Decentralized control and state estimation of a flying parallel robot interacting with the environment

Shiyu Liu, Isabelle Fantoni, Abdelhamid Chriette

Decentralized control and state estimation of a flying parallel robot interacting with the environment

Shiyu Liu[a], Isabelle Fantoni[a,*] and Abdelhamid Chriette[a]

[a]Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, Nantes F-44000, France

ARTICLE INFO

Keywords:
Unmanned Aerial Vehicles (UAVs)
Decentralized control
Multi-robot systems
Parallel robots
Aerial manipulation

ABSTRACT

Unmanned Aerial Vehicles (UAVs) have great potential to achieve a variety of tasks remotely such as aerial grasping, transporting and manipulating objects. Architectures with multiple UAVs have further enhanced the payload capacity and manipulability of these robots, for instance a Flying Parallel Robot (FPR) where a moving platform is cooperatively supported by multiple quadrotors with passive rigid links. In this paper, we address the vision-based state estimation and decentralized control applied to the multi-UAV parallel robot, taking the FPR as an example. An ArUco marker system is applied to estimate the relative pose of each UAV with respect to the common platform frame, along with the Extended Kalman Filter to reconstruct the robot state without the dependence on any external localization system. The interaction controller is then deployed in a decentralized manner, which is potentially more robust to communication delays or interruptions. The proposed methodology has been validated by real-time experiments demonstrating the teleoperation of the FPR interacting with the environment.

1. Introduction

Unmanned Aerial Vehicles (UAVs) equipped with additional tools or end-effectors have great potential to accomplish a variety of tasks in remote locations or places considered dangerous to human operators. These aerial robots commonly known as Aerial Manipulators (Ollero, Tognon, Suarez, Lee and Franchi, 2021) have been shown capable of performing tasks like grasping, inspection, and transportation, in which the physical interaction with the environment is often involved. Among these aerial manipulators, the ones with multiple UAVs cooperatively supporting a payload or a moving platform have aroused growing interests due to its potential advantages. In fact, the cooperation of multiple UAVs has greatly increased the payload capacity as in cable-suspended aerial robots (Bernard, Kondak, Maza and Ollero, 2011; Masone, Bülthoff and Stegagno, 2016; Gassner, Cieslewski and Scaramuzza, 2017; Loianno and Kumar, 2018; Sanalitro, Savino, Tognon, Cortes and Franchi, 2020) and can achieve the full manipulability in 3-dimensional space such as the designs shown in (Nguyen, Park, Park and Lee, 2018; Park, Lee, Heo and Lee, 2019; Li, Song, Bégoc, Chriette and Fantoni, 2021b). Similarly to these works, a Flying Parallel Robot (FPR) has been proposed in Six, Briot, Chriette and Martinet (2018) and validated by experiments in Six, Briot, Chriette and Martinet (2021), which is composed of a passive parallel architecture supported by a number of quadrotors rigidly connected to the structure by means of spherical joints. By combining the advantages of parallel mechanisms and aerial vehicles, the FPR has shown its potential of accomplishing various tasks by exerting force onto the external objects in the environment (Liu, Fantoni, Chriette and Six, 2022). A lot of industrial applications can be envisaged such as infrastructure inspection, repairing and pick-and-place operations in remote locations (Ruggiero, Lippiello and Ollero, 2018). However, the previous works on the FPR and most of other multi-UAV parallel robots still depend on the high-rate exteroceptive measurements of the robot state, which means the necessity of using Motion Capture (MOCAP) systems that are impractical in real-world applications.

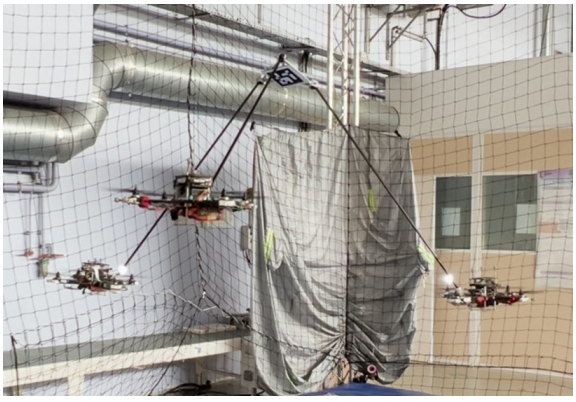


Figure 1: A flying parallel robot composed of a moving platform supported by three quadrotors with rigid legs.

The robot localization in multi-UAV parallel robots is an important topic especially for maintaining good inter-UAV positions. Indeed, the access to Global Navigation Satellite Systems (GNSS) in outdoor scenarios has been a solution applied to relatively large-scale systems as in Bernard et al. (2011); Park et al. (2019). However, the centimetric precision of the GPS is not satisfactory for sufficient calculation of internal configuration avoiding collisions between multiple UAVs, not to mention the degenerated cases where the aerial robots are close to the building or under structures to perform the tasks, or even in GPS-denied environments. Several

*Corresponding author
shiyu.liu@ls2n.fr (S. Liu); isabelle.fantoni@ls2n.fr (I. Fantoni); abdelhamid.chriette@ls2n.fr (A. Chriette)
ORCID(s): 0000-0003-1558-2835 (S. Liu); 0000-0003-3472-3023 (I. Fantoni); 0000-0003-4457-1561 (A. Chriette)

vision-based solutions have therefore been proposed such as Loianno and Kumar (2018); Li, Ge and Loianno (2021a) using monocular vision and Inertial Measurement Unit (IMU) sensing to localize each individual UAV with respect to the payload. In Loianno and Kumar (2018), multiple quadrotors are rigidly connected to a bar-like structure, each estimating its own pose and that of the structure using Visual-Inertial Odometry (VIO) technique. In Li et al. (2021a) for a cable-suspended aerial system, apart from onboard VIO, the information from the embedded monocular camera is further used to estimate the payload and cable states. The vision-based state estimation technique can therefore be applied to other multi-UAV parallel robots as the FPR, where each individual UAV can localize itself relative to the moving platform using intrinsic measurements from the camera and the IMU.

Based on the localization of each UAV with respect to a common body (the payload or the moving platform), a control law can be designed which can potentially be deployed onboard each UAV to reduce the information exchanges and increase the robustness of the whole system. This control methodology is often referred to as a decentralized control scheme extensively applied in swarm robotics and multi-robot control community, where each robot is desirably treated as an independent agent running its own perception and control algorithms making the whole system more robust to communication delays or interruptions (Coppola, McGuire, De Wagter and de Croon, 2020). In these systems, the decentralization is often made possible by finding some consensus or maintaining a virtual structure among the agents (Chung, Paranjape, Dames, Shen and Kumar, 2018). The decentralized controller employed in multi-UAV parallel robots is still an open problem, with examples found in the control of a cable-suspended payload by quadrotors using leader-follower (Gassner et al., 2017; Gabellieri, Tognon, Sanalitro, Pallottino and Franchi, 2020) or force-consensus (Thapa, Bai and Acosta, 2020) methods.

In this paper, we propose a state estimation method distributed on each UAV using vision-based technique with ArUco marker system and a decentralized control approach that can also be deployed onboard each UAV with a minimal set of information shared among the UAVs. Our approach allows each individual UAV in the FPR to compute its control based on the onboard measurements and achieve cooperatively an interaction control with sufficiently good estimation of external wrench acting on the robot. This work is therefore taking advantage of the decentralized scheme in control problems of the FPR interacting with the environment based on intrinsic measurements. We have implemented and validated the approach by real-world experiments. It is remarked that in a real-world implementation, a set of onboard sensors is needed so as to estimate certain variables (i.e. attitude and linear velocity) of the quadrotors. In the experiments we conducted, we used the MOCAP system to improve the linear velocity estimates in quadrotor's local frame, which can eventually be replaced by other onboard sensors. The methodology proposed in this paper does not necessarily require any MOCAP system, as the main contribution is to use the vision-based pose estimation technique and decentralized method to control the FPR without dependency on any external localization system.

The rest of the paper is organized as follows. In Section 2, we present the modelling and control of the FPR taking into account the interaction with the environment. Section 3 presents how to estimate the robot state (i.e. pose and velocity) by intrinsic measurements without the need of using any external localization systems. Section 4 addresses the decentralized control scheme for accomplishing the interaction with the environment using only the accessible robot states recovered by the onboard measurements. We present the experimental results of the proposed methodology in Section 5. Finally, we conclude this work and discuss the future works in Section 6.

2. Dynamic modelling and control

We firstly introduce the modelling and controller for the FPR system, where a moving platform is supported from n quadrotors by rigid links (or called legs). The top of each leg is attached to the platform using one Degree-of-Freedom (DoF) revolute joint, while the other end is connected to the quadrotor by means of spherical joint, which is considered attached to the Center of Mass (CoM) of each quadrotor and allows a dynamic decoupling regarding the quadrotor's rotational movements. The kinematic and dynamic models are presented in Section 2.1, based on which external wrench estimation and interaction control methods are formulated in Section 2.2 and Section 2.3.

