Decentralized Control and Teleoperation of a Multi-UAV Parallel Robot Based on Intrinsic Measurements

Shiyu Liu, Julian Erskine, Abdelhamid Chriette, Isabelle Fantoni

Decentralized Control and Teleoperation of a Multi-UAV Parallel Robot Based on Intrinsic Measurements

Shiyu Liu[1], Julian Erskine[1], Abdelhamid Chriette[1] and Isabelle Fantoni[2]

***Abstract*— Aerial manipulators have great potential in accomplishing a variety of aerial tasks. One class of aerial manipulators, multi-UAV parallel robots, consists of multiple UAVs connected to a payload or an end-effector by passive kinematic chains. The primary limitation of such aerial manipulators is the dependence on motion capture (MOCAP) systems that provide precise and high-rate exteroceptive pose measurements of all bodies in a common inertial frame, but which are impractical in the majority of real applications.**

This paper proposes a novel methodology of controlling multi-UAV parallel robots, using a Flying Parallel Robot (FPR) as a case study, that could be deployed without a system of external localisation. Intrinsic measurements acquired onboard the UAVs are used to recover a set of robot states that avoid using coordinates derived from a global frame and allow control of the robot by teleoperation. Two decentralized control methods are proposed, based on inter-UAV communicating or non-communicating scenarios. Experiments with intrinsic measurements emulated by MOCAP are carried out to show the performance of the proposed method.

***Index Terms*— Aerial Systems: Mechanics and Control, Multi-Robot Systems, Parallel Robots, Aerial Manipulation**

I. Introduction

Aerial manipulation is an emerging domain where Unmanned Aerial Vehicles (UAVs) are used for grasping, transporting, positioning, assembling and disassembling objects [1]. The ability to perform these tasks considerably expands the range of applications that can be carried out remotely, such as inspection and maintenance in industrial settings, structure construction, or other applications considered dangerous for a human operator [2]. A variety of aerial-manipulator frameworks have been proposed in the literature, such as a single UAV equipped with a robotic arm [3]. Such architectures are restricted by the limited payload of a single UAV [4], and thus feasible tasks are limited to contact-based inspection, grasping and transporting of light objects.

Most recently, new types of aerial manipulators based on parallel mechanical connections between multiple UAVs have been proposed in order to improve the payload capacity and manipulability [4]–[7]. In such designs, UAVs are physically connected to the payload or an end-effector, without adding additional actuators (and thus mass) to the system. The work [5] shows strategies for supporting the payload with cables between a team of quadrotors, however the tensile-only nature of cables reduces the wrench-feasible configurations of these types of aerial manipulators [8]. Multiple UAVs can also be attached physically to a moving platform using rigid mechanical joints to achieve manipulation tasks, as investigated in [4], [6]. To take advantage of full manipulability in $SE(3)$ using quadrotors, a novel Flying Parallel Robot (FPR) was proposed in [7] based on a parallel architecture of rigid links and passive joints.

Most multi-UAV parallel robots are still in the stage of laboratory development and indoor validation, and depend on the use of motion capture (MOCAP) systems for the robot pose measurements in their controls. An exception is [9] where onboard visual and inertial cues are used for leader-follower control of a suspended payload, however this was for a simple bar suspended between two quadrotors, a relatively simple system. Multi-UAV parallel robots in outdoor environments are rare, due to the relatively inaccurate GPS localisation (particularly near buildings or under cover) being insufficient for precise inter-UAV positioning. In order to extend the potential application fields of such aerial manipulators, control schemes based on intrinsic or onboard measurements must be developed to eliminate the need for precise external localisation systems.

The lack of localisation in a common inertial frame is an issue that is treated extensively by the multi-robot control communities for swarming and formation control [10]. In these fields, it is often desirable to treat each robot as an independent agent running its own perception and control algorithms (a so-called decentralized architecture) making for a more intrinsically dependant and robust system [11]. Inspired by this, we take advantage of the fact that all UAVs in multi-UAV parallel robots are connected to a common body (the payload or the end-effector mounted on a moving platform) to assign a common frame with respect to which the individual UAVs can localise and control themselves using onboard measurements.

The novelty of this work is the development of a methodology for controlling aerial manipulators with a multi-UAV parallel architecture (as in [4]–[7]) without exteroceptive localisation. This paper shows

- A method for controlling multi-UAV parallel robots without exteroceptive measurements.
- A comparison of communicating and non-communicating decentralized control laws for the above-mentioned method for the FPR.
- That good positional accuracy can be achieved by teleoperation from a human operator.

The rest of the paper is organized as follows: we begin

Videos of experiments: https://youtu.be/s1on54vG_c8

Julian Erskine is the corresponding author.

[1] École Centrale de Nantes, Laboratoire des Sciences du Numérique de Nantes (LS2N), UMR CNRS 6004, 1 rue de la Noe, 44321 Nantes, France.

[2] LS2N, Centre National de la Recherche Scientifique, France.

Emails: {shiyu.liu, julian.erskine, abdelhamid.chriette, isabelle.fantoni}@ls2n.fr

by presenting the modelling and control of the FPR with a centralized scheme in Section II. We then show in Section III, how each quadrotor is able to reconstruct a partial set of robot states by intrinsic measurements, which together are sufficient to control the FPR. Two decentralized control methods are proposed in Section IV, which allows the platform to be controlled by an human operator. Experiments (with intrinsic measurements emulated by MOCAP) in Section V demonstrate the effectiveness of this solution. Finally, in Section VI, we discuss the results and potential extensions of this work to more robust and generic cases.

II. Modelling and Control of the FPR

We present the modelling and control of the FPR proposed in [7], but it is similar for most multi-UAV parallel robots.

