to obtain a decoupling between the drone rotation dynamics and the passive architecture dynamics. This decoupling is a key component of the control law design (see section 4). Other joints could be proposed for the connection between a drone and a leg, but they would add a considerable complexity to the controller design, and are therefore considered to be out of the scope of this paper.

In what follows, the *passive architecture* refers to the tree structure composed by the platform, associated to the legs, without the quadcopters and the associated spherical joints.

The following frames are also defined in the form $\mathcal{F}(O, \mathbf{x}, \mathbf{y}, \mathbf{z})$, with O being the origin of the frame and $\mathbf{x}, \mathbf{y}, \mathbf{z}$ its three main axes:

- $\mathcal{F}_w(O_w, \mathbf{x}_w, \mathbf{y}_w, \mathbf{z}_w)$ a reference world frame.
- $\mathcal{F}_p(O_p, \mathbf{x}_p, \mathbf{y}_p, \mathbf{z}_p)$ a frame attached to the platform $\mathcal{B}_p$.
- $\mathcal{F}_i(O_i, \mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i)$ a frame attached to the quadcopter $\mathcal{D}_i$ at its CoM.
- $\mathcal{F}'_i(O_i, \mathbf{x}'_i, \mathbf{y}'_i, \mathbf{z}'_i)$ a frame attached to the last body $\mathcal{B}_{n_i}$ of each leg at point O_i.

The position and orientation of any frame $\mathcal{F}$ relative to the world frame $\mathcal{F}_w$ are defined by the position of its origin in the world frame given by $\boldsymbol{\xi}^T = \begin{bmatrix} x & y & z \end{bmatrix}$ and a unit quaternion $\mathbf{h} \in \mathbb{H}$ allowing a representation of the rotations without any singularity (see (Campa and Camarillo, 2008; Kuipers, 1999) and appendix Appendix A.1). Subsequently, the variables $\boldsymbol{\xi}_p$ and $\mathbf{h}_p$ ($\boldsymbol{\xi}_i$ and $\mathbf{h}_i$, $\boldsymbol{\xi}'_i$ and $\mathbf{h}'_i$, resp.) will parameterize the position and orientation of the frame $\mathcal{F}_p$ ($\mathcal{F}_i$, $\mathcal{F}'_i$, resp.) with respect to the world frame $\mathcal{F}_w$.

Table 1. Kinematic parameters of a flying parallel robot

Element	Frame	Coordinates	Velocity
Platform (end effector)	$\mathcal{F}_p$	$\boldsymbol{\xi}_p$ $\mathbf{h}_p$	$\dot{\boldsymbol{\xi}}_p$ $\boldsymbol{\omega}_p$
Leg i	$\mathcal{F}'_i$ (last body)	$\mathbf{q}_i$	$\dot{\mathbf{q}}_i$
Drone i	$\mathcal{F}_i$	$\boldsymbol{\xi}_i$ $\mathbf{h}_i$	$\dot{\boldsymbol{\xi}}_i$ $\boldsymbol{\omega}_i$

The passive architecture configuration is then defined by a vector $\mathbf{q} = [\boldsymbol{\xi}_p^T \ \mathbf{h}_p^T \ \mathbf{q}_1^T \ \dots \ \mathbf{q}_n^T]^T$ composed of the following coordinates:

- The position $\boldsymbol{\xi}_p^T = \begin{bmatrix} x_p & y_p & z_p \end{bmatrix}$ of the platform $\mathcal{B}_p$.
- The coordinates of the unit quaternion $\mathbf{h}_p$ defining the platform frame $\mathcal{F}_p$ orientation.
- All passive joint coordinates: for each leg $\mathcal{L}_i$ ($i = 1, \dots, n$), the joint coordinates are grouped in the vector denoted $\mathbf{q}_i$. The coordinates of the spherical joint linking the leg to its associated quadrotor are not included in this vector.

The variable n_q defines the number of degrees of freedom of the passive architecture, i.e. the kinematic chain without the drones. The vector $\mathbf{q}$ contains $n_q + 1$ components because the quaternion defining the platform orientation is composed of four (dependent) coordinates.

For what follows, $\mathbf{v}_q$ denotes the velocity vector composed of the following n_q components:

$$\mathbf{v}_q^T = \begin{bmatrix} \dot{\boldsymbol{\xi}}_p^T & \boldsymbol{\omega}_p^T & \dot{\mathbf{q}}_1^T & \dots & \dot{\mathbf{q}}_n^T \end{bmatrix} \tag{1}$$

where $\boldsymbol{\omega}_p$ denote the angular velocity of the platform frame $\mathcal{F}_p$.

The Table 1 provides an overall view of the parameters introduced in this section.

2.2. Geometric and kinematic models of the passive architecture

In order to study the robot from a kinematic point of view, it is necessary to obtain the relations between the position and velocity of the drones CoM relatively to the configuration and coordinates velocity of the passive architecture. Some details are provided here on the computation of the geometric and kinematic models.

2.2.1. Inverse geometric model

The inverse geometric model is the model expressing the Cartesian coordinates $\boldsymbol{\xi}_i$ of each point O_i, i.e. the drone $\mathcal{D}_i$'s CoM, as a function of the passive architecture coordinate vector $\mathbf{q}$. This model is provided by $3n$ equations given by:

$$\boldsymbol{\xi}_i = \boldsymbol{\xi}_p + \mathbf{r}_{PA_i} + \mathbf{r}_{A_iO_i} \quad i = 1, \dots, n \tag{2}$$

where $\mathbf{r}_{AB} = \overrightarrow{AB}$ for any points A and B. $\mathbf{r}_{PA_i}$ is a constant in the platform frame, and $\mathbf{r}_{A_iO_i} = \mathbf{r}_{A_iO_i}(\mathbf{q})$ is a function of the

passive architecture coordinates $\mathbf{q}$. This analytical expression can be obtained by using the conventional method for computing the forward geometric model of any tree structure robot, as explained for instance in (Khalil and Dombre, 2002). Based on this approach, Eq. (2) can be rewritten under the generic form:

$$\boldsymbol{\xi}_i = \mathbf{f}_i(\mathbf{q}) \tag{3}$$

2.2.2. Direct geometric model

The direct geometric model gives the configuration of the passive architecture as a function of the drones position in space.

$$\mathbf{q} = \mathbf{f}(\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_n) \tag{4}$$

As for any parallel robot, the analytical computation of the direct geometric model is not straightforward, even potentially impossible, and will require a specific study for each architecture (Merlet, 2004). Some numerical schemes based on Newton-Raphson algorithm or interval analysis, such as the ones proposed in (Merlet, 2004), could be defined or adapted. However, in the experimental setup (see Section 5), the platform pose is directly measured by exteroceptive sensors. Therefore, providing more details on the computation of the direct geometric model is out of the scope of this work.

2.2.3. Kinematic model

The kinematic model of the flying parallel robot describes the relation between the velocity of the drones CoM at O_i and the vector of the coordinates velocity $\mathbf{v}_q$. The velocity $\dot{\boldsymbol{\xi}}_i$ of each drone CoM O_i is given by the derivative with respect to time of equation (2):

$$\dot{\boldsymbol{\xi}}_i = \dot{\boldsymbol{\xi}}_p + \dot{\mathbf{r}}_{PA_i} + \dot{\mathbf{r}}_{A_iO_i} \tag{5}$$

For each leg $\mathcal{L}_i$, by using the conventional methododology for computing the forward kinematic model of any tree structure robot, as explained for instance in (Khalil and Dombre, 2002), it is possible to define the Jacobian kinematic matrix $\mathbf{J}_i$ giving the twist ${}^{A_i}\mathbf{t}_i$ of the frame $\mathcal{F}_i'$ relative to the point A_i as a linear function of the leg velocities $\dot{\mathbf{q}}_i$:

$${}^{A_i}\mathbf{t}_i = \begin{bmatrix} \dot{\mathbf{r}}_{A_iO_i} \\ {}_{A_i}\boldsymbol{\omega}_i \end{bmatrix} = \mathbf{J}_i\dot{\mathbf{q}}_i = \begin{bmatrix} \mathbf{J}_{\dot{r},i} \\ \mathbf{J}_{\omega,i} \end{bmatrix} \dot{\mathbf{q}}_i \tag{6}$$

where $\mathbf{J}_{\dot{r},i}$ (respectively $\mathbf{J}_{\omega,i}$) is the the sub-matrix of the Jacobian $\mathbf{J}_i$ relative to the translational velocity (respectively the angular velocity) of the frame $\mathcal{F}_i'$.

Introducing (6) into (5), then recalling that $\mathbf{r}_{PA_i}$ is constant in the platform frame, and using the fact that $\dot{\mathbf{r}}_{PA_i} = \boldsymbol{\omega}_p \times \mathbf{r}_{PA_i}$, the following expression is obtained for the leg i:

$$\dot{\boldsymbol{\xi}}_i = \dot{\boldsymbol{\xi}}_p + \boldsymbol{\omega}_p \times \mathbf{r}_{PA_i} + \mathbf{J}_{\dot{r},i}\dot{\mathbf{q}}_i \tag{7}$$

Now, by stacking the equations of all legs, the complete kinematic model of the passive architecture is obtained under the following matrix form:

$$\mathbf{v} = \mathbf{J}\mathbf{v}_q \tag{8}$$

where $\mathbf{v}^T = \begin{bmatrix} \dot{\boldsymbol{\xi}}_1^T & \ldots & \dot{\boldsymbol{\xi}}_n^T \end{bmatrix}$ is the vector of all drone CoM velocities and:

$$\mathbf{J} = \begin{bmatrix} \mathbb{1}_3 & -\mathbf{r}^{\times}_{O_pA_1} & \mathbf{J}_{\dot{r},1} & \mathbb{0} & \ldots & \mathbb{0} \\ \mathbb{1}_3 & -\mathbf{r}^{\times}_{O_pA_2} & \mathbb{0} & \mathbf{J}_{\dot{r},i} & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \mathbb{0} \\ \mathbb{1}_3 & -\mathbf{r}^{\times}_{O_pA_n} & \mathbb{0} & \ldots & \mathbb{0} & \mathbf{J}_{\dot{r},n} \end{bmatrix} \tag{9}$$

where $\mathbf{r}^{\times}$ is the cross product matrix of the vector $\mathbf{r}$ defined by:

$$\mathbf{r}^{\times}\boldsymbol{\omega} = \mathbf{r} \times \boldsymbol{\omega} = -\boldsymbol{\omega} \times \mathbf{r}$$

for any vectors $\mathbf{r}$ and $\boldsymbol{\omega}$.

