Decentralized Control and Estimation of a Flying Parallel Robot Interacting with the Environment

Shiyu Liu

THÈSE DE DOCTORAT DE

L'ÉCOLE CENTRALE DE NANTES

ECOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies
de l'Information et de la Communication
Spécialité : *Automatique, Productique et Robotique*

Par
Shiyu LIU

Decentralized Control and Estimation of a Flying Parallel Robot Interacting with the Environment

Thèse présentée et soutenue à l'École Centrale de Nantes, le 29 novembre 2022
Unité de recherche : UMR 6004, Laboratoire des Sciences du Numérique de Nantes (LS2N)

Rapporteurs avant soutenance :

Tarek HAMEL — Professeur des universités - Université Côte d'Azur, Sophia Antipolis
Nicolas MARCHAND — Directeur de recherche CNRS - Université Grenoble Alpes

Composition du Jury :

Président :	Nicolas ANDREFF	Professeur des universités - Université de Franche-Comté
Examinateur :	Stéphane CARO	Directeur de recherche CNRS - École Centrale de Nantes
Dir. de thèse :	Isabelle FANTONI	Directrice de recherche CNRS - École Centrale de Nantes
Co-encadrant :	Abdelhamid CHRIETTE	Maître de conférences - École Centrale de Nantes

To my family,

To my love.

Acknowledgement

Before presenting the research topic I tackled in my Ph.D. thesis, I would like to acknowledge the ones who have played an essential role in my research work.

First of all, I would like to express my great gratitude towards my research supervisors, Dr. Isabelle Fantoni and Dr. Abdelhamid Chriette, for their continued confidence in me, their tremendous support and the great amount of liberty given to me for diversifying my research directions. Thanks to their rigour, patience, gentleness and encouragement, I have been through a perfect research experience during the past three years.

Secondly, I am grateful to my thesis defence committee for their valuable questions and comments that have contributed to the quality of my work and this manuscript. I would like to thank Prof. Tarek Hamel and Dr. Nicolas Marchand for reviewing my manuscript and for their detailed remarks which have helped complement the weakness of my research work. Particular thanks are due to Prof. Nicolas Andreff who has also participated in my CSI committee and has been providing useful insights and interesting research directions during our CSI meetings and my thesis defence. Thank Dr. Stéphane Caro for the insightful discussion and his wise advice related to my research topic.

Then, I would like to address a particular acknowledgement to the colleagues and friends with whom I have been working together in ARMEN team, LS2N. Thank Dr. Julian Erskine, Dr. Zhongmou Li and Dr. Rafael Balderas-Hill, with whom I worked closely in the drone team at the beginning of my thesis. My research would not have been so smooth without their help and inspiration. Thank Dr. Damien Six for his support on the experiments. His talent in development has always been impressive to me. Thank Haixin Sun and Dr. Songming Chen the only two colleagues in the team who started their thesis at the same time as mine, and their family Dr. Chunxiao Yang and Dr. Xiaoxiao Song, for our valuable and amiable moments spent together in the past three years.

In addition, I would also like to express my gratefulness to all the people I have met in LS2N, especially the colleagues in office S514, including Dr. David Pérez-Morales, Dr. Franco Fusco, Dr. Guillaume Jeanneau, Dr. Minglei Zhu, Dr. Shaoqi Wu, Dr. Valentin Le Mesle, Dr. Zhiping Wang, Dr. Zhiqiang Wang, Alessandro Colotti, Andrea Gotelli, Bozhao Wang, Charlotte Beaune, Farhan Ahmed, Federico Zaccaria, Giovanni Juin-Gauthier,

Hanbang Gao, Honglu Sun, Hugo Bouvier-Lambert, Mathilde Theunissen, Minh-Quan Dao, Peidong Zhang, Salman Hamadi, Selma Oubouabdellah, Yuxin Zhang. It is my pleasure to have discovered the research life and shared some wonderful moments together.

Last, I would like to acknowledge the ones I love in my life. Thank my maternal grandparents, my parents Mr. Ronglu Liu and Prof. Yaping Shi, and my sister Shixuan Liu (yoyo) for having been supporting me materially and spiritually with their endless efforts. I wouldn't be able to complete my studies with a doctoral degree without their understanding and support. Special thanks go to Yu Zhang, my lifelong partner, for having accompanied me, supporting me and encouraging me over the past few years. You are my light no matter on the happiest or the toughest days. Will you marry me?

Abstract

Aerial manipulation is an emerging domain where multirotor Unmanned Aerial Vehicles (UAVs) are equipped with onboard end-effectors for grasping, transporting and manipulating objects. To enhance the payload capacity and achieve full manipulability in 3-dimensional space, a flying parallel robot (FPR) was previously proposed in which a number of UAVs are used to collectively support a passive parallel architecture. While the previous works have focused on design, modelling and motion control of the robot, it is significant to further investigate the FPR interacting with the environment.

In this thesis, the estimation and control methods dealing with the interaction with the environment are presented, which are applied to a specific FPR composed of a moving platform and a number of rigid legs attached with quadrotors actuating the system. Several momentum-based observers are implemented to estimate the external wrench exerted on the robot. An impedance-based controller is designed with the desired wrench-tracking capability. Experiments show that the overall estimation and control methods can deal with modelling uncertainties, and external disturbances such as additional payload and wind perturbations, as well as accomplish contact-based interaction tasks.

The second main contribution of this thesis is the proposal of decentralized strategies based on onboard and intrinsic measurements of the UAVs. A vision-based estimation technique using the ArUco marker system is firstly presented, shown to be capable of reconstructing partially the robot pose sufficient for the control in absence of any external localisation system. The previously proposed motion or interaction control algorithms can then be deployed in a decentralized manner, allowing each UAV to perform its own control based on its own measurements and information shared within all the UAVs. Experiments show the effectiveness of the proposed methods in regulating the robot configuration, achieving precise positioning tasks through teleoperation and performing contact-based interactions with an object in the environment.

In addition, a detailed analysis of the wrench feasibility of the FPR is presented. Notions of Available Thrust Set and Available Wrench Set are introduced, with detailed computation and case studies conducted. A quantitative metric named feasibility margin is furthermore adopted, which is applied to determine the optimal leg configurations of the FPR in different platform orientations maximising the wrench feasibility of the robot accordingly to specific task requirements.

Résumé

La manipulation aérienne est un nouveau domaine où les drones multirotors sont équipés d'effecteurs pour la saisie, le transport et la manipulation des objets. Afin d' améliorer la capacité de charge et la manipulabilité dans l'espace 3D, un robot parallèle volant (FPR) a été proposé, dans lequel un certain nombre de drones sont utilisés pour soutenir coopérativement une architecture parallèle et passive. Alors que les travaux précédents se sont concentrés sur la conception, la modélisation et le contrôle de mouvement du robot, il est important d'étudier davantage le FPR en interaction avec l'environnement.

Dans cette thèse, les méthodes d'estimation et de contrôle considérant l'interaction avec l'environnement sont présentées et appliquées à un FPR spécifique composé d'une plateforme mobile et des jambes rigides attachées à des quadrirotors qui actionnent le système. Plusieurs observateurs basés sur la quantité de mouvement sont mis en œuvre pour estimer les torseurs externes exercés sur le robot. Un contrôle d'impédance est conçu avec la capacité de suivi de torseurs d'interaction désirés. Les expériences montrent que les méthodes de contrôle et d'estimation peuvent traiter les incertitudes de modélisation, les perturbations externes telles que la charge supplémentaire et les perturbations du vent, ainsi que les tâches d'interaction au contact avec un objet dans l'environnement.

La deuxième contribution de cette thèse est la proposition de stratégies décentralisées basées sur des mesures embarquées et intrinsèques des drones. Une technique d'estimation utilisant la vision et le système de marqueurs ArUco est tout d'abord présentée. Elle s'avère capable de reconstruire partiellement la pose du robot en l'absence de système de localisation externe. Les algorithmes de contrôle de mouvement ou d'interaction peuvent ensuite être déployés de manière décentralisée, permettant à chaque drone d'effectuer sa propre loi de commande sur la base de ses propres mesures et des informations partagées au sein de tous les drones. Les expériences montrent l'efficacité des méthodes proposées pour la régulation de la configuration du robot, la réalisation des tâches de positionnement précis par téléopération et des interactions basées sur le contact.

En outre, une analyse détaillée sur les torseurs admissibles du FPR est présentée. Les notions d'espace de force de poussée et d'espace de torseurs admissibles sont introduites, avec des calculs détaillés et des études de cas réalisées. Une métrique quantitative appelée marge de faisabilité est ensuite présentée, et appliquée pour déterminer les configurations optimales des jambes du FPR dans différentes orientations de la plateforme, maximisant la faisabilité de torseurs du robot en fonction des exigences spécifiques de la tâche.

Table of Contents

List of Figures

List of Tables

List of Symbols

General Symbols

x	A scalar variable
$\mathbf{x}$	A column vector
$\mathbf{X}$	A matrix
$\mathbb{Z}$	The set of integers
$\mathbb{R}$	The set of real numbers
$\mathbb{R}^n$	The set of n-dimensional vectors in real numbers
$\mathbb{R}^{m\times n}$	The set of $(m \times n)$ matrices in real numbers
$\mathbb{H}$	The quaternion set
$\mathbf{1}_{n\times n}$	An identity matrix of dimension $(n \times n)$
$\mathbf{0}_n$	A n-dimensional column vector of zeros
$\mathbf{0}_{n\times n}$	A $(n \times n)$ matrix of zeros

Symbols for Modelling of the Flying Parallel Robot (FPR)

$\mathfrak{F}_0$	An inertial reference frame, also referred to as a global frame or a world frame
$\mathfrak{F}_j$	A local frame attached to an arbitrary body j
$\mathfrak{F}_{Fj}$	The flat frame attached to an arbitrary body j
n	Total number of legs or multirotors
i	An index for the number of legs or multirotors, $i \in \{1, 2, ..., n\}$
$\mathbf{x}_i, \mathbf{x}_p$	An variable associated to leg/multirotor i and the platform expressed in $\mathfrak{F}_0$
${}^j\mathbf{x}_i, {}^j\mathbf{x}_p$	An variable associated to leg/multirotor i and the platform expressed in an arbitrary frame $\mathfrak{F}_j$
${}^j\mathbf{r}_{AB}$	A vector defining the translation of $\overrightarrow{AB}$ expressed in an arbitrary frame $\mathfrak{F}_j$
$\mathbf{p}_j$	3-dimensional position vector of a body j relative to $\mathfrak{F}_0$
$\mathfrak{q}_j$	Unit quaternion of a body j relative to $\mathfrak{F}_0$
$\mathbf{R}_j$	Rotation matrix of an arbitrary body j relative to $\mathfrak{F}_0$
${}^k\mathbf{R}_j$	Rotation matrix of a body j relative to an arbitrary frame $\mathfrak{F}_k$
$\mathbf{v}_j$	Linear velocity of a body j relative to $\mathfrak{F}_0$
$\boldsymbol{\omega}_j$	Angular velocity of a body j relative to $\mathfrak{F}_0$

${}^{j}\boldsymbol{\omega}_{j/k}$	Angular velocity of a body j relative to $\mathfrak{F}_k$ and expressed in $\mathfrak{F}_j$
${}^{j}\mathbf{v}_j, {}^{j}\boldsymbol{\omega}_j$	Linear and angular velocities of a body j in its body-fixed frame
$\dot{\mathbf{v}}_j, \mathbf{a}_j$	Linear acceleration of a body i relative to $\mathfrak{F}_0$
${}^{j}\dot{\mathbf{v}}_j$	Linear acceleration of a body i relative to $\mathfrak{F}_j$
$\dot{\boldsymbol{\omega}}_j, {}^{j}\dot{\boldsymbol{\omega}}_j$	Angular acceleration of a body j expressed in $\mathfrak{F}_0$ and $\mathfrak{F}_j$
$\mathbf{q}$	Generalised coordinates (robot pose) of the FPR
$\boldsymbol{\nu}$	Generalised velocity vector of the FPR
$\dot{\boldsymbol{\nu}}$	Generalised acceleration vector of the FPR
$\boldsymbol{\tau}$	Actuation wrench of the FPR
$\mathbf{f}_i$	3-dimensional thrust force of multirotor i
$\mathbf{f}$	A high-dimensional vector concatenating the thrust forces of all the multirotors
$\mathbf{f}_j, \mathbf{m}_j$	3-dimensional force and moment exerted on a body j
$\mathfrak{g}$	Gravity vector
$\mathbf{J}$	Jacobian matrix
$\mathbf{M}(\mathbf{q})$	Generalised inertia matrix
$\mathbf{C}(\mathbf{q}, \boldsymbol{\nu})$	Generalised Coriolis matrix
$\mathbf{c}(\mathbf{q}, \boldsymbol{\nu})$	Generalised vector of Coriolis, centrifugal and gravitational terms
$\mathbf{g}(\mathbf{q})$	Generalised vector of gravity wrench

Spatial Vector Notation

$\mathbb{F}$	Space of 6-dimensional spatial force vectors
$\mathbb{M}$	Space of 6-dimensional spatial motion vectors
$\tilde{\mathbf{v}}_j$	6-dimensional spatial velocity of a body j
$\tilde{\mathbf{a}}_j$	6-dimensional spatial acceleration of a body j
$\tilde{\mathbf{f}}_j$	6-dimensional spatial force of a body j
$\tilde{\mathbf{X}}_j$	A spatial matrix of dimension (6×6) associated to a body j

Symbols for Estimation and Control of the FPR

$\boldsymbol{\tau}_e$	External wrench of the FPR
$\boldsymbol{\tau}_e^d$	Desired interaction wrench of the FPR
$\hat{\boldsymbol{\tau}}_e$	An estimation on the external wrench of the FPR
$\mathbf{q}^d$	Desired values on the generalized coordinates
$\boldsymbol{\nu}^d$	Desired velocity
$\dot{\boldsymbol{\nu}}^d$	Desired acceleration
$\boldsymbol{\mathcal{P}}$	Momentum variable

$\mathcal{G}_j(s)$	j-th element transfer function in Laplace domain
$\boldsymbol{\varepsilon}_o$	Orientation error defined by unit quaternions
$\boldsymbol{\varepsilon}_q$	Tracking error of the robot pose
$\boldsymbol{\varepsilon}_\tau$	Tracking error of the interaction wrench
$\boldsymbol{\varepsilon}_\omega$	Tracking error of the multirotor's angular velocity
$\mathbf{u}$	An auxiliary control input
$f_{t,i}^d$	Setpoint of total thrust magnitude of multirotor i
$\mathbf{m}_i^d$	Setpoint of body-frame moments of multirotor i
$\mathfrak{q}_i^d$	Attitude setpoint of multirotor i
$\mathbf{M}^d$	Matrix of the desired mass coefficients
$\mathbf{K}^d$	Matrix of the desired spring coefficients
$\mathbf{B}^d$	Matrix of the desired damping coefficients
$\mathbf{K}$	A matrix of estimation or control gains
$\boldsymbol{\chi}$	Teleoperable set of coordinates of the FPR
$\mathbf{u}_i$	Control vector of multirotor i
$\boldsymbol{\zeta}_i$	Information vector of multirotor i shared with other multirotors

Symbols for Wrench Feasibility Analysis of the FPR

$\mathbf{w}$	Generalised wrench exerted on the robot
$\mathbf{w}_g$	Generalised gravity wrench
$\mathbf{W}$	Wrench matrix
f_{max}	Maximum total thrust of a multirotor
γ	Inclination angle limit of a multirotor
$\mathbf{v}$	A vertex defining the polytope
$\mathcal{T}_i$	Feasible Thrust Space (FTS) of a single multirotor i
$\mathcal{T}$	Augmented FTS of all the multirotors
$\mathcal{W}$	Feasible Wrench Space (FWS) of the FPR
$\mathcal{W}_i$	FWS mapped from the FTS of a single multirotor i
$\mathcal{W}_{\mathbf{f}_p}^s$	Static FWS evaluated on the platform force
$\mathcal{W}_{\mathbf{m}_p}^s$	Static FWS evaluated on the platform moment
$\mathcal{F}_p^s$	3-dimensional projection of the static FWS $\mathcal{W}_{\mathbf{f}_p}^s$ onto the platform force space
$\mathcal{M}_p^s$	3-dimensional projection of the static FWS $\mathcal{W}_{\mathbf{m}_p}^s$ onto the platform moment space

List of Acronyms

ACTS	Aerial Cable-Towed System
CDPR	Cable-Driven Parallel Robot
CoM	Centre of Mass
DGM	Direct Geometric Model
DoF	Degrees of Freedom
EKF	Extended Kalman Filter
ESC	Electronic Speed Control
FKM	Forward Kinematic Model
FOMO	First-Order Momentum Observer
FPR	Flying Parallel Robot
FTS	Feasible Thrust Space
FWS	Feasible Wrench Space
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IKM	Inverse Kinematic Model
IGM	Inverse Geometric Model
IMU	Inertial Measurement Unit
MDH	Modified Denavit-Hartenberg
MOCAP	Motion Capture System
MoI	Moment of Inertia
PID	Proportional, Integral and Derivative
QP	Quadratic Programming
RMSE	Root-Mean-Square Error

ROS	Robot Operating System
SMMO	Sliding-Mode Momentum Observer
SOMO	Second-Order Momentum Observer
UAV	Unmanned Aerial Vehicle
VTOL	Vertical Take-Off and Landing

INTRODUCTION

Research Context

In the past two decades, advances in equipment, electronics, sensors and batteries have supported the development of micro Unmanned Aerial Vehicle (UAV) platforms. The emergence of small and inexpensive rotor-based platforms has dramatically increased research topics and civilian applications of UAVs from the public to professionals, such as aerial photography, building and infrastructure inspection, agricultural monitoring, security and surveillance. These application scenarios are evolving with an increase in research into micro UAVs, particularly for:

- Sensor-based estimation or control of a single UAV [Fraundorfer, 2012; Zhou, 2015; Bourquardez, 2009; Serra, 2016];
- Formation flight of a fleet of UAVs [Turpin, 2012; Schiano, 2016; Hou, 2018; Guerrero-Castellanos, 2019];
- Aerial manipulations achieved either by a single UAV equipped with an onboard actuator (see Section 1.1.1) or by the cooperation of multiple UAVs for the transport and handling of loads using cables or rigidly-connected links (see Section 1.1.2).

The application for which the aerial vehicles need to physically interact with the environment presents various challenges such as flight stability, autonomy, accuracy and payload capacity, etc. A variety of novel architectures and prototypes involving UAVs have been developed and tested with their industrial applications being in the development phase [Ollero, 2018]. However, existing aerial vehicle platforms generally face the following difficulties:

- A low autonomy due to the limited payload capacity and energy consumption of onboard actuators;
- Unfavourable flight stability and accuracy during interactions with the environment;
- A disruption in manipulation due to the blast of UAVs;
- Limited manipulation and payload capacity.

To overcome some of these limitations, a concept of an aerial robot actuated by multiple UAVs named Flying Parallel Robot (FPR) has been proposed in [Six, 2018a]. It is a poly-articulated passive architecture composed of a moving platform on which an arbitrary number of passive legs and UAVs are attached (see Section 1.2 for more details). This

concept has combined the advantages of parallel mechanism and aerial vehicles, offering potentially manipulation capabilities such as enhanced payload capacity and full manipulability in 3-dimensional space that do not exist compared to other aerial manipulators with single or multiple UAVs.

While the previous works in [Six, 2018a] have been carried out to tackle several scientific challenges, such as the generic dynamic modelling and control of the FPR with n UAVs, there are still scientific and technological obstacles remain to be overcome, which are also the main objectives of this thesis, namely:

- **Objective 1:** robot control considering the interaction with the environment (action on an object/wall for example);
- **Objective 2:** decentralized control and estimation strategies that can be deployed onboard each UAV to improve control of the entire system;
- **Objective 3:** sensor-based observers/controllers to avoid as much as possible using Motion Capture (MOCAP) systems for exteroceptive measurements.

Contributions

In this thesis, the control and estimation techniques applied to a multi-UAV parallel robot named Flying Parallel Robot (FPR) taking into account the interaction with the environment are investigated, which extend potentially the capability of the FPR to achieve aerial manipulation tasks in the real-world scenario. The specific FPR studied within the thesis is composed of a moving platform, on which a number of legs are attached with one Degree-of-Freedom (DoF) revolute joints, and multirotors attached on the other end of the legs by means of spherical joints.

With the initial design, generic modelling and control of the FPR previously proposed, the main contributions of this thesis mainly consist of

- ❖ An in-depth study on the modelling of the specific FPR (**Chapter 2**), including the geometric relations, kinematics and dynamics modelling of the robot. This involves analytical expressions of geometric and kinematic models with several numerical algorithms for deriving the dynamic models. The numerical validation of all the models has been done to verify the modelling process served as a basis for the control and estimation techniques studied in the rest of the thesis.
- ❖ Several momentum-based observers applied to the FPR (**Section 3.2**), namely first-order wrench observer (FOWO), second-order wrench observer (SOWO) and sliding-

mode wrench/momentum observer (SMWO), for estimating the external wrench exerted on the robot. The momentum-based approaches have been constructed according to the dynamic properties (and models) of the robot, and experimentally validated (**Section 3.5**).

- ❖ An impedance-based control applied to the FPR dealing with the interaction with the environment (**Section 3.3**), which is completed with the external wrench estimation performed by the momentum-based observer. The proposed controller is capable of tracking desired trajectories of the robot pose and the desired wrench expected to exert on the environment. The overall control and estimation techniques (**Section 3.4**) are experimentally validated with modelling uncertainties, external perturbations and contact-based interactions with the environment (**Section 3.5**).