A. Parametrization

The studied FPR is a multi-body aerial manipulator composed of a moving platform and 3 legs attached to the platform using revolute joints. A UAV is mounted at each leg tip by spherical joint. Any multirotor-based UAV could be applied and we choose quadrotor in our case. We make the assumption that the attached point of each quadrotor is located at its center of mass (COM). All the articulations mentioned above are passive (i.e. no actuation). Fig. 1 shows the following frames:

- $\mathcal{F}_0$: an inertial reference frame.
- $\mathcal{F}_p$: a frame attached to platform's COM position P.
- $\mathcal{F}_{Li}$: a frame attached to the center of the revolute joint O_i of leg i, with z axis defined about the physical joint axis, and $i = 1, 2, 3$ for all i in the rest of this paper.
- $\mathcal{F}_i$: a frame attached to quadrotor i's COM position A_i.

The generalized coordinates of the FPR are given by

$$\mathbf{q} = \begin{bmatrix}\mathbf{p}_p^T & \mathbf{h}_p^T & \boldsymbol{\theta}^T\end{bmatrix}^T \in \mathbb{R}^{10\times 1} \tag{1}$$

where $\mathbf{p}_p$ is the platform position in $\mathcal{F}_0$, $\mathbf{h}_p$ is a unit quaternion defining the orientation of the platform relative to $\mathcal{F}_0$, and $\boldsymbol{\theta} = [\theta_1\ \theta_2\ \theta_3]^T$ denotes the passive leg angles, with $\theta_i = 0$ indicating the leg's direction vector $\overrightarrow{O_iA_i}$ lies in the platform plane. The generalized velocity of the FPR is defined by

$$\boldsymbol{\nu} = \begin{bmatrix}\mathbf{v}_p^T & {}^p\boldsymbol{\omega}_p^T & {}^{Li}\dot{\boldsymbol{\theta}}^T\end{bmatrix}^T \in \mathbb{R}^{9\times 1} \tag{2}$$

where $\mathbf{v}_p$ is the linear velocity of the platform expressed in $\mathcal{F}_0$, ${}^p\boldsymbol{\omega}_p$ is the body-frame angular velocity of the platform in $\mathcal{F}_p$, and ${}^{Li}\dot{\boldsymbol{\theta}} = [\dot\theta_1\ \dot\theta_2\ \dot\theta_3]^T$ represents the angular velocity of each passive leg expressed in $\mathcal{F}_{Li}$.

B. Kinematic Models

The Inverse Kinematic Model (IKM) of the FPR relates the linear velocities of the quadrotors to the generalized velocity vector. Let $\mathbf{v}_i$ be the linear velocity of quadrotor i expressed in $\mathcal{F}_0$, the IKM can thus be expressed as

$$\begin{bmatrix}\mathbf{v}_1^T & \mathbf{v}_2^T & \mathbf{v}_3^T\end{bmatrix}^T = \mathbf{J}\boldsymbol{\nu} \tag{3}$$

where $\mathbf{J} \in \mathbb{R}^{9\times 9}$ is the Jacobian matrix obtained by differentiating the geometric relations between the platform pose and positions of the quadrotors [12]. Note that $\mathbf{v}_i$ can be calculated by the body-frame velocity ${}^i\mathbf{v}_i$ and the rotation matrix $\mathbf{R}_i$ representing quadrotor i's attitude relative to $\mathcal{F}_0$ as

$$\mathbf{v}_i = \mathbf{R}_i\,{}^i\mathbf{v}_i \tag{4}$$

A dual model to the IKM is the Forward Kinematic Model (FKM) computing the generalized velocity given the linear velocities of the quadrotors, i.e.

$$\boldsymbol{\nu} = \mathbf{J}^{-1}\begin{bmatrix}\mathbf{v}_1^T & \mathbf{v}_2^T & \mathbf{v}_3^T\end{bmatrix}^T \tag{5}$$

supposing $\det(\mathbf{J}) \neq 0$. If $\det(\mathbf{J}) = 0$, it refers to a singular configuration of the FPR, implying a singularity-crossing problem which is beyond the scope of this study (see [7]).

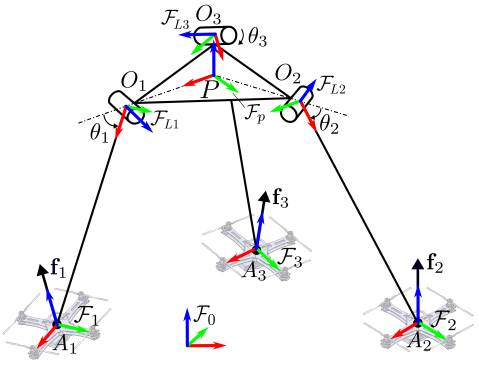


Fig. 1: General scheme of the FPR

C. Dynamics

As each quadrotor's COM is assumed to be located at the center of the spherical joint linked to the leg tip, there is a decoupling between the rotational dynamics of the quadrotors and the dynamics of the FPR. The quadrotors can thus be considered as rotating thrust generators, which together form the actuation force

$$\mathbf{f} = \begin{bmatrix}\mathbf{f}_1^T & \mathbf{f}_2^T & \mathbf{f}_3^T\end{bmatrix}^T \in \mathbb{R}^{9\times 1} \tag{6}$$

where each $\mathbf{f}_i \in \mathbb{R}^{3\times 1}$ is quadrotor i's thrust force in $\mathcal{F}_0$, i.e.

$$\mathbf{f}_i = \mathbf{R}_i\begin{bmatrix}0 & 0 & f_i\end{bmatrix}^T \tag{7}$$

with f_i being the scalar value of quadrotor i's total thrust. The dynamics of the FPR can be derived using the recursive Newton-Euler method [13], which results in a dynamic model described in matrix form

$$\mathbf{M}(\mathbf{q})\dot{\boldsymbol{\nu}} + \mathbf{c}(\mathbf{q}, \boldsymbol{\nu}) = \mathbf{J}^\mathbf{T}\mathbf{f} \tag{8}$$

where $\dot{\boldsymbol{\nu}} \in \mathbb{R}^{9\times 1}$ is the acceleration of the FPR, $\mathbf{M} \in \mathbb{R}^{9\times 9}$ is the generalized inertia matrix and $\mathbf{c} \in \mathbb{R}^{9\times 1}$ includes Coriolis, centrifugal and gravity terms.