$\mathbf{J}$ is a $(3n \times n_q)$ matrix. The dimensions and rank of the matrix $\mathbf{J}$ can be interpreted, from a kinematic point of view, as follows:

- If $\text{rank}(\mathbf{J}) < n_q$, then there is not a unique solution to the inverse kinematic model. The robot will have one or several uncontrolled degrees of freedom, which is out of the scope of this study. This situation arises if $3n < n_q$ or if the matrix $\mathbf{J}$ is singular. Finding singularities of this matrix is also out of the scope of this paper. However, for the interested reader, an example of a study of the singularities of a flying parallel robot with three drones and three legs is proposed in (Six et al., 2018).
- If $\text{rank}(\mathbf{J}) = n_q$, then any solution to the inverse kinematic model is unique and there is no uncontrolled degrees of freedom, assuming that the robot can be assembled. Note that in this case, there is $3n \geq n_q$. If $3n = n_q$ then the solution is guaranteed, while if $3n > n_q$ there are more coordinates in the vector $\mathbf{x}^T_{drone} = [\mathbf{x}_1^T \ldots \mathbf{x}_n^T]$ of all drone CoM than degrees of freedom in the passive architecture. This case is similar parallel robots encountering overactuation (Merlet, 2006).

The scope of this study is restrained to the case $\text{rank}(\mathbf{J}) = n_q$. There is as much degrees of freedom in all drone translations than in the passive architecture.

The computation of this Jacobian matrix has its importance both in the computation of the kinematics of the flying parallel robot but also for its dynamic model as the transpose of this Jacobian matrix links the forces applied at the leg extremity to the efforts in the passive architecture as it will be shown in the next section that develops the computation of the dynamic model for the generic flying parallel robot.

3. Dynamic model of flying parallel robots

As mentioned in section 2.1, the following hypothesis is made: the last body of the leg $\mathcal{L}_i$ is considered to be attached to the drone $\mathcal{D}_i$ CoM with a passive spherical joint (for $i = 1, \ldots, n$). This gives the following properties for the dynamics of the robot:

- No torques are transmitted between the passive architecture and the drones, only interaction forces.

- The interaction forces between the passive architecture and each drone are applied at the drone CoM.

As a consequence, the drones rotations are not affected by the passive structure and remains similar as if the drones were moving in free air. Also, the passive architecture dynamics is only affected by the forces acting at the drones CoM, i.e. the thrust force, the weight, and the acceleration of the drone.

Considering those properties, the dynamic model of the robot is split in two parts. A first part considers the dynamic model of the quadrotors rotations. Then, a second part focuses on the passive architecture under the influence of the interaction forces with each drone.

For the computation of the full dynamic model of the FPR, the hypothesis of a quasi-stationary flight configuration of the robot is considered. This hypothesis, widely adopted in the literature (see (Mahony et al., 2012), for example), will allow us to neglect several friction and aerodynamic effects in the calculation of the dynamic model, effects which will be partially compensated by the controller (see section 4). In section 5, it will be shown that the controller remains stable even in the context of dynamic trajectory tracking.

3.1. Quadrotor rotation dynamics

In the scope of this study, all the drones are assumed to be in a near hovering configuration. The propellers inertia and ground and ceiling effects are also neglected. Given these standard hypotheses (see for instance (Bouabdallah et al., 2004; Bristeau et al., 2009; Mahony et al., 2012)), the resultant forces and torques provided by the four propellers of a quadrotor are composed of a thrust force aligned with the propeller axes (in this study, $\mathbf{z}_i$ for the drone $\mathcal{D}_i$) and three moments that allow a full control of the three rotations of the quadrotor. Those forces are linearly mapped to the square of each propeller angular velocity (Mahony et al., 2012). Thus, the thrust $\mathbf{f}_i = f_i\mathbf{z}_i$ and three independent moments $\mathbf{m}_i$ will be considered as the drone inputs for control purpose.

Because there is no torque transmitted via the spherical joints between the passive architecture and the drone, the dynamic equation of the drone rotation around its CoM is obtained by straightforwardly applying the Newton's law to the drone $\mathcal{D}_i$:

$$\mathbf{m}_i = \mathbf{\Sigma}_i\dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \mathbf{\Sigma}_i\boldsymbol{\omega}_i \tag{10}$$

where $\mathbf{\Sigma}_i$ and $\boldsymbol{\omega}_i$ are the inertia matrix and the angular velocity of the drone $\mathcal{D}_i$ about its CoM O_i, respectively.

It is observed that, in (10), there is no variable related to the passive architecture kinematics or dynamics. Therefore, the dynamics of the drone rotations is not affected by the passive architecture. This result will be exploited in Section 4 for the development of a cascaded controller for the flying parallel robot.

In the next section the dynamic model of the passive architecture is computed.

3.2. Dynamic model of the passive architecture

The passive architecture of the flying parallel robot is actuated by the total interaction force $\mathbf{f}_{d,i}$ $(i = 1, ..., n)$ produced by the drone at O_i and acting on the last body of each leg (Fig: 2a). In order to compute the dynamic model of the passive architecture, three steps will be followed:

- First, the dynamic model of a virtual architecture, which is similar to the passive architecture, except that all joints and the platform will be considered virtually actuated (Fig: 2b), will be developed (Section 3.2.1).
- Then, in order to link the dynamics of this virtual architecture to the dynamics of the real robot, actuated by the interaction forces $\mathbf{f}_{d,i}$ between the passive architecture and the drones, the principle of virtual powers (Khalil and Dombre, 2002) will be used (Section 3.2.2).
- Finally, the interaction forces $\mathbf{f}_{d,i}$ will be expressed as a function of the drone thrust forces $\mathbf{f}_i$ in order to obtain the complete dynamic model of the passive architecture (Section 3.2.3).

3.2.1. Dynamic model of the virtual architecture

The virtual architecture shown in Fig. 2b is similar to the passive architecture, except that it is virtually considered that it is a fully actuated free floating tree structure with all joints being active. The vector $\boldsymbol{\tau_e}$ is defined as the vector grouping all the efforts on this virtual architecture composed of:

- A virtual 6-dimensional wrench acting on the platform.
- A vector stacking all virtual wrenches providing actuation on each joint. The dimension of this vector is corresponding to the total number of degrees of freedom in the joints.

The dynamic model of the fully actuated virtual tree architecture takes the general form (Khalil, 2010):

$$\boldsymbol{\tau}_e = \mathbf{M}_e(\mathbf{q})\dot{\mathbf{v}}_q + \mathbf{c}_e(\mathbf{q}, \mathbf{v}_q) \tag{11}$$

where $\mathbf{M}_e(\mathbf{q})$ is the $(n_q \times n_q)$ generalized inertia matrix and $\mathbf{c}_e(\mathbf{q}, \mathbf{v}_q)$ the n_q-dimensional vector of Coriolis, centrifugal and gravitational effects. As explained in the introduction of this section, the system is considered in a near-hovering configuration. Then the friction effects, either aerodynamic or in the passive joints, are negligible compared to actuation forces and, therefore, are not considered in this model. Details for computing the matrix $\mathbf{M}_e$ and the vector $\mathbf{c}_e$ for any tree structure robot are provided in (Khalil, 2010) and are not recalled here for reasons of brevity.

3.2.2. Computation of the drone interaction forces

The interaction wrench applied at each point O_i by the drone $\mathcal{D}_i$ to the real passive architecture is denoted $\mathbf{w}_{d,i}$. In order to compute how those wrenches affect the dynamics of the real passive architecture, an arbitrary virtual displacement of all points O_i is considered. Then, the Principle of Virtual Powers is used to link the interaction wrench to the virtual efforts $\boldsymbol{\tau}_e$ applied in the virtual fully actuated architecture.

Since each drone $\mathcal{D}_i$ is linked to the passive architecture through a spherical joint in point O_i (the friction in the joint is

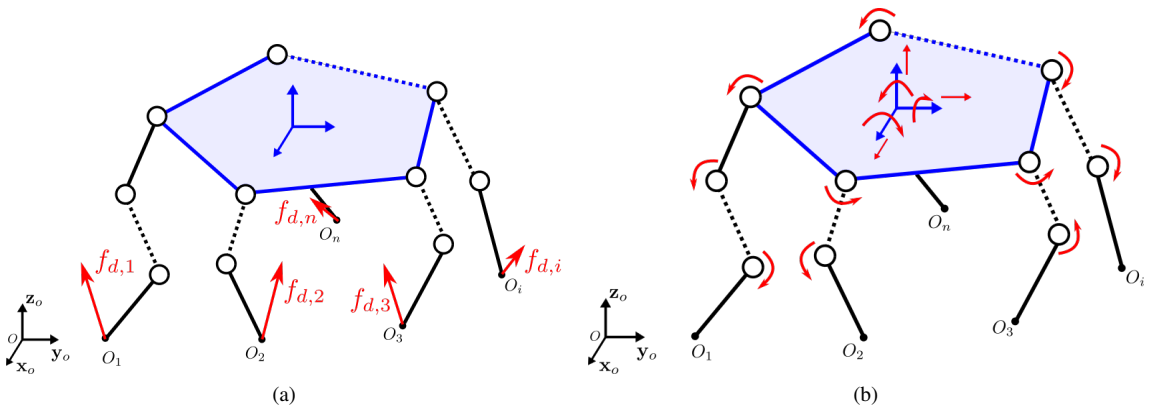

Figure 2. The passive architecture actuated by the interaction forces with the drones (a) and an equivalent fully actuated tree structure (b). In these pictures, the red arrows stand for the actuation forces or torques of the passive architecture.

neglected), it is possible to consider that the interaction wrenches are pure forces at each point O_i. Let us consider an arbitrary virtual velocity ${\mathbf{v}_q}^*$ of the passive architecture leading to a virtual velocity $\mathbf{v}_i^*$ of the center of mass of each drone O_i. The principle of virtual powers can be written in the virtual equivalent free-floating tree-structure as:

$$\mathcal{P}_p + \mathcal{P}_{\tau_e} = 0 \tag{12}$$

where

- $\mathcal{P}_p$ is the power developed by the inertial effect and the gravitational forces on the passive architecture.
- $\mathcal{P}_{\tau_e}$ is the power developed by the virtual joint actuation in the virtual equivalent free-floating tree-structure.

Coming back now to the real robot, the principle of virtual powers gives:

$$\mathcal{P}_p + \mathcal{P}_i = 0 \tag{13}$$

where $\mathcal{P}_i$ is the power developed by the interaction forces between the drones and the passive architecture. Equations (12) and (13) allow us to write the equivalence between the power provided by the interaction forces $\mathbf{f}_{d,i}$ applied to the passive architecture and the power provided by the virtual actuation $\boldsymbol{\tau}_e$ in the virtual architecture, giving

$$\sum_{i=1}^{n} {\mathbf{v}_i^*}^T \mathbf{f}_{d,i} = {\mathbf{v}_q}^{*T} \boldsymbol{\tau}_e \tag{14}$$

Equation (14) can be rewritten as

$${\mathbf{v}^*}^T \mathbf{f}_d = {\mathbf{v}_q}^{*T} \boldsymbol{\tau}_e \tag{15}$$

with

- ${\mathbf{v}^*}^T = \begin{bmatrix} {\mathbf{v}_1^*}^T & \dots & {\mathbf{v}_n^*}^T \end{bmatrix}$
- $\mathbf{f}_d^T = \begin{bmatrix} \mathbf{f}_{d,1}^T & \dots & \mathbf{f}_{d,n}^T \end{bmatrix}$

Introducing the kinematic relation (8) in (15) leads to

$$\mathbf{v}_q^{*T} \mathbf{J}^T \mathbf{f}_d = \mathbf{v}_q^{*T} \boldsymbol{\tau}_e \tag{16}$$

This relation is valid for any virtual velocity $\mathbf{v}_q^*$ which leads, by identification of both sides of the equation (16), to the following relationship:

$$\mathbf{J}^T \mathbf{f}_d = \boldsymbol{\tau}_e \tag{17}$$

Introducing (17) into the dynamic model (11) leads to

$$\mathbf{J}^T \mathbf{f}_d = \mathbf{M}_e(\mathbf{q})\dot{\mathbf{v}}_q + \mathbf{c}_e(\mathbf{q}, \mathbf{v}_q) \tag{18}$$

3.2.3. *Final form of the dynamic model*

Recall that in the scope of this study, it is considered that the drones are only near hovering case. Thus, the aerodynamic forces acting on each body $\mathcal{D}_i$ such as aerodynamic friction are neglected. For each leg $\mathcal{L}_i$, the interaction force applied in point O_i can be obtained computing the force resultant applied on the drone $\mathcal{D}_i$ (Mahony et al., 2012):

$$\mathbf{f}_i - \mathbf{f}_{d,i} + m_i \mathbf{g} = m_i \ddot{\boldsymbol{\xi}}_i \tag{19}$$

where

- $\mathbf{f}_i$ is the actuation force applied by the propellers (see section 3.1).
- m_i is the mass of the drone $\mathcal{D}_i$.
- $\mathbf{g}$ is the gravity vector.