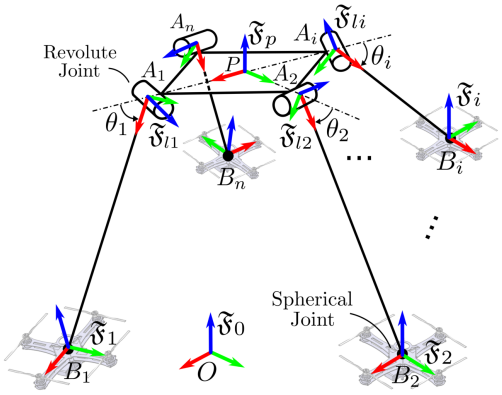


Figure 2: System description of a generic flying parallel robot. The frames are defined by axes in red (x), green (y), blue (z).

2.1. Kinematic and dynamic models

As shown in Fig. 2, we consider n quadrotors that cooperatively support a moving platform (on which an end-effector can be mounted) by rigid legs with passive (i.e. not actuated) joints. The robot configuration can be exclusively represented by the platform's 6D pose and additional DoFs provided by the revolute-joint motions. Therefore, the robot pose is given by a vector

$$\mathbf{q} = \begin{bmatrix} \mathbf{p}_p^T & \mathbf{q}_p^T & \boldsymbol{\theta}_l^T \end{bmatrix}^T \in \mathbb{R}^{7+n} \tag{1}$$

Table 1
Notation and symbols

$\mathfrak{F}_0$	global reference frame	$\mathbf{R}_p \in SO(3)$	rotation matrix of the platform relative to $\mathfrak{F}_0$
$\mathfrak{F}_p$	platform frame, with CoM located at point P	$\mathbf{R}_i \in SO(3)$	rotation matrix of quadrotor i relative to $\mathfrak{F}_0$
$\mathfrak{F}_{li}$, $\mathfrak{F}_i$	leg i's frame and quadrotor i's frame, with CoM located respectively at point A_i and B_i	$\mathbf{v}_p, \mathbf{v}_i \in \mathbb{R}^3$	linear velocity of the platform, quadrotor i in $\mathfrak{F}_0$
		${}^p\boldsymbol{\omega}_p \in \mathbb{R}^3$	body-frame angular velocity of the platform
$\mathfrak{F}_{F*}$	flat frames attached to an arbitrary body	${}^i\boldsymbol{\omega}_i \in \mathbb{R}^3$	body-frame angular velocity of quadrotor i
$\mathbf{p}_p, \mathbf{p}_i \in \mathbb{R}^3$	position of the platform, quadrotor i in $\mathfrak{F}_0$	$\mathbf{f}_p \in \mathbb{R}^3$	force exerted on the platform expressed in $\mathfrak{F}_0$
$\mathfrak{q}_p, \mathfrak{q}_i \in \mathbb{H}$	unit quaternion of the platform, quadrotor i relative to $\mathfrak{F}_0$	${}^p\mathbf{m}_p \in \mathbb{R}^3$	moment exerted on the platform expressed in $\mathfrak{F}_p$
		$\mathbf{m}_l \in \mathbb{R}^n$	a vector concatenating moments around the legs
$\boldsymbol{\theta}_l \in \mathbb{R}^n$	a vector concatenating the leg angles θ_i	$\mathbf{f}_i \in \mathbb{R}^3$	thrust force of quadrotor i expressed in $\mathfrak{F}_0$
$\dot{\boldsymbol{\theta}}_l \in \mathbb{R}^n$	a vector concatenating the rates of leg angles	$f_i \in \mathbb{R}$	thrust force magnitude of quadrotor i
$\mathbf{r}_i, \mathbf{l}_i \in \mathbb{R}^3$	unit vectors of directions $\overrightarrow{PA_i}$, $\overrightarrow{A_iB_i}$	$\mathbf{0}_k$	a k-dimensional vector of zeros
$r, l \in \mathbb{R}$	platform radius (length of $\overrightarrow{PA_i}$) and leg length	$\mathbf{1}_{k\times k}$	an identity matrix of dimension $(k \times k)$

where $\mathbf{p}_p \in \mathbb{R}^3$ and $\mathfrak{q}_p \in \mathbb{H}$ are respectively the 3-dimensional position and unit quaternion of the platform with respect to $\mathfrak{F}_0$, $\boldsymbol{\theta}_l = [\theta_1, \theta_2, ..., \theta_n]^T \in \mathbb{R}^n$ is a vector concatenating the revolute-joint angles of the legs. Notation for relevant variables in this paper is given in Table 1. Assuming that the center of the spherical joint is located the CoM of the quadrotor, the geometric relationship between the robot pose and the position of the i^{th} quadrotor can be derived as

$$\mathbf{p}_i = \mathbf{p}_p + r \cdot \mathbf{r}_i + l \cdot \mathbf{l}_i \tag{2}$$

where the unit vectors $\mathbf{r}_i$ and $\mathbf{l}_i$ are expressed in $\mathfrak{F}_0$. They can be calculated respectively by $\mathbf{r}_i = \mathbf{R}_p {}^p\mathbf{r}_i$ and $\mathbf{l}_i = \mathbf{R}_p {}^p\mathbf{R}_{li} {}^{li}\mathbf{l}_i$, with ${}^p\mathbf{r}_i$ and ${}^{li}\mathbf{l}_i$ being two constant vectors, ${}^p\mathbf{R}_{li}$ a rotation matrix between $\mathfrak{F}_p$ and $\mathfrak{F}_{li}$ as a function of θ_i, and $\mathbf{R}_p$ the rotation matrix of the platform converted from the unit quaternion $\mathfrak{q}_p$. By concatenating relationships Eq. (2) for all the quadrotors, the Inverse Geometric Model (IGM) of the FPR can be expressed by

$$\mathbf{p} = IGM(\mathbf{q}) \tag{3}$$

with $\mathbf{p} = [\mathbf{p}_1^T, \mathbf{p}_2^T, ..., \mathbf{p}_n^T]^T \in \mathbb{R}^{3n}$ a vector concatenating the 3-dimensional positions of all the quadrotors in $\mathfrak{F}_0$.

Deriving Eq. (3), the Inverse Kinematic Model (IKM) can be obtained as

$$\mathbf{v} = \mathbf{J}(\mathbf{q})\boldsymbol{\nu} \tag{4}$$

where $\mathbf{v} = [\mathbf{v}_1^T, \mathbf{v}_2^T, ..., \mathbf{v}_n^T]^T \in \mathbb{R}^{3n}$ is a vector of linear velocities of all the quadrotors in $\mathfrak{F}_0$, $\mathbf{J}(\mathbf{q}) \in \mathbb{R}^{3n\times(6+n)}$ is a Jacobian matrix, and $\boldsymbol{\nu}$ is the robot velocity given by

$$\boldsymbol{\nu} = \begin{bmatrix} \mathbf{v}_p^T & {}^p\boldsymbol{\omega}_p^T & \dot{\boldsymbol{\theta}}_l^T \end{bmatrix}^T \in \mathbb{R}^{6+n} \tag{5}$$

with $\mathbf{v}_p, {}^p\boldsymbol{\omega}_p \in \mathbb{R}^3$ being the linear and angular velocity of the platform expressed respectively in $\mathfrak{F}_0$ and $\mathfrak{F}_p$, and $\dot{\boldsymbol{\theta}}_l \in \mathbb{R}^n$ the derivatives of the leg angles. Knowing the relationship of Eq. (3), the robot velocity can be computed from the measurements of quadrotors' linear velocities by the Forward Kinematic Model (FKM) as

$$\boldsymbol{\nu} = \mathbf{J}(\mathbf{q})^{\dagger}\mathbf{v} \tag{6}$$

with $\mathbf{J}(\mathbf{q})^{\dagger}$ representing the pseudo-inverse of the Jacobian matrix. Note that to fully actuate the system, the condition of $n \geq 3$ should be satisfied. When the Jacobian matrix is square ($n = 3$), the relation holds true if and only if the Jacobian matrix is not singular (i.e. $\det(\mathbf{J}) \neq 0$).

The dynamic model of the FPR taking into account the external wrench denoted by $\boldsymbol{\tau}_e$ can be described in a matrix form

$$\mathbf{M}(\mathbf{q})\dot{\boldsymbol{\nu}} + \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} + \boldsymbol{\tau}_e \tag{7}$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{(6+n)\times(6+n)}$ is the generalized inertia matrix, $\mathbf{C}(\mathbf{q}, \boldsymbol{\nu}) \in \mathbb{R}^{(6+n)\times(6+n)}$ is the Coriolis matrix, $\mathbf{g}(\mathbf{q}) \in \mathbb{R}^{6+n}$ is the gravity vector, and $\boldsymbol{\tau}$ is the actuation wrench defined by $\boldsymbol{\tau} = [\mathbf{f}_p^T, {}^p\mathbf{m}_p^T, \mathbf{m}_l^T]^T \in \mathbb{R}^{6+n}$. The actuation wrench can be linked to the thrust forces of the quadrotors by

$$\boldsymbol{\tau} = \mathbf{J}(\mathbf{q})^T\mathbf{f} \tag{8}$$

with $\mathbf{f} = [\mathbf{f}_1^T, \mathbf{f}_2^T, ..., \mathbf{f}_n^T]^T \in \mathbb{R}^{3n}$ a vector concatenating the 3-dimensional thrust forces of all the quadrotors in $\mathfrak{F}_0$. Note that the dynamic modelling is accomplished using Spatial Vector notation (Featherstone, 2008), with the Coriolis matrix computed based on a numerical algorithm detailed in Echeandia and Wensing (2021).