- ❖ A vision-based pose estimation method that can be deployed onboard each UAV to reconstruct partially the robot pose without depending on any external localisation system (**Section 4.2**). The real-time pose estimation is based on ArUco marker detection system, along with the measurements from the Inertial Measurement Unit (IMU) and the Extended Kalman Filter (EKF) technique to reconstruct the robot pose and velocity.

- ❖ Several decentralized control algorithms proposed for motion control and interaction control schemes (**Section 4.3**). These controllers have been implemented with emulated or real-time pose estimation results, and validated in a variety of experimenting scenarios such as precise positioning of the platform, pick-up of payload with teleoperation and contact-based interactions with the environment (**Section 4.4**).

- ❖ A detailed analysis on wrench feasibility of the FPR (**Chapter 5**), which computes the feasible wrench the robot can exert or resist taking into account the actuation limits of UAVs (constrained by the maximum thrust magnitude and rotational movements of UAVs). Then the analysis of the quantitative metrics has been applied to determine the optimal leg configurations to maximise the wrench feasibility in certain platform orientations.

In summary, the main contents of this manuscript will be organised as follows: Theoretical background and basis regarding the modelling of the FPR will be presented respectively in **Chapter 1** and **Chapter 2**. The scientific problems related to Objective 1 for the interaction with the environment will be addressed in **Chapter 3**. Then, decentralized control and estimation strategies of Objective 2 will be discussed in **Chapter 4**, in which the real-world challenges in Objective 3 for sensor-based estimation have also been partially tackled. Finally, **Chapter 5** complements the wrench analysis of the FPR interacting with the environment.

Chapter 1

State of the Art

This chapter presents the state of the art regarding the thesis subject on the Flying Parallel Robot (FPR) as an aerial manipulator with multiple Unmanned Aerial Vehicles (UAVs). In the first section, the domain of aerial manipulation is introduced, with a literature review on the existing architectures and designs for all kinds of aerial manipulators, which consist mainly of single-UAV designs and cooperative or rigidly connected systems with multiple UAVs. The second section presents the FPR design, which has been proposed in order to combine the advantages and capacities of a parallel mechanism and aerial vehicles. A general modelling and motion control of FPR are also detailed.

1.1 Aerial Manipulation

In recent years, the development of micro Unmanned Aerial Vehicles (UAVs) based on powerful and inexpensive multirotor-based platforms has dramatically increased the research topics on UAVs. The applications of modern UAVs are not only for aerial photography but extended to totally novel domains, among which a topic that has been rapidly developed is **aerial manipulation**. It is a domain where the UAVs are equipped with robotic arms or/and end-effectors for accomplishing aerial grasping, transportation and manipulation tasks [Ruggiero, 2018; Ollero, 2021]. The proposal of such novel aerial workers is essential because they can easily access remote locations or other places that are dangerous or inaccessible for human operators. For instance, aerial manipulators can quickly reach and operate in high-altitude workspaces, for the applications like inspection and maintenance of infrastructure, grasping and transporting objects where aerial robots need to physically interact with the environment [Ollero, 2018].

The topic of aerial manipulation presents many scientific and technological challenges, such as vehicle autonomy, flight stability, manipulation accuracy and payload capability, etc. It is therefore a cutting-edge research topic which attempts to extend the ability of aerial robots for performing dexterous and complex manipulation tasks in the air. To enhance the capability and versatility of aerial robots, problems on the mechanical and structural designs, advanced sensing and perception as well as robot planning and control have been addressed. Different UAV platforms have been analysed and used in the aerial

manipulation field, ranging from helicopters and classical under-actuated multirotors, to fully-actuated multirotors with tilted/tilting rotors. A variety of aerial manipulator designs have also been developed and studied over the past decade, which can be generally classified into two genres:

- Single-UAV manipulator: single UAV attached with additional equipment for aerial manipulation;
- Multi-UAV manipulator: multiple UAVs performing cooperatively the aerial task or rigidly connected to a moving platform or an articulated structure.

In the following sections, a literature review on the development of different generations of aerial manipulators is given, along with the state of the art on the perception, estimation and control of such aerial robots for accomplishing aerial manipulation tasks in both indoor and outdoor scenarios.

1.1.1 Single-UAV Manipulators: From Helicopter to Multirotor-based Platforms

The early development of aerial manipulators has been based on a variety of mature aerial vehicles such as helicopters, multirotor-based platforms and other vehicles that have been derived from classical multirotors. The types of onboard equipment to perform the manipulation tasks have also evolved from simple rigid/passive links to more sophisticated grippers or articulated robotic arms. The development of the mechanical designs of end-effectors, onboard sensing and control algorithms has allowed a single aerial vehicle to achieve novel applications and extend its potential capacities in real-world scenarios.

Helicopters with Industrial Arms

When referring to the aerial manipulator, the helicopter has played an important role in the development of early works, which is a classical aerial system composed of a main horizontal rotor for providing vertical lifting and a tail rotor to counteract the torque generated by the main rotor and control the heading of the aircraft. It has been taken as one of the first platforms to embed robotic manipulators or end-effectors to achieve grasping, transportation and manipulation in outdoor environments. For instance, [Bernard, 2010; Bernard, 2011] have investigated the use of helicopters for autonomous slung-load transportation and deployment (see Fig. 1.1(a)); the grasping of payloads by a small-size helicopter attached with a self-designed gripper or manipulator has been studied in [Pounds, 2011; Pounds, 2012; Pounds, 2014] (see Fig. 1.1(b)) and [Kondak, 2013] (see Fig. 1.1(d)); the helicopters can also be equipped with an industrial and large-size robotic manipulator, applied to manipulation and operation of objects remotely, as investigated in [Huber, 2013; Kondak, 2014; Laiacker, 2016] (as shown in Fig. 1.1(c)). These designs

(a) Helicopter for autonomous load transportation and deployment [Bernard, 2011]

(b) Helicopter with a gripper for grasping and transporting objects [Pounds, 2012]

(c) Helicopter equipped with a KUKA LWR industrial manipulator [Laiacker, 2016]

(d) Helicopter equipped with a robotic arm for grasping and manipulation [Kondak, 2013]

Figure 1.1 – Aerial manipulators composed of a helicopter and an end-effector/industrial manipulator.

often use GNSS (Global Navigation Satellite System) for the positioning of the robot as they must work outdoors due to the dimension requirement of helicopters, and equip with other onboard sensors such as IMU and cameras for pose estimation.

The use of helicopters is advantageous in terms of payload capacity and autonomy during the manipulation tasks. However, the mechanical construction and control of such complex and large-size platforms are difficult to handle, along with the safety issues associated with the blades of the helicopter's main rotor for flights close to external objects and surfaces. In contrast, multirotor-based UAVs are usually small-size aircraft and significantly simpler in terms of design, assembly and control, resulting in growing trends towards using multirotor as the platform for aerial manipulation in most of the recent works.

Classical Multirotors with Links, Grippers or Articulated Arms

The multirotor consists of a rigid frame and multiple propellers parallelly fixed to the frame, producing unidirectional thrust vertical to the main frame and generating three-

dimensional moments to rotate its attitude, which is therefore an under-actuated system moving in $SE(3)$ with vertical take-off and landing (VTOL) capability. The multirotors are commonly equipped with four, six or eight propellers, and thus characterised by the number of rotors and named respectively as *quadrotor*, *hexarotor* and *octorotor*. The second generation of aerial manipulators is thus generally composed of a single multirotor equipped with onboard end-effectors, which are usually customised using inexpensive materials and lightweight servo motors (when actuated). The achievable manipulation tasks depend on the type of embedded equipment, which can be summarised as follows:

- **actuated rigid links or tools**, with examples shown in Fig. 1.2(a)-(c);
- **grippers**, with examples in Fig. 1.2(d), (e);
- **passive links**, such as cables or tethers used in designs shown in Fig. 1.2(f), (g);
- **articulated robotic arms**, including single serial robotic arm as in Fig. 1.2(h)-(j), dual-arm system in Fig. 1.2(k) and parallel manipulators in Fig. 1.2(l), (m).

First of all, the employment of **actuated rigid links or tools** has the objective to achieve specific and unitary interaction tasks. For instance, designs where a rigid link is attached on the quadrotor have been proposed in [Bartelds, 2016] [Alexis, 2016; Yüksel, 2016; Wopereis, 2017; Hamaza, 2018; Yüksel, 2019], showing the composed system capable of applying a sustained force on the external surfaces [Bartelds, 2016] (see Fig. 1.2(a)), [Yüksel, 2016; Wopereis, 2017], and dedicated to contact-based inspection [Alexis, 2016] or painting-like tasks [Yüksel, 2019] (see Fig. 1.2(b)). Customised tools can also be embedded on a single UAV to achieve the specific task, such as "PaintCopter" a quadrotor fitted with a spray mechanism for autonomous spray painting [Vempati, 2018] (see Fig. 1.2(c)), a quadrotor equipped with suction cups capable of perching and opening the door [Tsukagoshi, 2015], and a nailgun-equipped octorotor for nail deployment [Romano, 2021].

Similarly, custom and small-size **grippers** can be deployed onboard the quadrotors for the grasp and transportation of objects, such as gripping of a beam-like object and piece of wood using impactive and ingressive gripper [Mellinger, 2011b] (see Fig. 1.2(d)), grasp of a stuffed toy ensured by the compliance and under-actuation of the self-designed gripper [Ghadiok, 2011] (see Fig. 1.2(e)), pickup of a cylinder-form object by avian-inspired gripper [Thomas, 2013], as well as grasping and transportation of magnetic objects by a team of quadrotors equipped with self-designed gripper [Loianno, 2018b].

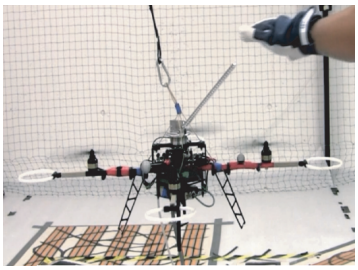
In addition, in some of the aerial manipulator designs, **passive links** like cables or tethers are integrated within the multirotor model for agile transportation of cable-suspended load as investigated in [Palunko, 2012; Palunko, 2013; Sreenath, 2013b] and [Foehn, 2017] (see Fig. 1.2(f)), for inspection and surveillance tasks in case of tethered to a moving

platform [Tognon, 2016], and for assembling rope structures in inaccessible locations [Augugliaro, 2013] (see Fig. 1.2(g)).

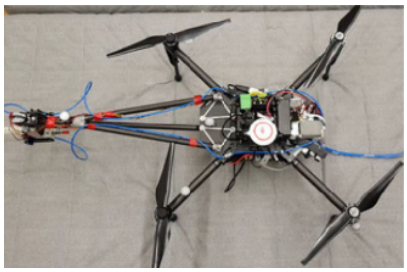
Most importantly, designs with different types of **articulated robotic arms** embedded on the UAV are commonly seen in a large number of works, because the onboard robotic manipulators can provide additional degrees of freedom allowing to accomplish more complex manipulation tasks and compensate the perturbations on the multirotors. The employed robotic arms range from a single serial manipulator (up to 6 DoFs) to dual-arm systems and parallel mechanisms. For instance, a single multirotor has been equipped with multi-DoF serial robotic arm, including: a 2-DoF arm for object manipulation [Kim, 2013] (see Fig. 1.2(h)), grasping [Tognon, 2017] (see Fig. 1.2(i)) and catch/release application [Morton, 2016]; a 3-DoF arm for contact-based inspection [Nayak, 2018], object grasping [Kim, 2016] and assembly tasks [Jimenez-Cano, 2013]; a 4-DoF arm for hinged-door opening application [Lee, 2020]; a 5-DoF arm for object grasping and manipulation task [Bellicoso, 2015]; and a 6-DoF robotic arm for precise positioning task [Ruggiero, 2015a] (see Fig. 1.2(j)) [Lippiello, 2016; Rossi, 2017], interaction task with ropes and flexible bars [Cataldi, 2016] and assembly of bar structures [Cano, 2013]. As the shortcomings of serial arms like the increased inertia and shifted centre of mass (CoM) might further affect the stability of the system, interest in applying parallel designs with multi-arm systems and parallel manipulators has been increasingly aroused in recent works. On one hand, the dual-arm systems are particularly studied for grasping pipe-like objects [Suarez, 2017a] (see Fig. 1.2(k)) [Suarez, 2018], for transporting objects along with long-reach link [Caballero, 2018], for valve tuning [Orsag, 2014] and peg-in-hole insertion task [Car, 2018]. A three-arm onboard manipulator system has also been proposed in [Paul, 2019] which is capable of gripping objects with multiple forms and landing on an uneven surface. On the other hand, the use of parallel manipulators has greatly increased the achievable accuracy of the composed aerial manipulators, for which examples can be seen in contact-based inspection [Fumagalli, 2012; Scholten, 2013; Fumagalli, 2016], surface cleaning application [Kamel, 2016] (see Fig. 1.2(l)), precise positioning and manipulation of objects [Danko, 2015], and even aerial writing task [Tzoumanikas, 2020] (see Fig. 1.2(m)).

The application scenarios involve both indoor and outdoor operations, integrating outdoor navigation systems based on GNSS [Jimenez-Cano, 2013; Ruggiero, 2015a; Morton, 2016; Suarez, 2017a], indoor navigation features using Motion Capture (MOCAP) systems [Alexis, 2016; Hamaza, 2018; Romano, 2021; Palunko, 2012; Rossi, 2017; Orsag, 2014; Car, 2018; Danko, 2015; Kamel, 2016; Tzoumanikas, 2020], and onboard vision-based approaches [Ghadiok, 2011; Kim, 2016; Lippiello, 2016; Suarez, 2017b; Vempati, 2018].

(a) Quadrotor attached with an actuated rigid link for contact-based inspection task [Bartelds, 2016]

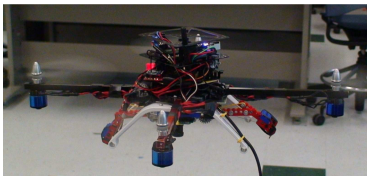


(b) Quadrotor mounted with a rigid bar for physical interaction with the environment [Yüksel, 2019]

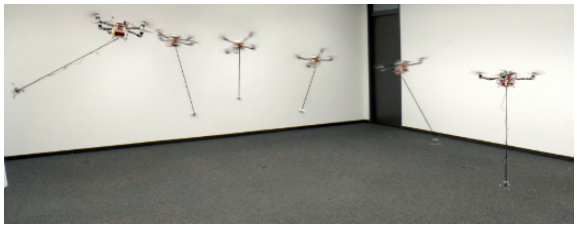


(c) Quadrotor equipped with a custom spray gun mechanism for spray painting [Vempati, 2018]

(d) Quadrotor equipped with impactive and ingressive gripper for grasping and transporting bar structures and pieces of wood [Mellinger, 2011b]



(e) Quadrotor with a compliant gripper for indoor aerial gripping [Ghadiok, 2011]

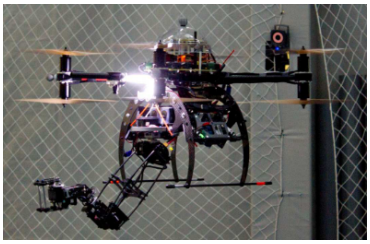


(f) Agile quadrotor manoeuvres with a cable-suspended payload [Foehn, 2017]

(g) Quadrotor assembling tensile structures [Augugliaro, 2013]

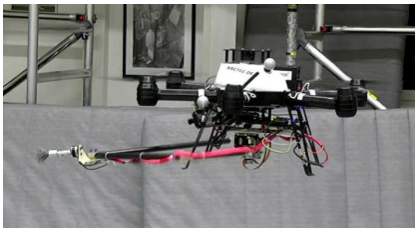
(h) Quadrotor with a 2-DoF arm for object manipulation [Kim, 2013]

(i) Quadrotor with a 2-DoF manipulator and gripper for aerial grasping [Tognon, 2017]

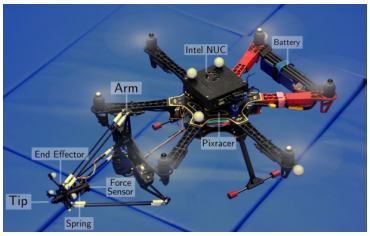


(j) Quadrotor endowed with a 6-DoF robotic arm for assembly task [Ruggiero, 2015a]

(k) Hexarotor equipped with a dual-arm system for grasping bar-like objects [Suarez, 2017a]



(l) Quadrotor mounted with a five-bar parallel mechanism for surface cleaning [Kamel, 2016]



(m) Hexarotor with a 3-DoF parallel arm for aerial writing task [Tzoumanikas, 2020]

Figure 1.2 – Aerial manipulator frameworks with a single multirotor platform and attached equipment for physical interaction.

Non-Conventional Multirotors with Arms

Apart from classical multirotors, there exists some derivations from these conventional aerial vehicles based on which novel aerial manipulator frameworks are established as illustrated in Fig. 1.3. One similar platform to the multirotor is the **ducted fan**, in which a main horizontal propeller is mounted within a cylindrical duct to generate the total thrust while a set of control vanes is available on the duct surface to achieve full controllability of the vehicle's attitude. It is therefore an aerial vehicle possessing the equivalent under-actuation and VTOL characteristics of multirotors. Similar frameworks have been proposed for physical interaction with the environment using a rigid link [Gentili, 2008; Marconi, 2011; Marconi, 2012; Forte, 2014], and a parallel robotic arm as in [Keemink, 2012; Torre, 2012] and [Naldi, 2018] (see Fig. 1.3(a)).

Besides, as the multirotor and ducted-fan aircraft are typically under-actuated systems, implying that the vehicle's attitude dynamics is coupled with its translational movements, which has limited the manipulability of such aerial manipulators in 3-dimensional space, recent works have also focused on developing **multirotors with tilted or tiltable propellers**. These novel and usually customised platforms consist of a main rigid frame and multiple tilted propellers fixed to the rigid frame [Rajappa, 2015; Rashad, 2019] or movable elements actuated by servo motors for tilting them [Ryll, 2016; Staub, 2018], showing their possibility of being fully-actuated, over-actuated or even omnidirectional systems that can exert total thrust along any direction with any value in a spherical shell independently from the total moment [Tognon, 2018a; Hamandi, 2021]. The special actuation characteristics of the tilted or tiltable multirotor makes it more versatile and efficient in accomplishing physical interaction tasks as they benefit from the increased aerodynamic behaviour due to non-parallel rotor configuration [Abbaraju, 2021].

The examples of these aerial manipulators can be seen in: "Tilt-Hex" robot with a fully-actuated tilted-propeller hexarotor for 6D physical interaction [Ryll, 2017; Franchi, 2018; Ryll, 2019] (see Fig. 1.3(b)); "OTHex" (Open-Tilted Hexarotor) platform endowed with a gripper for grasping bar structures [Staub, 2018] and a 2-DoF robotic arm for push-and-slide inspection [Tognon, 2019; Nava, 2020]; "AeroX" adopting an eight-tilted rotor and a 6-DoF robotic arm for physical contact inspection in the oil and gas industry [Trujillo, 2019] (see Fig. 1.3(c)); omnidirectional vehicles such as "Voliro" platform capable of wall interaction [Kamel, 2018]; "ODAR" system with eight non-aligned bidirectional rotors for peg-in-hole operation [Park, 2018]; an omnidirectional octorotor equipped with a pouch for catching a thrown ball [Brescianini, 2018] (see Fig. 1.3(d)); a tilting-hexarotor platform equipped with a rigid end-effector for contact inspection [Bodie, 2019; Bodie, 2021a], or mounted with a delta-based parallel manipulator for aerial pick-and-place ap-

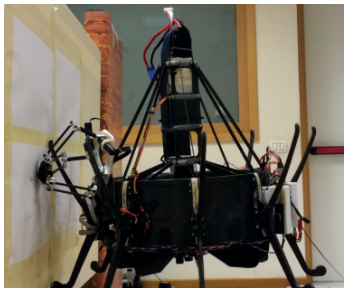
plication [Bodie, 2021b] (see Fig. 1.3(e)), and even with a soft robotic arm [Szasz, 2022] for applications in aerial construction, delivery, etc. Another interesting design with omnidirectional multirotors is "dextAIR" where an aerial manipulator suspended by elastic spring [Yiğit, 2021b; Perozo, 2022] (see Fig. 1.3(f)) or from a cable-driven parallel robot (CDPR) [Yiğit, 2021c] has been proposed to enhance the ability of this kind of aerial manipulators like increasing the accuracy, autonomy and especially reducing the energy consumption due to gravity compensation from the suspension system.

Last but not least, another type of derived platform worth mentioning is **morphing multirotors**, where the main body of the vehicle is composed of actuated links that are movable and equipped each with one or multiple propellers. These morphing designs allow for very high adaptability to the environment and interaction tasks, since the vehicle can not only modify the propeller directions but also its shape and thus the relative positions of propellers thanks to the actuated multi-link structures. For instance, [Falanga, 2019] has proposed a morphing quadrotor capable of folding itself with several possible morphologies for surface inspection and rescue application (see Fig. 1.3(g)); a similar morphing quadrotor that can optimise its morphology for transportation has been studied in [Kim, 2021]; a transformable quadrotor with two-dimensional multi-links capable of grasping, carrying and dropping objects has been proposed by [Zhao, 2017] (see Fig. 1.3(h)), which has then been extended to a fully-actuated tilted-octorotor design for aerial grasping of large-size objects [Anzai, 2018], and a fully-actuated quadrotor with tiltable propellers capable of opening the door [Sugito, 2022].