The expressions (19) can be grouped for all the drones, giving

$$\mathbf{f}_d = \mathbf{f} + \mathbf{M}_d \mathbf{g}_n - \mathbf{M}_d \dot{\mathbf{v}} \tag{20}$$

where

- $\mathbf{M}_d$ is a $(3n \times 3n)$ diagonal matrix which takes the form

$$\mathbf{M}_d = \begin{bmatrix} m_1 \mathbb{1}_3 & & \mathbb{0} \\ & \ddots & \\ \mathbb{0} & & m_n \mathbb{1}_3 \end{bmatrix}$$

- $\mathbf{f} = \begin{bmatrix} \mathbf{f}_1^T & \dots \mathbf{f}_n^T \end{bmatrix}^T$
- $\mathbf{g}_n$ is the gravity vector expressed n times $\mathbf{g}_n = \begin{bmatrix} \mathbf{g}^T \dots \mathbf{g}^T \end{bmatrix}^T$.

Introducing the derivative of the kinematic model (8) into (20) allows the computation of the vector $\mathbf{f}_d$ grouping all interaction forces. The expression of $\mathbf{f}_d$ becomes then a function of the drone actuation forces $\mathbf{f}$, the passive architecture coordinates $\mathbf{q}$, velocities $\mathbf{v}_q$ and accelerations $\dot{\mathbf{v}}_q$:

$$\mathbf{f}_d = \mathbf{f} + \mathbf{M}_d\mathbf{g}_n - \mathbf{M}_d(\mathbf{J}(\mathbf{q})\dot{\mathbf{v}}_q + \dot{\mathbf{J}}(\mathbf{q}, \mathbf{v}_q)\mathbf{v}_q) \tag{21}$$

Introducing (21) into the dynamic model (18) gives then the expression of the relation between the drone actuation forces $\mathbf{f}$ and the passive architecture coordinates $\mathbf{q}$, velocities $\mathbf{v}_q$ and accelerations $\dot{\mathbf{v}}_q$:

$$\mathbf{J}^T\mathbf{f} = (\mathbf{M}_e + \mathbf{J}^T\mathbf{M}_d\mathbf{J})\dot{\mathbf{v}}_q + \mathbf{c}_e + \mathbf{J}^T\mathbf{M}_d(\dot{\mathbf{J}}\mathbf{v}_q - \mathbf{g}_n) \tag{22}$$

The expression (22) of the dynamic model of the passive architecture can be then rewritten under the form

$$\mathbf{J}^T\mathbf{f} = \mathbf{M}(\mathbf{q})\dot{\mathbf{v}}_q + \mathbf{c}(\mathbf{q}, \mathbf{v}_q) \tag{23}$$

where $\mathbf{M} = (\mathbf{M}_e + \mathbf{J}^T\mathbf{M}_d\mathbf{J})$ and $\mathbf{c} = \mathbf{c}_e + \mathbf{J}^T\mathbf{M}_d(\dot{\mathbf{J}}\mathbf{v}_q - \mathbf{g}_n)$. Equation (23) is the dynamic model of the flying parallel robot.

This structure of the dynamic model is close to the form of the dynamic model of conventional robots. This structure will inspire the feedback linearization strategy for the control law of the passive architecture developed in section 4.

4. Control law

In this part dedicated to the control strategy, several layers of control will be defined using auxiliary inputs. Thus, an auxiliary input of the control law associated to a variable x, resp. a vector $\mathbf{x}$, is denoted by α_x, resp. $\boldsymbol{\alpha}_\mathbf{x}$.

The aim of the control law design is to obtain the convergence of the passive architecture coordinates $\mathbf{q}$ towards a desired configuration ${}^d\mathbf{q}$. In section 2.2, the hypothesis of a full rank Jacobian matrix $\mathbf{J}$ is made. Then, given the dynamic model of the passive architecture (23), the flying robot can be controlled through the vector $\mathbf{f}$ grouping all the actuation forces provided by the drones. In this case, to be fully actuated, the flying parallel robot would require the action of three independent forces at each point O_i. However, each quadrotor $\mathcal{D}_i$ is underactuated and can only provide a single unidirectional force in the direction of its axis $\mathbf{z}_i$.

To handle this underactuation, the strategy that is proposed is similar to some approaches developed in (Bouabdallah et al., 2004; Castillo et al., 2005; Voos, 2009; Das et al., 2009; Lee et al., 2010; Mahony et al., 2012), applied to classic quadrotors control. The principle of control in this case consists in separating the translational control from the rotational control. Indeed, in all cases, a first control law determines the vertical thrust and the orientation necessary to stabilize the translation dynamics. This orientation is then considered as the reference to be followed by a second control law that stabilizes the rotation dynamics. Thus, the outputs of the high level control serve as set points for the low level control, hence the notion of hierarchical control. It is interesting to note that each control law can be developed separately. The main difficulty of this control structure is then to demonstrate the stability of the overall closed-loop system, to ensure that the actuators are not saturated and to guarantee good performance and robust behavior with respect to disturbances.

Applying a similar approach for the Flying Parallel Robot, the main point of the control strategy is to use a cascaded control (see Fig. 3). An outer loop computes the required force α_{f_i} and orientation $\boldsymbol{\alpha}_{\mathbf{h}_i}$ of each drone $\mathcal{D}_i$ to obtain the convergence of the passive architecture coordinates $\mathbf{q}$ toward the trajectory ${}^d\mathbf{q}$. This outer loop contains a PD controller (1) associated to a dynamic model (2) that computes the desired force vector $\boldsymbol{\alpha}_\mathbf{f}$ to be provided by the drones (see Fig. 3). As the thrust provided by each drone is unidirectional, the appropriate drone orientation $\boldsymbol{\alpha}_{\mathbf{h}_i}$ must be also computed (3). Then, on each drone, an internal control loop (4) ensures that it provides the appropriate thrust with the desired orientation.

The designed control law will be presented using the following approach. First, the control law of the passive architecture will be expressed assuming that the drones can provide forces in any direction. Then the cascaded control strategy for the control of the overall system will be developed.

4.1. Passive architecture control law

This section describes the control approaches used in Blocks (1) and (2) of Fig. 3.

4.1.1. Control of the orientation errors

As the quaternions, used for the representation of orientations have four dependant coordinates, a specific control law must be established to ensure the convergence of the orientation coordinates. Let us denote as $\mathbf{h}_i^T = \begin{bmatrix}\eta_i & \boldsymbol{\epsilon}_i^T\end{bmatrix}$ a unit quaternion, where η_i is its scalar part and $\boldsymbol{\epsilon}_i$ its vector part. As mentioned in appendix Appendix A.1, the Hamilton product of $\mathbf{h}_1$ by its inverse $\bar{\mathbf{h}}_1$ is the identity quaternion (see (A.5)). Reciprocally, two quaternions $\mathbf{h}_1$ and $\mathbf{h}_2$ verifying

$$\mathbf{h}_1 \circ \bar{\mathbf{h}}_2 = \begin{bmatrix}1\\ \mathbf{0}\end{bmatrix} \tag{24}$$

represent the same rotation. Thus, the quaternion error $\delta\mathbf{h}(\mathbf{h}_1, \mathbf{h}_2) = \mathbf{h}_1 \circ \bar{\mathbf{h}}_2$ is a natural way to represent the orientation error between two frames. As orientations are represented by unit quaternions, the condition $\delta\boldsymbol{\epsilon}(\mathbf{h}_1, \mathbf{h}_2) = \mathbf{0}$, where $\delta\boldsymbol{\epsilon}$ is the vector part of $\delta\mathbf{h}$, is a sufficient condition to guarantee that $\mathbf{h}_1$ and $\mathbf{h}_2$ represent the same rotation. Therefore, aligning frames which respective orientations are represented by $\mathbf{h}_1$ and $\mathbf{h}_2$ is equivalent to make $\delta\boldsymbol{\epsilon}(\mathbf{h}_1, \mathbf{h}_2)$ converge toward zero, with $\delta\boldsymbol{\epsilon}(\mathbf{h}_1, \mathbf{h}_2)$ obtained using (A.3) in Appendix Appendix A.1 applied to $\mathbf{h}_1 \circ \bar{\mathbf{h}}_2$

$$\delta\boldsymbol{\epsilon}(\mathbf{h}_1, \mathbf{h}_2) = -\eta_1\boldsymbol{\epsilon}_2 + \eta_2\boldsymbol{\epsilon}_1 - \boldsymbol{\epsilon}_1 \times \boldsymbol{\epsilon}_2 \tag{25}$$

Let us now consider a C^2 trajectory of desired orientation of a rigid body ${}^d\mathbf{h}$ and an auxiliary input $\boldsymbol{\alpha}_{\dot{\omega}}$ associated to the body

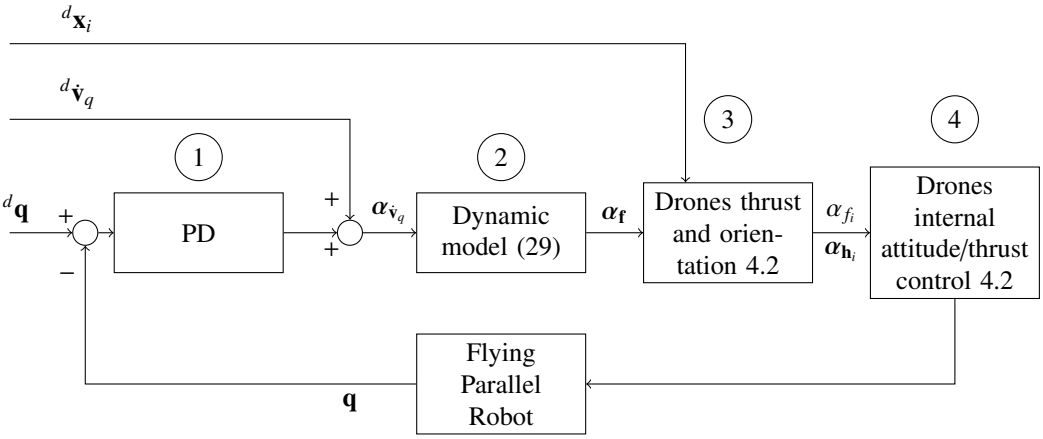

Figure 3. General control scheme for a flying parallel robot

angular acceleration. The following PD control law is defined for controlling the orientation of the considered rigid body:

$$\boldsymbol{\alpha}_{\dot{\omega}} = {}^d\dot{\boldsymbol{\omega}} + {}^h\mathbf{K}_d({}^d\boldsymbol{\omega} - \boldsymbol{\omega}) + {}^h\mathbf{K}_p\delta\boldsymbol{\epsilon}({}^d\mathbf{h}, \mathbf{h}) \tag{26}$$

with

- ${}^d\boldsymbol{\omega}$ and ${}^d\dot{\boldsymbol{\omega}}$ respectively the desired angular velocity and acceleration computed from the trajectory using (A.6).
- ${}^h\mathbf{K}_p$ and ${}^h\mathbf{K}_d$ diagonal positive gains matrices.