2.2. External wrench estimation

Before constructing the interaction control law, a momentum based technique is applied to build an estimate of the external wrench acting on the robot. A momentum-based wrench estimation technique can be adopted according to Liu et al. (2022), which is summarized as follows.

Let $\hat{\boldsymbol{\tau}}_e$ denote an estimate of the external wrench and $\boldsymbol{\mathcal{P}} = \mathbf{M}(\mathbf{q})\boldsymbol{\nu}$ be the generalized momentum of the robot. The time-domain expression of the estimated external wrench can be determined based on the computation of the momentum as

$$\begin{aligned}\hat{\boldsymbol{\tau}}_e(t) = \mathbf{K}_O\Big[\boldsymbol{\mathcal{P}}(t) - \int_{t_0}^{t} \Big(\mathbf{C}(\mathbf{q}, \boldsymbol{\nu})^T\boldsymbol{\nu} - \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau}(t) \\ + \hat{\boldsymbol{\tau}}_e(t - \Delta t)\Big)dt - \boldsymbol{\mathcal{P}}(t_0)\Big]\end{aligned} \tag{9}$$

where $\boldsymbol{\mathcal{P}}(t) = \mathbf{M}\big(\mathbf{q}(t)\big)\boldsymbol{\nu}(t)$ is the momentum at time instant t with $\boldsymbol{\mathcal{P}}(t_0)$ initialized by zero, $\hat{\boldsymbol{\tau}}_e(t)$ is the estimate at current timestamp which depends on the previous estimate $\hat{\boldsymbol{\tau}}_e(t-\Delta t)$, and $\mathbf{K}_O$ is a positive-definite diagonal matrix for the estimation gains. By approximating any two consecutive time steps (supposing Δt small enough), the estimation of external wrench by Eq. (9) is equivalent to a first-order filtering towards the real values (Liu et al., 2022).

2.3. Interaction controller

Based on the estimated external wrench, a reaction behavior of the FPR can be designed during the interaction with the environment. An impedance-based control law with desired wrench tracking ability is implemented in Liu et al. (2022), in which a generalized form of the desired impedance behavior of the robot can be chosen as

$$\mathbf{M}^d(\dot{\boldsymbol{\nu}}^d - \dot{\boldsymbol{\nu}}) + \mathbf{B}^d(\boldsymbol{\nu}^d - \boldsymbol{\nu}) + \mathbf{K}^d\boldsymbol{\varepsilon}_q(\mathbf{q}^d, \mathbf{q}) = \boldsymbol{\varepsilon}_\tau \quad (10)$$

with $\mathbf{M}^d$, $\mathbf{B}^d$ and $\mathbf{K}^d \in \mathbb{R}^{(6+n)\times(6+n)}$ being the positive-definite diagonal matrices respectively for the desired mass, damping and stiffness coefficients, $\boldsymbol{\varepsilon}_\tau = -\hat{\boldsymbol{\tau}}_e - \boldsymbol{\tau}_e^d$ representing the tracking error of the interaction wrench, and $\boldsymbol{\varepsilon}_q(\mathbf{q}^d, \mathbf{q})$ the tracking error of the robot pose determined in Liu, Erskine, Chriette and Fantoni (2021).

Based on the desired impedance in Eq. (10), an auxiliary input corresponding to the acceleration of the system can be determined by

$$\mathbf{u} = \dot{\boldsymbol{\nu}}^d + (\mathbf{M}^d)^{-1}\big(\mathbf{B}^d(\boldsymbol{\nu}^d - \boldsymbol{\nu}) + \mathbf{K}^d\boldsymbol{\varepsilon}_q(\mathbf{q}^d, \mathbf{q}) - \boldsymbol{\varepsilon}_\tau\big) \quad (11)$$

which is then applied to the dynamic model in Eq. (7) to obtain the thrust force commands as

$$\mathbf{f} = \big[\mathbf{J}(\mathbf{q})^T\big]^\dagger\Big(\mathbf{M}(\mathbf{q})\mathbf{u} + \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\mathbf{q}) - \hat{\boldsymbol{\tau}}_e\Big) \quad (12)$$

with $\big[\mathbf{J}(\mathbf{q})^T\big]^\dagger$ being the pseudo-inverse of the transpose of the Jacobian matrix. When $n = 3$, the Jacobian matrix is square and the pseudo-inverse is replaced by a simple matrix inversion, which necessitates $\det(\mathbf{J}) \neq 0$.

We remark that the achievement of thrust force for each quadrotor is handled by the onboard flight controller tracking the attitude and thrust magnitude commands $(\mathfrak{q}_i^d, f_i^d)$ computed from each 3-dimensional thrust force vector $\mathbf{f}_i$ using the method presented in Liu et al. (2021). It is additionally remarked that this control method requires high-frequency and stable measurements of the robot pose and velocity relative to the global reference frame $\mathfrak{F}_0$, which is usually known by external localization systems, meaning for indoor applications the dependence on Motion Capture (MOCAP) system.

3. Vision-based state estimation

In this section, we present the robot state estimation based on the vision-based method using ArUco marker system that can be deployed onboard each quadrotor to avoid using any external localization systems. We hereby demonstrate how to construct the state estimation using real-world data from the onboard visual and inertial measurements.

3.1. Relative pose measurement

Supposing that each quadrotor can estimate its pose relative to the platform frame $\mathfrak{F}_p$, the robot pose can be partially reconstructed, consisting in roll and pitch of the platform and the internal configuration (i.e. leg angles). We therefore implement a real-time pose estimation process using ArUco marker detection algorithm (Romero-Ramirez, Muñoz-Salinas and Medina-Carnicer, 2021). This vision-based estimation method requires the visual marker to be always visible to the camera, an assumption satisfied by carefully mounting the camera on the quadrotor pointing towards the marker.

Consider that an ArUco marker is attached below the platform, which has a marker frame denoted by $\mathfrak{F}_M$ (see Fig. 3(a)). The marker can be detected and tracked by the ArUco detection algorithm using the monocular camera attached onboard each quadrotor, for which the relative pose of the marker expressed in the camera frame (given by $\mathfrak{F}_{Ci}$) is estimated and can be represented by a homogeneous matrix written as

$${}^{Ci}\mathbf{T}_M = \begin{bmatrix} {}^{Ci}\mathbf{R}_M & {}^{Ci}\mathbf{p}_M \\ \mathbf{0}_3^T & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \quad (13)$$

where ${}^{Ci}\mathbf{R}_M \in SO(3)$ and ${}^{Ci}\mathbf{p}_M \in \mathbb{R}^3$ are respectively the rotation matrix and the translation vector for the relative pose of the marker expressed in quadrotor i's camera frame. Note that $\mathbf{0}_3$ is a 3-dimensional column vector of zeros.

Then, by further multiplying two fixed transformations ${}^p\mathbf{T}_M$ and ${}^{Ci}\mathbf{T}_i$ calibrated beforehand, which respectively represent the marker pose relative to $\mathfrak{F}_p$ and the camera pose relative to $\mathfrak{F}_i$, we can obtain an estimate of the platform pose relative to the body-fixed frame $\mathfrak{F}_i$ of quadrotor i by

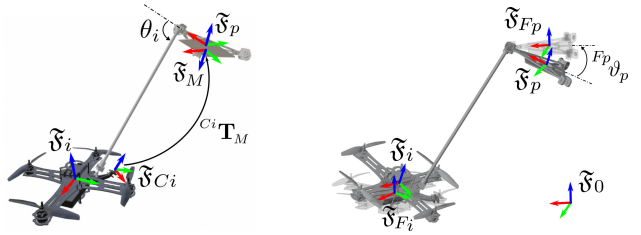
$${}^i\mathbf{T}_p = ({}^{Ci}\mathbf{T}_i^{-1}){}^{Ci}\mathbf{T}_M({}^p\mathbf{T}_M^{-1}) \quad (14)$$

with ${}^{Ci}\mathbf{T}_i^{-1}$ and ${}^p\mathbf{T}_M^{-1}$ being the inverse transformations, knowing that for a homogeneous transformation ${}^A\mathbf{T}_B$ between two arbitrary frames $\mathfrak{F}_A$ and $\mathfrak{F}_B$, its inverse is given by

$${}^A\mathbf{T}_B^{-1} = {}^B\mathbf{T}_A = \begin{bmatrix} {}^A\mathbf{R}_B^T & -{}^A\mathbf{R}_B^T{}^A\mathbf{p}_B \\ \mathbf{0}_3^T & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \quad (15)$$

The inverse transformation of Eq. (14), ${}^p\mathbf{T}_i = {}^i\mathbf{T}_p^{-1}$, allows each quadrotor to locate itself commonly in the platform frame.