D. Centralized Control Law

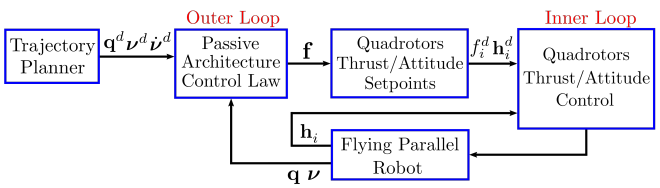


Fig. 2: Centralized control scheme of the FPR

The dynamic properties of the FPR allow to design a cascade control strategy (shown in Fig. 2): 1) An outer loop that drives the FPR to follow a desired trajectory; 2) An inner loop run at a higher frequency that ensures the convergence of each quadrotor's thrust and attitude towards desired setpoints. The outer-loop control error is

$$\mathbf{e} = \begin{bmatrix} \mathbf{e}_p \\ \mathbf{e}_o \\ \mathbf{e}_\theta \end{bmatrix} = \begin{bmatrix} \mathbf{p}_p^d - \mathbf{p}_p \\ \boldsymbol{\epsilon}(\mathbf{h}_p^d, \mathbf{h}_p) \\ \boldsymbol{\theta}^d - \boldsymbol{\theta} \end{bmatrix} \in \mathbb{R}^{9\times 1} \tag{9}$$

where $\mathbf{p}_p^d$, $\mathbf{h}_p^d$, $\boldsymbol{\theta}^d$ are the desired values of the generalized coordinates, $\mathbf{e}_o = \boldsymbol{\epsilon}(\mathbf{h}_p^d, \mathbf{h}_p)$ is the orientation error of the platform computed by the quaternion error as in [14]. From Eq. (8), we define the outer-loop control law $\mathbf{f}$ as

$$\mathbf{f} = \mathbf{J}^{-T}(\mathbf{M}(\mathbf{q})\mathbf{u} + \mathbf{c}(\mathbf{q}, \boldsymbol{\nu})) \tag{10}$$

with an auxiliary input $\mathbf{u}$ defined to minimize the control error, which is usually regulated by a PID control law with desired acceleration feedforward as

$$\mathbf{u} = \dot{\boldsymbol{\nu}}^d + \mathbf{K}_d(\boldsymbol{\nu}^d - \boldsymbol{\nu}) + \mathbf{K}_p\mathbf{e} + \mathbf{K}_i \int \mathbf{e}\, dt \tag{11}$$

where $\boldsymbol{\nu}^d$, $\dot{\boldsymbol{\nu}}^d$ are the desired velocity and acceleration, $\mathbf{K}_p$, $\mathbf{K}_i$, $\mathbf{K}_d$ are positive diagonal matrices respectively for PID gains. Then the desired thrust f_i^d of each quadrotor can be calculated simply by the norm of each $\mathbf{f}_i$ and the desired attitude $\mathbf{h}_i^d$ is calculated as in [15], choosing the yaw to avoid collisions between the propellers and the leg. The inner-loop controller ensures the convergence towards the desired setpoint $\mathbf{u}_i = [f_i^d, \mathbf{h}_i^{dT}]^T$, in which the attitude of each quadrotor is regulated using a controller such as [14].

III. Distributed State Measurement

The main drawback of the previous work on the FPR has been that the centralized controller required reliable, high frequency measurements of the robot pose expressed in a common reference frame $\mathcal{F}_0$. Until now, for multi-UAV parallel robots this has meant the necessity to use MOCAP systems. These systems are bulky, expensive, and require unobstructed lines of sight from multiple angles to each rigid body. Along with the high cost, the obvious impracticality in outdoor, cluttered, or un-mastered environments is a disappointing limitation for an aerial robot. The solution we propose in this paper is to recover the robot state using intrinsic measurements and the sharing of information between UAVs, allowing each UAV to sufficiently reconstruct a partial set of robot state needed for the control.

We start by measuring the passive leg angles $\boldsymbol{\theta}$ within the generalized coordinates $\mathbf{q}$ presented in Section II. Previously this has been done using the geometric model given the pose of the platform and position of the quadrotors with respect to $\mathcal{F}_0$, which are no longer available. Two options have been considered for measuring the leg angles:

1) Pinhole cameras mounted on the quadrotors;

2) Potentiometers (or encoders) at the passive leg joints.

As cameras are now very lightweight, and potentiometers would require much added weight due to the necessary additional stiffening of the joint to maintain an accurate alignment, we choose to use pinhole cameras. We therefore assume that there is a camera mounted on each quadrotor frame, which is able to estimate the pose of the platform relative to the camera, either with a marker system such as ARUCO [16], or using model-fitting visual techniques that are rapidly becoming more computationally feasible [17]. We denote $\mathcal{F}_{Ci}$ as a frame attached to the camera mounted on quadrotor i, for which the fixed transformation of $\mathcal{F}_i$ relative to $\mathcal{F}_{Ci}$ is known and defined by the homogeneous matrix ${}^{Ci}\mathbf{T}_i$. Note that the homogeneous transformation matrix between two frames $\mathcal{F}_A$, $\mathcal{F}_B$ is given by

$${}^A\mathbf{T}_B = \begin{bmatrix} {}^A\mathbf{R}_B & {}^A\mathbf{p}_B \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4} \tag{12}$$

where ${}^A\mathbf{R}_B$ and ${}^A\mathbf{p}_B$ are respectively the rotation matrix and displacement vector of $\mathcal{F}_B$ expressed in $\mathcal{F}_A$.

If the camera is able to measure the pose of the platform expressed in $\mathcal{F}_{Ci}$ as ${}^{Ci}\mathbf{T}_p$, then each quadrotor will have knowledge of its pose

$${}^p\mathbf{T}_i = {}^p\mathbf{T}_{Ci}\,{}^{Ci}\mathbf{T}_i \tag{13}$$

expressed in $\mathcal{F}_p$, which is sufficient information to reconstruct the internal configuration of the FPR (i.e. the passive leg angle θ_i) by the geometric model presented in [12].