It should be noted that the asymptotic stability of this control law is proven in appendix Appendix A.2. Then, the obtained convergence of $\delta\boldsymbol{\epsilon}({}^d\mathbf{h}, \mathbf{h})$ toward zero ensures the alignment of the current frame with the desired one. As a result, this control law on quaternions will be exploited in the PD control law designed for the passive architecture coordinates in the following section.

4.1.2. Passive architecture control law

Let us consider a desired passive architecture configuration ${}^d\mathbf{q}$. The error vector $\mathbf{e}_q$ on the robot configuration $\mathbf{q}$ is defined by:

$$\mathbf{e}_q = \begin{bmatrix} {}^d\mathbf{x}_p - \mathbf{x}_p \\ \delta\boldsymbol{\epsilon}({}^d\mathbf{h}_p, \mathbf{h_p}) \\ {}^d\mathbf{q}_1 - \mathbf{q}_1 \\ \vdots \\ {}^d\mathbf{q}_n - \mathbf{q}_n \end{bmatrix} \tag{27}$$

The design of the control law of the passive architecture is based on feedback linearization. An auxiliary input $\boldsymbol{\alpha}_{\dot{\mathbf{v}}_q}$ associated to the passive coordinate acceleration will be first considered. The PD control law, corresponding to the block (1) in Fig. 3, is defined as

$$\boldsymbol{\alpha}_{\dot{\mathbf{v}}_q} = {}^d\dot{\mathbf{v}}_q + \mathbf{K}_d({}^d\mathbf{v}_q - \mathbf{v}_q) + \mathbf{K}_p\mathbf{e}_q \tag{28}$$

with:

- ${}^d\mathbf{v}_q$ and ${}^d\dot{\mathbf{v}}_q$ respectively the desired coordinate velocities and accelerations.
- $\mathbf{K}_p$ and $\mathbf{K}_d$ diagonal positive gains matrices.

This control law leads to a convergence of all the coordinates when $\dot{\mathbf{v}}_q = \boldsymbol{\alpha}_{\dot{\mathbf{v}}_q}$:

- for the quaternion orientation coordinates as discussed in section 4.1.1.
- for the other coordinates as it leads to a dynamic equation under the form of a second order ordinary differential equation.

Then, the auxiliary input related to the propeller thrusts $\boldsymbol{\alpha}_\mathbf{f}$ is computed using the inverse dynamic model of the passive architecture (23). It is assumed in Section 2.2.3 that the Jacobian matrix $\mathbf{J}$ defined in (9) was square. Then, if the passive architecture is not in a singular configuration, the matrix $\mathbf{J}$ is invertible and it is possible to compute the drone actuation force auxiliary input $\boldsymbol{\alpha}_\mathbf{f}$ from (23) with:

$$\boldsymbol{\alpha}_\mathbf{f} = \mathbf{J}^{-T}(\mathbf{M}(\mathbf{q})\boldsymbol{\alpha}_{\dot{\mathbf{v}}_q} + \mathbf{c}(\mathbf{q}, \mathbf{v}_q)) \tag{29}$$

This equation corresponds to the block (2) of the control scheme (Fig. 3).

Assuming that the desired input forces are obtained, i.e. $\mathbf{f} = \boldsymbol{\alpha}_\mathbf{f}$, and combining the dynamic model (23) with the control law (29), this ensures that $\dot{\mathbf{v}}_q = \boldsymbol{\alpha}_{\dot{\mathbf{v}}_q}$. Then, the convergence of the passive architecture coordinates is obtained from (28). Imperfections in the estimation of the dynamic model can be compensated by the choice of appropriate gains to maintain a local convergence of the controlled system as discussed in (Samson, 1987).

The cases where the Jacobian matrix $\mathbf{J}$ is rectangular were assumed to be out of the scope of this study. Nevertheless, it should be mentioned that, for an overactuated robot, i.e. if $3n > n_q$, the control of the over-actuation could be handled by applying conventional techniques, for instance, by using the Moore–Penrose pseudo-inverse $\mathbf{J}^+$ (Merlet, 1996):

$$\boldsymbol{\alpha}_\mathbf{f} = \mathbf{J}^+(\mathbf{M}(\mathbf{q})\boldsymbol{\alpha}_{\dot{\mathbf{v}}_q} + \mathbf{c}(\mathbf{q}, \mathbf{v}_q)) \tag{30}$$

In this case, the kernel of the matrix $\mathbf{J}$ might be exploited for secondary tasks.

4.2. Drone orientation and thrust auxiliary inputs

This section describes the control approaches used in Blocks (3) and (4) of Fig. 3.

At this point, it is important to recall that the $3n$ coordinates of the auxiliary input vector $\boldsymbol{\alpha}_\mathbf{f}$ are not independent, as each drones can only provide a single actuation force in the direction of the propeller axes. A given vector $\boldsymbol{\alpha}_\mathbf{f}$ can be obtained only if the drones are properly oriented and provide the appropriate thrust. Each drone $\mathcal{D}_i$ must be oriented such that the frame axis $\mathbf{z}_i$ is aligned with its force auxiliary input vector $\boldsymbol{\alpha}_{\mathbf{f}_i}$, i.e.

$$\mathbf{z}_i = \frac{\boldsymbol{\alpha}_{\mathbf{f}_i}}{|\boldsymbol{\alpha}_{\mathbf{f}_i}|} \tag{31}$$

Note that, this equation suppose $|\boldsymbol{\alpha}_{\mathbf{f}_i}| > 0$. $|\boldsymbol{\alpha}_{\mathbf{f}_i}| = 0$ is a singular configuration that corresponds to acrobatic flight, out of the scope of this study.

The vector $\mathbf{z}_i$ is not enough to constrain the orientation of the drone. From a valid orientation, any rotation along $\mathbf{z_i}$ might be used to control the robot. However, some configurations of the drone are preferred to avoid a collision between the drones arms and the legs of the passive architecture. To do this, it is possible to choose the direction a second vector of the drone local frame, in this specific case $\mathbf{x}_i$. Given a computed orientation of the vector $\mathbf{z}_i$ (31) and a desired orientation of the vector ${}^d\mathbf{x}_i$, it is possible to compute the rotation matrix of the drone frame $\mathcal{F}_i$ by

$$\mathbf{R}_i = \begin{bmatrix} {}^d\mathbf{x}_i & \mathbf{z}_i \times {}^d\mathbf{x}_i & \mathbf{z}_i \end{bmatrix} \tag{32}$$

Then, the quaternion input $\boldsymbol{\alpha}_{\mathbf{h}_i}^T$ to obtain the desired orientation of the drone i can be computed from the rotation matrix $\mathbf{R}_i$ using (Campa and Camarillo, 2008)

$$\boldsymbol{\alpha}_{\mathbf{h}_i}^T = \frac{1}{2\sqrt{r_{11}+r_{22}+r_{33}+1}} \begin{bmatrix} r_{11}+r_{22}+r_{33}+1 \\ r_{32}-r_{23} \\ r_{13}-r_{31} \\ r_{21}-r_{12} \end{bmatrix} \tag{33}$$

where r_{kl} is the coefficient of the matrix $\mathbf{R}$ located at the kth row and the lth column.

The thrust input α_{f_i} for each drone is then given by

$$\alpha_{\mathbf{f}_i} = |\boldsymbol{\alpha}_{\mathbf{f}_i}| \tag{34}$$

Equations (33) and (34), performed in the block (3) of the controller (Fig. 3), give the desired orientation and force output for each drone i.

Concerning each quadrotor internal control law (block (4) in Fig. 3), any thrust/attitude control law that is used for classic quadrotor can be implemented on each quadrotor to reach the desired auxiliary inputs α_{f_i} and $\boldsymbol{\alpha}_{\mathbf{h}_i}$. Such a control law, is available in the autopilot the Pixhawk controllers used for the experiments. The autopilot control law in the Pixhawk controller is based on (Bresciani et al., 2013).

As a summary, the overall controller will then be a cascaded controller inspired by quadrotor control laws (Mahony et al., 2012)

- An external loop ensures the convergence of the passive architecture coordinates using a target thrust force α_{f_i} and orientation quaternion $\boldsymbol{\alpha}_{\mathbf{h}_i}$ for each drone (blocks (1) (2) and (3) in Fig. 3).
- A internal loop for each drone ensures the convergence of the thrust forces f_i and orientation quaternions $\mathbf{h}_i$ toward the targets (block (4) in Fig. 3).

5. Experiments

In this section, some experimental results of trajectory tracking with a prototype of flying parallel robots, made of three legs, developed at LS2N, are presented. To perform these experiments, the previous modelling approaches and control strategies are applied.

5.1. Robot architecture

In order to evaluate the efficiency of the control law presented in Section 4, a prototype of a flying parallel robot made compose of three legs and a platform was designed. Each leg is made of a rigid link whose extremities are connected to two passive joints: a passive spherical joint connects the leg to the drone, and a passive revolute joint connects the leg to the platform (see Fig. 4). This architecture is denoted as a *3-$\underline{D}$SR* robot, where $\underline{D}$ stands for an actuated drone, *S* for a passive spherical joint and *R* for a passive revolute joint. The simulated kinematics and dynamics of this particular architecture have been shown in (Six et al., 2018).

Based on the proposed architecture, a prototype, described below, is designed. It should however be mentioned that this prototype is a proof of concept: it was made at low cost for validating the ability of a FPR to fly, and to track a trajectory. It was not designed for accuracy nor manipulability purpose. The characterization of such performance requires the use of a better designed robot, with more expensive hardware, and it is left as part of the future works.