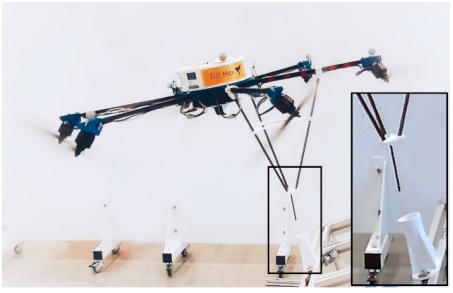
These derived multirotor-based platforms have shown their respective advantages such as fully- or even over-actuation, flexibility and configurability, which are interesting characteristics to enhance the capacity of the constructed aerial manipulators. On the other hand, there are further challenges in the mechanical design, motion planning and control of such systems due to their complexity.

As mentioned beforehand, all these aerial manipulator designs have been based on a variety of UAV platforms for the purpose of performing physical interaction tasks, which has opened up the field of aerial manipulation. The transition from helicopters to multirotor-based platforms has been due to the consideration of safety and stability issues that might be associated with aerial manipulators using helicopters during the flight and the manipulation task. Then a large branch of additional tools has been employed onboard a single multirotor to achieve all kinds of possible aerial applications. Similar designs can also be found using the ducted-fan vehicle, which is however a less interesting platform due to the complexity of the system and the similarity to conventional multirotors in their

characteristics. To further enhance the manipulability of aerial vehicles, frameworks with tilted- or tiltable propellers have overcome the under-actuation of multirotors, and morphing designs have shown their largely improved flexibility and adaptability to interact with the environment, which however further complicates the mechanical design and control of the system. Despite all these efforts put in for the enhancement of aerial vehicles, the single-UAV manipulator still presents a major drawback related to the limited payload capacity, resulting in tasks infeasible for a single manipulator such as handling a load too heavy or too large, or the required interaction wrench beyond the range a single UAV can exert. Therefore, there is a growing tendency towards multi-UAV manipulator designs, which will be presented in the next section.



(a) Ducted fan with a parallel arm for docking manoeuvre [Naldi, 2018]



(b) Tilt-Hex: a fully-actuated hexarotor with tilted rotors performing peg-in-hole task [Ryll, 2019]

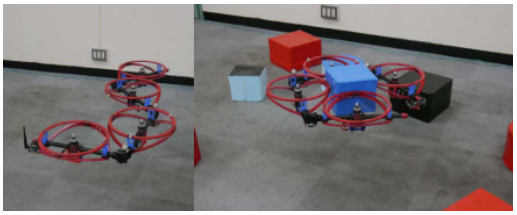
(c) AeroX: an eight-tilted rotor platform with an 6-DoF robotic arm for contact inspection [Trujillo, 2019]

(d) A tilted octorotor with a pouch mounted to catch ball [Brescianini, 2018]

(e) An omnidirectional hexarotor with a parallel arm for pick-and-place operation [Bodie, 2021b]

(f) dextAIR: an omnidirectional octorotor with elastic suspension [Perozo, 2022]

(g) A morphing quadrotor capable of folding and squeezing [Falanga, 2019]



(h) A transformable quadrotor with two-dimensional multi-links [Zhao, 2017]

Figure 1.3 – Aerial manipulator frameworks with derived multirotor-based platforms. (a): ducted-fan vehicle; (b)-(f): multirotor with tilted or tiltable propellers; (g) and (h): morphing multirotor designs.

1.1.2 Multi-UAV Manipulators: From Cooperative Systems to Rigidly Connected Systems

The designs where multiple UAVs are deployed to cooperatively perform an aerial task or to compose a multi-body system have been increasingly seen in recent works, which can be a potential solution for the transportation of heavy or large-size objects and the tasks infeasible by a single UAV. During a cooperative aerial manipulation, every single aerial manipulator can perform the task independently but they cooperate to enhance the manipulation ability, while in a multi-body system, all the UAVs are rigidly attached to a structure or a moving platform with articulated links for more challenging application scenarios. Each aerial vehicle consisting the cooperative or multi-body systems can be based on the main types of platforms previously described in Section 1.1.1, i.e. helicopter, conventional multirotor, ducted fan, and multirotor with tilted or tiltable propellers, etc.

Cooperative Systems with Multiple Independent Manipulators

An instinctive design for cooperative aerial manipulation is the cooperation of single-UAV manipulators with actuated equipment previously proposed as an aerial vehicle platform equipped with a rigid tool, a gripper or a robotic arm. Examples of such studies can be seen in cooperative grasping and transportation of objects using a team of quadrotors equipped with simple grippers such as [Mellinger, 2013] (see Fig. 1.4(a)), [Loianno, 2018a] (see Fig. 1.4(b)) and [Tagliabue, 2019], or with multi-DoF robotic arms [Kim, 2017] (see Fig. 1.4(c)) [Caccavale, 2015; Yang, 2015; Lee, 2015; Kim, 2018a; Kim, 2018b], and cooperative manipulation with multiple tool-embedded quadrotors [Gioioso, 2014; Mohammadi, 2016; Thapa, 2020]. An interesting design can also be noted where multiple quadrotors with docking mechanisms physically connect and form a whole system to a flying gripper [Gabrich, 2018].

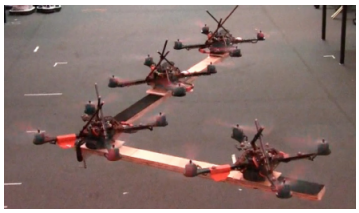
As each individual manipulator is rigidly attached to the objects when the grasp is engaged or the contact with the object is performed, the internal dynamics due to the interaction between the object and each manipulator should be carefully considered. This can be handled by modelling the cooperative multi-agent system as a single composed system and controlling it in a centralized manner [Caccavale, 2015; Lee, 2015; Mohammadi, 2016; Loianno, 2018a] using either control allocation or optimal control methods to determine the input of each agent. Decentralized controllers have also been investigated where each agent computes its own control law with only knowledge of its own states. Such a controller can be established either along with a higher-level planner such as an optimal contact force distribution process [Yang, 2015; Mohammadi, 2016] and a trajectory planner with each agent control enhanced by an onboard estimation of internal forces

[Kim, 2018b; Tagliabue, 2019], or by employing a consensus algorithm to ensure an equal share of estimated payload mass on each agent [Thapa, 2020]. Besides, motion planning of such multi-agent systems is significantly important in order to perform cooperatively the aerial grasping and transportation task, especially in an obstacle environment, as investigated in [Lee, 2015; Kim, 2017; Kim, 2018a].

Cable-Suspended Cooperative Systems

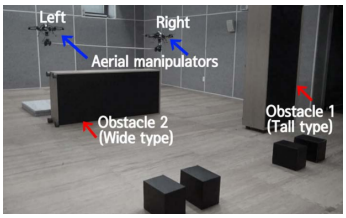
Another design of cooperative manipulation by aerial robots that has been intensively studied is the cable-suspended aerial system for transportation or manipulation using a number of UAVs, which can be considered as a composition of multiple aerial manipulators with passive links shown in Fig. 1.2(g). One possibly the earliest example of such cooperative systems is the aerial cable-towed system (ACTS) investigated for the first time in [Michael, 2011; Fink, 2011; Jiang, 2013; Sreenath, 2013a; Manubens, 2013] as an analogy to classical CDPR, during the time when mechanical design, motion planning, modelling and control problems of this aerial cooperative system have been addressed. Later, the works have been extended to a variable cable-towed system [Li, 2020b] (see Fig. 1.4(d)) in terms of mechanical design, a cooperative transportation system using only onboard sensing (i.e. monocular vision and IMU) [Li, 2021a] for the perception and pose estimation, an analysis on configuration planning of such systems based on wrench feasibility [Erskine, 2019a] in the aspect of planning, and a more robust ACTS under uncertainties [Sanalitro, 2020] regarding the control. The cable suspension designs can also be found in dual-UAV co-manipulation and transportation of rod-like objects [Tagliabue, 2017] (see Fig. 1.4(e)) [Gassner, 2017; Tognon, 2018b; Mohiuddin, 2020]. A catenary robot for non-prehensile manipulation of cuboid objects using a non-stretchable cable connected to two quadrotors has been proposed in [Cardona, 2021] (see Fig. 1.4(f)).

Unlike the cooperation of manipulators with onboard actuated equipment, this type of cooperative system often requires a preparation phase before the transportation or manipulation task during when the cables and the load (in transportation tasks) need to be assembled to the UAVs in advance, and therefore they are often modelled as a single robot including the states of all the UAVs and controlled in a centralized way. Exceptions can be found in several works using decentralized controllers based on the leader-follower paradigm [Tagliabue, 2017; Tognon, 2018b] or guided by onboard vision [Gassner, 2017]. The advantage of multiple UAVs supporting the payload by suspended cables is the large workspace and high payload capacity. However, the tensile-only nature of cables reduces the wrench feasibility of the robot [Erskine, 2018; Erskine, 2019b], making the cable-suspended systems only for transporting objects. As a result, designs on multiple UAVs with rigidly connected links have been more recently investigated.

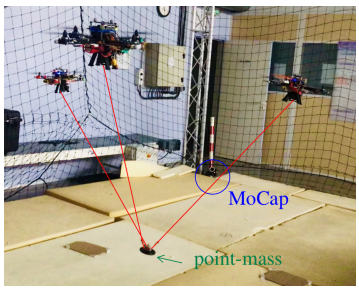


(a) Cooperative grasping and transportation by a team of manipulators with impactive grippers [Mellinger, 2013]

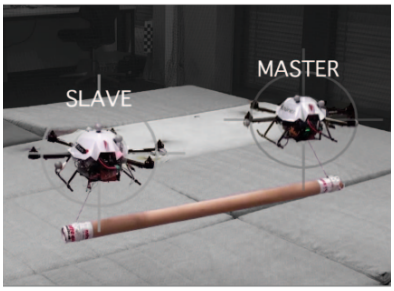
(b) Cooperative transportation by two quadrotors with electromagnet [Loianno, 2018a]



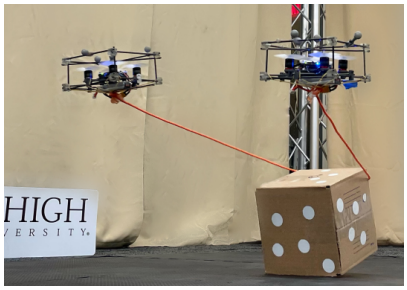
(c) Cooperative transportation of a rod by two quadrotors with robotic arm [Kim, 2017]



(d) Variable aerial cable-towed system for flexible cooperative transportation [Li, 2020b]



(e) Collaborative transportation of beam-like objects suspended by cables using two hexarotors [Tagliabue, 2017]



(f) Two quadrotors attached with a non-stretchable cable for manipulating cuboid objects [Cardona, 2021]

Figure 1.4 – Cooperative aerial manipulation frameworks. (a)-(c): cooperation of multiple aerial manipulators with onboard actuated equipment; (d)-(f): cable-suspended transportation or manipulation collaborated by multiple UAVs.

Rigidly Connected Systems with Multiple UAVs

Similarly to the ACTS, the rigidly connected system with multiple UAVs is considered as a single robot and required to be pre-assembled. According to the assembly configurations, the UAVs can be attached to a single rigid frame (as referred to Fig. 1.5(a)-(c)), or to an articulated architecture with actuated or passive joints to compose a multi-body system (seen in Fig. 1.5(d), (e)). In these rigidly connected systems, additional mechanisms are commonly used in order to maximise motion dexterity and improve the manipulability of the robot.

An earliest and successful demonstration of such systems is the SmQ (spherically-connected multi-quadrotor) platform [Nguyen, 2015; Nguyen, 2018] (see Fig. 1.5(a)). In this design, multiple quadrotors are connected to a rigid frame (i.e. a moving platform) for cooperative manipulation. The attachment of quadrotors onto the rigid frame is achieved via a passive spherical joint to decouple the rotational dynamics of each quadrotor with the movements of the rigid body, rendering the whole system fully actuated (i.e. full 6-dimensional pose of the rigid frame independently controllable). It has been proven that fully actuation can

be ensured with at least three well-located UAVs whose attaching points to the CoM of the platform (or the tool) are not collinear with each other [Nguyen, 2015]. The feasible wrench analysis and optimal control of this SmQ system with three quadrotors are investigated afterwards in [Nguyen, 2018].

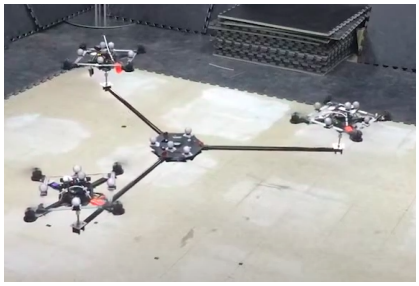
A similar design is an over-actuated multi-rotor aerial vehicle with more tiny quadrotors attached to a rigid frame via a two-DoF passive gimbal mechanism [Yu, 2021] (see Fig. 1.5(b)). The platform can achieve full 6-dimensional motion with redundancies from four quadrotors seen as vectoring force actuators. A nullspace-based control allocation method of this over-actuated system has been proposed in [Su, 2021] allowing better performance of trajectory tracking by the rigid platform.

Another interesting design is the flying gripper in which multiple UAVs are physically fixed to a rigid frame with well-arranged orientations to allow fully actuation of the rigid body. An under-actuated finger is further attached beneath each quadrotor and actuated by its yaw rotation using a non-back-drivable worm-gear mechanism permitting the opening and closing motion of four fingers for aerial grasping and transportation of large-size objects [Li, 2021c] (see Fig. 1.5(c)). The original design has been proposed in [Saint-Sevin, 2018] and optimisation on the geometric arrangements of quadrotors has also been addressed to maximise the robot's manipulability and its capacity to produce form-closed grasps. The dynamic control algorithm using Model Predictive Control (MPC) of the flying gripper has been investigated in [Li, 2021c]. A flying gripper design with mobile quadrotor attitudes has later proposed during the thesis of [Li, 2021b], in which the attachment of quadrotors to the rigid frame is achieved by means of universal joints, showing a more robust behaviour of the robot during the grasp due to the decoupling on quadrotor roll and pitch dynamics.

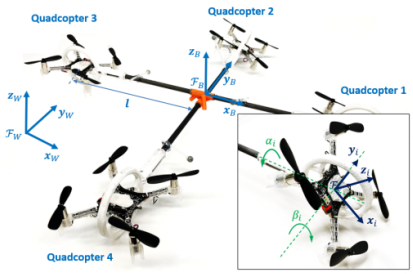
Moving forward from these designs with multiple UAVs attached to a single rigid structure, the multi-body system is a more recently proposed architecture composed of an articulated structure with passive or actuated joints and multirotors distributed over multiple links. Similarly to morphing multirotors with transformable multi-links, the use of articulated structure has largely increased the flexibility and the manipulability of the robot, and the whole system is usually fully-actuated due to the collaboration of multiple multirotors. One example of multi-body systems is the "DRAGON" platform composed of a transformable multi-body structure and ducted-fan vehicles attached to each link with servo motors to actuate the system. The robot has shown its capability of flying through a narrow window by changing its structure [Zhao, 2018], performing a squeezing task under a plate with an embedded soft end-effector [Zhao, 2020], and being applied to forceful

valve manipulation task [Zhao, 2022] (see Fig. 1.5(d)). Another example can be seen in "LASDRA" (large-size aerial skeleton system with distributed rotor actuation) initially proposed in [Yang, 2018]. The system consists of multiple links connected to each other using passive spherical joints, with each link fully actuated by distributed multi-rotors. A joint locking strategy has been proposed to increase the robot's loading capacity and a decentralized impedance control scheme has been studied to allow for compliant operation by such a large-size dexterously-articulated robot [Yang, 2018]. Further investigation of LASDRA system working in the outdoor environment has been done, in which a pose and posture estimation method has been proposed based on IMU and GNSS modules [Park, 2019] (see Fig. 1.5(e)).

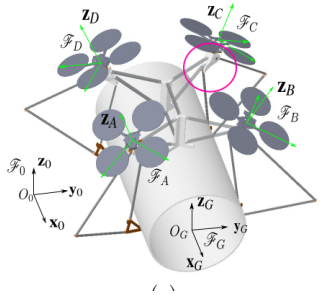
In conclusion, the cooperation of multiple aerial manipulator platforms has been more appropriate for the manipulation task unfeasible by a single UAV because of the payload capacity and actuation property. Such advantages have been particularly illustrated by the aerial cable-towed systems (ACTS) where a payload is supported by multiple UAVs using suspended cables. As a cable can only generate unilateral force, the system is only dedicated to aerial transportation tasks. Therefore, designs with multiple UAVs rigidly connected to a moving platform seem to be more appealing for full-pose manipulation of an object by the tool/end-effector attached to the rigid frame. To further enhance the flexibility and configurability of these aerial robots, multi-body systems with an articulated structure actuated by multiple UAVs have been investigated, showing their potential in accomplishing more complex tasks in unconstrained environments. Inspired by some of these designs, a flying parallel robot with a passive architecture actuated by multiple UAVs has been proposed, which will be presented in the next section.



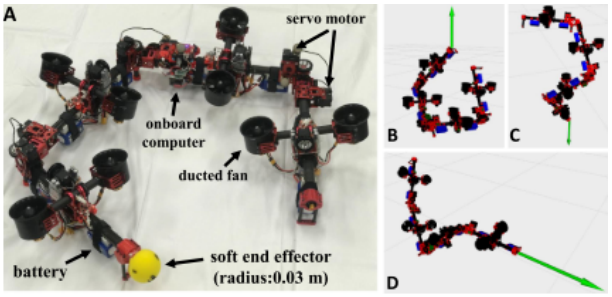
(a) SmQ: spherically connected multi-quadrotor system [Nguyen, 2018]



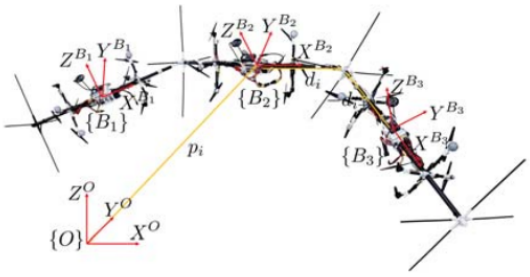
(b) An over-actuated UAV platform with four quadrotors rigidly connected to the frame using gimbal mechanism [Yu, 2021]



(c) Flying gripper with four quadrotors connected to fingers using gear-worm mechanisms [Li, 2021c]



(d) DRAGON platform: multi-link robot actuated by ducted fans [Zhao, 2020]



(e) LASDRA: large-size aerial skeleton system with distributed rotor actuation [Park, 2019]

Figure 1.5 – Aerial manipulators with rigidly connected structures. (a)-(c): multiple UAVs connected to a rigid frame using mechanical mechanisms; (d) and (e): multi-body systems actuated by multiple UAVs connected by means of passive joints.

1.2 Design of Flying Parallel Robots

Belonging to the class of multi-UAV parallel manipulators, the design of Flying Parallel Robot (FPR) has been proposed in the thesis of [Six, 2018a] where multiple multirotor-based UAVs are rigidly connected to an articulated structure via mechanical joints. The structure on which the multiple UAVs are attached is analogous to a classical parallel mechanism, and therefore it is named as *passive architecture* in the FPR design, implying that the articulated structure includes only the passive (i.e. not actuated) joints. The passive architecture is commonly composed of a moving platform and multiple kinematic chains connecting the multirotors as actuators to the platform via rigid bodies and passive joints. Fig. 1.6 shows examples of flying parallel robots with 2 or 3 UAVs and their analogies as a Bi-glide mechanism and a 3-RPS parallel mechanism. As it can be seen, the fixed base and the motors (within active joints) in parallel mechanisms are replaced by multirotors in the design of FPR using a similar architecture. The only actuation of the system is thus exclusively provided by the thrust forces of multirotors while the attach-

ments of multirotors using spherical joints ensure a decoupling property between their rotational dynamics and the dynamics of the passive architecture, making it possible to consider the multirotors as vectoring thrust generators in 3-dimensional space. One can also discover the same property through the use of spherical or universal joints in the design of SmQ platform [Nguyen, 2018], the over-actuated UAV platform [Yu, 2021] and the flying gripper with mobile-attitude quadrotors [Li, 2021b].

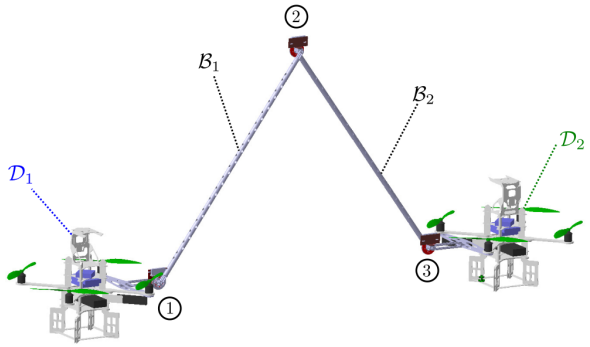
The original idea of this proposal is to combine a parallel architecture with the capacity provided by multiple UAVs, allowing to obtain several interesting properties such as:

- **High payload capacity** due to the distribution of the load on multiple UAVs;
- **No additional actuators** needed apart from the multirotors, which is advantageous in terms of energy consumption;
- **Low perturbations** on the end-effector from the airflow generated by the propellers of UAVs, which can be ensured by an adapted passive architecture that supports the end-effector far away from locations of UAVs;
- **High versatility and configurability** to the aerial tasks, as the use of rigid links and the possibility of reconfiguring the architecture allow to perform the task above or below the level of flight of UAVs by a single robot;
- **Large amount of possible architectures** presented in the community of parallel robots, offering the possibility of constructing the flying parallel robot with different kinematics and dynamics properties.