The orientation of the platform has been previously defined by the unit quaternion $\mathbf{h}_p$, which loses much of its meaning under the assumption that there is no inertial reference frame $\mathcal{F}_0$. We remark however that each quadrotor necessarily estimates its attitude using onboard IMU measurements and state estimation techniques such as Kalman filtering such that it can provide additional attitude information. The orientation of any frame with respect to $\mathcal{F}_0$ can be represented by the Z-Y-X Euler angles, denoted by ψ (yaw), ϑ (pitch) and ϕ (roll), where the roll and pitch are defined about the axes orthogonal to the gravity vector and are necessarily well estimated onboard the quadrotor for stable flight. However, the quadrotors do not necessarily share a common frame for the yaw estimates, as in practice this measurement suffers in the presence of magnetic field perturbations, which may easily arise in operations near metallic structures or strong electrical currents. We can then assume that each quadrotor estimates and expresses its attitude with respect to an unknown reference frame $\mathcal{F}_{0i}$, which is constrained such that its z axis is aligned with the z axis of $\mathcal{F}_0$ (i.e. ${}^0\mathbf{z}_{0i} = \mathbf{z}_0$), but with an unknown yaw ${}^0\psi_{0i}$ relative to $\mathcal{F}_0$. We therefore introduce a flat frame attached to each quadrotor i, denoted by $\mathcal{F}_{Fi}$, to represent an orientation with $\phi_i = 0$, $\vartheta_i = 0$ and a yaw ${}^{0i}\psi_i$ measured onboard with respect to the unknown frame $\mathcal{F}_{0i}$. The orientation of quadrotor i in its flat frame $\mathcal{F}_{Fi}$ can then be expressed as

$${}^{Fi}\mathbf{R}_i = \mathbf{R}_y(\vartheta_i)\mathbf{R}_x(\phi_i). \tag{14}$$

As the orientation of the platform with respect to $\mathcal{F}_i$ is measured and described by ${}^i\mathbf{R}_p = {}^p\mathbf{R}_i^T$ from Eq. (13), we can further express the orientation of the platform in $\mathcal{F}_{Fi}$ by

$${}^{Fi}\mathbf{R}_p = {}^{Fi}\mathbf{R}_i\,{}^i\mathbf{R}_p \tag{15}$$

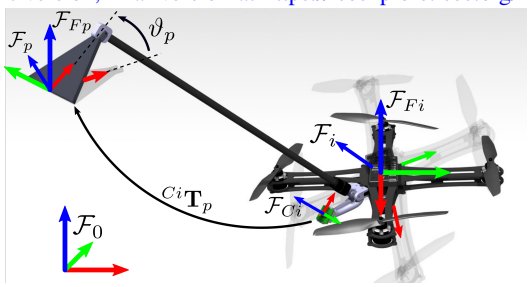


Fig. 3: The frames of one leg of the FPR. The real quadrotor pose is transparent, the quadrotor and leg in the flat frame are solid, and the flat platform frame is transparent.

which allows to obtain the roll and pitch of the platform expressed in $\mathcal{F}_{Fi}$ independently to any yaw of the platform, and their measurements on each individual quadrotor must be the same. To simplify the derivation, similar to the flat frame $\mathcal{F}_{Fi}$, we create a flat platform frame $\mathcal{F}_{Fp}$ which shares a common z axis with $\mathcal{F}_{Fi}$ (as shown in Fig. 3). The roll and pitch of the platform in $\mathcal{F}_{Fp}$ are then given by

$$ {}^{Fp}\phi_p = \text{atan2}\left({}^{Fi}\mathbf{R}_p(3,2), {}^{Fi}\mathbf{R}_p(3,3)\right) \tag{16a} $$

$$ {}^{Fp}\vartheta_p = -\text{asin}\left({}^{Fi}\mathbf{R}_p(3,1)\right) \tag{16b} $$

where $\mathbf{R}(r,c)$ refers to the r^{th} row and c^{th} column of the rotation matrix. This permits us to reconstruct the quaternion of the platform relative to $\mathcal{F}_{Fp}$ and express the attitude and linear velocity of quadrotor i in $\mathcal{F}_{Fp}$ which is common to all quadrotors by the transformation

$$ {}^{Fp}\mathbf{T}_i = {}^{Fp}\mathbf{T}_p \, {}^{p}\mathbf{T}_i \tag{17} $$

This flat platform frame $\mathcal{F}_{Fp}$ is therefore considered as a common reference frame for the FPR, which allows us to reconstruct the generalized coordinates $\mathbf{q}$ of the FPR up to an unknown translation $\mathbf{p}_p$ and an unknown yaw ψ_p of the platform with respect to any inertial frame $\mathcal{F}_0$.

IV. Decentralized Control of the FPR

Having shown in the previous section that sufficient information can be recovered from onboard sensors to reconstruct at least some of the generalized coordinates, we now show how to control the robot.

A. Control Problem Formulation

The reconstructed generalized coordinates may be effectively separated into two groups, one with known values (i.e. ϕ_p, ϑ_p, $\boldsymbol{\theta}$), and the other with unknown values ($\mathbf{p}_p$ and ψ_p). We remark however that the unknown coordinates are the same as the flat outputs of a quadrotor which are easily controlled with a joystick. Furthermore, the derivatives of the unknown coordinates ${}^{Fp}\mathbf{v}_p = {}^{Fp}\mathbf{R}_0 \, {}^{0}\dot{\mathbf{p}}_p$ and ${}^{Fp}\dot{\psi}_p$ in $\mathcal{F}_{Fp}$ can be calculated as a function of the measured body-frame velocities of the quadrotors using the FKM from Eq. (5) and expressing the linear velocities of all quadrotors in the common frame $\mathcal{F}_{Fp}$ with ${}^{Fp}\mathbf{R}_i$ by Eq. (4). The generalized velocity computed by the FKM is then $\boldsymbol{\nu} = [{}^{Fp}\mathbf{v}_p^T \ {}^{p}\boldsymbol{\omega}_p^T \ {}^{Li}\dot{\boldsymbol{\theta}}^T]^T$, where ${}^{Fp}\dot{\psi}_p$ can be calculated from ${}^{p}\boldsymbol{\omega}_p$, ${}^{Fp}\phi_p$ and ${}^{Fp}\vartheta_p$. We are thus able to control the set of coordinates defined as:

$$ \boldsymbol{\chi} = [{}^{Fp}\mathbf{v}_p^T \ {}^{Fp}\dot{\psi}_p \ {}^{Fp}\vartheta_p \ {}^{Fp}\phi_p \ \boldsymbol{\theta}^T]^T \tag{18} $$

without relying on any external measurement. This is furthermore an easily teleoperable system, as the leg angles, roll and pitch of the platform (which are susceptible to causing physically or numerically unstable configurations when the values are too large or small) may be regulated by value (instead of by rate). This allows an operator to set the desired values for those states, and then focus on the positioning and heading of the platform by controlling the linear velocity and yaw rate in $\mathcal{F}_{Fp}$, independent to changes in roll and pitch of the platform. We therefore assume that an operator can pilot the FPR by specifying the desired values of $\boldsymbol{\chi}^d$, as well as providing the desired derivatives $\dot{\boldsymbol{\chi}}^d$ to ensure the smoothness of the robot state evolution.

B. Control Architecture

As described in Section II-D, the control of the FPR fundamentally computes a desired thrust and attitude setpoint $\mathbf{u}_i = [f_i^d, \mathbf{h}_i^{dT}]^T$ for each quadrotor, which drives the generalized coordinates $\mathbf{q}$ of the FPR to some desired values $\mathbf{q}^d$. This has been done in the past with a centralized controller which has knowledge of all the states, and computes an input command $\mathbf{u}_i$ for each quadrotor. This however relies on an highly reliable bidirectional communication channel, and the interruption of more than a few control iterations (several tenths of a second) almost inevitably result in crashes. As each quadrotor is able to make use of onboard measurements, it becomes possible to implement a decentralized controller that is more robust to communication interruptions.

The control law run on each quadrotor does not differ from the centralized controller [7] and detailed in Section II-D, however it is deployed in a decentralized manner and receives feedback from decentralized sources, while listening to the common teleoperation command sent by the operator. As this command (i.e. $\boldsymbol{\chi}^d$ and $\dot{\boldsymbol{\chi}}^d$) includes only a partial set of the desired trajectory for computing the control error in Eq. (9), the errors on unknown position and yaw variables are always set to zero. We are then able to compute the full vector of the actuation force ${}^{Fp}\mathbf{f}$ expressed in $\mathcal{F}_{Fp}$ using the control law Eq. (10) on each quadrotor. Then only the thrust vector ${}^{Fp}\mathbf{f}_i$ corresponding to the given quadrotor i is applied to compute the desired setpoint $\mathbf{u}_i = [f_i^d, {}^{0i}\mathbf{h}_i^{dT}]^T$ in the attitude reference frame $\mathcal{F}_{0i}$ of quadrotor i described in Section III. This can be done by computing the desired attitude ${}^{Fp}\mathbf{h}_i^d$ in $\mathcal{F}_{Fp}$ from ${}^{Fp}\mathbf{f}_i$, then deriving the corresponding desired Euler angles in $\mathcal{F}_{Fp}$, denoted by ${}^{Fp}\phi_i^d$, ${}^{Fp}\vartheta_i^d$ and ${}^{Fp}\psi_i^d$, where the desired roll and pitch, ${}^{0i}\phi_i^d$, ${}^{0i}\vartheta_i^d$ in $\mathcal{F}_{0i}$ are the same as ${}^{Fp}\phi_i^d$ and ${}^{Fp}\vartheta_i^d$, while the desired yaw in $\mathcal{F}_{0i}$ can be further calculated if we know an onboard IMU estimate of ${}^{0i}\psi_i$ and a measurement of ${}^{p}\psi_i$ extracted from ${}^{p}\mathbf{R}_i$, i.e.

$$ {}^{0i}\psi_i^d = {}^{0i}\psi_i + ({}^{Fp}\psi_i^d - {}^{p}\psi_i) \tag{19} $$

where ${}^{p}\psi_i = {}^{Fp}\psi_i$ as the relative yaw between $\mathcal{F}_p$ and $\mathcal{F}_{Fp}$ is zero. The desired attitude of quadrotor i expressed in $\mathcal{F}_{0i}$

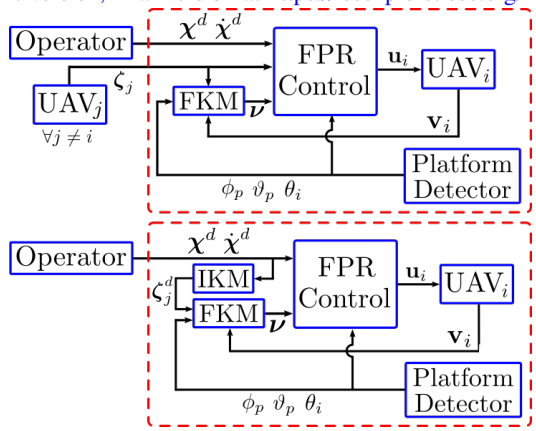


Fig. 4: The communicating (top) and non-communicating (bottom) decentralized control schemes. The red dashed line indicates modules that are embedded on UAV i.

can finally be determined by ${}^{0i}\phi_i^d$, ${}^{0i}\vartheta_i^d$ and ${}^{0i}\psi_i^d$.