The inertial parameters of the experimental prototype and gains of the control law are:

- Distance from platform center to revolute joint (PA_i): 0.127 m
- Leg length: 0.99 m
- Mass of the platform: 0.185 kg
- Mass of each leg: 0.139 kg
- Mass of each drone: 1.05 kg
- Control law gain P: 7.0
- Control law gain D: 5.0
- Control law gain I: 1.75

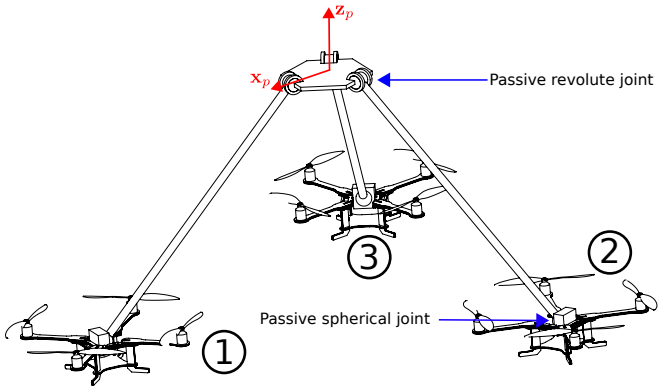

Figure 4. Kinematic scheme of the *3-DSR* flying parallel robot

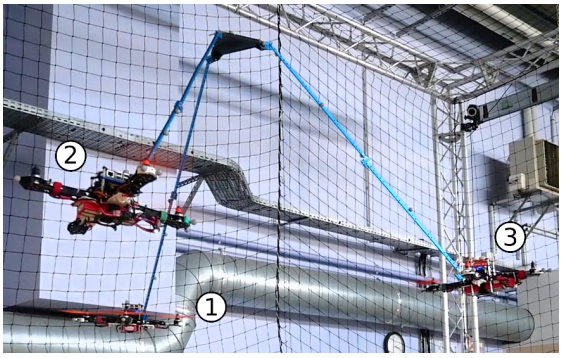

Figure 5. The *3-DSR* prototype

5.2. Hardware and software implementation

To design the experimental prototype (Fig. 5), the platform was 3D-printed. The leg elements are all from a modular robotic construction kit. Three aluminium bars were attached through revolute joints to the platform. On the drone side, custom quadrotors based on a Lynxmotion Crazy2fly frame, with MT2208 brushless DC motors, 12A ESCs, and plastic 8045 dual-blade propellers, are used. The legs are attached to the drones using a spherical joint for each drone. However, due to the arrangement of the drone frame, it was not possible to align precisely the drone COM and the center of the spherical joints, despite the assumption made for the computation of the dynamic model (Section 3). This leads to additional coupling moments between the drones and the passive architecture. The controller is expected to be sufficiently robust to compensate for those coupling moments not included in the dynamic model as well as for the others unmodelled dynamic effects (especially aerodynamic effects).

The controller on each drone is a Pixhawk 2.4 running PX4-v1.7.3. Each Pixhwak controller computes the attitude/thrust inner control loop for its drone. A Raspberry PI 3B+ is also embedded on each drone to communicate using WIFI with the command center. The pose of the passive architecture was gathered using a Qualisys MOCAP system with eight cameras. All data were nominally gathered at 200 Hz. A scheme of the hardware implementation is given in Fig. 6.

The control software system is composed of several ROS

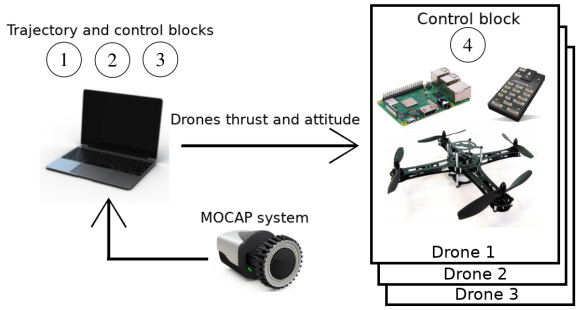

Figure 6. Hardware implementation of the experimental setup

nodes (Quigley et al., 2009) designed to:

- Ensure the collection of the MOCAP data. This node is run on a central computer.
- Compute the outer loop of the control law and provide the desired thrust/attitude for each drone. This node is also run on the central computer.
- Relay the desired thrust/attitude to each Pixhawk controller. One node is run on each Raspberry PI.

Given the low frequency of the data collected with the MOCAP, a Kalman Filter is implemented. It fuses the measured data with an estimation of the state accelerations obtained from the dynamic model of the flying robot (23). The technique is similar to (Sarim et al., 2015) for the position estimation but using the robot dynamic model instead of a single quadrotor model.

5.3. Compensation of the steady state error

The control law designed for the passive architecture in section 4.1 is a PD computed-torque control. In order to compensate for the steady-state error, it is possible to add an integral term to the PD law in (28) while keeping the convergence under the condition $\mathbf{K}_i < \mathbf{K}_p\mathbf{K}_d$ with $\mathbf{K}_i$ being the matrix associated to the integral term (Lewis et al., 2004). Thus, a PID computed-torque control law with anti-windup was implemented in the context of the experimental setup.

5.4. Experimental trajectory design

A trajectory was designed to provide motions along several degrees of freedom of the platform. The trajectory lasts 80 seconds and is composed of the following steps (see also Fig. 8 and Table 2)

- From 0 to 15 seconds: take-off, pure translation of the platform along the $\mathbf{z}$ axis.
- From 15 to 30 seconds: pure translation along $\mathbf{y}$ axis.
- From 30 to 45 seconds: motion of the leg 1 and the leg 2 while maintaining the platform in position and orientation.

- From 45 to 60 seconds: pure rotation of the platform along the **z** axis
- From 60 to 75 seconds: rotation of the platform along the **y** axis and change of the orientation of the leg 1
- From 75 to 80 seconds: landing

A video of the experiment is available in the additional content and snapshots of this video are given in Fig. 7.

The desired trajectory is defined using 6-degrees polynomials to ensure the C^2 continuity of the trajectory in any point. For this indoor experiments, the safe flight area available is 6 m long $\times$ 4 m width $\times$ 5 m height. Therefore, the trajectory is designed such that it exploits the dynamic capability of the benchmark given the environmental constraints. It will be seen further that, along this trajectory, the thrust saturation on one drone was reached.

5.5. Experimental results

To illustrate the measures obtained with the experimental setup, the Fig. 8 shows the trajectory designed and the tracking performed by the experimental setup along the nine degrees of freedom of the passive architecture. The Fig. 9 shows the error vector defined in Eq. (27) as an indication of the controller performances. The ability of the controller to remain stable along the different degrees of freedom can be observed, with the mean and maximum absolute errors along the nine degrees of freedom given in Table 3.

Good mean error performances in the tracking of the desired trajectory can be observed:

- Around 2 cm for the x and y coordinate of the platform and 4 cm for the z coordinate.
- From 5° to 8° for the orientation coordinates of the platform.
- A mean error around 4° for the orientation of the legs.

The flight has been reproduced dozens of times with similar results. Those tracking results were obtained despite the suboptimal position of the spherical joint linking the drone to the passive architecture and the several unmodelled aerodynamic effects. This shows promising performances in general. However, It can also be seen that some configurations required during the flight were more challenging to the system and led to an increased tracking error but without loosing the overall stability. The controller was able to converge back to the trajectory out of those specific configurations. In these configurations, one of the following issues appears:

- Ground effects when taking off and landing affect the tracking of the trajectory along the z axis.
- The yaw rotation of the platform from 45 to 60 seconds is too fast for the system and leads to a delay in the tracking and an overshoot of the maximal value.
- When reaching the configuration around 70 seconds, the thrust of the drone 3 (Fig. 10) is really close to the maximum thrust at full battery load (18 N). Thus, the tracking performances are locally decreased. The platform pitch angle in this configuration is such that the weight distribution between the drones is not as symmetric as in the other configurations. In future works, a wrench analysis similar to Erskine et al. (2019) must be performed to deeply analyse the practical workspace of such robot. This configuration was kept in those results to put light on the limits of the configurations that can be accurately performed by the prototype. Also, in such configuration, the drone 3 is more affected by the propeller wash of the other drones as shown in Fig. 11.

As the objective of such robot is to perform aerial manipulation, this precision might not be enough depending on the application. However, it is recalled that for this work, a low cost prototype was made for validating the ability of a FPR to fly and track a trajectory. Then, several options can be explored in order to improve the overall precision of the control strategy:

- Taking into account a more complete model of the propellers (Bristeau et al., 2009), that might also take into consideration the transient states.
- A better model of the aerodynamical effects (ground effects, interaction between the drones) (Powers et al., 2012) or the implementation of advanced control techniques to compensate coupling and disturbances as in (Castillo-Zamora et al., 2019).
- A better mechanical design to obtain the full decoupling of the dynamic model on the experimental prototype.
- A study of the admissible configurations given the drones thrust limit defining an achievable wrench space in an approach inspired by works performed on aerial cable towed systems (Erskine et al., 2019).

Those leads will be explored in the future work to improve the performances of such robot.

6. Conclusions and future works

In this paper, a generic architecture of a flying parallel robot with rigid links was presented. This robot is made of an articulated passive architecture, composed of several legs, actuated with several UAVs (in this study, quadricopters). This new flying robot offers the ability to control a platform position and orientation in space. The generic dynamic model of this robot has been established. This model shows decoupling properties between the dynamics of the generalized coordinates of the passive architecture and the dynamics of the rotation of the UAVs. This decoupling is exploited to establish a cascade controller that ensures the stability of the system and allows the tracking of a trajectory imposed on the coordinates of the passive structure.

time (s)	x_p (m)	y_p (m)	z_p (m)	roll (°)	pitch (°)	yaw(°)	q_1 (°)	q_2 (°)	q_3 (°)
0	0	0	0.8	0	0	0	-54.4	-54.4	-54.4
8	0	0	2.5	0	0	0	-54.4	-54.4	-54.4
15	0	0	2.5	0	0	0	-54.4	-54.4	-54.4
20	0	1.5	2.5	0	0	0	-54.4	-54.4	-54.4
25	0	-1.25	2.5	0	0	0	-54.4	-54.4	-54.4
30	0	0	2.5	0	0	0	-54.4	-54.4	-54.4
35	0	0	2.5	0	0	0	-37.2	-71.6	-54.4
40	0	0	2.5	0	0	0	-71.6	-37.2	-54.4
45	0	0	2.5	0	0	0	-54.4	-54.4	-54.4
53	0	0	2.5	0	0	180	-54.4	-54.4	-54.4
60	0	0	2.5	0	0	0	-54.4	-54.4	-54.4
65	0	0	2.5	0	-30	0	-71.6	-54.4	-54.4
70	0	0	2.5	0	-30	0	-71.6	-54.4	-54.4
75	0	0	2.5	0	0	0	-54.4	-54.4	-54.4
80	0	0	0.8	0	0	0	-54.4	-54.4	-54.4

Table 2. Experimental trajectory way-points: each way-points is reached using a 9-degree polynomial trajectory.