Before reconstructing the robot pose from knowledge of ${}^p\mathbf{T}_i$, the estimated relative pose needs to be further filtered to be able to output at a constant frequency with reduced estimation noises and errors that might occur due to the issues commonly found in vision systems such as motion blur, unfavourable lightening conditions and loss of objects in the camera's angle of view. To achieve this, an Extended Kalman Filter (EKF)-based algorithm is adopted and detailed as follows.



(a) (b)

Figure 3: Definition of frames for the pose estimation on each quadrotor. (a) frames for the detected relative pose and the fixed transformations; (b) flat frames attached to the platform and each quadrotor i. Note that the transparent bodies represent the poses in their respective flat frames.

3.2. EKF-based pose estimation

The relative pose initially given by ${}^p\mathbf{T}_i$ is rearranged by a vector defined as the system state of the EKF

$$\mathbf{x} = \begin{bmatrix} {}^p\mathbf{p}_i^T & {}^p\mathbf{q}_i^T \end{bmatrix}^T \in \mathbb{R}^7 \tag{16}$$

with ${}^p\mathbf{q}_i \in \mathbb{H}$ a unit quaternion converted from the rotation matrix ${}^p\mathbf{R}_i$. To develop the prediction model in the EKF, two reasonable hypotheses are made:

- ${}^p\dot{\mathbf{p}}_i$ is supposed to be zero at each instant, since each quadrotor is rigidly attached to the passive structure, and its relative position is constant with non-changing (or slow movements of) leg angles.
- ${}^p\dot{\mathbf{q}}_i$ is only affected by the angular velocity of each quadrotor, as the rotational movements are fully decoupled with the passive architecture and the angular velocity of the platform is negligible compared to the fast rotational dynamic response of quadrotors.

Consider now the system equations for the prediction and correction models of the EKF written as (Sabatini, 2006)

$$\begin{aligned} \mathbf{x}_k &= \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_{k-1} \\ \mathbf{z}_k &= \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \end{aligned} \tag{17}$$

where $\mathbf{u}_k$ is the system input being the angular velocity ${}^i\boldsymbol{\omega}_i$ of the quadrotor i for each time instant k, and $\mathbf{w}_{k-1}$, $\mathbf{v}_k$ are vectors respectively for the prediction and measurement noises modeled by Gaussian noises as $\mathbf{w}_{k-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1})$ and $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$ with $\mathbf{Q}_{k-1}, \mathbf{R}_k$ the covariance matrices.

Based on the hypotheses, the prediction model and the state transition matrix can be determined by

$$\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k) = \mathbf{x}_{k-1} + \begin{bmatrix} \mathbf{0} \\ \frac{1}{2}\, {}^p\mathbf{q}_i \circ \begin{vmatrix} 0 \\ {}^i\boldsymbol{\omega}_i \end{vmatrix} \end{bmatrix} T_s \tag{18}$$

$$\mathbf{F}_k = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}_{k-1}, \mathbf{u}_k} = \begin{bmatrix} \mathbf{1}_{3\times3} & \mathbf{0}_{3\times4} \\ \mathbf{0}^T_{3\times4} & \mathbf{1}_{4\times4} + \boldsymbol{\Omega}({}^i\boldsymbol{\omega}_i) T_s \end{bmatrix} \tag{19}$$

where T_s is the time step of the prediction process, and $\boldsymbol{\Omega}(.)$ is a (4×4) skew-symmetric matrix associated to a vector of angular velocity $\boldsymbol{\omega}$ commonly found in quaternion-based EKF algorithms (Sabatini, 2006), given by

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) = \frac{1}{2} \begin{bmatrix} 0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & -[\boldsymbol{\omega}]_\times \end{bmatrix} \in \mathbb{R}^{4\times4} \tag{20}$$

with $[.]_\times \in \mathbb{R}^{3\times3}$ being a conventional skew-symmetric matrix of a 3-dimensional vector. Remark that the evolution of quaternion between two iterations in Eq. (18) and Eq. (19) is written by the first-order and second-order of Taylor series expansion of $\exp\left(\boldsymbol{\Omega}({}^i\boldsymbol{\omega}_i) T_s\right)$, as $\boldsymbol{\Omega}({}^i\boldsymbol{\omega}_i)$ is assumed to be constant within the time interval T_s, when T_s is small enough.

The correction model and the observation matrix is by the way easy to derive as the full vector of the system state is considered to be directly measured, which gives

$$\mathbf{h}(\mathbf{x}_k) = \mathbf{x}_k, \quad \mathbf{H}_k = \frac{\partial \mathbf{h}}{\partial \mathbf{x}}\Big|_{\mathbf{x}_k} = \mathbf{1}_{7\times7} \tag{21}$$

3.3. Robot state reconstruction

After having estimated and filtered the relative pose of each quadrotor expressed in $\mathfrak{F}_p$, we now focus on how to recover some variables of the robot state that could be sufficient for the control.

The internal configuration (i.e. the leg angle) for each quadrotor was previously computed using the geometric model knowing the global pose of the platform and global positions of quadrotors. It can now be computed with only knowledge of relative position ${}^p\mathbf{p}_i$ by

$$\theta_i = \arccos\left(\frac{{}^p\mathbf{r}_{A_iB_i} \cdot {}^p\mathbf{p}_{A_i}}{\|{}^p\mathbf{r}_{A_iB_i}\| \cdot \|{}^p\mathbf{p}_{A_i}\|} \right) \tag{22}$$

with ${}^p\mathbf{p}_{A_i}$ representing the position of the revolute joint's center expressed in $\mathfrak{F}_p$, and ${}^p\mathbf{r}_{A_iB_i} = {}^p\mathbf{p}_i - {}^p\mathbf{p}_{A_i}$ being a vector describing the leg's direction expressed in $\mathfrak{F}_p$.

For representing the orientation of a rigid body without any global reference frame, we introduce a notion of flat frames similarly as in Liu et al. (2021). It is a frame (denoted by $\mathfrak{F}_{Fj}$ attached to an arbitrary body j) defined with zero roll and pitch relative to a global reference frame $\mathfrak{F}_0$, remaining an unknown yaw with respect to $\mathfrak{F}_0$ (as shown in Fig. 3(b)). Therefore, the roll and pitch of a body can be expressed with respect to its flat frame, whose values are the same as the ones represented in any global reference frame $\mathfrak{F}_0$. We remark that the roll and pitch of each quadrotor can be sufficiently well estimated by the onboard IMU sensor and the filtering technique, which are denoted by ${}^{Fi}\boldsymbol{\phi}_i$ and ${}^{Fi}\vartheta_i$ expressed in its flat frame $\mathfrak{F}_{Fi}$. Along with the relative orientation between $\mathfrak{F}_p$ and $\mathfrak{F}_i$ estimated and filtered onboard, the platform's orientation is known by the relationship

$${}^{Fi}\mathbf{R}_p = {}^{Fi}\mathbf{R}_i\, {}^i\mathbf{R}_p \tag{23}$$

where ${}^i\mathbf{R}_p$ is the rotation matrix extracted from ${}^i\mathbf{T}_p$ and ${}^{Fi}\mathbf{R}_i = \mathbf{R}_y({}^{Fi}\vartheta_i)\mathbf{R}_x({}^{Fi}\boldsymbol{\phi}_i)$ is the rotation matrix computed from unit rotations (in x and y axes) for roll and pitch angles

of quadrotor i. Finally, the roll and pitch of the platform expressed in its flat frame can be reconstructed by

$$\begin{aligned} {}^{Fp}\phi_p &= \operatorname{atan2}\left({}^{Fi}\mathbf{R}_p(3,2), {}^{Fi}\mathbf{R}_p(3,3)\right) \\ {}^{Fp}\vartheta_p &= -\operatorname{asin}\left({}^{Fi}\mathbf{R}_p(3,1)\right) \end{aligned} \tag{24}$$

where $\mathbf{R}(r,c)$ refers to the element at the r-th row and c-th column of the matrix.

We notice that the flat frame attached to the platform can be served as a common frame to express all the other states of each quadrotor i, for which the transformation matrix from $\mathfrak{F}_i$ to $\mathfrak{F}_{Fp}$ is computed by

$${}^{Fp}\mathbf{T}_i = {}^{Fp}\mathbf{T}_p {}^{p}\mathbf{T}_i \tag{25}$$

where the translation vector of ${}^{Fp}\mathbf{T}_p$ is zero and the rotation matrix of ${}^{Fp}\mathbf{T}_p$ is similarly computed by ${}^{Fp}\mathbf{R}_p = \mathbf{R}_y({}^{Fp}\vartheta_p)\mathbf{R}_x({}^{Fp}\phi_p)$. From the rotation matrix ${}^{Fp}\mathbf{R}_i$, the quadrotor i's attitude expressed in $\mathfrak{F}_{Fp}$ represented by the unit quaternion ${}^{Fp}\mathbf{q}_i$ can be also computed. This further allows to express the linear velocities of all the quadrotors commonly in $\mathfrak{F}_{Fp}$ by

$${}^{Fp}\mathbf{v}_i = {}^{Fp}\mathbf{R}_i {}^{i}\mathbf{v}_i \tag{26}$$

which are then used to compute the velocity vector of the FPR by the FKM given in Eq. (6), in which the linear velocity of the platform is now expressed in $\mathfrak{F}_{Fp}$.