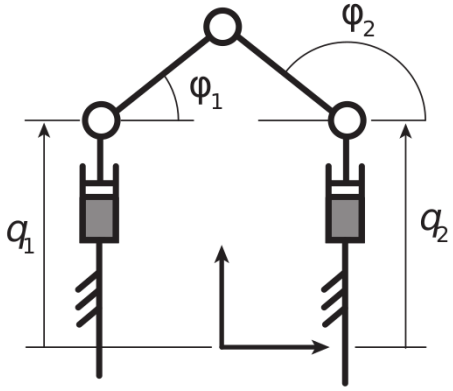
These advantageous properties allow the FPR to become a potential candidate for a versatile aerial manipulator capable of performing multiple manipulation tasks with a single design. In [Six, 2018a], analytical studies on the general modelling and motion control of flying parallel robots were conducted, with preliminary implementations on a 2D FPR actuated by two UAVs presented in [Six, 2017b] and a 3-D FPR with three UAVs referred to [Six, 2018b]. In the following parts, the literature review on the general modelling of the flying parallel robot and the motion control of a general FPR tracking a prescribed trajectory will be detailed.

1.2.1 General Modelling of Flying Parallel Robots

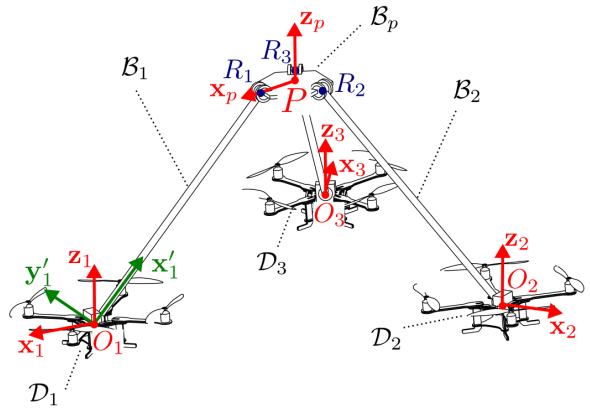
In this section, a general methodology regarding the dynamic modelling of flying parallel robots is presented. To keep the generality, the modelling of the FPR is developed on a general design composed of a moving platform and n legs attached with multirotors (as illustrated in Fig. 1.7) without specifying the number of legs. The general modelling can thus be applied to a specific design such as the ones shown in Fig. 1.6.



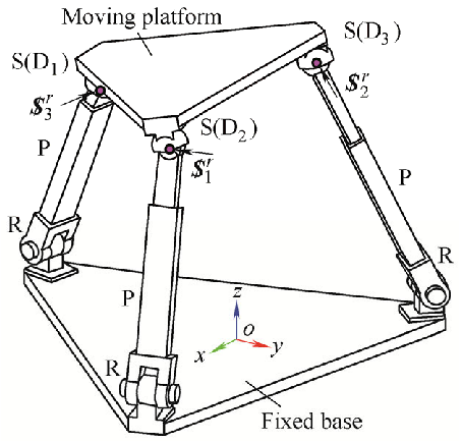
(a) Example of a 2D flying parallel robot with 2 UAVs [Six, 2018a]



(b) A Bi-glide parallel mechanism, with the active joints in grey.



(c) Example of a flying parallel robot with 3 UAVs [Six, 2018b]



(d) A 3-RPS mechanism [Qu, 2015]. The prismatic joints are actuated.

Figure 1.6 – Example of flying parallel robots with 2 UAVs analogous to a Bi-glide parallel mechanism and with 3 UAVs to a 3-RPS parallel mechanism.

In the general design of FPR, the CoM positions of multirotors are considered to be attached to the end of each leg by means of spherical joints. This hypothesis ensures the previously discussed decoupling property between the rotational movements of multirotors and that of the passive architecture, which is indispensable for designing a cascaded control law presented afterwards in Section 1.2.2. It has to be mentioned by the way that, the revolute joints are adequate for connecting the multirotors to the rigid structure in a 2D FPR design such as Fig. 1.6(a), which ensures the decoupling property in 2-dimensional space. The passive architecture in the FPR can be characterised by:

- the 6-dimensional pose of the moving platform (in case of 2D FPR design, its dimension is reduced to be expressed in 2D space);
- the additional coordinates associated with the internal configuration of each leg, which are specified according to different architecture designs.

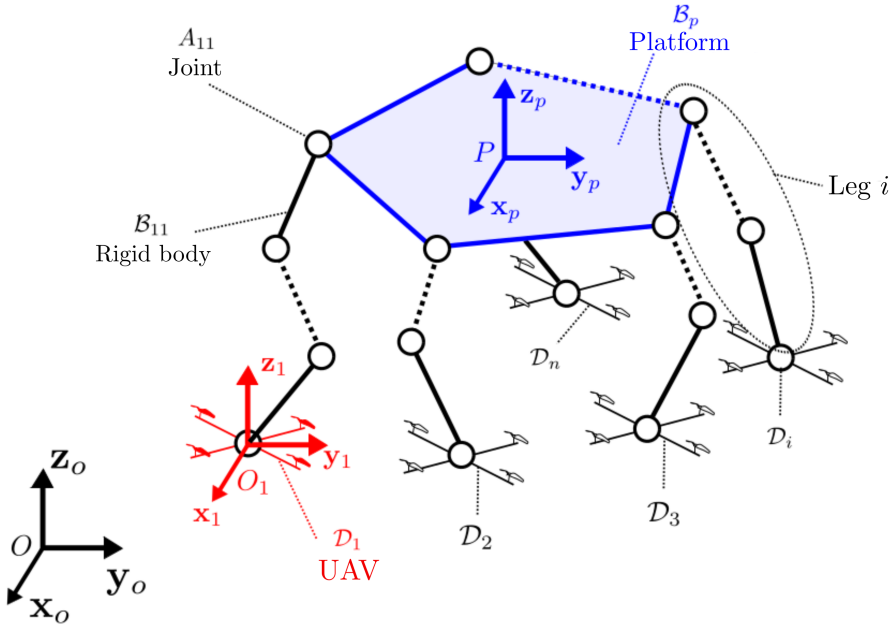


Figure 1.7 – General schema of the flying parallel robot design with n UAVs [Six, 2018a].

The 6-dimensional pose of the platform is commonly expressed by the position and orientation of the platform with respect to a global reference frame (denoted by $\mathfrak{F}_0$), using six coordinates $(x, y, z, \psi, \theta, \phi)$. The first three components define the CoM position of the platform in the global frame, and the last three components represent the orientation of the platform relative to the global frame using the convention of Euler-Bryant angles [Diebel, 2006]. Therefore, the vector of the generalised coordinates for the passive architecture can be denoted by $\mathbf{q}_p$ grouping the pose of the platform and the coordinates characterising the joint variables in the kinematic chains. The total number of DoFs of the FPR is thus characterised by the dimension of $\mathbf{q}_p$, denoted by n_p.

The modelling process can be inspired by the analytical methods well studied for parallel robots, which generally implies a virtual separation of the platform from the rest of the robot to compose a tree structure. In the general modelling of FPR, the multirotors are virtually separated from the passive architecture, which allows obtaining a tree structure from the moving platform to the extremities of the legs (as seen in Fig. 1.8). Similarly to the dynamic modelling of parallel robots, the dynamic model is then calculated for each of the virtually separated elements. The overall dynamics of the FPR can finally be established by expressing the interaction force between the extremity of each element and the attached multirotor through the spherical joint.

To model a tree structure, it is common to use the Modified Denavit-Hartenberg (MDH) convention [Khalil, 2002b], which has been extensively adopted in modelling serial or

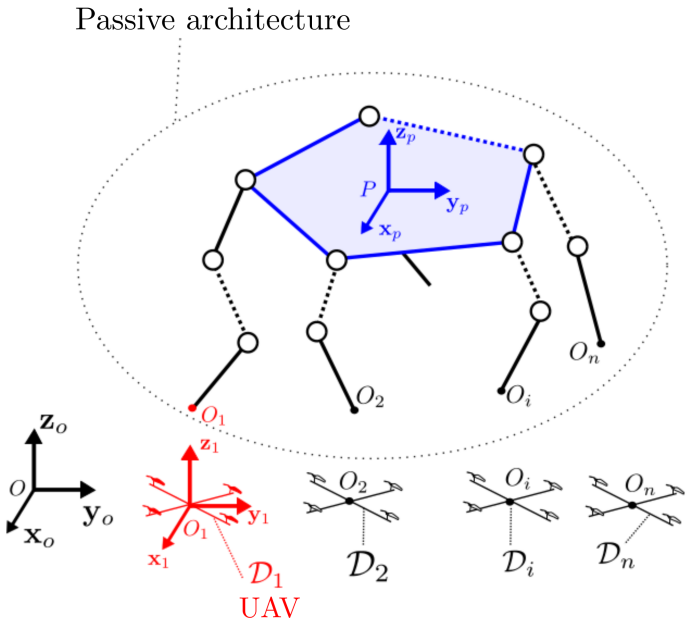


Figure 1.8 – Virtual separation of UAVs with the passive architecture [Six, 2018a].

tree-structure robots. The MDH convention uses a number of parameters to describe the position and orientation of the downstream body according to the antecedent body, dependent on the type of joint for the connection (usually classified by revolute joint or prismatic joint). More complex joints can be decomposed and represented by a combination of several simple joints, for instance, a spherical joint can be seen as three revolute joints with orthogonal axes. More details about the modelling of a tree-structure robot can be found in [Khalil, 2002b, Chapter 7]. The parameters of MDH convention for an arbitrary body i are given by:

- ant_i: the antecedent body of the body i in a single kinematic chain;
- μ_i: equals to 0 (or 1) if the joint between two bodies is passive (or active);
- σ_i: equals to 0 (or 1) if the joint type is revolute (or prismatic);
- $\alpha_i, d_i, \theta_i, r_i$: four parameters defining the position and orientation of the current joint i from the frame of its antecedent body;
- γ_i, b_i: supplementary parameters for the definition of frame in a tree structure.

To be able to model the mobility of the moving platform using the MDH convention, it is necessary to define a virtual serial architecture to relate the platform frame $\mathfrak{F}_p$ to the global frame $\mathfrak{F}_0$ using six coordinates of the platform pose. This virtual architecture can then be modelled by a 3P3R serial architecture (as illustrated in Fig. 1.9). The platform body $\mathcal{B}_p$ is supposed to be aligned with the last body in the 3P3R architecture while the intermediary virtual bodies (denoted as $\mathcal{V}_1$ to $\mathcal{V}_5$) are considered to have zero mass and inertia, and to be coincident with each other. The order of three revolute joints

depends on the rotation order defined by the chosen representation of orientation, e.g. for Euler-Bryant angles, the revolute joints are defined successively about x, y and z axis in the global frame. The parameters for the virtual architecture equivalent to the moving platform are given by Fig. 1.10.

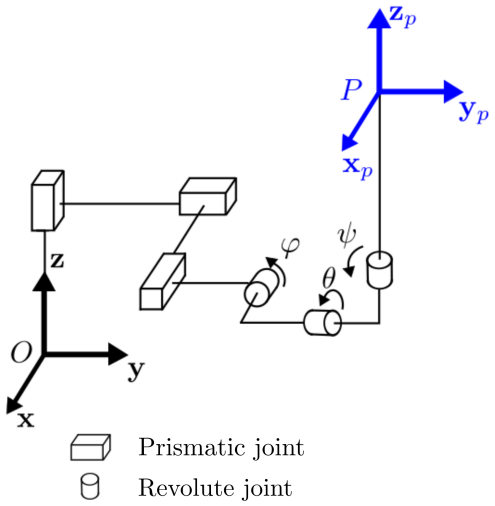


Figure 1.9 – Equivalent 3P3R architecture to the mobility of the moving platform [Six, 2018a].

i	ant$_i$	μ_i	σ_i	γ_i	b_i	α_i	d_i	θ_i	r_i
$\mathcal{V}_1$	$\mathcal{F}_o$	0	1	0	0	0	0	0	z
$\mathcal{V}_2$	$\mathcal{V}_1$	0	1	0	0	$-\frac{\pi}{2}$	0	$-\frac{\pi}{2}$	y
$\mathcal{V}_3$	$\mathcal{V}_2$	0	1	0	0	$-\frac{\pi}{2}$	0	0	x
$\mathcal{V}_4$	$\mathcal{V}_3$	0	0	0	0	0	0	φ	0
$\mathcal{V}_5$	$\mathcal{V}_4$	0	0	0	0	$\frac{\pi}{2}$	0	$\theta + \frac{\pi}{2}$	0
$\mathcal{B}_p$	$\mathcal{V}_5$	0	0	0	0	$\frac{\pi}{2}$	0	ψ	0

Figure 1.10 – Table of MDH parameters associated with the virtual 3P3R architecture corresponding to the mobility of the moving platform [Six, 2018a]

This MDH table will be completed according to the type of joints used in the legs and the geometric parameters from the platform to the extremities where the multirotors are located. After having fully determined the MDH table of the whole passive architecture, the computation of the geometric relation and kinematics of the FPR can be achieved by applying the well-structured analytical algorithms presented in [Khalil, 2002b], which results in a geometric model determining the positions of the leg extremities knowing the generalised coordinates $\mathbf{q}_p$ and a kinematic model defining the velocity relationship as follows:

$$^0\mathbf{t}_i = \mathbf{J}_i(\mathbf{q}_p)\dot{\mathbf{q}}_p \tag{1.1}$$

where:

- $\dot{\mathbf{q}}_p$ is the generalised velocity vector of the passive architecture;
- ${}^0\mathbf{t}_i$ is the twist [1] of the point O_i (i.e. the extremity of each leg i) expressed in the global frame $\mathfrak{F}_0$;
- $\mathbf{J}_i(\mathbf{q}_p) \in \mathbb{R}^{6\times n_p}$ is the Jacobian matrix associated to the leg i which is dependent on the generalised coordinates $\mathbf{q}_p$.

One may refer to [Khalil, 2002b] for more details on how to derive the geometric relationship and kinematic model using the MDH convention.

By separating the linear velocity $\mathbf{v}_i$ and angular velocity $\boldsymbol{\omega}_i$ of the twist, (1.1) can be further expressed by

$$\begin{bmatrix} {}^0\mathbf{v}_i \\ {}^0\boldsymbol{\omega}_i \end{bmatrix} = \begin{bmatrix} \mathbf{J}_{v,i} \\ \mathbf{J}_{\omega,i} \end{bmatrix} \dot{\mathbf{q}}_p \tag{1.2}$$

For the dynamic modelling of the FPR, the first step is to model the dynamics of the passive architecture excluding the multirotors, as it has been virtually decoupled with the multirotors to compose a tree structure. Then, by introducing the multirotor's dynamics and the interaction forces between the multirotors and the legs, the complete dynamic model of the FPR can be derived.

First of all, the dynamics of the passive architecture can be computed using the Euler-Lagrange formula. The Lagrangian associated to the passive architecture is given by

$$L_p(\mathbf{q}_p, \dot{\mathbf{q}}_p) = E_p(\mathbf{q}_p, \dot{\mathbf{q}}_p) - U_p(\mathbf{q}_p) \tag{1.3}$$

where E_p and U_p are respectively the kinematic and potential energy of the passive architecture. The equation of Euler-Lagrange is written as [Khalil, 2002b]

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{\mathbf{q}}_p}\right) - \left(\frac{\partial L}{\partial \mathbf{q}}\right) = \boldsymbol{\tau}_p \tag{1.4}$$

where the virtual actuation wrench $\boldsymbol{\tau}_p$ can be expressed as a function of the interaction wrench $\mathbf{w}_{p,i} = \begin{bmatrix} \mathbf{f}_{p,i}^T & \mathbf{m}_{p,i}^T \end{bmatrix}^T$ between the multirotor i and the passive architecture architecture, where $\mathbf{f}_{p,i}$ and $\mathbf{m}_{p,i}$ are the force and moment. This is done by defining a virtual velocity $\dot{\mathbf{q}}_p^*$ and a virtual twist $\mathbf{t}_i^*$ of the last body on each leg i. Then the principle of virtual power indicates that the virtual power of the passive architecture should be equivalent to the sum of power provided by the interaction wrench transmitted at the extremity

1. A *twist* is a 6-dimensional vector, representing the linear and angular velocity of a rigid body.

of each leg, i.e.

$$\dot{\mathbf{q}}_p^{*T} \boldsymbol{\tau}_p = \sum_{i=1}^{n} \mathbf{t}_i^{*T} \mathbf{w}_{p,i} \tag{1.5}$$

By replacing the term $\mathbf{t}_i^*$ by the kinematic relationship of (1.1), (1.5) becomes

$$\boldsymbol{\tau}_p = \sum_{i=1}^{n} \mathbf{J}_i^T \mathbf{w}_{p,i} \tag{1.6}$$

Introducing this relation in (1.4) and linearising the equation given the acceleration vector of the generalised coordinates $\ddot{\mathbf{q}}_p$, the expression of the dynamic model in matrix form can be finally obtained as

$$\mathbf{M}_p \ddot{\mathbf{q}}_p + \mathbf{c}_p = \sum_{i=1}^{n} \mathbf{J}_i^T \mathbf{w}_{p,i} \tag{1.7}$$

with $\mathbf{M}_p$ the generalised inertia matrix of the passive architecture, and $\mathbf{c}_p$ a vector including the Coriolis, centrifugal and gravitational effects.

Then, the second step involves the calculation of the interaction wrench based on the development of the dynamics of the virtually separated multirotors. Each multirotor produces a thrust force acting on its frame, being subject to the interaction wrench $\mathbf{w}_{p,i}$ from the legs and the gravity. Since the spherical joints are used to connect the multirotors to the passive architecture, the moments are not transmitted, i.e. $\mathbf{m}_{p,i} = \mathbf{0}$. The translational dynamics of a single multirotor i can thus be derived, which is expressed in the reference frame $\mathfrak{F}_0$ as follows

$$m_i \ddot{\mathbf{r}}_i = \mathbf{f}_i + m_i \mathbf{g} - \mathbf{f}_{p,i} \tag{1.8}$$

where:

- $\mathbf{r}_i$ is the CoM position of multirotor i expressed in $\mathfrak{F}_0$, and thus $\ddot{\mathbf{r}}_i$ is its linear acceleration;
- $\mathbf{f}_i$ is the 3-dimensional thrust force produced by the propellers of the multirotor i expressed in the global frame $\mathfrak{F}_0$;
- m_i is the mass of multirotor i;
- $\mathbf{g} = \begin{bmatrix} 0 & 0 & -g \end{bmatrix}^T$ is the gravity vector expressed in $\mathfrak{F}_0$.

By further expressing the acceleration of each multirotor (i.e. the acceleration of the leg extremity $\dot{\mathbf{v}}_i$) using the derivatives of (1.2), it is possible to express the interaction force between the multirotor and the passive architecture by

$$\mathbf{f}_{p,i} = \mathbf{f}_i + m_i \mathbf{g} - m_i (\mathbf{J}_{v,i} \ddot{\mathbf{q}}_p + \dot{\mathbf{J}}_{v,i} \dot{\mathbf{q}}_p) \tag{1.9}$$

After knowing the interaction force established by the translational dynamics of mul-

tirotors, this internal term in (1.7) can be cancelled to reach

$$\mathbf{M}_p\ddot{\mathbf{q}}_p + \mathbf{c}_p = \sum_{i=1}^{n} \mathbf{J}_{v,i}^T(\mathbf{f}_i - m_i\mathbf{J}_{v,i}\ddot{\mathbf{q}}_p - m_i\dot{\mathbf{J}}_{v,i}\dot{\mathbf{q}}_p + m_i\mathbf{g}) \tag{1.10}$$

reminding that $\mathbf{J}_{v,i}$ is the linear components of the Jacobian matrix, i.e. first three rows of $\mathbf{J}_i$, since no moments are transmitted to the passive architecture and thus the angular components (last three rows) of the Jacobian matrix in (1.2) are omitted. Knowing the rotation matrix associated to each multirotor's orientation, the 3-dimensional thrust force $\mathbf{f}_i$ in (1.10) can be expressed by

$$\mathbf{f}_i = \mathbf{R}_i{}^i\mathbf{f}_i \tag{1.11}$$

with ${}^i\mathbf{f}_i = \begin{bmatrix} 0 & 0 & f_{z,i} \end{bmatrix}^T$ the body-frame thrust force and $\mathbf{R}_i$ the rotation matrix dependent on the chosen convention of Euler angles. For the convention of Euler-Bryant angles (XYZ convention) defined by $(\phi_i, \theta_i, \psi_i)$, the rotation matrix $\mathbf{R}_i$ is given by

$$\mathbf{R}_i = \begin{bmatrix} \cos\theta_i\cos\psi_i & -\cos\theta_i\sin\psi_i & \sin\theta_i \\ \cos\phi_i\sin\psi_i + \cos\phi_i\sin\theta_i\sin\psi_i & \cos\phi_i\cos\psi_i - \sin\phi_i\sin\theta_i\sin\psi_i & -\sin\phi_i\cos\theta_i \\ \sin\phi_i\sin\psi_i - \cos\phi_i\sin\theta_i\cos\psi_i & \sin\phi_i\cos\psi_i + \cos\phi_i\sin\theta_i\sin\psi_i & \cos\phi_i\cos\theta_i \end{bmatrix} \tag{1.12}$$

Introducing (1.11) in (1.10) and reformulating the equation, the following relation can be obtained

$$\Big(\mathbf{M}_p + \sum_{i=1}^{n} m_i\mathbf{J}_{v,i}^T\mathbf{J}_{v,i}\Big)\ddot{\mathbf{q}}_p + \Big(\mathbf{c}_p + \sum_{i=1}^{n} m_i\mathbf{J}_{v,i}^T(\dot{\mathbf{J}}_{v,i}\dot{\mathbf{q}}_p - \mathbf{g})\Big) = \sum_{i=1}^{n} \mathbf{J}_{v,i}^T\mathbf{R}_i{}^i\mathbf{f}_i \tag{1.13}$$

By further grouping the relations for all the legs and multirotors, the dynamic model of the FPR can be written in matrix form as

$$\mathbf{M}_t\ddot{\mathbf{q}}_p + \mathbf{c}_t = \mathbf{J}_p^T\mathbf{R}_t\mathbf{f}_t \tag{1.14}$$

where:

- $\mathbf{M}_t = \mathbf{M}_p + \sum_{i=1}^{n} m_i\mathbf{J}_{v,i}^T\mathbf{J}_{v,i} \in \mathbb{R}^{n_p \times n_p}$
  This matrix is the composed generalised inertia matrix, which has taken into account the inertia matrix of the passive architecture and the sum of inertial effects due to the mass of multirotors attached at the extremities of legs.