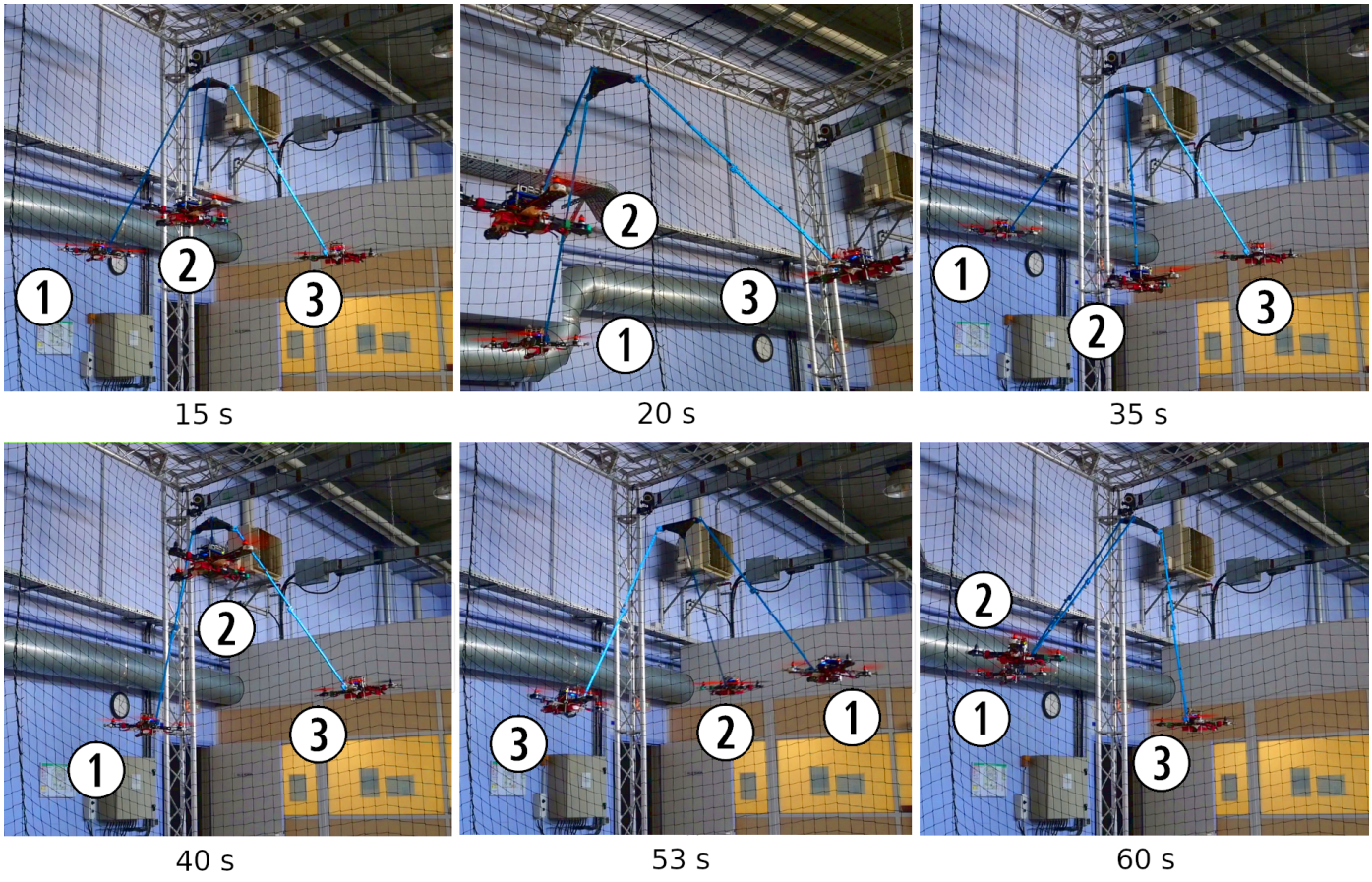

Figure 7. Snapshots of the experiment.

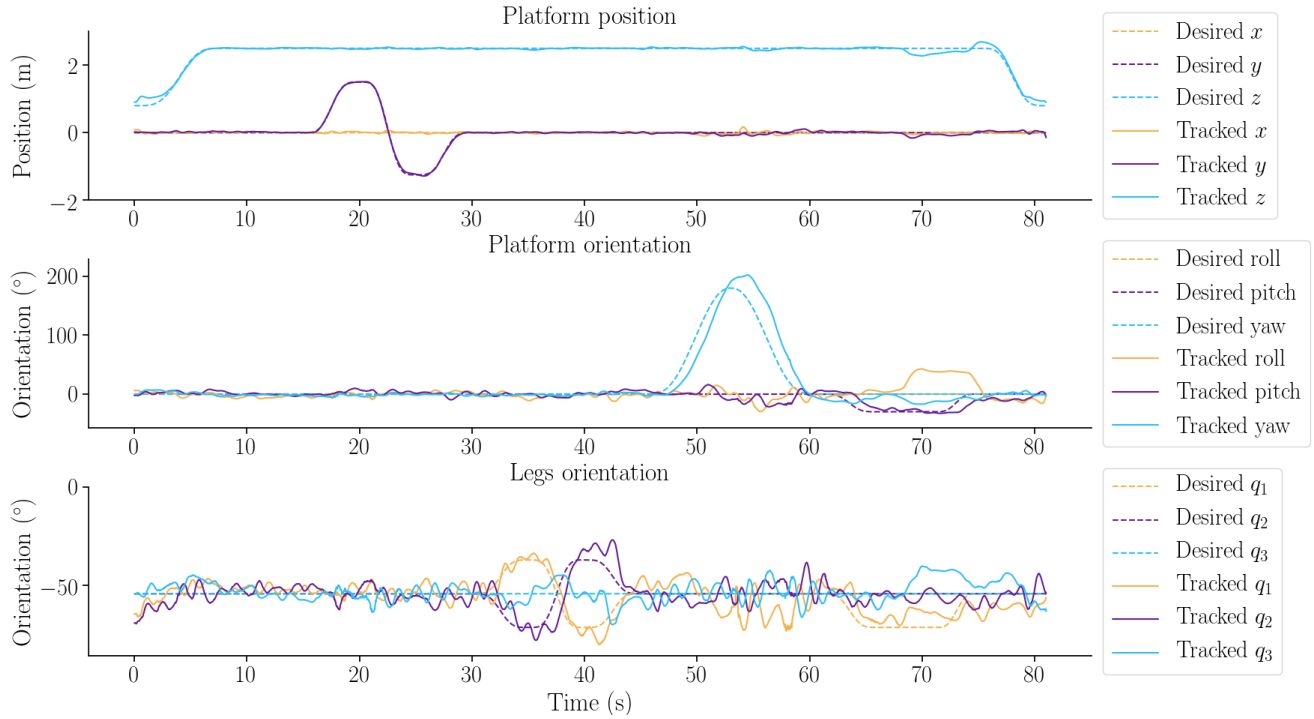

Figure 8. Reference and tracked trajectories obtained for the *3-DSR*. For a better readability, the tracking of the platform is displayed using the Roll/Pitch/Yaw representation even if the controller uses the quaternion representation.

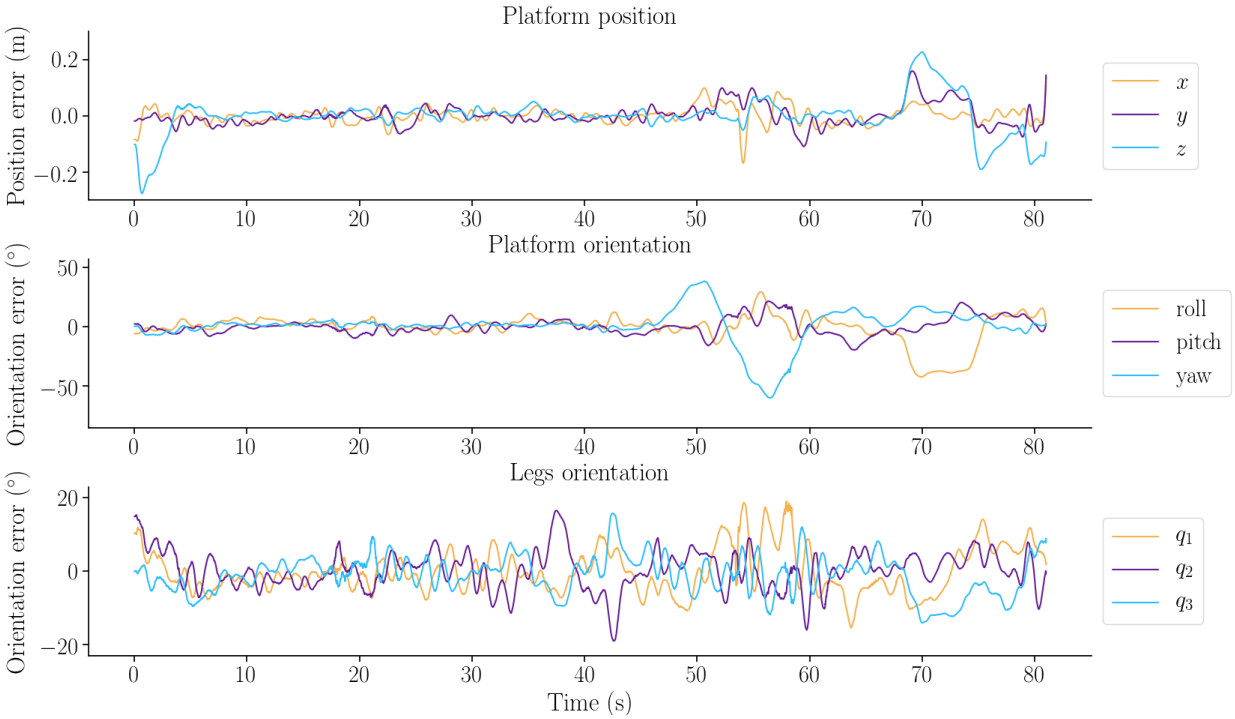

Figure 9. Measured tracking errors. For a better readability, the platform errors are displayed using the Roll/Pitch/Yaw representation even if the controller computes the quaternion errors.

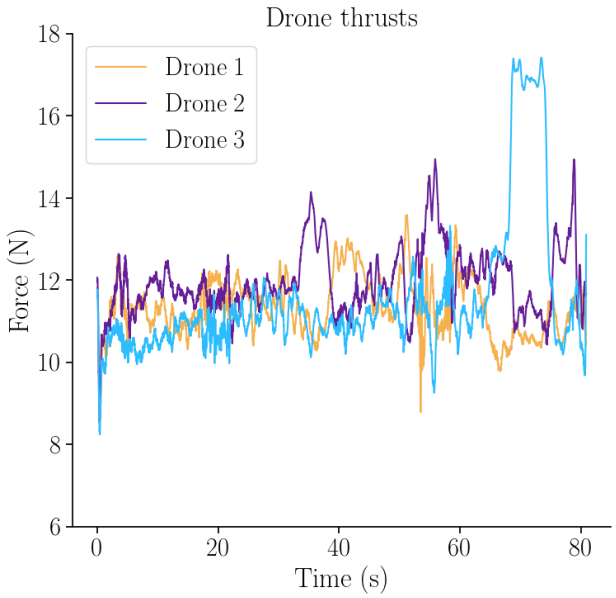

Figure 10. Drone thrusts for each drone during the experiment.