C. C- and NC-Type Controllers

After having adapted the control law to a decentralized scheme, we assume that each quadrotor has a measurement of the leg angle and the linear velocity expressed in $\mathcal{F}_{Fp}$, denoted by $\boldsymbol{\zeta}_i = [\theta_i, {}^{Fp}\mathbf{v}_i^T]^T$, as well as ${}^{Fp}\phi_p$, ${}^{Fp}\vartheta_p$, and can thus apply the control law if it has knowledge of the other leg angles and the other quadrotors' velocities expressed in $\mathcal{F}_{Fp}$ (i.e. $\boldsymbol{\zeta}_j = [\theta_j, {}^{Fp}\mathbf{v}_j^T]^T, \quad \forall j \neq i$). To get the information about the unknown states $\boldsymbol{\zeta}_j$, $\forall j \neq i$ on any given quadrotor i, we propose and test two different decentralized controllers (shown in Fig. 4):

1) C-controller: use of inter-UAV communication to exchange the most recent measurements of $\boldsymbol{\zeta}_j = [\theta_j, {}^{Fp}\mathbf{v}_j^T]^T$ for each quadrotor.
2) NC-controller: a communication-less method for which each quadrotor assumes that each of the other quadrotors perfectly tracks the desired state of the system, thus use of desired values denoted by $\boldsymbol{\zeta}_j^d = [\theta_j^d, {}^{Fp}\mathbf{v}_j^{dT}]^T$. The desired values θ_j^d of the other legs are obtained directly by the teleoperation command that quadrotor i receives, while ${}^{Fp}\mathbf{v}_j^d$ can be computed by the IKM as in Eq. (3) given the desired velocity $\boldsymbol{\nu}^d$.

While the C-controller benefits from a dynamic model that is closer to that of the real robot (deviating only by measurement noise), it may also have a noisier (thus less precise) dynamic model near the converged state. It is furthermore not robust to communication failures, and may have time delays, mitigating the potential advantage of the more accurate dynamic model. The NC-controller shows good potential in maintaining the robot configuration in converged and equilibrium state, however it might be less accurate when tracking varying trajectories, especially sudden changes in trajectory cause differences between the desired and actual states large enough to affect the controller's robustness.

V. Experiments

In this section, we detail the implementation of the FPR, and present experiments to validate our methodology.

A. Implementation

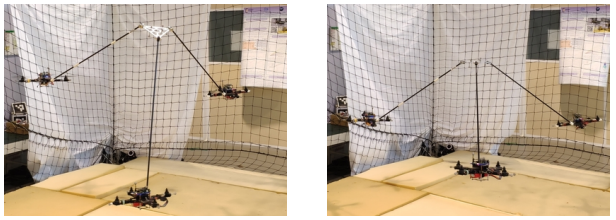
The FPR prototype is shown in Fig. 5 and Fig. 7. The platform is in the form of an equilateral triangle with side lengths of 20.3 cm and a mass of 215 g. The legs are 1.05 m long with a mass of 73 g. The legs are mounted at an unmodelled horizontal offset of 6.7 cm from the COM of the quadrotor, which acts as a disturbance to the quadrotor attitude control. The quadrotors we use are based on a 34 cm frame, with SunnySky X2212 1250kV brushless DC motors, 8045 plastic dual-blade plastic propellers and a 3-cell LiPo battery. Each quadrotor weighs about 1.03 kg and generates up to 25 N of thrust.

The quadrotors use a Pixhawk autopilot with PX4 v1.10.1 firmware [18], and are tuned with aggressive attitude and attitude rate gains to reject unmodelled disturbances caused by the offset of the leg from the quadrotors' COM. The decentralized controllers are executed on Raspberry Pi 3B+ computers mounted on each quadrotor, with ROS Melodic handling communications between computers, and with the communication between the Pixhawk and the onboard computer being handled by MAVROS.

The FPR was flown by teleoperation with a 4-axis gamepad controller, in a enclosed $4 \times 6 \times 3.5$ m flight arena equipped with an 8-camera Qualisys MOCAP system streaming data over a 5 GHz wifi network. The MOCAP system is **only** used for:

- Extracting the pose of the platform in the quadrotors local frames (i.e. extracting ${}^{Ci}\mathbf{T}_p$). It could be replaced by a monocular camera on each UAV observing the platform. Gaussian noise (with a standard deviation of 2 cm for translations and 2° for rotations) is added along/around each axis of this measurement, representative of noise in existing marker-based [19] and markerless [20] real-time visual pose estimation methods.
- Sensor fusion onboard the UAVs to improve the estimate of their body-frame linear velocities. It could thus be replaced by onboard optical flow sensors.
- Providing a ground truth in post-flight analysis.

The MOCAP may thus be fully replaced with small onboard sensors for any real application, although with additional challenges as discussed in Section VI.



(a) Large platform roll **(b)** Wide legs configuration

Fig. 5: Testing the orientation and leg angle control

B. Experiment 1: Orientation and Internal Configuring

The first experiment demonstrating the validity of our control method is simply to fly the FPR without any specific task, and track the desired configuration ϕ_p^d, ϑ_p^d, and $\boldsymbol{\theta}^d$

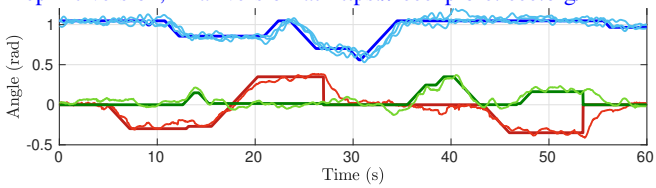


Fig. 6: The results for Experiment 1 with the C-controller. The top curves show the desired value for all leg angles $\boldsymbol{\theta}^d$ (dark blue line) and measured $\boldsymbol{\theta}$ (light blue lines). The bottom curves show the tracking of the orientation, with desired roll ϕ_p^d (dark red line) and actual roll ϕ_p (light red line), desired pitch ϑ_p^d (dark green line) and actual pitch ϑ_p (light green line).

given by the operator. Fig. 5a shows a configuration with a large roll, while Fig. 5b shows a configuration with low leg angles. The results with the centralized controller making full use of MOCAP and computes all control in $\mathcal{F}_0$ is provided for comparison. The desired and measured states for the experiment using the C-controller are plotted in Fig. 6 *(plots for NC-controller can be found in the video*[1]*)*, and the results of both controllers as well as that of the centralized controller tracking a trajectory with similar ranges of values and continuous accelerations are summarized in Table I.