- $\mathbf{c}_t = \mathbf{c}_p + \sum_{i=1}^{n} m_i\mathbf{J}_{v,i}^T(\dot{\mathbf{J}}_{v,i}\dot{\mathbf{q}}_p - \mathbf{g}) \in \mathbb{R}^{n_p}$
  This vector is composed of the Coriolis, centrifugal and gravitational terms corresponding to the passive architecture $\mathbf{c}_p$ and those effects generated by the mass of multirotors.

- $\mathbf{J}_p = \begin{bmatrix} \mathbf{J}_{v,1} \\ \mathbf{J}_{v,2} \\ \vdots \\ \mathbf{J}_{v,n} \end{bmatrix} \in \mathbb{R}^{3n \times n_p}$

  The complete Jacobian matrix concatenates the individual Jacobian matrices corresponding to each multirotor. Note that this Jacobian matrix is a priori not square.

- $\mathbf{R}_t = \begin{bmatrix} \mathbf{R}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{R}_n \end{bmatrix} \in \mathbb{R}^{3n \times 3n}$

  This is a square matrix concatenating the rotation matrices of all the multirotors.

- $\mathbf{f}_t = \begin{bmatrix} {}^1\mathbf{f}_1 \\ {}^2\mathbf{f}_2 \\ \vdots \\ {}^n\mathbf{f}_n \end{bmatrix} \in \mathbb{R}^{3n}$

  This is a vector concatenating the 3-dimensional thrust forces produced by all the multirotors, expressed in the local frame of each multirotor.

The rotational dynamics of multirotors are fully decoupled through the use of spherical joints, with the hypothesis being implicitly made that the centre of spherical joints is located at the CoM position of each multirotor. It can thus be expressed by a well-known formula such as the one presented in [Mahony, 2012]. The orientation of the multirotor can also be characterised by Euler-Bryant angles, denoted by $\boldsymbol{\eta}_i = \begin{bmatrix} \phi_i & \theta_i & \psi_i \end{bmatrix}^T$, with its body-frame rotational velocity and acceleration defined respectively by ${}^i\boldsymbol{\omega}_i$ and ${}^i\dot{\boldsymbol{\omega}}_i$. Then the rotational dynamics of a single multirotor can be given by

$$\mathbf{I}_i {}^i\dot{\boldsymbol{\omega}}_i = -{}^i\boldsymbol{\omega}_i \times (\mathbf{I}_i {}^i\boldsymbol{\omega}_i) + {}^i\mathbf{m}_i \tag{1.15}$$

with $\mathbf{I}_i$ the moment of inertia of the multirotor expressed in its body frame and ${}^i\mathbf{m}_i$ the moment generated by the multirotor about its own frame. The angular velocity of each multirotor i can be related to the rates of the Euler angles via the equation

$${}^i\boldsymbol{\omega}_i = \mathbf{D}_i \dot{\boldsymbol{\eta}}_i \tag{1.16}$$

where the matrix $\mathbf{D}_i$ depends on the convention of the Euler angles, which can be given by the following equation coherent with the rotation matrix defined in (1.12).

$$\mathbf{D}_i = \begin{bmatrix} \cos\theta_i \cos\psi_i & \sin\psi_i & 0 \\ -\cos\theta_i \sin\psi_i & \cos\psi_i & 0 \\ \sin\theta_i & 0 & 1 \end{bmatrix} \tag{1.17}$$

Note that for small angles, ${}^{i}\boldsymbol{\omega}_i \approx \dot{\boldsymbol{\eta}}_i$. This approximation is however not considered in the model as multirotors might have large inclination angles. By introducing the relation of (1.16) and its derivative into (1.15), the following equation can be obtained

$$\mathbf{I}_i\mathbf{D}_i\ddot{\boldsymbol{\eta}}_i = -\mathbf{I}_i\dot{\mathbf{D}}_i\dot{\boldsymbol{\eta}}_i - \mathbf{D}_i\dot{\boldsymbol{\eta}}_i \times (\mathbf{I}_i\mathbf{D}_i\dot{\boldsymbol{\eta}}_i) + {}^{i}\mathbf{m}_i \tag{1.18}$$

The rotational dynamics of all the multirotors can be grouped and written in the same matrix form as in (1.14) by

$$\mathbf{M}_a\ddot{\mathbf{q}}_a + \mathbf{c}_a = \mathbf{m} \tag{1.19}$$

where:

- $\mathbf{q}_a = \begin{bmatrix} \boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \\ \vdots \\ \boldsymbol{\eta}_n \end{bmatrix} \in \mathbb{R}^{3n}$
  The vector concatenates the orientation coordinates of the multirotors.
- $\mathbf{M}_a = \begin{bmatrix} \mathbf{I}_1\mathbf{D}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_2\mathbf{D}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{I}_n\mathbf{D}_n \end{bmatrix} \in \mathbb{R}^{3n\times 3n}$
  The matrix concatenates the product between the moment of inertia and the matrix $\mathbf{D}$ for each multirotor in diagonal elements.
- $\mathbf{c}_a = \begin{bmatrix} \mathbf{I}_1\dot{\mathbf{D}}_1\dot{\boldsymbol{\eta}}_1 + \mathbf{D}_1\dot{\boldsymbol{\eta}}_1 \times (\mathbf{I}_1\mathbf{D}_1\dot{\boldsymbol{\eta}}_1) \\ \vdots \\ \mathbf{I}_n\dot{\mathbf{D}}_n\dot{\boldsymbol{\eta}}_n + \mathbf{D}_n\dot{\boldsymbol{\eta}}_n \times (\mathbf{I}_n\mathbf{D}_n\dot{\boldsymbol{\eta}}_n) \end{bmatrix} \in \mathbb{R}^{3n}$
  This vector groups the Coriolis and centrifugal effects associated with the rotational movements of multirotors.
- $\mathbf{m} = \begin{bmatrix} {}^{1}\mathbf{m}_1 \\ \vdots \\ {}^{n}\mathbf{m}_n \end{bmatrix} \in \mathbb{R}^{3n}$
  The vector concatenates the moments applied by the propellers of each multirotor on its body frame.

In general, the dynamic model of the FPR is composed of the dynamics of passive architecture including the effects due to the masses of multirotors depicted by (1.14) and the rotational dynamics of the multirotors in (1.19), formulating a decoupled system in two levels. The derivation of this dynamic model and its property is indispensable to designing the control algorithms, such as the motion control presented as follows.

1.2.2 Motion Control of Flying Parallel Robots

As mentioned above, the decoupling between the rotational dynamics of multirotors and the dynamics of the passive architecture allows to construct a cascaded control law, which is composed of:

- A high-level control loop for the trajectory tracking of the passive architecture;
- A low-level attitude control of multirotors to achieve the thrust forces required by the high-level loop.

This arrangement is achievable and commonly seen in other rigidly connected multi-UAV designs such as [Nguyen, 2018], because the response of multirotor's rotational dynamics is much faster than the dynamics of the rigid structure.

For the control of the passive architecture, considering the dynamic model given by (1.14), an auxiliary control input $\boldsymbol{\nu}_p$ can be defined such that the inverse dynamic model can be written as a double integrator system, i.e.

$$\ddot{\mathbf{q}}_p = \boldsymbol{\nu}_p = \mathbf{M}_t^{-1}(\mathbf{J}_p^T \mathbf{R}_t \mathbf{f}_t - \mathbf{c}_t) \tag{1.20}$$

Given a desired trajectory on the coordinates of the passive architecture to follow, denoted by $(\mathbf{q}_p^d, \dot{\mathbf{q}}_p^d, \ddot{\mathbf{q}}_p^d)$, the auxiliary input can be established by a PID (Proportional-Integral-Derivative) control law for minimising the tracking error defined by $\mathbf{e}_p = \mathbf{q}_p - \mathbf{q}_p^d$, which is written as

$$\boldsymbol{\nu}_p = \ddot{\mathbf{q}}_p^d - \mathbf{K}_p \mathbf{e}_p - \mathbf{K}_d \dot{\mathbf{e}}_p - \mathbf{K}_i \int \mathbf{e}_p \tag{1.21}$$

with $\mathbf{K}_p$, $\mathbf{K}_d$ and $\mathbf{K}_i$ the positive-definite diagonal matrices respectively for the proportional, derivative and integral gains. These gains should be properly determined to make sure the convergence of the double integrator system of (1.20). In general cases, a PD regulation is sufficient to ensure the convergence towards the desired trajectory. However, the integral term is sometimes mandatory to compensate for the static error due to the inevitable modelling errors, which in contrast can not be handled by a PD controller. Then, combining equations (1.20) and (1.21), the control input of the passive architecture denoted by an auxiliary variable $\boldsymbol{\nu}_J$ of dimension $(3n \times 1)$ is known

$$\boldsymbol{\nu}_J = (\mathbf{J}_p^T)^\dagger \Big(\mathbf{M}_t \big(\ddot{\mathbf{q}}_p^d - \mathbf{K}_p \mathbf{e}_p - \mathbf{K}_d \dot{\mathbf{e}}_p - \mathbf{K}_i \int \mathbf{e}_p \big) + \mathbf{c}_t \Big) \tag{1.22}$$

which is the vector of thrust forces required to actuate the system expressed in $\mathfrak{F}_0$, i.e.

$$\boldsymbol{\nu}_J = \mathbf{R}_t \mathbf{f}_t = \begin{bmatrix} \mathbf{f}_1^T & \mathbf{f}_2^T & \cdots & \mathbf{f}_n^T \end{bmatrix}^T \tag{1.23}$$

with $\mathbf{f}_i$ the 3-dimensional thrust force of multirotor i expressed in $\mathfrak{F}_0$. Note that $(\mathbf{J}_p^T)^\dagger$

is the pseudo-inverse of the transpose of the Jacobian matrix $\mathbf{J}_p$, since the matrix $\mathbf{J}_p$ is generally not a square matrix, i.e. $3n \neq n_p$.

Once the commands on the vector of thrust forces are known, the setpoints for the low-level system can be determined, consisting in the total thrust and the desired attitude of each multirotor. It can be done by simplifying the expression of (1.23) to keep only the z components of each ${}^i\mathbf{f}_i$, as the thrust forces produced by multirotors are always along the z axis of its own frame. Therefore, the expression of (1.23) can be reduced to be expressed in the form of

$$\boldsymbol{\nu}_J = \mathbf{R}_r \mathbf{f}_z \tag{1.24}$$

with $\mathbf{f}_z = \begin{bmatrix} f_{z,1} & f_{z,2} & \cdots & f_{z,n} \end{bmatrix}^T$ being the vector concatenating the z components of each ${}^i\mathbf{f}_i$ as the total thrust produced by each multirotor, and $\mathbf{R}_r$ the reduced matrix of $\mathbf{R}_t$ having the dimension of $(3n \times n)$ which is composed of the third columns of each rotation matrix $\mathbf{R}_i$. From (1.12) and (1.14), it can be written by

$$\mathbf{R}_r = \begin{bmatrix} \begin{matrix} \sin\theta_1 \\ -\sin\phi_1\cos\theta_1 \\ \cos\phi_1\cos\theta_1 \end{matrix} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \begin{matrix} \sin\theta_2 \\ -\sin\phi_2\cos\theta_2 \\ \cos\phi_2\cos\theta_2 \end{matrix} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \begin{matrix} \sin\theta_n \\ -\sin\phi_n\cos\theta_n \\ \cos\phi_n\cos\theta_n \end{matrix} \end{bmatrix} \tag{1.25}$$

which brings the direct relationship of the auxiliary variable $\boldsymbol{\nu}_J$ with the total thrust and attitude for each multirotor i as

$$\begin{aligned} \sin\phi_i f_{z,i} &= \nu_{J,3i-2} \\ -\sin\phi_i\cos\theta_i f_{z,i} &= \nu_{J,3i-1} \\ \cos\phi_i\cos\theta_i f_{z,i} &= \nu_{J,3i} \end{aligned} \tag{1.26}$$

with $\nu_{J,k}$ the k-th component of the vector $\boldsymbol{\nu}_J$. Therefore, the total thrust and the desired attitude (roll and pitch) of each multirotor can be determined by

$$\begin{aligned} f_{z,i} &= \sqrt{\nu_{J,3i-2}^2 + \nu_{J,3i-1}^2 + \nu_{J,3i}^2} \\ \phi_i &= -\text{atan2}(\nu_{J,3i-1}, \nu_{J,3i}) \\ \theta_i &= \text{asin}(\frac{\nu_{J,3i-2}}{f_{z,i}}) \end{aligned} \tag{1.27}$$

However, the yaw angle ψ_i of each multirotor cannot be determined from (1.24). According to (1.16) and (1.17), the multirotor's yaw movement will have an effect on its inclination angle, and thus the 3-dimensional thrust it produces. To produce the correct thrust, a predefined constant yaw for each multirotor should be maintained.

In the low-level control loop running at a higher frequency, the objective is to achieve the desired setpoints computed beforehand. Considering the rotational dynamics of multirotors depicted by (1.19), the control input of the attitude controller consists of the moments generated on each multirotor's frame, i.e. the vector of $\mathbf{m}$. The desired setpoints of all the multirotors can be grouped by a vector $\mathbf{q}_a^d$, with the tracking error defined by $\mathbf{e}_a = \mathbf{q}_a - \mathbf{q}_a^d$. Similarly to the process in the high-level loop, the inversion of the dynamic model allows to construct a double integrator system, by defining an auxiliary input $\boldsymbol{\nu}_a$ such that

$$\ddot{\mathbf{q}}_a = \boldsymbol{\nu}_a = \mathbf{M}_a^{-1}(\mathbf{m} - \mathbf{c}_a) \tag{1.28}$$

A control law can therefore be designed based on the auxiliary input $\boldsymbol{\nu}_a$ which is corresponding to the angular accelerations of multirotors. To make up the concept of a cascaded control algorithm in two levels, rapid convergence and good stability of attitude tracking are crucial in the low-level controller. The sliding mode control is thus chosen to ensure these two features, which in addition is robust to the modelling errors and perturbations. The vector of the sliding surface variable is firstly defined by [Six, 2018a]

$$\boldsymbol{\sigma} = \dot{\mathbf{e}}_a + \boldsymbol{\Lambda}_1 \mathbf{e}_a + \boldsymbol{\Lambda}_2 \int \mathbf{e}_a \tag{1.29}$$

with $\boldsymbol{\Lambda}_1$ and $\boldsymbol{\Lambda}_2$ two positive-definite diagonal matrices. The auxiliary input can then be computed by sliding mode as

$$\boldsymbol{\nu}_a = \ddot{\mathbf{q}}_a^d - \boldsymbol{\Lambda}_1 \dot{\mathbf{e}}_a - \boldsymbol{\Lambda}_2 \mathbf{e}_a - \mathbf{K}_s \text{sign}(\boldsymbol{\sigma}) \tag{1.30}$$

with $\mathbf{K}_s$ a positive-definite diagonal matrix for the gains. The output of the function sign(.) is defined by a vector $\mathbf{s}$ of the same dimension

$$\text{sign}(\sigma_k) = \begin{cases} 1 & \text{if} \quad \sigma_k > 0 \\ 0 & \text{if} \quad \sigma_k = 0 \\ -1 & \text{if} \quad \sigma_k < 0 \end{cases} \tag{1.31}$$

The stability of the sliding mode control can be proven using the Lyapunov theory [Khalil, 2002b]. Considering a Lyapunov function defined by $V = \boldsymbol{\sigma}^T \boldsymbol{\sigma}$, its derivative is written by

$$\dot{V} = -2\boldsymbol{\sigma}^T \dot{\boldsymbol{\sigma}} \tag{1.32}$$

By introducing (1.29), there is

$$\dot{V} = -2\boldsymbol{\sigma}^T(\ddot{\mathbf{q}}_a - \ddot{\mathbf{q}}_a^d + \boldsymbol{\Lambda}\dot{\mathbf{e}}_a + \boldsymbol{\Lambda}_i\mathbf{e}_a) \tag{1.33}$$

Combining (1.30) and (1.33), the following relation can be found

$$\dot{V} = -2\boldsymbol{\sigma}^T\mathbf{K}_s\text{sign}(\boldsymbol{\sigma}) \tag{1.34}$$

which is strictly negative for $\boldsymbol{\sigma}$ not zero. Therefore, $\boldsymbol{\sigma}$ converges asymptotically towards zero in infinite time, which ensures the stability of the closed-loop system. Once the convergence of the sliding surface variable is achieved, the evolution of the tracking error can be written by the error dynamics

$$\dot{\mathbf{e}}_a + \boldsymbol{\Lambda}\mathbf{e}_a + \boldsymbol{\Lambda}_i \int \mathbf{e}_a = \mathbf{0} \tag{1.35}$$

The convergence towards the desired attitude $\mathbf{q}_a^d$ is guaranteed, since $\boldsymbol{\Lambda}$ and $\boldsymbol{\Lambda}_i$ are the matrices with coefficients strictly positive. However, despite its robustness, the sliding mode control often suffers from the chattering issue, which is an undesired phenomenon caused by the non-continuous output of sign(.) function. A solution to limit this problem is to replace the sign(.) by a saturation function sat(.) which outputs continuously and is given by

$$\text{sat}(\sigma_k) = \begin{cases} 1 & \text{if} \quad \sigma_k > \epsilon_k \\ \sigma_k/\epsilon_k & \text{if} \quad \|\sigma_k\| \leq \epsilon_k \\ -1 & \text{if} \quad \sigma_k < -\epsilon_k \end{cases} \tag{1.36}$$

with ϵ_k an arbitrarily small value defining a boundary around the sliding surface.

Finally, the control input for the attitude controller is obtained by the combination of sliding mode control law and the dynamic model as follows

$$\mathbf{m} = \mathbf{M}_a\Big(\ddot{\mathbf{q}}_a^d - \boldsymbol{\Lambda}\dot{\mathbf{e}}_a - \boldsymbol{\Lambda}_i\mathbf{e}_a - \mathbf{K}_s\text{sat}(\boldsymbol{\sigma})\Big) + \mathbf{c}_a \tag{1.37}$$

The control input of multirotors is then completed by $\mathbf{f}_z$ and $\mathbf{m}$, i.e. the vector of total thrusts and the vector of moments. These commands can be achieved by an onboard flight controller that distributes the force and moment commands to the rotor speeds. It needs to be mentioned that for quadrotors, the commands of $\mathbf{f}_z$ and $\mathbf{m}$ are sufficient to fully determine the motor inputs of each propeller, while in the case of hexarotor or octorotor, an additional allocation method should be applied to allocate the control commands since the number of actuators is greater than the dimension of the commands (i.e. the system is over-actuated).

The overall control algorithm of the flying parallel robot can be summarised by the diagram shown in Fig. 1.11. The high-level loop regulates the coordinates of the passive architecture, with its output determining the total thrust and the attitude (roll and pitch) for each multirotor required to actuate the system. The desired roll and pitch angles from the output of the high-level loop are usually filtered to avoid high-frequency noises, and the desired yaw is predefined to fully compose the attitude setpoints. Then the low-level loop running at a higher frequency stabilises the attitude of the multirotors and ensures the production of desired thrust forces using the sliding mode control law. The stability of the complete control procedure for trajectory tracking is ensured by the arrangement of the cascaded control algorithm.

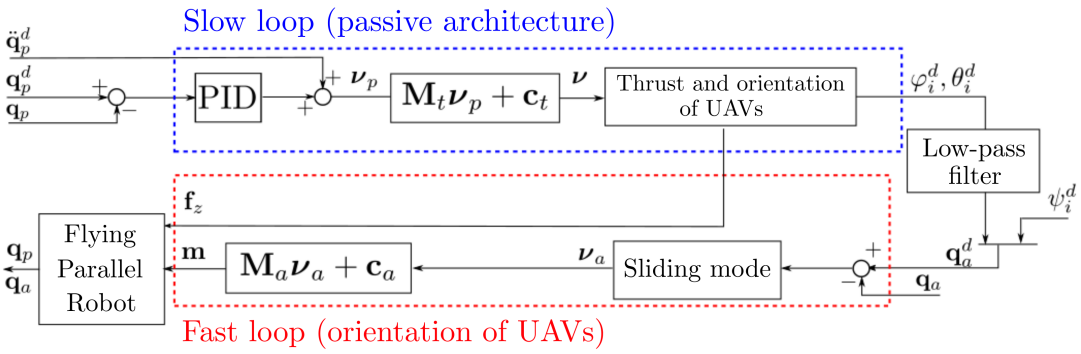


Figure 1.11 – Overall control diagram of the flying parallel robot [Six, 2018a].