Variable	Mean absolute error	Maximum absolute error
x_p (m)	0.021	0.17
y_p (m)	0.0227	0.16
z_p (m)	0.039	0.28
roll (°)	7.25	42.73
pitch (°)	5.08	21.56
yaw (°)	7.99	60.54
q_1 (°)	4.79	18.99
q_2 (°)	4.04	18.96
q_3 (°)	4.04	15.77

Table 3. Mean and maximum error absolute values during the experimental flight. For a better comprehension, the platform orientation errors are computed using the Roll/Pitch/Yaw representation even if the controller computes the quaternion errors.

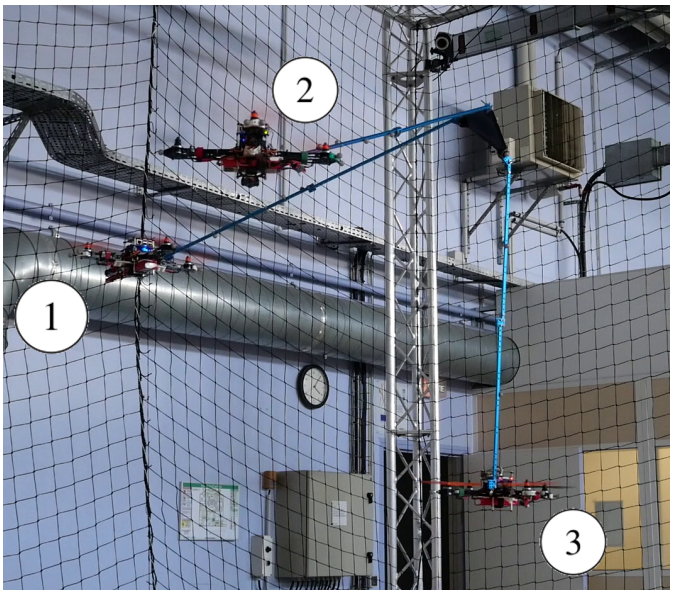

Figure 11. Configuration obtained at the 70th second of the trajectory

Experimental results showed controller performance against noisy pose estimation and perturbations due to interaction between drones (e.g. propeller wind disturbance) and unmodelled effects (e.g. aerodynamic effects). Despite some specific configurations, where the tracking performances are degraded, the controller showed the ability to perform a stabilized flight with the prototype. The performances of position tracking are quite good on average despite a few critical points where the maximum error reaches 20 cm. This is mainly due to the fact that at that moment the thrust of one of the drones is very close to the maximum thrust, in other words close to motor saturation.

As future works, several research directions can be explored to improve the overall accuracy of the control strategy: a better consideration of the propellers dynamics, better modeling of aerodynamic effects (ground effects, interaction between UAVs) and a study of the admissible configurations given the drones thrust limit defining an achievable wrench space.

Finally, to show the robot's ability to act or perform a task in the environment, another very important aspect is to take into account the physical interaction of the FPR with the environment (object/wall). Research efforts will be devoted to the design of control strategies taking into account the interaction between the drones and the legs, but also the interaction with the environment (e.g. force applied on a wall) through a measurement of the forces on the end-effector or through the design of an *ad hoc* observer. With the help of this controller, manipulating tasks in experimental conditions could be considered.

Appendix A. Appendix

Appendix A.1. Recalls on quaternions

Unit quaternions can be used to represent frame orientations without any singularity of representation. Moreover, they are useful for feedback control of robot frame orientations (Yuan,

1988). In this section, some properties useful in the context of this paper will be recalled.

A quaternion is represented with a scalar part η and a vector part $\boldsymbol{\epsilon}$ such that $\mathbf{h}^T = \begin{bmatrix}\eta & \boldsymbol{\epsilon}^T\end{bmatrix}$. A unit quaternion verify the following normality property

$$\eta^2 + \boldsymbol{\epsilon}^T\boldsymbol{\epsilon} = 1 \tag{A.1}$$

Each rotation can be represented by two quaternions. The quaternion are linked to an axis $\mathbf{u}$ - angle θ representation by the following formula

$$\eta = \pm\cos(\theta/2) \quad \boldsymbol{\epsilon} = \pm\sin(\theta/2)\mathbf{u} \tag{A.2}$$

The quaternions $\begin{bmatrix}\eta & \boldsymbol{\epsilon}^T\end{bmatrix}$ and $\begin{bmatrix}-\eta & -\boldsymbol{\epsilon}^T\end{bmatrix}$ represent the same rotation. Note that even if quaternions have four coordinates, only three of them are independent due to the normality condition (A.1).

If $\mathbf{h}^T = \begin{bmatrix}\eta & \boldsymbol{\epsilon}^T\end{bmatrix}$ represent the rotation from $\mathcal{F}_0$ to $\mathcal{F}_1$, then the rotation from $\mathcal{F}_1$ to $\mathcal{F}_0$ is represented by the quaternion $\bar{\mathbf{h}}^T = \begin{bmatrix}\eta & -\boldsymbol{\epsilon}^T\end{bmatrix}$.

The Hamilton product of two quaternions $\mathbf{h}_1$ and $\mathbf{h}_2$ is defined by

$$\mathbf{h}_1 \circ \mathbf{h}_2 = \begin{bmatrix}\eta_1\eta_2 - \epsilon_1^T\epsilon_2 \\ \eta_1\epsilon_2 + \eta_2\epsilon_1 + \epsilon_1 \times \epsilon_2\end{bmatrix} \tag{A.3}$$

The rotation of a vector $\mathbf{p}$ by a rotation represented by a quaternion $\mathbf{h}$ is given by the conjugation of $\mathbf{p}$ by $\mathbf{h}$ using the Hamilton product, as in the following expression:

$$\begin{bmatrix}0 \\ \mathbf{p}'\end{bmatrix} = \mathbf{h} \circ \begin{bmatrix}0 \\ \mathbf{p}\end{bmatrix} \circ \bar{\mathbf{h}} \tag{A.4}$$

where $\mathbf{p}'$ is the rotated vector. The Hamilton product allows to combine quaternion. If $\mathbf{h}_1$ represents the rotation from the frame $\mathcal{F}_0$ to $\mathcal{F}_1$ and $\mathbf{h}_2$ the rotation from the frame $\mathcal{F}_1$ to $\mathcal{F}_2$ then $\mathbf{h}_1 \circ \mathbf{h}_2$ it the rotation from the frame $\mathcal{F}_0$ to $\mathcal{F}_2$. The product of a quaternion by its inverse is the identity quaternion representing a zero angle rotation

$$\mathbf{h} \circ \bar{\mathbf{h}} = \begin{bmatrix}1 \\ \mathbf{0}\end{bmatrix} \tag{A.5}$$

The time derivative of the quaternion vector $\mathbf{h}$ associated to a frame $\mathcal{F}$ is related to the frame angular velocity $\boldsymbol{\omega}$ by the following relations (Campa and Camarillo, 2008)

$$\begin{aligned}\dot{\mathbf{h}} &= \frac{1}{2}\begin{bmatrix}-\boldsymbol{\epsilon}^T \\ \eta\mathbb{1}_3 - \boldsymbol{\epsilon}^\times\end{bmatrix}\boldsymbol{\omega} = \frac{1}{2}\begin{bmatrix}0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & -\boldsymbol{\omega}^\times\end{bmatrix}\mathbf{h} \\ \boldsymbol{\omega} &= 2\begin{bmatrix}-\boldsymbol{\epsilon} & \eta\mathbb{1}_3 + \boldsymbol{\epsilon}^\times\end{bmatrix}\dot{\mathbf{h}}\end{aligned} \tag{A.6}$$

with $\mathbb{1}_3$ the dimension identity matrix in $\mathbb{R}^{3\times3}$. For any vectors $\boldsymbol{\epsilon}$ and $\boldsymbol{\omega}$. The expression (A.6) can be used to compute the passive architecture velocity vector $\mathbf{v}_q$ from the derivative of the coordinate vector $\mathbf{q}$.

Appendix A.2. Proof of convergence of the orientation control law

The convergence of the control law can be proven using the Lyapunov's second method for stability (Khalil, 2002). The Lyapunov candidate is defined as

$$V_2 = \frac{1}{2}({}^d\boldsymbol{\omega} - \boldsymbol{\omega})^T({}^d\boldsymbol{\omega} - \boldsymbol{\omega}) + {}^h\mathbf{K}_p V_1 \tag{A.7}$$

where

$$V_1 = ({}^d\eta - \eta)^2 + ({}^d\boldsymbol{\epsilon} - \boldsymbol{\epsilon})^T({}^d\boldsymbol{\epsilon} - \boldsymbol{\epsilon}) \tag{A.8}$$

V_2 is positive and strictly positive if ${}^d\boldsymbol{\omega} \neq \boldsymbol{\omega}$ or ${}^d\mathbf{h} \neq \mathbf{h}$. Introducing (A.6) and (25) in the first derivative of (A.8) and skipping the mathematical derivation, it is obtained

$$\dot{V}_1 = \delta\boldsymbol{\epsilon}({}^d\mathbf{h}, \mathbf{h})^T({}^d\boldsymbol{\omega} - \boldsymbol{\omega}) \tag{A.9}$$

Assuming that the angular acceleration match the auxiliary input $\alpha_{\dot{\omega}} = \dot{\boldsymbol{\omega}}$ and introducing (26) and (A.9) in the first derivative of the Lyapunov candidate (A.7) gives

$$\dot{V}_2 = -{}^h\mathbf{K}_d({}^d\boldsymbol{\omega} - \boldsymbol{\omega})^T({}^d\boldsymbol{\omega} - \boldsymbol{\omega}) \tag{A.10}$$

$\dot{V}_2$ is negative semidefinite and the trajectories such that $\dot{V}_2 = 0$ verify $\omega = {}^d\omega$, therefore $\dot{\omega} = {}^d\dot{\omega}$ and using (26) $\delta\epsilon = 0$. LaSalle's invariant principle ensures the asymptotic convergence of $\delta\epsilon$ towards 0.

References

Anzai, T., Zhao, M., Chen, X., Shi, F., Kawasaki, K., Okada, K., Inaba, M., 2017. Multilinked multirotor with internal communication system for multiple objects transportation based on form optimization method, in: IEEE International Conference on Intelligent Robots and Systems, Vancouver, BC, Canada. pp. 5977–5984.

Backus, S.B., Odhner, L.U., Dollar, A.M., 2014. Design of hands for aerial manipulation: actuator number and routing for grasping and perching, in: Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), Chicago, Illinois, USA. pp. 34–40.

Bernard, M., Kondak, K., Maza, I., Ollero, A., 2011. Autonomous Transportation and Deployment with Aerial Robot for Search and Rescue Missions. Journal of Field Robotics 28, 914–931.

Bouabdallah, S., Murrieri, P., Siegwart, R., 2004. Design and control of an indoor micro quadrotor, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), New Orleans, LA, USA. pp. 4393–4398.

Bresciani, D., Hehn, M., D'Andrea, R., 2013. Nonlinear Quadrocopter Attitude Control. Technical Report. ETH Zurich.