4. Decentralized control

In this section, we focus on how to deploy the estimation and control methods in a decentralized manner allowing each quadrotor to perform the state estimation and its control based on the onboard and intrinsic measurements.

4.1. Control problem formulation

From the previous section, we have reconstructed a partial set of robot pose (i.e. the roll and pitch of the platform and the leg angles), with the position and yaw of the platform relative to any global reference frame remained unknown. We remark however that these unknown variables are equivalent to the flat outputs of a single quadrotor (Mellinger and Kumar, 2011), which are commonly controllable by their derivatives (such as linear velocity and yaw rate) for a single quadrotor. We further remark that the first-order derivatives of the unknown variables of the platform can be computed using the kinematic model as presented in Section 3.3. Therefore, we choose the variables on which we have the direct control as a vector

$$\boldsymbol{\chi} = \begin{bmatrix} {}^{Fp}\mathbf{v}_p^T & {}^{Fp}\dot{\psi}_p & {}^{Fp}\vartheta_p & {}^{Fp}\phi_p & \boldsymbol{\theta}_l^T \end{bmatrix}^T \tag{27}$$

where the yaw rate of the platform ${}^{Fp}\dot{\psi}_p$ can be computed from the angular velocity $\boldsymbol{\omega}_p$ for which the reconstruction is detailed in Section 3.

We remark that the control problem related to driving the controllable variables of Eq. (27) to some desired values can be referred to as an easily teleoperable system, since the platform roll and pitch and the leg angles can be regulated by desired values, while a human operator may focus on control of the 3-dimensional position and heading of the platform by sending the velocity commands in $\mathfrak{F}_{Fp}$ independent of changes in platform angles. To further ensure the smoothness of the evolution of the robot states, the teleoperation commands (denoted by $\boldsymbol{\chi}^d$) sent by the operator are further processed by a simple low-pass filter, which outputs the desired derivative commands $\dot{\boldsymbol{\chi}}^d$ to the controller as well. Once received the teleoperation commands composed of $\boldsymbol{\chi}^d$ and $\dot{\boldsymbol{\chi}}^d$ on the controller side, these commands can be parsed to extract the desired trajectory data (i.e. $\mathbf{q}^d$, $\boldsymbol{\nu}^d$ and $\dot{\boldsymbol{\nu}}^d$). Meanwhile, it is also expected that the operator provides the desired interaction wrench $\boldsymbol{\tau}_e^d$ to the robot.

4.2. Communication-based decentralized control

On each quadrotor, the control law itself is based on the interaction controller presented in Section 2.3. However, the wrench estimation and impedance controller require the knowledge of some states of the other quadrotors. One may remark that this information can be known from the desired values and setpoints for these variables, as discussed in Non-Communicating controller in Liu et al. (2021). However, the sharing of information is obligated to build an unbiased estimation of the external wrench, which is necessary to ensure the stability of the overall control system. Therefore, we introduce for each quadrotor i an information vector composed of the minimum set of variables to share with the other quadrotors

$$\boldsymbol{\zeta}_i = \begin{bmatrix} \theta_i & {}^{Fp}\mathbf{v}_i^T & {}^{Fp}\mathbf{q}_i^T & f_i^d \end{bmatrix}^T \tag{28}$$

where θ_i is the estimated leg angle attached with the quadrotor i, ${}^{Fp}\mathbf{v}_i$ and ${}^{Fp}\mathbf{q}_i$ are measurements of quadrotor i's linear velocity and attitude expressed in $\mathfrak{F}_{Fp}$, and f_i^d is the thrust magnitude command that has been most recently sent to the low-level flight controller.

From f_i^d and ${}^{Fp}\mathbf{q}_i$ shared among the quadrotors, the system input $\mathbf{f}$ computed in the latest control iteration can be recovered and used to estimate the external wrench in the momentum-based observer. It is noted that to reduce the estimation error, we share the actual attitude of the quadrotor instead of the desired values. The actual thrust magnitude produced by each quadrotor might not perfectly corresponds to the desired command, resulting that any mismatch between the actual and desired thrust will be considered as a disturbance and appear in the external wrench estimates. Regarding the potential communication delays when sharing the information among the UAVs, we should make sure that the communication frequency must be higher enough (at least equal to or greater than the control frequency) such that the synchronization of measurements can be negligible.

Once the 3-dimensional thrust force ${}^{Fp}\mathbf{f}_i$ (now expressed in $\mathfrak{F}_{Fp}$) for each quadrotor is computed from the high-level

control law in Eq. (12), the desired commands for the low-level flight controller can be determined (Liu et al., 2021), which is given by a vector

$$\mathbf{u}_i = \begin{bmatrix} f_i^d & {}^{Fp}\mathbf{q}_i^{dT} \end{bmatrix}^T \tag{29}$$

and achieved by the onboard thrust/attitude controller such as the one presented in Brescianini, Hehn and D'Andrea (2013).

4.3. Adjustment on attitude commands

As all the quadrotors do not share a common reference frame now, the attitude command for each quadrotor ${}^{Fp}\mathbf{q}_i^d$ expressed in $\mathfrak{F}_{Fp}$ should be further adjusted to be coherent with its onboard attitude. We consider that the onboard attitude of each quadrotor is expressed in an unknown inertial frame denoted by $\mathfrak{F}_{0i}$, which shares common roll and pitch angles with those expressed in $\mathfrak{F}_{Fp}$ (angles along axes perpendicular to the gravity vector) but a different yaw. Therefore, the correction of attitude commands is simply to correct the yaw setpoints which should be coherent with those of onboard measurements.

We can know from ${}^{Fp}\mathbf{R}_i$ a relative yaw between quadrotor i's frame and $\mathfrak{F}_{Fp}$, denoted by ${}^{Fp}\psi_i$, and a measurement of quadrotor i's yaw denoted by ${}^{0i}\psi_i$ expressed in its unknown reference frame. The corrected yaw setpoint expressed in $\mathfrak{F}_{0i}$ can therefore be computed by

$${}^{0i}\psi_i^d = {}^{0i}\psi_i + ({}^{Fp}\psi_i^d - {}^{Fp}\psi_i) \tag{30}$$

with ${}^{Fp}\psi_i^d$ the desired yaw originally obtained from the desired attitude ${}^{Fp}\mathbf{q}_i^d$.

4.4. Discussion on the overall control

The overall control system is distributed onboard each quadrotor, with the decentralized control scheme depicted in Fig. 4. The stability of the overall control system to the external disturbances can be ensured as long as the estimation errors remain bounded (Liu et al., 2022). This assumption can be considered true as all the UAVs share their most recent attitudes and thrust magnitude commands, making the estimation error performed by each individual quadrotor being bounded. The estimation errors due to measurement noises or communication delays on the robot states might further affect the stability of the system, which is however not studied in this paper.

5. Experimental results

In this section, we present the implementation and experimental results of the proposed methods applied to the FPR for interacting with the environment. A video of experiments can be found at `https://youtu.be/eTaze1PvIP0`.

5.1. Experimental setup

The prototype of the FPR studied within this work is composed of a triangle-form platform and three legs attached with quadrotors, as shown in Fig. 1. The aluminium platform

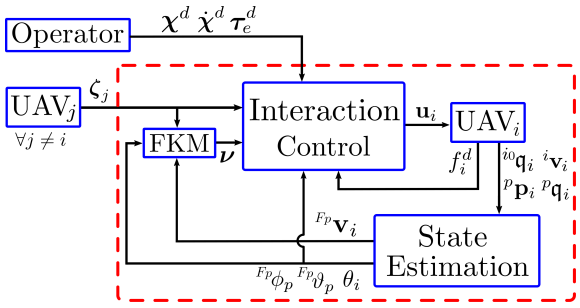


Figure 4: Decentralized control diagram. The blocks within the red dashed rectangle are implemented on each UAV. The interaction control block includes the external wrench estimation and impedance-based controller.

weighs 250 g and the rigid legs are each 1.043 m long with a mass of 66 g. The custom quadrotors used in the FPR are based on a 34-cm Lynxmotion Crazy2Fly frame, controlled by an open-source flight autopilot which is based on a custom-built PX4 firmware (software) and Pixhawk 5 (hardware) (Meier, Honegger and Pollefeys, 2015). The onboard flight controller has been experimentally tuned with aggressive attitude and angular rate gains to ensure rapid convergence to the attitude setpoints. On each quadrotor, a Raspberry Pi 4B is mounted as a companion computer to handle the communication with the Pixhawk 5 via Ethernet by UDP protocol. The communication between quadrotors is handled by ROS2 Galactic, through a 5-GHz Wifi network.

The ArUco marker with ID 0 and the size of 0.109 m is selected from the dictionary "ARUCO_MIP_36h12". Images are captured by the onboard Raspberry Pi V2 cameras at 20 Hz with the relative pose estimated by ArUco detection algorithm and filtered by EKF at the same rate. The interaction controller with external wrench estimator is run onboard each quadrotor at 50 Hz, with information shared among the quadrotors at the same rate. The onboard flight controller is run at 250 Hz.