1.3 Conclusion

In this chapter, the topics of aerial manipulation and the design of the Flying Parallel Robot (FPR) have been discussed. A literature review on the existing aerial manipulator architectures is firstly conducted, in which a variety of aerial vehicle platforms have been adopted to achieve the manipulation tasks, and the transition of designs has been found from single-UAV manipulators equipped with onboard end-effectors to multi-UAV designs as cooperation systems or rigidly connected systems. Belonging to the class of multi-UAV manipulators, the FPR presents a passive architecture analogous to a parallel mechanism supported by multiple UAVs. This design has the potential of achieving various and possibly complex manipulation tasks with a single architecture due to the advantages of combining the parallel mechanism design with aerial vehicles.

General modelling of the FPR has then been presented, and the decoupling property between the rotational dynamics of multirotors and the dynamics of the passive architecture allows the design of a cascaded control algorithm which has the objective to track a desired trajectory of the passive architecture in the high-level loop and stabilise

the attitudes of multirotors in the low-level loop. These state-of-the-art studies on aerial manipulators and the general design of FPR serve as a solid basis for this thesis. The modelling process and the motion control algorithm have also been the preliminaries for the studies on a specific FPR presented in the rest of the manuscript.

Chapter 2

Geometric, Kinematic and Dynamic Modelling

This chapter presents the geometric relations, kinematics and dynamics of a specific Flying Parallel Robot (FPR) composed of a moving platform supported by a number of multirotor-based Unmanned Aerial Vehicles (UAVs) with passive kinematic chains. An introduction to this particular design of the FPR is given in the first section. The second section presents the system description such as the coordinate frames and the robot states. The third section demonstrates the geometric relations, which are then derived to compute the kinematics in the fourth section. The dynamic modelling of the FPR is detailed in the fifth section, in which the dynamics of the passive architecture is modelled using two numerical algorithms and the decoupled rotational dynamics for each multirotor is discussed. Numerical validation of the kinematics and dynamics is presented in the sixth section.

2.1 Introduction

The Flying Parallel Robot (FPR) studied within the scope of this thesis is a specific case of the general design presented in Section 1.2, which is explicitly composed of

- a **moving platform** potentially equipped with an end-effector;
- a number of **rigid legs** connected to the platform using one-DoF revolute joints;
- **multirotors** connected at the extremity of each leg by means of spherical joints.

All the joints in the FPR are not actuated and the only actuation of the system is provided by the multirotors. The location of the spherical joints connecting the multirotors to the rigid legs is assumed to be the Centre of Mass (CoM) position of each multirotor, which is a reasonable assumption commonly found in other multi-UAV robots with rigidly-connected links as in [Nguyen, 2018; Li, 2021b]. The spherical joint is a constraint element allowing the relative rotation of two bodies and transmitting only force but not moment. This mechanical property makes it possible to decouple the rotational dynamics of multirotors from the dynamics of the *passive architecture*[1], similarly to what has been discussed in

1. Recall that *passive architecture* refers to the rigid articulated structure including the moving platform, the rigid legs and the passive (i.e. not actuated) joints, as presented in Section 1.2

Section 1.2. The multirotors can thus be considered as rotating thrust generators in 3-dimensional space, providing together thrust forces to actuate the passive architecture of the FPR. As a result, it is possible to consider the FPR as a cascade system composed of

- the passive architecture with relatively slow dynamics that is driven by the thrust forces of multirotors transmitted to the leg tips via spherical joints;
- multirotors that generate the corresponding thrust forces, with fast rotational dynamics decoupled from the dynamics of the passive architecture.

The translational dynamics of each multirotors is however dependent on that of the passive architecture as they are rigidly connected to the legs. In the state of the art for the general modelling of FPRs (referred to Section 1.2.1), the overall dynamics of the FPR can be derived by introducing the interaction force between the multirotor and the corresponding leg. Then by cancelling the interaction force that appears both in the translational dynamics of the multirotor and dynamics of the passive architecture, a generalised dynamic model of the FPR is obtained. This methodology can be applied to model the specific FPR studied in this thesis. However, another solution is adopted to simplify the modelling process, without the requirement of introducing the interaction force. Actually, the multirotors can be considered as additional mass points added at the extremities of the legs, by which the translational dynamics of the multirotors are systematically taken into account in the model of the passive architecture, while the rotational dynamics of each multirotor remains decoupled with this dynamic model. In addition, more intuitive and numerically efficient methods are used regarding the geometric, kinematic and dynamic modelling of this FPR, instead of applying the state-of-the-art methods based on the MDH convention and Euler-Lagrange formula.

2.2 System Description

A general scheme of the FPR is shown in Fig. 2.1, demonstrating a platform supported by several multirotor-based UAV via the rigid legs. Note that the subscript i is denoted as an index for the number of the leg and the multirotor, which remains the same meaning in the rest of the manuscript. In addition, several frames are presented, and respectively denoted by:

- $\mathfrak{F}_0$ an inertial reference frame fixed in a global localisation system with its origin denoted by O;
- $\mathfrak{F}_p$ a body-fixed frame of the platform located at its CoM position denoted by P;
- $\mathfrak{F}_{li}$ a frame attached to the centre of leg i's revolute joint A_i, in which the joint angle is expressed around the z axis;
- $\mathfrak{F}_{bi}$ a body-fixed frame attached to the CoM position B_i of the multirotor i.

As discussed in Section 1.2.1, the generalised coordinates of the FPR consist of the 6-dimensional pose of the platform and the additional mobility in the passive kinematic chains. For the orientation of the platform, the representation of unit quaternion is chosen, in order to avoid the potential singularity issue associated with the Euler angles. The mobility in the passive chains is characterised by the angle of the respective revolute joint. Therefore, the vector of the generalised coordinates can be defined by

$$\mathbf{q} = \begin{bmatrix} \mathbf{p}_p \\ \mathfrak{q}_p \\ \boldsymbol{\theta}_l \end{bmatrix} \in \mathbb{R}^{7+n} \tag{2.1}$$

where $\mathbf{p}_p \in \mathbb{R}^3$ is the 3-dimensional position of the platform, $\mathfrak{q}_p \in \mathbb{H}$ is the unit quaternion of the platform, which is defined in the quaternion space $\mathbb{H}$, and $\boldsymbol{\theta}_l = \begin{bmatrix} \theta_1 & \theta_2 & ... & \theta_n \end{bmatrix}^T \in \mathbb{R}^n$ is a vector concatenating the leg angles. n is the total number of legs (and multirotors) and is supposed to be equal to or greater than 3. If $n < 3$, it can be proven that the system is under-actuated such that the 6-dimensional pose of the platform is not fully controllable [Six, 2018a], which is undesired and out of the research interests in this thesis. The generalised velocity of the FPR is defined by a vector

$$\boldsymbol{\nu} = \begin{bmatrix} \mathbf{v}_p \\ {}^p\boldsymbol{\omega}_p \\ \dot{\boldsymbol{\theta}}_l \end{bmatrix} \in \mathbb{R}^{6+n} \tag{2.2}$$

including the linear velocity of the platform $\mathbf{v}_p \in \mathbb{R}^3$ expressed in $\mathfrak{F}_0$, the body-frame angular velocity of the platform ${}^p\boldsymbol{\omega}_p \in \mathbb{R}^3$ in $\mathfrak{F}_p$, and a vector of the leg angle rates $\dot{\boldsymbol{\theta}}_l = \begin{bmatrix} \dot{\theta}_1, & \dot{\theta}_2, & ..., & \dot{\theta}_n \end{bmatrix}^T \in \mathbb{R}^n$. The generalised acceleration of the FPR is given by a vector

$$\dot{\boldsymbol{\nu}} = \begin{bmatrix} \mathbf{a}_p \\ {}^p\dot{\boldsymbol{\omega}}_p \\ \ddot{\boldsymbol{\theta}}_l \end{bmatrix} \in \mathbb{R}^{6+n} \tag{2.3}$$

where $\mathbf{a}_p$ is the linear acceleration of the platform with respect to $\mathfrak{F}_0$ and expressed in $\mathfrak{F}_0$, ${}^p\dot{\boldsymbol{\omega}}_p$ is the angular acceleration expressed in $\mathfrak{F}_p$ and $\ddot{\boldsymbol{\theta}}_l$ concatenates the joint angle accelerations. Note that $\mathbf{a}_p$ can be calculated by the time derivative of the velocity as $\dot{\mathbf{v}}_p$ since the linear velocity of the platform is also expressed in $\mathfrak{F}_0$.

The coordinates of the actuators (i.e. each multirotor i) can be defined by the position $\mathbf{p}_i \in \mathbb{R}^3$ and orientation $\mathfrak{q}_i \in \mathbb{H}$ with respect to the reference frame $\mathfrak{F}_0$, while its linear and angular velocities are denoted by $\mathbf{v}_i \in \mathbb{R}^3$ expressed in $\mathfrak{F}_0$ and ${}^{bi}\boldsymbol{\omega}_i \in \mathbb{R}^3$ expressed in $\mathfrak{F}_{bi}$. Each multirotor produces a thrust force to actuate the FPR, represented by a vector $\mathbf{f}_i \in \mathbb{R}^3$ expressed in $\mathfrak{F}_0$ which can be written as ${}^{bi}\mathbf{f}_i = \begin{bmatrix} 0 & 0 & f_{t,i} \end{bmatrix}^T$ in its body-fixed frame

$\mathfrak{F}_{bi}$, with $f_{t,i}$ being the total thrust magnitude.

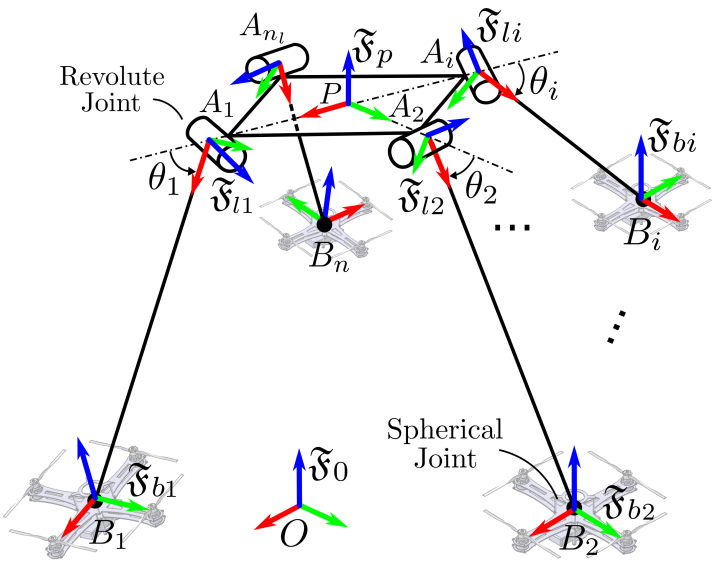


Figure 2.1 – A general scheme of the Flying Parallel Robot, with x, y, z axes of the frames defined respectively by red, green and blue arrows.

It should be remarked that any multirotor-based UAV can be applied as the actuators in the FPR design, among which the quadrotor (as shown in Fig. 2.1) has been chosen in the works done during this thesis for several reasons, including

- **Fully actuation**: Given a thrust force vector $\mathbf{f}_i$ and desired yaw as commands, quadrotor's inputs are fully determined, i.e. thrust magnitude and orientation in 3-dimensional space derived from $\mathbf{f}_i$ can deterministically define the four inputs of the quadrotor's motors.
- **Implementation simplicity**: Quadrotor's control has been intensively studied in the literature, and open-source autopilot software is available.

To not to lose the generality in the following sections, the term *multirotor* is still used to present the multirotor-based UAVs as the quadrotors used in the FPR.

It is additionally noted that the orientation of any given body j can also be defined by Euler angles using ZYX convention as $\boldsymbol{\eta}_j = \begin{bmatrix}\phi_j & \vartheta_j & \psi_j\end{bmatrix}^T \in \mathbb{R}^3$, with $\phi_j, \vartheta_j, \psi_j$ respectively representing the roll, pitch and yaw angle of a body (see [Diebel, 2006] and Appendix A.1). This representation is beneficial to simplify the trajectory generation and data analysis, which might however encounter singularity problems and is thus not preferable in the control. Remark that the unit quaternion is the only representation adopted in the control algorithms presented in this thesis, while the Euler angles are used when it is needed (such as data analysis). The conversion between Euler angles and the corresponding unit quaternion can be found in Appendix A.2.

2.3 Geometric Relations

The geometric relations in robotics involve generally the Direct Geometric Model (DGM) which determines the end-effector pose from the values of the actuated joints, and the Inverse Geometric Model (IGM) as an inverse problem of the DGM. For parallel robots, the DGM is a complex problem in general, as it cannot be easily expressed in analytical form and might have multiple solutions. In contrast, the IGM is simple to derive and remains unique for parallel robots.

The IGM of the FPR is specifically referred to as the model computing the positions of multirotors knowing the coordinates of the passive architecture, since the actuators are the multirotors and the robot pose is composed of not only the pose of the end-effector (i.e. the platform), but also the leg angles. This model can then be applied to compute the internal configuration of the FPR (i.e. leg angles), supposing that the positions of multirotors and the pose of the platform are known by exteroceptive sensors. Finally, the additional consideration on the location of spherical joints attached to the multirotor that is away from the multirotor's CoM will be presented.

2.3.1 Inverse Geometric Model (IGM)

The location of the multirotor i's CoM position can be determined via the kinematic chain linking the moving platform to each multirotor, as

$$\overrightarrow{OB_i} = \overrightarrow{OP} + \overrightarrow{PA_i} + \overrightarrow{A_iB_i} \tag{2.4}$$

This relation can be written by introducing the system's coordinates as

$$\mathbf{p}_i = \mathbf{p}_p + r \cdot \mathbf{r}_i + l \cdot \mathbf{l}_i \tag{2.5}$$

where $\mathbf{p}_i$ is the 3-dimensional position of the multirotor i, r is denoted as the radius of the platform (distance from its CoM position to the centre of the revolute joint), and l as the length of the leg, $\mathbf{r}_i$, $\mathbf{l}_i$ are two unit vectors defining the direction of $\overrightarrow{PA_i}$ and $\overrightarrow{A_iB_i}$. These two unit vectors are expressed in $\mathfrak{F}_0$, which is however constant if being expressed respectively in $\mathfrak{F}_p$ and $\mathfrak{F}_{li}$. Thus, the following transformations of frames can be found

$$\begin{cases} \mathbf{r}_i = \mathbf{R}_p {}^p\mathbf{r}_i \\ \mathbf{l}_i = \mathbf{R}_p {}^p\mathbf{R}_{li} {}^{li}\mathbf{l}_i \end{cases} \tag{2.6}$$

where $\mathbf{R}_p$ is the rotation matrix associated to the quaternion $\mathfrak{q}_p$ (referred to (A.60) in Appendix A.2), ${}^p\mathbf{R}_{li}$ the rotation matrix defining the transformation from $\mathfrak{F}_{li}$ to $\mathfrak{F}_p$, and

${}^{p}\mathbf{r}_i$, ${}^{li}\mathbf{l}_i$ are obviously two constant vectors that can be defined by

$$ {}^{p}\mathbf{r}_i = \begin{bmatrix} \cos\alpha_i \\ \sin\alpha_i \\ 0 \end{bmatrix}, \text{ with } \alpha_i = (i-1)\cdot\frac{(n-2)\pi}{n}; \quad {}^{li}\mathbf{l}_i = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{2.7} $$

α_i is a constant parameter defining the inclined angle between $\overrightarrow{PA_i}$ and the x axis of the platform frame in $\mathfrak{F}_p$, with α_1 initialized as zero (i.e. $\overrightarrow{PA_1}$ aligned with platform's x axis). It is recalled that the total number of the legs satisfies $n \geq 3$ to ensure the expression of α_i in (2.7) being valid. The transformation between $\mathfrak{F}_{li}$ and $\mathfrak{F}_p$ given by ${}^{p}\mathbf{R}_{li}$ can be defined with three sequential rotations similarly as the definition of Euler angles in ZXZ convention (see Appendix A.1), by firstly rotating an angle of α_i around z axis of the frame $\mathfrak{F}_p$, then following with a rotation of $-\pi/2$ around the x axis of the rotated frame, and rotating an angle of θ_i around the new z axis to finally obtain the frame $\mathfrak{F}_{li}$ (as shown in Fig. 2.2, with $\mathfrak{F}'$ and $\mathfrak{F}''$ being two intermediary frames).

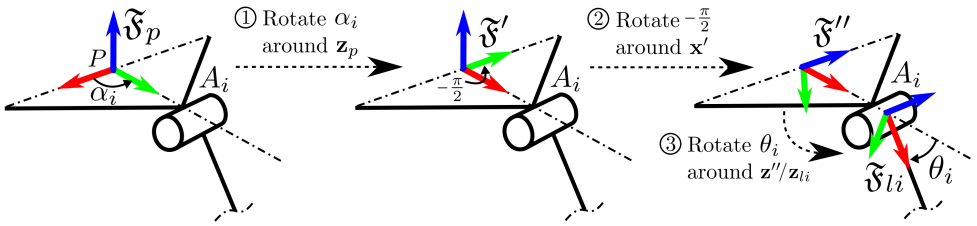


Figure 2.2 – Sequential rotations for defining the transformation between the platform frame and leg i's frame.

By multiplying the corresponding unit coordinate rotations given by (A.40), (A.41) and (A.42) in Appendix A.1, the rotation matrix ${}^{p}\mathbf{R}_{li}$ can be derived as

$$ \begin{aligned} {}^{p}\mathbf{R}_{li} &= \mathbf{R}_z(\alpha_i)\mathbf{R}_x(-\frac{\pi}{2})\mathbf{R}_z(\theta_i) \\ &= \begin{bmatrix} \cos\alpha_i & -\sin\alpha_i & 0 \\ \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos\alpha_i\cos\theta_i & -\cos\alpha_i\sin\theta_i & -\sin\alpha_i \\ \sin\alpha_i\cos\theta_i & -\sin\alpha_i\sin\theta_i & \cos\alpha_i \\ -\sin\theta_i & -\cos\theta_i & 0 \end{bmatrix} \end{aligned} \tag{2.8} $$

Finally, by grouping the equation of (2.5) for each multirotor, the complete IGM model can be obtained, which can be summarised by

$$ \mathbf{p} = \texttt{IGM}(\mathbf{q}) \tag{2.9} $$

with $\mathbf{p} = \begin{bmatrix}\mathbf{p}_1^T & \mathbf{p}_2^T & \cdots & \mathbf{p}_n^T\end{bmatrix}^T \in \mathbb{R}^{3n}$ a vector concatenating the positions of all the multirotors, and the input $\mathbf{q}$ as the vector of generalised coordinates of the robot.

2.3.2 Calculation of Internal Configuration via Geometric Relation

The geometric relation derived above can be used to calculate the internal configuration of the robot, i.e. the inclined angles θ_i of the revolute joints on the legs, knowing the position of the multirotor $\mathbf{p}_i$ and the pose of the platform $\mathbf{p}_p$ and $\mathfrak{q}_p$. By definition of the dot product between two vectors, the following relationship can be found

$$\overrightarrow{PA_i} \cdot \overrightarrow{A_iB_i} = \|\overrightarrow{PA_i}\| \cdot \|\overrightarrow{A_iB_i}\| \cdot \cos\theta_i \tag{2.10}$$

from which the revolute joint angle of the leg i can be derived as

$$\theta_i = \operatorname{acos}\left(\frac{\overrightarrow{PA_i} \cdot \overrightarrow{A_iB_i}}{\|\overrightarrow{PA_i}\| \cdot \|\overrightarrow{A_iB_i}\|}\right) \tag{2.11}$$

where $\overrightarrow{PA_i}$ and $\overrightarrow{A_iB_i}$ are vectors expressed in $\mathfrak{F}_0$. Using the geometric relationship derived in the IGM, the expression of these two vectors can be given knowing the position of the multirotor i and the platform $\mathbf{p}_i$, $\mathbf{p}_p$, and the rotation matrix $\mathbf{R}_p$ associated with the platform orientation $\mathfrak{q}_p$ as follows

$$\begin{cases} \overrightarrow{PA_i} = \mathbf{R}_p{}^p\mathbf{r}_{PA_i} \\ \overrightarrow{A_iB_i} = \overrightarrow{OB_i} - \overrightarrow{OA_i} = \overrightarrow{OB_i} - (\overrightarrow{OP} + \overrightarrow{PA_i}) \\ \qquad\quad = \mathbf{p}_i - (\mathbf{p}_p + \mathbf{R}_p{}^p\mathbf{r}_{PA_i}) \end{cases} \tag{2.12}$$

where ${}^p\mathbf{r}_{PA_i}$ is a constant vector defining the revolution joint i's position in $\mathfrak{F}_p$ and apparently ${}^p\mathbf{r}_{PA_i} = r \cdot {}^p\mathbf{r}_i$, with ${}^p\mathbf{r}_i$ a constant unit vector defined in (2.7). Note that the calculation of the leg angles using (2.11) can also be done if all the vectors are expressed in a local frame, such as $\mathfrak{F}_p$.

2.3.3 Location of Spherical Joint attached on the Multirotor

The spherical joint attaching the multirotor to the leg's tip is not necessarily located at the multirotor's CoM position. In the design of the FPR within the range of this thesis, the location of the spherical joint lies in the plane defined by the x and y axes of the multirotor's body-fixed frame $\mathfrak{F}_{bi}$. This location can thus be characterised by two constant parameters, the relative distance d and the relative angle β with respect to the multirotor

i's frame as depicted in Fig. 2.3. A set of more generalised parameters can be defined such as the relative position of the joint's centre in the multirotor frame. However, only d and β are used to parameterise the location of the spherical joint to keep the coherence and simplicity of the actual mechanical design.