Bristeau, P.J., Martin, P., Salaün, E., Petit, N., 2009. The role of propeller aerodynamics in the model of a quadrotor UAV, in: Proceedings of the European Control Conference, Budapest, Hungary. pp. 683–688.

Campa, R., Camarillo, K., 2008. Unit Quaternions: A Mathematical Tool for Modeling, Path Planning and Control of Robot Manipulators, in: Robot Manipulator. IntechOpen. chapter 2, pp. 21–47.

Castillo, P., Lozano, R., Dzul, A., 2005. Stabilization of a mini rotorcraft with four rotors: Experimental implementation of linear and nonlinear control laws. IEEE Control Systems Magazine 25, 45–55.

Castillo-Zamora, J.J., Escareno, J., Alvarez, J., Stephant, J., Boussaada, I., 2019. Disturbances and coupling compensation for trajectory tracking of a multi-link aerial robot, in: 6th International Conference on Control, Decision and Information Technologies, CoDIT 2019, Paris, France. pp. 738–743.

Cataldi, E., Muscio, G., Trujillo, M.A., Rodriguez, Y., Pierri, F., Antonelli, G., Caccavale, F., Viguria, A., Chiaverini, S., Ollero, A., 2016. Impedance control of an aerial-manipulator: Preliminary results, in: Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea. pp. 3848–3853.

Danko, T.W., Oh, P.Y., 2014. Design and Control of a Hyper-Redundant Manipulator for Mobile Manipulating Unmanned Aerial Vehicles. Journal of Intelligent & Robotic Systems 73, 709–723.

Das, A., Lewis, F., Subbarao, K., 2009. Backstepping Approach for Controlling a Quadrotor Using Lagrange Form Dynamics. Journal of Intelligent and Robotic Systems 56, 127–151.

Erskine, J., Chriette, A., Caro, S., 2019. Wrench analysis of cable-suspended parallel robots actuated by quadrotor unmanned aerial vehicles. Journal of Mechanisms and Robotics 11, 020909:1–12.

Ghadiok, V., Goldin, J., Ren, W., 2011. Autonomous indoor aerial gripping using a quadrotor, in: Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), pp. 4645–4651.

Heredia, G., Jimenez-Cano, A.E., Sanchez, I., Llorente, D., Vega, V., Braga, J., Acosta, J.A., Ollero, A., 2014. Control of a multirotor outdoor aerial manipulator, in: Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), Chicago, Illinois, USA. pp. 3417–3422.

Huber, F., Kondak, K., Krieger, K., Sommer, D., Schwarzbach, M., Laiacker, M., Kossyk, I., Parusel, S., Haddadin, S., Albu-Schaffer, A., 2013. First analysis and experiments in aerial manipulation using fully actuated redundant robot arm, in: Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), Tokyo, Japan. pp. 3452–3457.

Jamshidifar, H., Khajepour, A., 2020. Static Workspace Optimization of Aerial Cable Towed Robots With Land-Fixed Winches. IEEE Transactions on Robotics , 1603–1610.

Jimenez-Cano, A., Martin, J., Heredia, G., Ollero, A., Cano, R., 2013. Control of an aerial robot with multi-link arm for assembly tasks, in: Proceedings of IEEE International Conference on Robotics and Automation, IEEE. pp. 4916–4921.

Khalil, H.K., 2002. Nonlinear systems (Third edition). Pearson.

Khalil, W., 2010. Dynamic modeling of robots using recursive Newton-Euler techniques, in: Proceedings of the International Conference on Informatics in Control, Automation and Robotics, Madeira, Portugal.

Khalil, W., Dombre, E., 2002. Modeling, identification and control of robots. Kogan Page Science.

Kondak, K., Huber, F., Schwarzbach, M., Laiacker, M., Sommer, D., Bejar, M., Ollero, A., 2014. Aerial manipulation robot composed of an autonomous helicopter and a 7 degrees of freedom industrial manipulator, in: 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 2107–2112.

Kondak, K., Krieger, K., Albu-Schaeffer, A., Schwarzbach, M., Laiacker, M., Maza, I., Rodriguez-Castano, A., Ollero, A., 2013. Closed-loop behavior of an autonomous helicopter equipped with a robotic arm for aerial manipulation tasks. International Journal of Advanced Robotic Systems 10.

Kuipers, J.B., 1999. Quaternions and rotation sequences, in: Geometry, Integrability and Quantization, Varna, Bulgaria. pp. 127–143.

Lee, J.Y., Leang, K.K., Yim, W., 2018. Design and control of a fully- actuated hexrotor for aerial manipulation applications. Journal of Mechanisms and Robotics 10, 1–10.

Lee, T., Leok, M., McClamroch, N.H., 2010. Geometric tracking control of a quadrotor UAV on SE(3) complement. Proceedings of the IEEE Conference on Decision and Control , 5420–5425.

Lewis, F.L., Dawson, D.M., T.Abdallah, C., 2004. Robot Manipulator Control: Theory and Practice (Second Edition). Marcel Dekker, Inc.

Lindsey, Q., Mellinger, D., Kumar, V., 2011. Construction of Cubic Structures with Quadrotor Teams, in: Proceedings of Robotics: Science and Systems (RSS), Los Angeles, California, USA.

Loianno, G., Spurny, V., Thomas, J., Baca, T., Thakur, D., Hert, D., Penicka, R., Krajnik, T., Zhou, A., Cho, A., Saska, M., Kumar, V., 2018. Localization, Grasping, and Transportation of Magnetic Objects by a Team of MAVs in Challenging Desert-Like Environments. IEEE Robotics and Automation Letters 3, 1576–1583.

Mahony, R., Kumar, V., Corke, P., 2012. Multirotor Aerial Vehicles: Modeling, estimation, and control of quadrotor. IEEE Robotics and Automation Magazine 19, 20–32.

Merlet, J.p., 1996. Redundant parallel manipulators. Laboratory Robotics and Automation 8, 17–24.

Merlet, J.P., 2004. Solving the Forward Kinematics of a Gough-Type Parallel Manipulator with Interval Analysis. The International Journal of Robotics Research 23, 221–235.

Merlet, J.P., 2006. Parallel Robots. Springer, Dordrecht, The Netherland.

Michael, N., Fink, J., Kumar, V., 2011. Cooperative manipulation and transportation with aerial robots. Autonomous Robots 30, 73–86.

Nguyen, H.N., Park, S., Lee, D., 2015. Aerial tool operation system using quadrotors as Rotating Thrust Generators, in: Proceedings of the International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany. pp. 1285–1291.

Nguyen, H.N., Park, S., Park, J., Lee, D., 2018. A Novel Robotic Platform for Aerial Manipulation Using Quadrotors as Rotating Thrust Generators. IEEE Transactions on Robotics 34, 353–369.

Nikou, A., Gavridis, G.C., Kyriakopoulos, K.J., 2015. Mechanical design, modelling and control of a novel aerial manipulator. Proceedings - IEEE International Conference on Robotics and Automation 2015-June, 4698–4703.

Ollero, A., Heredia, G., Franchi, A., Antonelli, G., Kondak, K., Sanfeliu, A., Viguria, A., Martinez-de Dios, J.R., Pierri, F., Cortes, J., Santamaria-Navarro, A., Trujillo Soto, M.A., Balachandran, R., Andrade-Cetto, J., Rodriguez, A., 2018. The AEROARMS Project: Aerial Robots with Advanced Manipulation Capabilities for Inspection and Maintenance. IEEE Robotics Automation Magazine 25, 12–23.

Pounds, P.E.I., Bersak, D.R., Dollar, A.M., 2011. Grasping from the air: Hovering capture and load stability, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shangai, China. pp. 2491–2498.

Powers, C., Mellinger, D., Kushleyev, A., Kothmann, B., Kumar, V., 2012. Influence of Aerodynamics and Proximity Effects in Quadrotor Flight, in: The 13th International Symposium on Experimental robotics, Québec, Canada. pp. 289–302.

Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., Ng, A., 2009. ROS: an open-source Robot Operating System Morgan, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan.

Rajappa, S., Ryll, M., Bulthoff, H.H., Franchi, A., 2015. Modeling, control and design optimization for a fully-actuated hexarotor aerial vehicle with tilted propellers, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA. pp. 4006–4013.

Rossi, E., Tognon, M., Zurich, E., Carli, R., Schenato, L., Cortés, J., Franchi, A., 2019. Cooperative Aerial Load Transportation via Sampled Communication. IEEE Control Systems Letters 2020, 277–282.

Ruggiero, F., Trujillo, M.A., Cano, R., Ascorbe, H., Viguria, A., Perez, C., Lippiello, V., Ollero, A., Siciliano, B., 2015. A multilayer control for multirotor UAVs equipped with a servo robot arm, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA. pp. 4014–4020.

Ryll, M., Bülthoff, H.H., Robuffo Giordano, P., 2015. A Novel Overactuated Quadrotor UAV: Modeling, Control and Experimental Validation. IEEE Transactions on Control Systems Technology 23, 510–556.

Samson, C., 1987. Robust control of a class of non-linear systems and applications to robotics. International Journal of Adaptive Control and Signal Processing 1, 49–68.

Sarim, M., Nemati, A., Kumar, M., Cohen, K., 2015. Extended kalman filter based quadrotor state estimation based on asynchronous multisensor data, in: ASME 2015 Dynamic Systems and Control Conference, Columbus, USA.

Six, D., Briot, S., Chriette, A., Martinet, P., 2018. The kinematics , dynamics and control of a flying parallel robot with three quadrotors. IEEE Robotics and Automation Letters 3, 559–566.

Sreenath, K., Kumar, V., 2013. Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots, in: Proceedings of Robotics: Science and Systems, Berlin, Germany.

Suarez, A., Soria, P.R., Heredia, G., Arrue, B.C., Ollero, A., 2017. Anthropomorphic, compliant and lightweight dual arm system for aerial manipulation, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada. pp. 992–997.

Suarez, A., Vega, V.M., Fernandez, M., Heredia, G., Ollero, A., 2020. Benchmarks for Aerial Manipulation. IEEE Robotics and Automation Letters 5, 2650–2657. doi:10.1109/LRA.2020.2972870.

Voos, H., 2009. Nonlinear control of a quadrotor micro-uav using feedback-linearization, in: Proceedings of the IEEE International Conference on Mechatronics, Malaga, Spain. pp. 1–6.

Yang, B., He, Y., Han, J., Liu, G., 2014. Rotor-Flying Manipulator: Modeling, Analysis, and Control. Mathematical Problems in Engineering 2014, 1–13.

Yuan, J.S., 1988. Closed-Loop Manipulator Control Using Quaternion Feedback. IEEE Journal on Robotics and Automation 4, 434–440.

Zhao, M., Anzai, T., Shi, F., Chen, X., Okada, K., Inaba, M., 2018. Design, modeling, and control of an aerial robot DRAGON: A dual-rotor-embedded

multilink robot with the ability of multi-degree-of-freedom aerial transformation. IEEE Robotics and Automation Letters 3, 1176–1183.

Zhou, Z., Zhang, Y., Zhou, D., 2016. Geometric modeling and control for the full-actuated aerial manipulating system, in: 2016 35th Chinese Control Conference (CCC), pp. 6178–6182.