It can be seen in the plot that the orientation and the internal configuration track quite well the desired reference. We also remark that both the C-controller and NC-controller can perform nicely the tracking of these variables, but C-controller is slightly better in terms of tracking errors. While we would expect the centralized controller to have better performance, it is not vastely superiour to the two decentralized methods. In fact, as the centralized controller doesn't have any measurement noise (outside that of MOCAP which is $\approx$ 1 mm) and follows smooth trajectories with continuous acceleration, the degradation when switching to decentralized control with simulated noisy intrinsic measurements and non-continuous desired states seems reasonable.

TABLE I: Results for Experiment 1. Note that "mean $\boldsymbol{\theta}$" refers to the mean RMS error.

	RMS Error (deg)		
State	**C-controller**	**NC-controller**	**Centralized controller**
ϕ_p	4.1	5	2.5
ϑ_p	2.9	4.2	3.3
mean $\boldsymbol{\theta}$	2.8	3.8	2.1

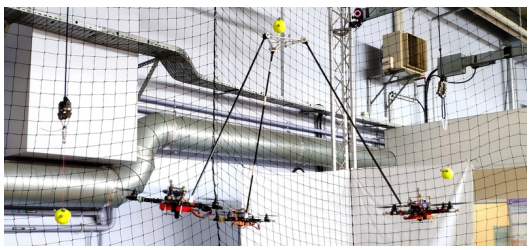
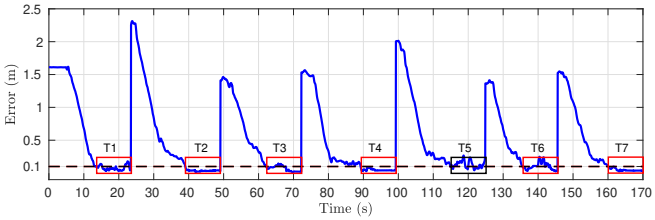
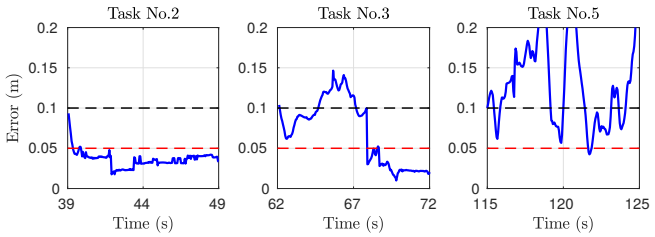


Fig. 7: Performing positioning tasks with teleoperation. The operator pilots the FPR to make the platform reach the target markers (the tennis balls) and attempts to pick up the balls.

[1] https://youtu.be/s1on54vG_c8



(a) Evolution of the norm of positioning errors over the course of Experiment 2, with 7 positioning tasks performed.



(b) Evolution of the norm of positioning errors during Task No.2, 3 and 5.

Fig. 8: Results for Experiment 2 with the NC-controller. The black dash lines represent the criteria for starting the precise positioning task, while red dash lines in (b) show the criteria for a successful task. The boxes in (a) refer to precise positioning tasks, with successful tasks in red boxes and failed one in black box.

C. Experiment 2: Precise Positioning with Teleoperation

The second experiment shows that a fine positioning using eye-to-platform teleoperation is achievable. Three markers (the tennis balls in Fig. 7) are suspended in the flight arena. An operator then flies the FPR with the task of positioning $\mathcal{F}_p$ at a marker. Once the positioning error falls below 10 cm, the operator is given 10 s to attempt to pick up the ball by placing it in the 4.5 cm diameter circular hole at the center of the platform (confirmed by a stabilization near zero of the positioning error). After the time is elapsed, a new marker is assigned and the process repeats. This experiment is performed with the platform at a configuration of zero roll and pitch, and leg angles of 60°, however there is no problem with performing it in other configurations, as seen in the video. An example of the positioning error of the platform using the NC-controller over the course of the full experiment (170 s) is shown in Fig. 8a, while Fig. 8b shows the positioning error for three separate tasks in the experiment. Note that task 5 in Fig. 8b was unsuccessful as the positioning error never stabilized. The large oscillations in error correspond to the ball swinging due to a collision with the platform early in the task.

There was little difference in performing the tasks between the C-controller and NC-controller. From the operator's perspective, both controllers were equivalent in terms of stability and positioning. Using the C-controller, 5/7 positioning tasks were successful, while with the NC-controller 6/7 tasks were accomplished. It must be recognized however that as teleoperation has human-in-the-loop feedback, it is difficult to standardize between tests.

As shown in Table II, the configuration states of the FPR were well tracked in both cases, with RMS error values almost half of what they were in Experiment 1 (as the

TABLE II: Results for Experiment 2, comparing the configuration tracking error as in Table I. The last row is the number of successful positioning tasks accomplished in the test flight.

RMS Error (deg)

State	C-controller	NC-controller
ϕ_p	2.4	2.1
ϑ_p	2.1	1.8
mean $\boldsymbol{\theta}$	1.8	1.6
success	5/7	6/7

configuration states had constant reference trajectories). A significant remark however is that in Experiment 1, the C-controller was better across all configuration states, while in Experiment 2 the NC-controller performs better. This is reasonable, as it is expected that the C-controller will allow a more accurate model of the robot compared to the NC-controller when the robot is not near to the desired state (during a step input for example). The C-controller however introduces noisy measurements and delays to each quadrotor's controller, thus for smooth flights it may be less precise. This is further supported by the observation during experiments that if a noisy velocity input was provided to the FPR, instability began in the C-controller but not the NC-controller, probably due to communication delays.