During the experiments, the FPR was flown by teleoperation using a 4-axis gamepad controller inside a 4×6×3.5 m flight arena equipped with an 8-camera Qualisys MOCAP system. The MOCAP provides the ground truth for post-flight analysis. We remark that MOCAP data is used for several technical reasons that can be totally avoided in future works

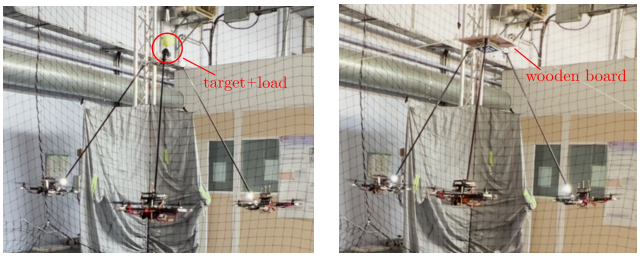
- For safety reason: emulating the relative pose ${}^{p}\mathbf{T}_i$ with representative noises in Liu et al. (2021) if the detection is lost for more than 0.5 s (corresponding to 10 control iterations) or a false detection is identified (with translation error >10 cm or rotation error >10°). It may be handled by redundant detection sources in the future.

- For practical reason: emulating high-level perception of the environment (replacing human operator) to obtain the platform velocity commands in x, y and yaw axes in Experiment II. It can be achieved by adding

a perception system with an additional camera on the platform.

- Sensor fusion onboard the quadrotors to improve the linear velocity estimates of each quadrotor with realistic noises added. It could thus be replaced by optical flow (estimating x, y-axis velocities) and distance (for correcting z-axis velocity) sensors. An additional analysis of the impact of velocity estimation noises and errors is given in the appendix.

It should be noted that MOCAP data is used in a way that only the local frame is needed (such as for improving the body-frame linear velocity estimates, and for emulating camera data), which can be replaced by other onboard sensors and cameras, making the proposed methodology still valid for controlling the robot without knowing any global frame associated to an external localization system.



(a) (b)

Figure 5: Experimenting scenarios of proposed state estimation and decentralized control methods. (a) precise positioning and pick-up of load; (b) contact-based interaction with a board.

5.2. Experiment I: precise positioning and pick-up of load

In the first experiment, the platform is teleoperated by a human operator for the precise positioning towards a target (tennis ball) hanging in the air on which a load is suspended corresponding to total weight of about 400 g, as shown in Fig. 5(a). The robot configuration is maintained with a flat platform orientation (zero roll and pitch) and leg angles of 50°, a configuration beneficial to ensure the visibility towards the marker from the onboard cameras of all the quadrotors. The operator thus attempts to pick up the load within 45 s after the take-off.

We first evaluate the real-time pose estimation by ArUco detection algorithm and EKF-based filtering. The results of the estimated, filtered values and the ground truth known by MOCAP are recorded and plotted in Fig. 6, with the root-mean-square error (RMSE) summarized in Table 2. Note that each axis of the relative position and orientation are separately analyzed. In Fig. 6, it can be noticed that the outliers of the estimated values emphasized within the red dots have been well filtered from the EKF outputs. The RMSE of the filtered values relative to the ground truth are all bounded within 3 cm for relative position and 3° for relative orientation, which has been improved compared to the initial estimation results from the ArUco detection algorithm.

The precise positioning of the platform and the pick-up of load are then evaluated. The operator was asked to perform the same positioning task 6 times to reduce the human factor affecting the results. However, from the pilot's point of view, the system is quite easy to manipulate, similar to the manual pilot of a single quadrotor without much difficulties. The results on the evolution of positioning errors for the 6 tasks (5 success and 1 failure) are shown in Fig. 7, with successful tasks well identified by the positioning error being stabilized below 15 cm. The z-axis external force of the platform estimated by one quadrotor during the successful task ① is plotted in Fig. 8, from which the pick-up of the load is illustrated with the estimated values converged to the ground truth of around -4 N. The overall success rate (5/6) of the pick-up tasks by teleoperation seems very encouraging.

In terms of attitude tracking of quadrotors, the results shown in Table 3 and Fig. 9 demonstrate that the low-level attitude control has been well performed. Note that the attitude tracking results are represented by ZYX (yaw-pitch-roll) Euler angles. The yaw angles of quadrotors are estimated from the outputs of the onboard pose estimation process. The magnitude of yaw tracking errors is around 10° due to the camera calibration and estimation errors, which is within acceptable range and doesn't affect the performance of the system due to the characteristic of quadrotors that yaw motion is decoupled with the thrust force the quadrotor can generate.

5.3. Experiment II: contact-based interaction with a board

In the second experiment, the FPR is controlled to interact with a 30×30 cm wooden board in the environment and exert desired contact force on the board. A region of dimension 10×10 cm is pointed out on the surface of the board, with its centre considered as the target for contact-based interactions. Fig. 5(b) illustrates the interaction experiment performed by the FPR. The teleoperation still involves the human in the loop, which is undesired for validating the methodology independent of the proficiency of human operators. Therefore, an emulated pose estimation between the target and the platform is set up in this experiment running at the highest level to generate the desired velocity on x and y axes ${}^{Fp}\mathbf{v}_{p,x/y}$ and yaw rate ${}^{Fp}\dot{\psi}_p$ using the MOCAP data, leaving the z-axis linear velocity of the platform, the desired contact force and robot configuration (i.e. platform orientation and leg angles) controlled by the human operator. This however may be done by adding an additional camera on the platform perceiving the contact surface and estimating the relative pose using model-based or learning-based algorithms.

Fig. 10 shows the evolution of the positioning error between the platform and the target, and Fig. 11 are the desired and estimated values of the contact force between the platform and the board. It can be seen that during the

UAV	Estimation Result	Relative Position [m]			Relative Orientation [°]		
		x	y	z	roll	pitch	yaw
1	ArUco	0.036	0.023	0.038	2.078	2.968	3.728
	EKF	0.017	0.012	0.026	1.272	1.571	3.305
2	ArUco	0.037	0.041	0.044	2.627	1.822	1.942
	EKF	0.019	0.026	0.029	1.302	1.407	1.363
3	ArUco	0.037	0.051	0.040	3.644	2.691	1.678
	EKF	0.014	0.033	0.023	3.048	1.128	0.891

Table 2
Root-mean-square error of the ArUco detection and EKF-based filtering results of the relative pose between each quadrotor and the platform. Results are calculated based on 6 repetitive tasks performed in Experiment I.

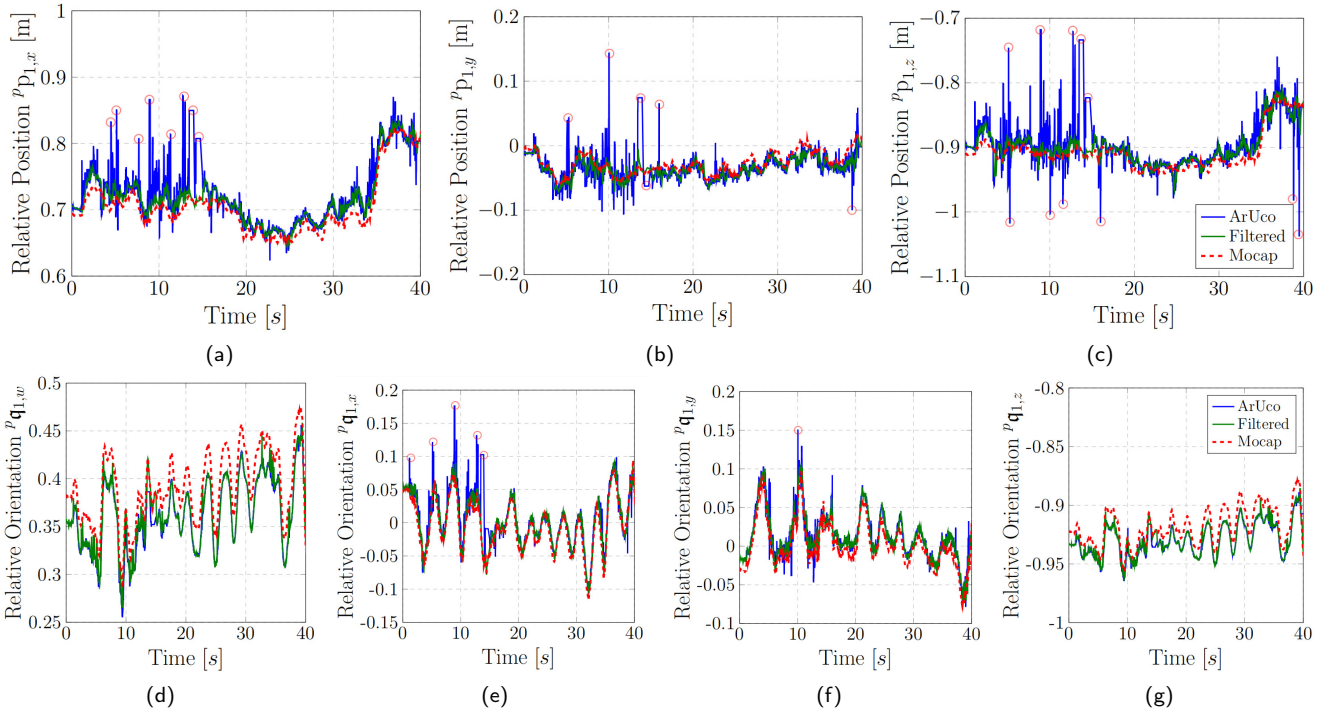


Figure 6: Relative pose of UAV 1 in the platform frame over the course of the first successful test in Experiment I, with ArUco estimates (blue curves), filtered outputs (green curves) and ground truth (red dashed curves). (a)-(c): x, y, z elements of the relative position ${}^p\mathbf{p}_1$, (d)-(g): w, x, y, z elements of the unit quaternion ${}^p\mathbf{q}_1$ representing the relative orientation.