From Fig. 2.3, it can be found that $d = \|\overrightarrow{B_iS_i}\|$ and β is the inclined angle between $\overrightarrow{B_iS_i}$ and x axis of the multirotor's body-fixed frame $\mathfrak{F}_{bi}$. Note that these two parameters d and β can be different for each multirotor, in particular the relative angle β where the spherical joint is connected. However, their values are considered as equal for all the multirotors in this FPR which is ensured by the assembly prerequisite. In addition, while a common assumption is that the centre of the spherical joint is located at the CoM position of the multirotor (i.e. $d = 0$) as discussed in Section 2.1, the fact that $d \neq 0$ makes the geometric relationship of (2.4) deviated by

$$\overrightarrow{OB_i} = \overrightarrow{OP} + \overrightarrow{PA_i} + \overrightarrow{A_iS_i} + \overrightarrow{S_iB_i} \tag{2.13}$$

which gives the following expression introducing the robot variables

$$\mathbf{p}_i = \mathbf{p}_p + r \cdot \mathbf{r}_i + l \cdot \mathbf{l}_i + d \cdot \mathbf{d}_i \tag{2.14}$$

with $\mathbf{d}_i$ representing the unit vector of $\overrightarrow{B_iS_i}$ expressed in $\mathfrak{F}_0$. However, the conditions that $r \gg d$ and $l \gg d$ allow this additional term in the geometric model to be neglected, as the radius of the platform r and the leg's length l are often required to be large enough such that no potential collision or interference between the multirotors would occur. In the following sections, the condition that $d = 0$ is *a priori* assumed, while additional discussions on the case where $d \neq 0$ will be made in Section 2.5.5 dealing with rotational dynamics of the multirotor.

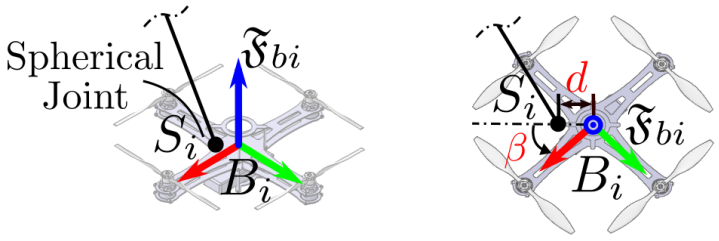


Figure 2.3 – Geometric constants defining the location S_i of the attached spherical joint on the multirotor.

The geometric relation given by (2.5) can be further derived to develop the kinematic model, which is detailed in the next section.

2.4 Kinematics

Similarly to geometric models, the Forward Kinematic Model (FKM) and the Inverse Kinematic Model (IKM) of the robot are dual models to construct the relationship between the velocity of the actuated variables and the velocity of the robot generalised coordinates. Similarly to the geometric model, the IKM of the FPR is easily computed which computes the linear velocity of each multirotor based on the knowledge of generalised velocity vector. The FKM is thus solved by the inverse of IKM. The second-order kinematics is also detailed as follows.

2.4.1 First-Order Kinematics

For the FPR, the first-order IKM can be developed by differentiating the geometric relationship in (2.5), which gives

$$\mathbf{v}_i = \mathbf{v}_p + r \cdot \dot{\mathbf{r}}_i + l \cdot \dot{\mathbf{l}}_i \tag{2.15}$$

with $\mathbf{v}_i \in \mathbb{R}^3$ being the linear velocity of multirotor i expressed in $\mathfrak{F}_0$. Note that $\mathbf{v}_p$ is the linear velocity of the platform. The derivatives $\dot{\mathbf{r}}_i$ and $\dot{\mathbf{l}}_i$ require however the differentiation of the rotation matrices, which hold

$$\begin{aligned} \dot{\mathbf{r}}_i &= \dot{\mathbf{R}}_p {}^p\mathbf{r}_i + \mathbf{R}_p {}^p\dot{\mathbf{r}}_i = \mathbf{R}_p[{}^p\boldsymbol{\omega}_p]_\times {}^p\mathbf{r}_i \\ &= -\mathbf{R}_p[{}^p\mathbf{r}_i]_\times {}^p\boldsymbol{\omega}_p \end{aligned} \tag{2.16}$$

$$\begin{aligned} \dot{\mathbf{l}}_i &= \dot{\mathbf{R}}_p {}^p\mathbf{R}_{li} {}^{li}\mathbf{l}_i + \mathbf{R}_p {}^p\dot{\mathbf{R}}_{li} {}^{li}\mathbf{l}_i + \mathbf{R}_p {}^p\mathbf{R}_{li} {}^{li}\dot{\mathbf{l}}_i \\ &= \mathbf{R}_p[{}^p\boldsymbol{\omega}_p]_\times {}^p\mathbf{R}_{li} {}^{li}\mathbf{l}_i + \mathbf{R}_p {}^p\mathbf{R}_{li}[{}^{li}\boldsymbol{\omega}_{li/p}]_\times {}^{li}\mathbf{l}_i \\ &= -\mathbf{R}_p[{}^p\mathbf{l}_i]_\times {}^p\boldsymbol{\omega}_p - \mathbf{R}_{li}[{}^{li}\mathbf{l}_i]_\times {}^{li}\boldsymbol{\omega}_{li/p} \end{aligned} \tag{2.17}$$

where ${}^p\boldsymbol{\omega}_p$ is the body-frame angular velocity of the platform, ${}^{li}\boldsymbol{\omega}_{li/p} \in \mathbb{R}^3$ is the angular velocity of the frame $\mathfrak{F}_{li}$ with respect to $\mathfrak{F}_p$ and expressed in $\mathfrak{F}_{li}$, which is by definition given as ${}^{li}\boldsymbol{\omega}_{li/p} = \dot{\theta}_i \boldsymbol{\delta}_i$, with $\dot{\theta}_i$ the leg angle rate and $\boldsymbol{\delta}_i = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ representing the revolute-joint axis of the leg i. Note that $[.]_\times$ represents the skew-symmetric matrix associated with an arbitrary 3-dimensional vector for the cross product operation (referred to (A.50) in Appendix A.1), ${}^p\mathbf{l}_i = {}^p\mathbf{R}_{li} {}^{li}\mathbf{l}_i$ is the vector $\mathbf{l}_i$ expressed in $\mathfrak{F}_p$, and $\mathbf{R}_{li} = \mathbf{R}_p {}^p\mathbf{R}_{li}$ represents the rotation matrix from $\mathfrak{F}_{li}$ to $\mathfrak{F}_0$.

The kinematics relationship of (2.15) can be computed by introducing (2.16) and (2.17) into (2.15) and taking ${}^{li}\boldsymbol{\omega}_{li/p} = \dot{\theta}_i \boldsymbol{\delta}_i$ into account, as

$$\mathbf{v}_i = \mathbf{v}_p - r \cdot \mathbf{R}_p[{}^p\mathbf{r}_i]_\times {}^p\boldsymbol{\omega}_p - l \cdot (\mathbf{R}_p[{}^p\mathbf{l}_i]_\times {}^p\boldsymbol{\omega}_p + \dot{\theta}_i \cdot \mathbf{R}_{li}[{}^{li}\mathbf{l}_i]_\times \boldsymbol{\delta}_i) \tag{2.18}$$

where the result of the cross product $[{}^{li}\mathbf{l}_i]_\times \boldsymbol{\delta}_i = {}^{li}\mathbf{l}_i \times \boldsymbol{\delta}_i = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^T$ can be further applied to develop (2.18), which gives

$$\mathbf{v}_i = \mathbf{v}_p + (r \cdot \mathbf{R}_p [{}^p\mathbf{r}_i]_\times^T + l \cdot \mathbf{R}_p [{}^p\mathbf{l}_i]_\times^T) {}^p\boldsymbol{\omega}_p + \dot{\theta}_i l \cdot \mathbf{y}_{li} \tag{2.19}$$

with $-[.]_\times = [.]_\times^T$ and $\mathbf{y}_{li}$ representing the y-axis vector (i.e. second column) of the rotation matrix $\mathbf{R}_{li}$. The physical interpretation of this relationship can be given as follows:

- the linear velocity of each multirotor $\mathbf{v}_i$ is directly linked to the linear velocity of the platform $\mathbf{v}_p$, both expressed in $\mathfrak{F}_0$.
- the angular velocity of the platform causes additionally translational movements at the leg extremities. Note that the term $(r \cdot \mathbf{R}_p [{}^p\mathbf{r}_i]_\times^T + l \cdot \mathbf{R}_p [{}^p\mathbf{l}_i]_\times^T) {}^p\boldsymbol{\omega}_p$ can be interpreted as ${}^p\boldsymbol{\omega}_p \times \mathbf{r}_{PB_i}$, with $\mathbf{r}_{PB_i}$ representing the vector of $\overrightarrow{PB_i}$.
- the rotational rate of the leg angle $\dot{\theta}_i$ generates translational movement of the multirotor only along y axis of the leg i's frame, according to the only possible DoF of the multirotor in leg's frame constrained by the revolute joint.

By concatenating the kinematics relationship in (2.19) for each multirotor i, the first-order IKM of the FPR can be finally derived by

$$\mathbf{v} = \begin{bmatrix} \mathbf{1}_{3\times3} & \mathbf{J}_{\boldsymbol{\omega}_1} & l \cdot \mathbf{y}_{l1} & \mathbf{0}_3 & \dots & \mathbf{0}_3 \\ \mathbf{1}_{3\times3} & \mathbf{J}_{\boldsymbol{\omega}_2} & \mathbf{0}_3 & l \cdot \mathbf{y}_{l2} & \dots & \mathbf{0}_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{1}_{3\times3} & \mathbf{J}_{\boldsymbol{\omega}_n} & \mathbf{0}_3 & \mathbf{0}_3 & \dots & l \cdot \mathbf{y}_{ln} \end{bmatrix} \begin{bmatrix} \mathbf{v}_p \\ {}^p\boldsymbol{\omega}_p \\ \dot{\theta}_1 \\ \dot{\theta}_2 \\ \vdots \\ \dot{\theta}_n \end{bmatrix} = \mathbf{J}\boldsymbol{\nu} \tag{2.20}$$

where $\mathbf{J}_{\boldsymbol{\omega}_i} = \left(r \cdot \mathbf{R}_p [{}^p\mathbf{r}_i]_\times^T + l \cdot \mathbf{R}_p [{}^p\mathbf{l}_i]_\times^T\right) \in \mathbb{R}^{3\times3}$, $\mathbf{1}_{3\times3}$ is the (3×3) identity matrix and $\mathbf{0}_3$ is a (3×1) vector of zeros. $\mathbf{J} \in \mathbb{R}^{(3n)\times(6+n)}$ is the Jacobian matrix linking the generalised velocity of the robot $\boldsymbol{\nu} \in \mathbb{R}^{6+n}$ to the vector of the multirotor linear velocities $\mathbf{v} = \begin{bmatrix} \mathbf{v}_1^T & \mathbf{v}_2^T & \dots & \mathbf{v}_n^T \end{bmatrix}^T \in \mathbb{R}^{3n}$.

The FKM of the FPR, also called Forward Kinematics, is dedicated to calculate the velocity coordinates from the linear velocities of the multirotors. For the FPR, this model can be derived by the inverse of (2.20) as

$$\boldsymbol{\nu} = \mathbf{J}^\dagger \mathbf{v} \tag{2.21}$$

with $\mathbf{J}^\dagger$ being the pseudo-inverse of the Jacobian matrix when the dimension of the actuated states (the linear velocities of multirotors, $3n$) is superior than the dimension of

robot coordinates (generalised velocity, $6+n$), i.e. the robot is over-actuated. Note that for the case where $n=3$, $\mathbf{J}^{\dagger}=\mathbf{J}^{-1}$ since $\mathbf{J}$ is square ($3n=6+n$), and the system is fully actuated. However, the inverse of Jacobian matrix necessitates $\mathbf{J}$ being invertible, i.e. $\det(\mathbf{J})\neq 0$. It has been proven that this assumption holds true if all the leg angles are within the range of $[0,\pi/2]$ rad [Six, 2018b].

2.4.2 Second-Order Kinematics

The second-order kinematic model depicts the relationship between the acceleration of the actuated variables and that of the robot configuration, which can be derived directly by the differentiation of the first-order kinematics in (2.20) as

$$\dot{\mathbf{v}}=\mathbf{J}\dot{\boldsymbol{\nu}}+\dot{\mathbf{J}}\boldsymbol{\nu} \tag{2.22}$$

where $\dot{\mathbf{v}}=\begin{bmatrix}\dot{\mathbf{v}}_1 & \dot{\mathbf{v}}_2 & \dots & \dot{\mathbf{v}}_n\end{bmatrix}^T\in\mathbb{R}^{3n}$ is a vector concatenating the linear accelerations of multirotors expressed in $\mathfrak{F}_0$. $\mathbf{J}$ is the Jacobian matrix given in (2.20) and $\dot{\mathbf{J}}\in\mathbb{R}^{(3n)\times(6+n)}$ is the derivative of the Jacobian matrix in form of

$$\dot{\mathbf{J}}=\begin{bmatrix}\mathbf{0}_{3\times3} & \dot{\mathbf{J}}_{\boldsymbol{\omega}_1} & l\cdot\dot{\mathbf{y}}_{l1} & \mathbf{0}_3 & \dots & \mathbf{0}_3\\ \mathbf{0}_{3\times3} & \dot{\mathbf{J}}_{\boldsymbol{\omega}_2} & \mathbf{0}_3 & l\cdot\dot{\mathbf{y}}_{l2} & \dots & \mathbf{0}_3\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ \mathbf{0}_{3\times3} & \dot{\mathbf{J}}_{\boldsymbol{\omega}_n} & \mathbf{0}_3 & \mathbf{0}_3 & \dots & l\cdot\dot{\mathbf{y}}_{ln}\end{bmatrix} \tag{2.23}$$

where $\dot{\mathbf{J}}_{\boldsymbol{\omega}_i}=\left(r\cdot\mathbf{R}_p[{}^p\boldsymbol{\omega}_p]_\times[{}^p\mathbf{r}_i]_\times^T+l\cdot\mathbf{R}_p[{}^p\boldsymbol{\omega}_p]_\times[{}^p\mathbf{l}_i]_\times^T+l\cdot\mathbf{R}_p[{}^p\mathbf{R}_{li}[\dot{\theta}_i\boldsymbol{\delta}_i]_\times{}^{li}\mathbf{l}_i]_\times^T\right)\in\mathbb{R}^{3\times3}$, $\dot{\mathbf{y}}_{li}$ is the second column of the matrix given by $\dot{\mathbf{R}}_{li}=\left(\mathbf{R}_p[{}^p\boldsymbol{\omega}_p]_\times{}^p\mathbf{R}_{li}+\mathbf{R}_p{}^p\mathbf{R}_{li}[\dot{\theta}_i\boldsymbol{\delta}_i]_\times\right)\in\mathbb{R}^{3\times3}$.

Similarly as in FKM, the acceleration of the FPR can be computed knowing the accelerations of the multirotors, i.e.

$$\dot{\boldsymbol{\nu}}=\mathbf{J}^{\dagger}(\dot{\mathbf{v}}-\dot{\mathbf{J}}\boldsymbol{\nu}) \tag{2.24}$$

$\mathbf{J}^{\dagger}=\mathbf{J}^{-1}$ when $n=3$, assuming that $\det(\mathbf{J})\neq 0$.

2.5 Dynamics

The dynamics model depicts the relationship between the robot motion and the applied actuation wrench by the equation of motion. It is necessary to derive an accurate dynamic model for the purpose of simulating the robot's behaviour using the Direct Dynamic Model (DDM) or constructing the control law with the Inverse Dynamic Model (IDM).

As discussed in Section 2.1, the dynamics of the FPR is decoupled by two-level systems: dynamics of the passive architecture and rotational dynamics of multirotors, which is ensured by the mechanical property of using spherical joints attached to the CoM of the multirotors. The effects due to the masses of multirotors are additionally considered within the development of the IDM for the passive architecture, which eliminates the need of introducing interaction force between the multirotor and the leg as in [Six, 2018a].

The IDM of the passive architecture can generally be written in matrix form as

$$\mathbf{M}(\mathbf{q})\dot{\boldsymbol{\nu}} + \mathbf{c}(\mathbf{q}, \boldsymbol{\nu}) = \boldsymbol{\tau} \tag{2.25}$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{(6+n)\times(6+n)}$ is the generalised inertia matrix, $\mathbf{c}(\mathbf{q}, \boldsymbol{\nu}) \in \mathbb{R}^{6+n}$ is a vector including Coriolis, centrifugal and gravitational effects, and $\boldsymbol{\tau} \in \mathbb{R}^{6+n}$ is the actuation wrench applied to the robot. This model can also be expressed as

$$\mathbf{M}(\mathbf{q})\dot{\boldsymbol{\nu}} + \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} \tag{2.26}$$

with $\mathbf{C}(\mathbf{q}, \boldsymbol{\nu}) \in \mathbb{R}^{(6+n)\times(6+n)}$ the Coriolis matrix factorizing the Coriolis and centrifugal terms, and $\mathbf{g}(\mathbf{q}) \in \mathbb{R}^{6+n}$ being the generalised gravity wrench, such that $\mathbf{c}(\mathbf{q}, \boldsymbol{\nu}) = \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\mathbf{q})$. Note that $\boldsymbol{\nu}$ and $\dot{\boldsymbol{\nu}}$ represent the generalised velocity and acceleration of the robot as defined in (2.2) and (2.3). For more general cases in classical robotics, they are often represented by the derivatives of the robot coordinates $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$.

The dynamic modelling consists in computing the terms in (2.25) or (2.26) either in analytical forms or using numerical algorithms. It is straightforward to get expressions of the dynamic model using Euler-Lagrange formulation as discussed in Section 1.2.1. However, the development of analytical expressions is often tedious and computationally heavy to be implemented in real time. Therefore, numerical methods are more commonly adopted, such as Khalil's method based on recursive Newton-Euler algorithm [Khalil, 2002b] and Featherstone's method using Spatial Vector notation [Featherstone, 2008]. Khalil's method is usually applied to obtain the dynamic model in form of (2.25), which is then used to construct a dynamic model-based control law. The Spatial Vector notation uses 6-dimensional vectors and matrices, and equations are thus more compact. A numerical algorithm for computing the Coriolis matrix in (2.26) is furthermore applied, which might be required in specific control and estimation methods. They are therefore indispensable for developing algorithms presented in the following chapters of this thesis.

In the following subsections, the computation of the actuation wrench for the passive architecture is firstly given. Then, two numerical methods for the dynamic modelling

respectively based on the recursive Newton-Euler algorithm and Spatial Vector notation will be detailed. In the fourth subsection, the gravity wrench can be computed directly by the static model, which complements the dynamic model written in form of (2.26). Finally, the decoupled rotational dynamics of multirotors will be presented.

2.5.1 Computation of Actuation Wrench

As discussed in Section 2.1, the FPR is a cascaded system in which the dynamics of the passive architecture is decoupled with the rotational dynamics of multirotors. The system is exclusively actuated by the thrust forces produced by the multirotors. The 3-dimensional thrust forces produced by the multirotors can be concatenated by a vector $\mathbf{f} = \begin{bmatrix}\mathbf{f}_1^T & \mathbf{f}_2^T & \cdots & \mathbf{f}_n^T\end{bmatrix}^T \in \mathbb{R}^{3n}$, with $\mathbf{f}_i \in \mathbb{R}^3$ the thrust force vector of the multirotor i expressed in $\mathfrak{F}_0$. The actuation wrench of the FPR given by $\boldsymbol{\tau}$ in the dynamic models can be expressed as

$$\boldsymbol{\tau} = \begin{bmatrix}\mathbf{f}_p \\ {}^p\mathbf{m}_p \\ \mathbf{m}_l\end{bmatrix} \in \mathbb{R}^{6+n} \tag{2.27}$$

with $\mathbf{f}_p \in \mathbb{R}^3$, ${}^p\mathbf{m}_p \in \mathbb{R}^3$ being the 3-dimensional force and moment actuated on the platform expressed respectively in $\mathfrak{F}_0$ and $\mathfrak{F}_p$, and $\mathbf{m}_l = \begin{bmatrix}\mathrm{m}_{l1} & \mathrm{m}_{l2} & \ldots & \mathrm{m}_{ln}\end{bmatrix}^T \in \mathbb{R}^n$ concatenating the moments actuated about the revolute-joint axis of the legs. The actuation wrench can be computed from $\mathbf{f}$ using the static relation such that

$$\boldsymbol{\tau} = \mathbf{J}^T\mathbf{f} \tag{2.28}$$

where $\mathbf{J}$ is the kinematic Jacobian matrix defined in (2.20) relating the generalised velocity of the FPR to the velocities of actuated multirotors. This relationship can be proven by the principle of virtual power as follows (referred to Section 1.2.1 and [Briot, 2015]).

Proof. Consider a virtual velocity of the robot $\boldsymbol{\nu}^*$ which is driven by the virtual velocities of the multirotors $\mathbf{v}^*$. In absence of any other effects, the thrust forces of the multirotors $\mathbf{f}$ lead to the robot's actuation wrench $\boldsymbol{\tau}$. Thus, power conservation states that

$$\boldsymbol{\nu}^{*T}\boldsymbol{\tau} = \mathbf{v}^{*T}\mathbf{f} \tag{2.29}$$

Replacing $\mathbf{v}^*$ with kinematic relation given by (2.20) gets

$$\boldsymbol{\nu}^{*T}\boldsymbol{\tau} = (\mathbf{J}\boldsymbol{\nu}^*)^T\mathbf{f} \tag{2.30}$$

which can be further simplified to obtain the relation (2.28) with the virtual velocity $\boldsymbol{\nu}^*$ being arbitrary.