While we have demonstrated centimeter-level precise for teleoperation, two factors likely contributed to the difficulty in accomplishing manipulation tasks:

1) The operator has to stand away from the robot for safety, thus small errors between the platform frame and the marker are hard to see. A solution could be eye-in-hand control with a camera on the platform.
2) The gamepad controller has a large deadzone making fine control difficult.

VI. Conclusion and Perspectives

We have presented a novel strategy based on onboard monocular vision to allow the teleoperation and internal configuration control of multi-UAV parallel robots without requiring any external localisation system. Each quadrotor estimates it own pose relative to the platform (or payload), and the robot is able to automatically regulate its roll, pitch, and other internal states, while a remote operator (or possibly a task-level controller such as eye-in-hand visual servoing controller) controls the velocity and yaw rate of the platform. Two decentralized control methods were tested in experiments and are shown to be able to regulate the configuration (platform roll and pitch, and leg angles) of the robot as well as being able to achieve precise positioning through eye-to-platform teleoperation.

Future work may extend this control methodology for tasks involving significant robot-environment interactions. Work is ongoing to evaluate and mitigate through sensor fusion the effects of computational delays, false detections and other real-time visual pose estimation issues that may occur in a fully MOCAP-independent version of this robot. Finally, while we have only validated this control method on the FPR, it seems to be applicable to many multi-UAV parallel robots to collaboratively manipulate a load. It is therefore an encouraging step towards environmentally unconstrained operation of multi-UAV parallel robots.

References

[1] F. Ruggiero, V. Lippiello, and A. Ollero, "Aerial manipulation: A literature review," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 1957–1964, 2018.

[2] A. Ollero, G. Heredia, A. Franchi, G. Antonelli, K. Kondak, A. Sanfeliu, A. Viguria, J. R. Martinez-de Dios, F. Pierri, J. Cortes, A. Santamaria-Navarro, M. A. Trujillo Soto, R. Balachandran, J. Andrade-Cetto, and A. Rodriguez, "The AEROARMS Project: Aerial Robots with Advanced Manipulation Capabilities for Inspection and Maintenance," *IEEE Robot. Autom. Mag.*, vol. 25, no. 4, pp. 12–23, 2018.

[3] A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, Present, and Future of Aerial Robotic Manipulators," *IEEE Transactions on Robotics*, pp. 1–20, 2021, ISSN: 1552-3098. DOI: 10.1109/tro.2021.3084395.

[4] H. N. Nguyen, S. Park, J. Park, and D. Lee, "A Novel Robotic Platform for Aerial Manipulation Using Quadrotors as Rotating Thrust Generators," *IEEE Trans. Robot.*, vol. 34, no. 2, pp. 353–369, 2018.

[5] D. Sanalitro, H. J. Savino, M. Tognon, J. Cortés, and A. Franchi, "Full-Pose Manipulation Control of a Cable-Suspended Load with Multiple UAVs under Uncertainties," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2185–2191, 2020.

[6] Z. Li, X. Song, V. Bégoc, A. Chriette, and I. Fantoni, "Dynamic modeling and controller design of a novel aerial grasping robot," in *ROMANSY 23 - Robot Design, Dynamics and Control*, Springer International Publishing, 2021, pp. 538–546.

[7] D. Six, S. Briot, A. Chriette, and P. Martinet, "The Kinematics, Dynamics and Control of a Flying Parallel Robot with Three Quadrotors," *IEEE Robot. Autom. Lett.*, vol. 3, no. 1, pp. 559–566, 2018.

[8] J. Erskine, A. Chriette, and S. Caro, "Wrench Analysis of Cable-Suspended Parallel Robots Actuated by Quadrotors UAVs," *ASME Journal of Mech. and Robot.*, vol. 11, no. 2, p. 020 909, 2019.

[9] M. Gassner, T. Cieslewski, and D. Scaramuzza, "Dynamic collaboration without communication: Vision-based cable-suspended load transport with two quadrotors," *Proceedings - IEEE Int. Conf. on Robotics and Automation*, pp. 5196–5202, 2017.

[10] M. Coppola, K. N. McGuire, C. De Wagter, and G. C. de Croon, "A Survey on Swarming With Micro Air Vehicles: Fundamental Challenges and Constraints," *Frontiers in Robotics and AI*, vol. 7, no. February, 2020.

[11] S.-J. Chung, A. A. Paranjape, P. Dames, S. Shen, and V. Kumar, "A Survey on Aerial Swarm Robotics," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 837–855, 2018.

[12] D. Six, "Conception et commande de robots parallèles volants," Ph.D. dissertation, L'Ecole Centrale de Nantes, 2018.

[13] S. Briot and W. Khalil, *Dynamics of Parallel Robots: From Rigid Bodies to Flexible Elements*. Springer, 2015, vol. 35.

[14] D. Brescianini, M. Hehn, and R. D'Andrea, "Nonlinear Quadrocopter Attitude Control," ETH Zurich, Tech. Rep., 2013.

[15] T. Lee, M. Leok, and H. McClamroch, "Geometric Tracking Control of a Quadrotor UAV on SE(3)," in *Proceedings of the 49th IEEE Conference on Decision and Contol*, Atlanta, GA, 2010, pp. 5420–5425.

[16] F. J. Romero-Ramirez, R. Muñoz-Salinas, and R. Medina-Carnicer, "Speeded up detection of squared fiducial markers," *Image and Vision Computing*, vol. 76, pp. 38–47, 2018.

[17] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," *CoRR*, 2017.

[18] L. Meier, D. Honegger, and M. Pollefeys, "PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms," *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 6235–6240, Jun. 2015.

[19] M. Kalaitzakis, S. Carroll, A. Ambrosi, C. Whitehead, and N. Vitzilaios, "Experimental Comparison of Fiducial Markers for Pose Estimation," in *Int. Conf. on Unmanned Aircraft Systems*, Athens, Greece, 2020, pp. 781–789.

[20] M. Pavliv, F. Schiano, C. Reardon, D. Floreano, and G. Loianno, "Tracking and Relative Localization of Drone Swarms with a Vision-based Headset," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1455–1462, 2021.