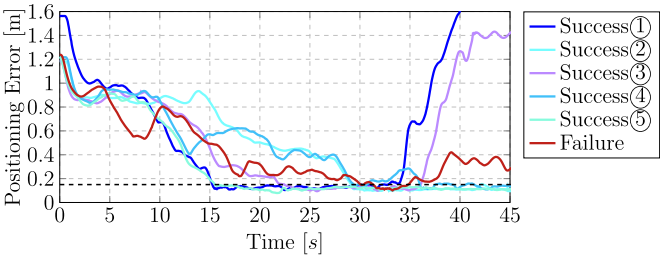


Figure 7: Evolution of 3-dimensional positioning errors of the platform with respect to the target in 6 performed tasks in Experiment I. Note that the black dashed line is the criterion for a successful task (≈ 15 cm). The blue and purple curves (Success ① and ③) are two successful cases where the load was released before the end of 45 s.

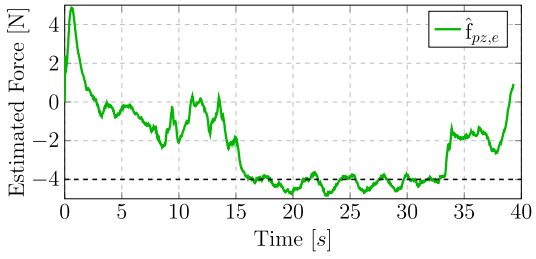
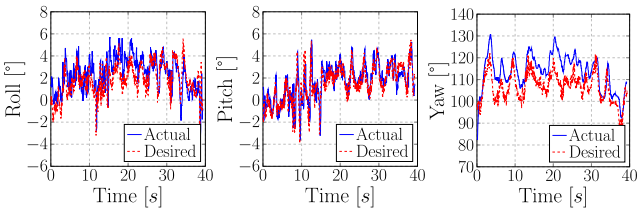


Figure 8: Evolution of the z-axis external force estimates of the platform during the successful experiment (Success ①). Note that the curves of the force estimates of all the quadrotors are overlapped.

steady contact (the time interval of [32, 55] s) the positioning error is stabilized at a value around 5 cm. From Fig. 11, the estimated contact force well tracks the desired values of 5 N during the interaction, except the peak between [27, 32] s probably because an impact occurs at the beginning of the

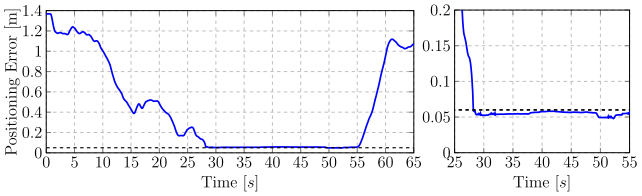


(a) roll (b) pitch (c) yaw

Figure 9: Attitude tracking results of quadrotor 3 during one instance of successful pick-up experiments.

UAV	Roll [°]	Pitch [°]	Yaw [°]
1	1.11	0.94	10.52
2	1.14	1.24	8.62
3	1.39	0.98	8.12

Table 3
Root-mean-square error on the attitude tracking of the quadrotors during one successful test in Experiment I.



(a) (b)

Figure 10: Evolution of 3-dimensional positioning error during the contact-based interaction. (a) positioning error over the whole flight; (b) positioning error during the steady contact.

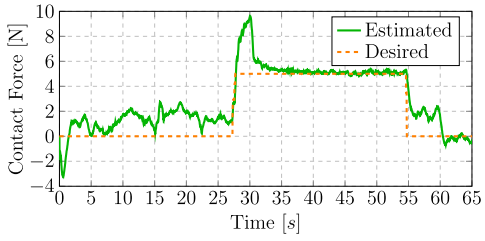


Figure 11: Tracking of desired contact force during the interaction experiment.

contact. The RMSE of the tracking of the other states (i.e. platform roll and pitch and leg angles) are summarized in Table 4, which demonstrates a good maintenance of robot configuration during the interaction experiments. Although the detection failures and interruptions have been notified in the experiments, the percentage of successful detection presented in Table 4 (from 88.3% to 95.3% for all the quadrotors) has been an encouraging progress applying the vision-based state estimation technique in the real-world scenario.

Robot State	Exp. 1	Exp. 2	UAV	Exp. 1	Exp. 2
ϕ_p [°]	5.4	4.6	1	93.1%	95.3%
ϑ_p [°]	5.9	5.9	2	88.3%	89.4%
mean θ_l [°]	5.4	5.3	3	88.7%	91.2%

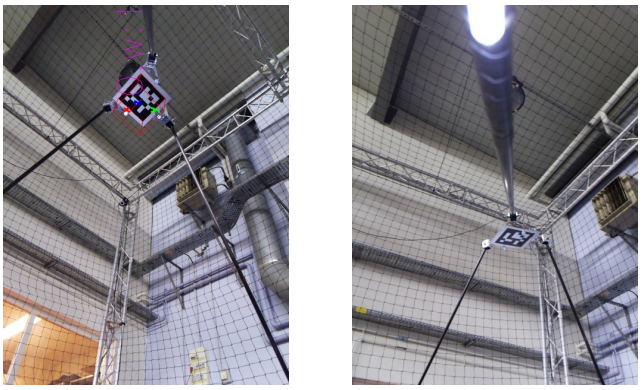
Table 4
Root-mean-square error on the tracking of the platform orientation and leg angles (left) and successful detection rate (right) in two experiments. Note that results of experiment 1 are computed by the average values of 6 performed tasks.

These results have therefore shown that the proposed decentralization of the state estimation and interaction control is capable of positioning the platform with good precision and accomplishing interactions with the environment by the platform. In addition, we have also observed in the experiment that the decentralized controller can better deal with the communication delays or interruptions compared to the centralized one especially in the emergency case where the communications among the quadrotors were lost. While a crash is expected and often experienced with a centralized controller, each quadrotor in a decentralized scheme is still capable of making use of desired states as investigated in Liu et al. (2021) to maintain its own control and land properly without crashing.

5.4. Analysis of failure cases

In real-world applications of our proposed method, there exists failure cases where the vision-based detection and pose estimation technique may be defeated. This is principally due to the fact that the ArUco marker detection system necessitates direct line-of-sight to the marker. In some specific leg configurations, such as when angle between the platform's plane and the leg's direction is large, the marker visibility from the camera attached on the quadrotor is worsen, which results in an increase of the marker detection losses by the ArUco detection algorithm. Fig. 12 demonstrates examples of captured images by the onboard camera of one quadrotor during the experiment, in which Fig. 12a is a successful detection case in normal robot configuration and Fig. 12b shows a failure case with a wide leg configuration.

The marker visibility issue may potentially be resolved by carefully regulating the leg angles within a safe interval (such as between 35° and 65°), which was validated during the experiments conducted in this work. This is a reasonable assumption since the potential application of the FPR consists in exerting forces onto the external environment with a configuration where the platform is above the quadrotors. An opposite configuration (platform underneath quadrotors) has less interests and the transitions between two sets of configuration require the crossing of singularity, which is totally out of the scope of this paper. However, the failure cases related to environmental conditions such as brightness changes still present the frustrating aspect of the vision-based method. In the experiments of this work, we must have favorable lightening conditions from the side way of



(a) (b)

Figure 12: Images captured by the onboard camera of one quadrotor. (a) successful detection instance; (b) failure instance.

the robot to ensure the vision-based detection algorithm to be working correctly.

6. Conclusion

In this paper, we present the distributed state estimation technique based on intrinsic measurements from the monocular camera and the IMU to recover the robot states without any external localization system. An impedance-based interaction control method enhanced with the external wrench estimation and desired wrench tracking capability has been deployed in a decentralized manner, allowing the whole system to interact with the environment. The experiments conducted on a flying parallel robot have shown the capability of this robot to accomplish real manipulation tasks, such as contact-based inspection or repair on the surface of industrial infrastructures.

Future works can continue on the implementation of the fully MOCAP-independent FPR equipped with a complete set of sensors to provide enough onboard information. A redundant detection algorithm might also be investigated, using multiple sources of information to recover the robot pose such as the IMU attached to the platform, and/or distance-based measurement units. The perception of environment may be done as well providing high-level commands to alleviate the difficulties related to the teleoperation by human operators.

Conflict of interest

The authors declared no potential conflict of interest with respect to the research, authorship and publication of this work.

Acknowledgment

The authors acknowledge Julian Erskine and Damien Six for their valuable suggestions and support on this work. This work was carried out within the framework of ROBOTEX 2.0, the French Infrastructure in Robotics under the grants ROBOTEX (EQUIPEX ANR-10-EQPX-44-01) and TIRREX (EQUIPEX+ grant ANR-21-ESRE-0015).

References

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

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

Chung, S.J., Paranjape, A.A., Dames, P., Shen, S., Kumar, V., 2018. A Survey on Aerial Swarm Robotics. IEEE Transactions on Robotics 34, 837–855.

Coppola, M., McGuire, K.N., De Wagter, C., de Croon, G.C., 2020. A Survey on Swarming With Micro Air Vehicles: Fundamental Challenges and Constraints. Frontiers in Robotics and AI 7.

Echeandia, S., Wensing, P., 2021. Numerical Methods to Compute the Coriolis Matrix and Christoffel Symbols for Rigid-Body Systems. Journal of Computational and Nonlinear Dynamics 16.

Featherstone, R., 2008. Rigid Body Dynamics Algorithms. Springer.

Gabellieri, C., Tognon, M., Sanalitro, D., Pallottino, L., Franchi, A., 2020. A study on force-based collaboration in swarms. Swarm Intelligence 14, 57–82.

Gassner, M., Cieslewski, T., Scaramuzza, D., 2017. Dynamic collaboration without communication: Vision-based cable-suspended load transport with two quadrotors, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 5196–5202.

Li, G., Ge, R., Loianno, G., 2021a. Cooperative transportation of cable suspended payloads with mavs using monocular vision and inertial sensing. IEEE Robotics and Automation Letters 6, 5316–5323.

Li, Z., Song, X., Bégoc, V., Chriette, A., Fantoni, I., 2021b. Dynamic Modeling and Controller Design of a Novel Aerial Grasping Robot. Springer. volume 601. pp. 538–546.

Liu, S., Erskine, J., Chriette, A., Fantoni, I., 2021. Decentralized Control and Teleoperation of a Multi-UAV Parallel Robot Based on Intrinsic Measurements, in: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Liu, S., Fantoni, I., Chriette, A., Six, D., 2022. Wrench Estimation and Impedance-based Control applied to a Flying Parallel Robot Interacting with the Environment, in: 11th IFAC Symposium on Intelligent Autonomous Vehicles (IAV 2022).

Loianno, G., Kumar, V., 2018. Cooperative transportation using small quadrotors using monocular vision and inertial sensing. IEEE Robotics and Automation Letters 3, 680–687.

Masone, C., Bülthoff, H.H., Stegagno, P., 2016. Cooperative transportation of a payload using quadrotors: A reconfigurable cable-driven parallel robot, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1623–1630.

Meier, L., Honegger, D., Pollefeys, M., 2015. PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms, in: IEEE International Conference on Robotics and Automation (ICRA), pp. 6235–6240.

Mellinger, D., Kumar, V., 2011. Minimum snap trajectory generation and control for quadrotors, in: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 2520–2525.

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.

Ollero, A., Tognon, M., Suarez, A., Lee, D., Franchi, A., 2021. Past, Present, and Future of Aerial Robotic Manipulators. IEEE Transactions on Robotics , 1–20.

Park, S., Lee, Y., Heo, J., Lee, D., 2019. Pose and posture estimation of aerial skeleton systems for outdoor flying, in: 2019 International Conference on Robotics and Automation (ICRA), pp. 704–710.

Romero-Ramirez, F.J., Muñoz-Salinas, R., Medina-Carnicer, R., 2021. Tracking fiducial markers with discriminative correlation filters. Image and Vision Computing 107, 104094.

Ruggiero, F., Lippiello, V., Ollero, A., 2018. Aerial manipulation: A literature review. IEEE Robotics and Automation Letters 3, 1957–1964.

Sabatini, A., 2006. Quaternion-based extended kalman filter for determining orientation by inertial and magnetic sensing. IEEE Transactions on Biomedical Engineering 53, 1346–1356.

Sanalitro, D., Savino, H.J., Tognon, M., Cortes, J., Franchi, A., 2020. Full-pose manipulation control of a cable-suspended load with multiple uavs under uncertainties. IEEE Robotics and Automation Letters 5, 2185 – 2191.

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.

Six, D., Briot, S., Chriette, A., Martinet, P., 2021. Dynamic modelling and control of flying parallel robots. Control Engineering Practice 117, 104953.

Thapa, S., Bai, H., Acosta, J., 2020. Cooperative Aerial Manipulation with Decentralized Adaptive Force-Consensus Control. Journal of Intelligent and Robotic Systems: Theory and Applications 97, 171–183.

Appendix

Analysis of velocity estimation errors

The experimental validation of the proposed method in the paper was still dependent on the Motion Capture (MOCAP) system. One of the major reason was to improve the linear velocity estimates locally measured by the quadrotors IMU sensors which inevitably present the drifts in velocity measurements. Although it is more a technical concern to replace the MOCAP by onboard sensors such as optical flow and distance sensor to correct the body-frame velocity estimates by sensor fusion techniques, it is still important to prove that our method is valid with acceptable range of estimation errors and noises of the quadrotor linear velocities. To achieve this, we conducted ROS-based real-time simulations under Gazebo environment (as shown in Fig. 13), with Gaussian noises added both on the pose estimation and velocity measurements of quadrotors. We considered the relative pose estimation errors as 4 cm in position and 4° in orientation, while the linear velocity estimates are disturbed by Gaussian noises with the standard derivation as 0.05m/s on each axis.

In the simulation, we controlled the FPR by sending the velocity commands to the decentralized controller, equivalently to the real-world experiments we analyzed in the paper. Fig. 14 shows the disturbed linear velocities of three quadrotors in the simulation. The platform velocity and its desired values are presented in Fig. 15, and the tracking of the platform orientation is demonstrated in Fig. 16, which together show that the robot is still well controlled by teleoperation even with the disturbed measurements. In reality, the measurements noises and drifts in the quadrotor linear velocities can be improved by equipping more complete set of onboard sensors, which allow the controller to maintain its performance that has been proven in this paper.

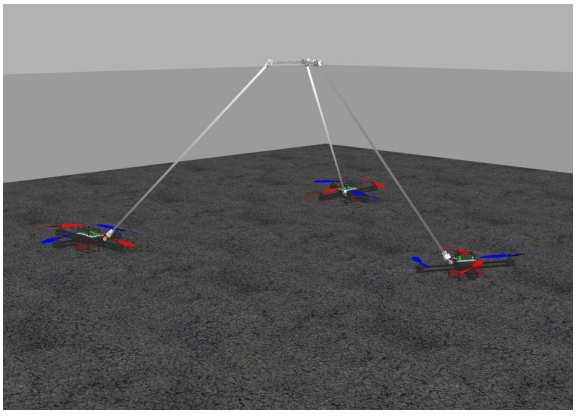
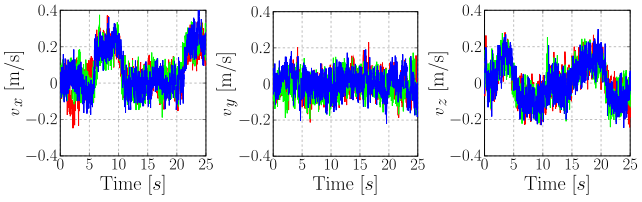
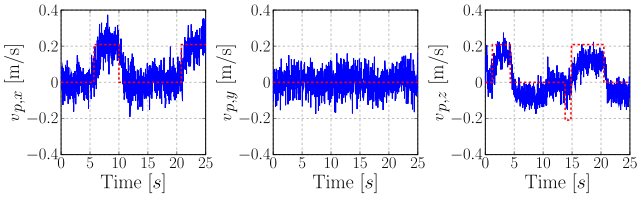


Figure 13: Real-time Gazebo simulation environment of the FPR



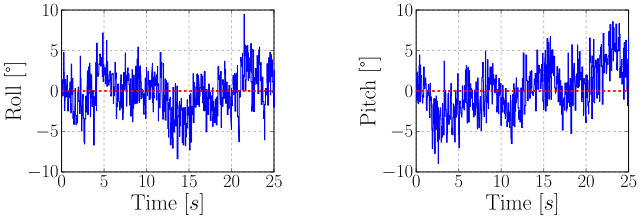
(a) v_x (b) v_y (c) v_z

Figure 14: Disturbed linear velocity estimates of quadrotors in the simulation. The curves in different colors represent results of three different quadrotors (red: quadrotor 1, green: quadrotor 2, blue: quadrotor 3).



(a) $v_{p,x}$ (b) $v_{p,y}$ (c) $v_{p,z}$

Figure 15: Platform linear velocity computed from the linear velocity estimates of quadrotors by kinematic model and its desired values (in red dashed lines).



(a) Platform roll angle (b) Platform pitch angle

Figure 16: Platform roll and pitch angles with their desired values (in red dashed lines).