2.5.2 Dynamic Modelling via Recursive Newton-Euler Algorithm

The recursive Newton-Euler algorithm uses the recursive computation of Newton-Euler equations allowing to decrease the computational complexity of the dynamic model [Khalil, 2010; Briot, 2015]. The algorithm starts with the formulation of the Newton-Euler equations for each rigid body in the robot, giving the total forces $\Sigma\mathbf{f}_j$ and moments $\Sigma\mathbf{m}_j$ on the body j at the origin of its frame $\mathfrak{F}_j$, i.e.

$$\begin{aligned}\Sigma\mathbf{f}_j &= m_j\mathbf{a}_j + \dot{\boldsymbol{\omega}}_j \times \mathbf{ms}_j + \boldsymbol{\omega}_j \times (\boldsymbol{\omega}_j \times \mathbf{ms}_j)\\ \Sigma\mathbf{m}_j &= \mathbf{I}_{O_j}\dot{\boldsymbol{\omega}}_j + \mathbf{ms}_j \times \mathbf{a}_j + \boldsymbol{\omega}_j \times (\mathbf{I}_{O_j}\boldsymbol{\omega}_j)\end{aligned} \tag{2.31}$$

where $\mathbf{a}_j$ is the linear acceleration of the origin of the frame $\mathfrak{F}_j$, $\boldsymbol{\omega}_j$ and $\dot{\boldsymbol{\omega}}_j$ are the angular velocity and acceleration of the link, $\mathbf{ms}_j \in \mathbb{R}^3$ is the vector of the first moment of inertia which is calculated by

$$\mathbf{ms}_j = m_j\mathbf{s}_j \tag{2.32}$$

as the mass of the body m_j multiplied by the CoM position $\mathbf{s}_j$ of the link with respect to its own frame $\mathfrak{F}_j$. $\mathbf{I}_{O_j} \in \mathbb{R}^{3\times3}$ is the rotational inertia matrix expressed at the origin of $\mathfrak{F}_j$.

Then, the dynamic model is obtained by applying two recursive algorithms sequentially, the forward recursive computation for the velocity and acceleration of the links, i.e. right side of the equations (2.31) and the backward recursive computation for the total forces and moments, i.e. the left side of the equations (2.31). The recursive algorithms for the modelling of the FPR are detailed as follows, which involve the computation of the inertial matrix and the Coriolis vector for the dynamics of the passive architecture considering additional masses of the multirotors attached at the leg tips. Note that the moving platform is considered as the (floating) base of the robot, required to define the forward and backward directions as in the modelling of classical robotic manipulators.

Forward Recursive Computation

The forward computation has the objective to compute the velocities and accelerations of the legs from those of the platform. It is therefore initialized by the knowledge of the platform twist (i.e. $\mathbf{v}_p$ and ${}^p\boldsymbol{\omega}_p$) and accelerations (i.e. $\mathbf{a}_p$ and ${}^p\dot{\boldsymbol{\omega}}_p$) with respect to $\mathfrak{F}_0$, where $\mathbf{v}_p$, $\mathbf{a}_p$ are further converted to be expressed in $\mathfrak{F}_p$ by ${}^p\mathbf{v}_p = \mathbf{R}_p^T\mathbf{v}_p$ and ${}^p\mathbf{a}_p = \mathbf{R}_p^T\mathbf{a}_p$.

Then, the velocity of the leg i's frame $\mathfrak{F}_{li}$ with respect to $\mathfrak{F}_0$ and expressed in its own frame can be computed from that of the platform (in the forward direction)

$$\begin{aligned}{}^{li}\mathbf{v}_{li} &= {}^p\mathbf{R}_{li}^{Tp}\mathbf{v}_{li} = {}^p\mathbf{R}_{li}^T({}^p\mathbf{v}_p + {}^p\boldsymbol{\omega}_p \times {}^p\mathbf{r}_{PA_i})\\ {}^{li}\boldsymbol{\omega}_{li} &= {}^p\mathbf{R}_{li}^{Tp}\boldsymbol{\omega}_p + {}^{li}\boldsymbol{\omega}_{li/p} = {}^p\mathbf{R}_{li}^{Tp}\boldsymbol{\omega}_p + \dot{\theta}_i\boldsymbol{\delta}_i\end{aligned} \tag{2.33}$$

Remark that ${}^{p}\mathbf{R}_{li}$ is the rotation matrix from the frame $\mathfrak{F}_{li}$ to the frame $\mathfrak{F}_{p}$, ${}^{p}\mathbf{r}_{PA_i}$ is the vector $\overrightarrow{PA_i}$ expressed in $\mathfrak{F}_{p}$ (i.e. ${}^{p}\mathbf{r}_{PA_i} = r \cdot {}^{p}\mathbf{r}_{i}$), ${}^{li}\boldsymbol{\omega}_{li/p}$ is the relative angular velocity of the frame $\mathfrak{F}_{li}$ with respect to $\mathfrak{F}_{p}$ and expressed in $\mathfrak{F}_{li}$, which is given by the leg angle rate $\dot{\theta}_i$ multiplied by its rotational axis $\boldsymbol{\delta}_i$ of the revolute joint. Regarding the acceleration of each leg's frame $\mathfrak{F}_{li}$, the following relations are given

$$\begin{aligned} {}^{li}\mathbf{a}_{li} &= {}^{p}\mathbf{R}_{li}^{Tp}\mathbf{a}_{li} = {}^{p}\mathbf{R}_{li}^{T}\Big({}^{p}\mathbf{a}_{p} + {}^{p}\dot{\boldsymbol{\omega}}_{p} \times {}^{p}\mathbf{r}_{PA_i} + {}^{p}\boldsymbol{\omega}_{p} \times ({}^{p}\boldsymbol{\omega}_{p} \times {}^{p}\mathbf{r}_{PA_i})\Big) \\ {}^{li}\boldsymbol{\omega}_{li} &= {}^{p}\mathbf{R}_{li}^{Tp}\boldsymbol{\omega}_{p} + {}^{li}\boldsymbol{\omega}_{li/p} = {}^{p}\mathbf{R}_{li}^{Tp}\boldsymbol{\omega}_{p} + \ddot{\theta}_i\boldsymbol{\delta}_i + \dot{\theta}_i({}^{li}\boldsymbol{\omega}_{p} \times \boldsymbol{\delta}_i) \end{aligned} \tag{2.34}$$

where $\ddot{\theta}_i$ is the angular acceleration of the revolute-joint angle, ${}^{li}\boldsymbol{\omega}_{p}$ is the angular velocity of the platform expressed in $\mathfrak{F}_{li}$ (i.e. ${}^{li}\boldsymbol{\omega}_{p} = {}^{p}\mathbf{R}_{li}^{Tp}\boldsymbol{\omega}_{p}$).

Backward Recursive Computation

The backward computation is then developed to compute the total forces and moments acting on each body of the robot (i.e. the left side of (2.31)), which is performed in the backward direction (from the legs to the platform).

For each leg i, the total forces and moments expressed in $\mathfrak{F}_{li}$ are given by

$$\begin{aligned} {}^{li}\Sigma\mathbf{f}_{i} &= {}^{li}\mathbf{f}_{p,i} + m_i\,{}^{li}\mathfrak{g} \\ {}^{li}\Sigma\mathbf{m}_{i} &= {}^{li}\mathbf{m}_{p,i} + {}^{li}\mathbf{ms}_{i} \times {}^{li}\mathfrak{g} \end{aligned} \tag{2.35}$$

where ${}^{li}\mathbf{f}_{i}$ and ${}^{li}\mathbf{m}_{i}$ are the reaction force and moment exerted on the leg i by the platform expressed in $\mathfrak{F}_{li}$, ${}^{li}\mathfrak{g}$ is a gravity vector expressed in $\mathfrak{F}_{li}$, i.e. ${}^{li}\mathfrak{g} = \mathbf{R}_{li}^{T}\mathfrak{g}$, with $\mathfrak{g} = \begin{bmatrix} 0 & 0 & -g \end{bmatrix}^{T}$ and $g = 9.81 m/s^2$. m_i and ${}^{li}\mathbf{ms}_{i}$ are constant mass parameters taking into account the leg's mass and the attached multirotor's mass, which are calculated by

$$\begin{cases} m_i = m_{l,i} + m_{b,i} \\ {}^{li}\mathbf{ms}_{i} = {}^{li}\mathbf{ms}_{l,i} + {}^{li}\mathbf{ms}_{b,i} \end{cases} \tag{2.36}$$

where $m_{l,i}$ and $m_{b,i}$ are the masses of the leg i and the multirotor i, ${}^{li}\mathbf{ms}_{l,i}$ is the vector of the leg i's first moment of inertia, which is computed by the multiplication between the leg i's mass and the CoM position in its own frame $\mathfrak{F}_{li}$, ${}^{li}\mathbf{ms}_{b,i}$ is the vector of the first moment of inertia corresponding to the multirotor's mass expressed in $\mathfrak{F}_{li}$, which is thus

$${}^{li}\mathbf{ms}_{b,i} = m_{b,i} \cdot {}^{li}\mathbf{r}_{A_iB_i} \tag{2.37}$$

with ${}^{li}\mathbf{r}_{A_iB_i} = l \cdot {}^{li}\mathbf{l}_{i}$ is the vector $\overrightarrow{A_iB_i}$ expressed in $\mathfrak{F}_{li}$.

The total forces and moments on the platform expressed in $\mathfrak{F}_p$ can then be computed by considering the forces and moments transmitted from the legs

$$\begin{aligned} {}^{p}\Sigma\mathbf{f}_p &= {}^{p}\mathbf{f}_p - \sum_{i=1}^{n} {}^{p}\mathbf{f}_{p,i} + m_p {}^{p}\mathfrak{g} \\ {}^{p}\Sigma\mathbf{m}_p &= {}^{p}\mathbf{m}_p - \sum_{i=1}^{n} ({}^{p}\mathbf{m}_{p,i} + {}^{p}\mathbf{r}_{PA_i} \times {}^{p}\mathbf{f}_{p,i}) + {}^{p}\mathbf{ms}_p \times {}^{p}\mathfrak{g} \end{aligned} \tag{2.38}$$

where ${}^{p}\mathbf{f}_p$ and ${}^{p}\mathbf{m}_p$ are the reaction force and moment exerted on the platform by the virtual base (considered as the actuation wrench in (2.27)) expressed in $\mathfrak{F}_p$, ${}^{p}\mathbf{f}_{p,i}$ and ${}^{p}\mathbf{m}_{p,i}$ are the reaction force and moment exerted on the platform by the individual leg expressed in $\mathfrak{F}_p$, ${}^{p}\mathfrak{g} = \mathbf{R}_p^T\mathfrak{g}$ is the gravity vector expressed in $\mathfrak{F}_p$, m_p and ${}^{p}\mathbf{ms}_p$ are constant mass parameters of the platform.

In practical computation, the gravity terms from (2.35) and (2.38) can be cancelled by taking into account their effects in the linear acceleration of the platform such that

$${}^{p}\mathbf{a}_p = \mathbf{R}_p^T(\mathbf{a}_p - \mathfrak{g}) \tag{2.39}$$

Thus, combining equations (2.33), (2.34) and (2.35), the expression of Newton-Euler equations (2.31) for the leg i can be written, which is then used to obtain the following relations

$$\begin{aligned} {}^{li}\mathbf{f}_{p,i} &= m_i {}^{li}\mathbf{a}_{li} + {}^{li}\dot{\boldsymbol{\omega}}_{li} \times {}^{li}\mathbf{ms}_i + {}^{li}\boldsymbol{\omega}_{li} \times ({}^{li}\boldsymbol{\omega}_{li} \times {}^{li}\mathbf{ms}_i) \\ {}^{li}\mathbf{m}_{p,i} &= {}^{li}\mathbf{I}_{O_i} {}^{li}\dot{\boldsymbol{\omega}}_{li} + {}^{li}\mathbf{ms}_i \times {}^{li}\mathbf{a}_{li} + {}^{li}\boldsymbol{\omega}_{li} \times ({}^{li}\mathbf{I}_{O_i} {}^{li}\boldsymbol{\omega}_{li}) \end{aligned} \tag{2.40}$$

where the composed mass m_i and first moment of inertia ${}^{li}\mathbf{ms}_i$ corresponding to the leg i with the attached multirotor i are given by (2.36), and the composed moment of inertia ${}^{li}\mathbf{I}_{O_i}$ is calculated by

$${}^{li}\mathbf{I}_{O_i} = {}^{li}\mathbf{I}_{O_{l,i}} + {}^{li}\mathbf{I}_{O_{b,i}} \tag{2.41}$$

with ${}^{li}\mathbf{I}_{O_{l,i}}$ being the leg's moment of inertia expressed at the origin of its frame $\mathfrak{F}_{li}$, and ${}^{li}\mathbf{I}_{O_{b,i}}$ the moment of inertia due to the mass of the multirotor attached at the extremity of the leg expressed in $\mathfrak{F}_{li}$, i.e.

$${}^{li}\mathbf{I}_{O_{b,i}} = m_{b,i} \cdot \begin{bmatrix} 0 & 0 & 0 \\ 0 & l^2 & 0 \\ 0 & 0 & l^2 \end{bmatrix} \tag{2.42}$$

l is the leg's length.

The actuation force and moment of the platform can be calculated by formulating the

Newton-Euler equations and considering relations in (2.38) and (2.40) as

$$\begin{aligned} {}^{p}\mathbf{f}_{p} &= m_{p}\,{}^{p}\mathbf{a}_{p} + {}^{p}\dot{\boldsymbol{\omega}}_{p} \times {}^{p}\mathbf{ms}_{p} + {}^{p}\boldsymbol{\omega}_{p} \times ({}^{p}\boldsymbol{\omega}_{p} \times {}^{p}\mathbf{ms}_{p}) + \sum_{i=1}^{n} {}^{p}\mathbf{f}_{p,i} \\ {}^{p}\mathbf{m}_{p} &= {}^{p}\mathbf{I}_{O_p}\,{}^{p}\dot{\boldsymbol{\omega}}_{p} + {}^{p}\mathbf{ms}_{p} \times {}^{p}\mathbf{a}_{p} + {}^{p}\boldsymbol{\omega}_{p} \times ({}^{p}\mathbf{I}_{O_p}\,{}^{p}\boldsymbol{\omega}_{p}) + \sum_{i=1}^{n} ({}^{p}\mathbf{m}_{p,i} + {}^{p}\mathbf{r}_{PA_i} \times {}^{p}\mathbf{f}_{p,i}) \end{aligned} \tag{2.43}$$

where m_p, ${}^{p}\mathbf{ms}_{p}$ and ${}^{p}\mathbf{I}_{O_p}$ are constant parameters corresponding to the platform's mass, first moment of inertia and moment of inertia expressed in $\mathfrak{F}_p$, respectively. The reaction force ${}^{p}\mathbf{f}_{p,i}$ and moment ${}^{p}\mathbf{m}_{p,i}$ exerted on the platform by each leg are computed in (2.40) and converted by

$$\begin{aligned} {}^{p}\mathbf{f}_{p,i} &= {}^{p}\mathbf{R}_{li}\,{}^{li}\mathbf{f}_{p,i} \\ {}^{p}\mathbf{m}_{p,i} &= {}^{p}\mathbf{R}_{li}\,{}^{li}\mathbf{m}_{p,i} \end{aligned} \tag{2.44}$$

Finally, the additional computation is performed to obtain the actuation wrench of the FPR aligned with its definition in (2.27) as follows

$$\boldsymbol{\tau} = \begin{bmatrix} \mathbf{f}_{p} \\ {}^{p}\mathbf{m}_{p} \\ \mathrm{m}_{l1} \\ \vdots \\ \mathrm{m}_{ln} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{p}\,{}^{p}\mathbf{f}_{p} \\ {}^{p}\mathbf{m}_{p} \\ \boldsymbol{\delta}_{1}^{T} \cdot {}^{l1}\mathbf{m}_{p,1} \\ \vdots \\ \boldsymbol{\delta}_{n}^{T} \cdot {}^{ln}\mathbf{m}_{p,n} \end{bmatrix} \tag{2.45}$$

Note that $\mathrm{m}_{l1}, ..., \mathrm{m}_{ln}$ represents the moment about the rotational axis of the revolute joints, $\boldsymbol{\delta}_1, ..., \boldsymbol{\delta}_n$ are the revolute-joint axis which are identical to be $\boldsymbol{\delta}_i = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$.

The recursive Newton-Euler formula for the dynamic modelling of the FPR is thus given by equations (2.33)–(2.45). By factorizing different terms that appear in (2.25), the inertia matrix and the Coriolis vector can be computed. A numerical algorithm can be implemented allowing an online computation of the dynamic model, as

- ❖ by imposing $\boldsymbol{\nu} = \mathbf{0}$ and $g = 0$, the result of $\boldsymbol{\tau}$ computed by the recursive algorithm is exclusively the term $\mathbf{M}(\mathbf{q})\dot{\boldsymbol{\nu}}$ in the dynamic model,
- ❖ then, by imposing $\dot{\boldsymbol{\nu}}_r = \begin{bmatrix} ... & 0 & 1_{(r)} & 0 & ... \end{bmatrix}^T$ the acceleration vector with only 1 for the r-th component and 0 elsewhere, the result of $\boldsymbol{\tau}$ corresponds to the r-th column of the inertia matrix $\mathbf{M}(\mathbf{q})$,
- ❖ the Coriolis vector $\mathbf{c}(\mathbf{q}, \boldsymbol{\nu})$ is resulted from the computations by imposing $\dot{\boldsymbol{\nu}} = \mathbf{0}$.

Therefore, the numerical computation for the dynamic modelling using the recursive Newton-Euler method can be summarized by the following algorithm, which has a time complexity of $O(Nn)$, with N being the total number of the degrees of freedom of the FPR, i.e. $N = 6 + n$, and n being the number of the legs (and the attached multirotors).

Algorithm. Numerical Computation of the Recursive Newton-Euler Method

Input: $\mathbf{q}$, $\boldsymbol{\nu}$ *Output:* $\mathbf{M}(\mathbf{q})$, $\mathbf{c}(\mathbf{q},\boldsymbol{\nu})$

$\mathbf{M} = \mathbf{0}_{N\times N}$

for $j = 1$ ***to*** N

$\dot{\boldsymbol{\nu}}_r = \mathbf{0}_N$

$\dot{\boldsymbol{\nu}}_r(j) = 1$

$\mathbf{M}.column(j) = recursiveNewtonEuler(\mathbf{q}, \boldsymbol{\nu} = \mathbf{0}, \dot{\boldsymbol{\nu}}_r, g = 0)$

end

$\mathbf{c} = recursiveNewtonEuler(\mathbf{q}, \boldsymbol{\nu}, \dot{\boldsymbol{\nu}} = \mathbf{0}, g = 9.81)$

return $\mathbf{M}$, $\mathbf{c}$

function $\boldsymbol{\tau} = (\mathbf{q}, \boldsymbol{\nu}, \dot{\boldsymbol{\nu}}, g)$

$^{p}\mathbf{a}_p = \mathbf{R}_p^T(\mathbf{a}_p + \begin{bmatrix}0 & 0 & g\end{bmatrix}^T) \quad (\mathbf{a}_p = \dot{\boldsymbol{\nu}}_{1:3})$

$^{p}\mathbf{f}_p = m_p{}^{p}\mathbf{a}_p + {}^{p}\dot{\boldsymbol{\omega}}_p \times {}^{p}\mathbf{ms}_p + {}^{p}\boldsymbol{\omega}_p \times ({}^{p}\boldsymbol{\omega}_p \times {}^{p}\mathbf{ms}_p) \quad ({}^{p}\boldsymbol{\omega}_p = \boldsymbol{\nu}_{4:6})$

$^{p}\mathbf{m}_p = {}^{p}\mathbf{I}_{O_p}{}^{p}\dot{\boldsymbol{\omega}}_p + {}^{p}\mathbf{ms}_p \times {}^{p}\mathbf{a}_p + {}^{p}\boldsymbol{\omega}_p \times ({}^{p}\mathbf{I}_{O_p}{}^{p}\boldsymbol{\omega}_p) \quad ({}^{p}\dot{\boldsymbol{\omega}}_p = \dot{\boldsymbol{\nu}}_{4:6})$

for $i = 1$ ***to*** n

$^{li}\boldsymbol{\omega}_{li} = {}^{p}\mathbf{R}_{li}^T{}^{p}\boldsymbol{\omega}_p + \dot{\theta}_i\boldsymbol{\delta}_i \quad (\dot{\theta}_i = \boldsymbol{\nu}_{6+i})$

$^{li}\mathbf{a}_{li} = {}^{p}\mathbf{R}_{li}^T\left({}^{p}\mathbf{a}_p + {}^{p}\dot{\boldsymbol{\omega}}_p \times {}^{p}\mathbf{r}_{PA_i} + {}^{p}\boldsymbol{\omega}_p \times ({}^{p}\boldsymbol{\omega}_p \times {}^{p}\mathbf{r}_{PA_i})\right)$

$^{li}\dot{\boldsymbol{\omega}}_{li} = {}^{p}\mathbf{R}_{li}^T{}^{p}\dot{\boldsymbol{\omega}}_p + \ddot{\theta}_i\boldsymbol{\delta}_i + \dot{\theta}_i({}^{li}\boldsymbol{\omega}_p \times \boldsymbol{\delta}_i) \quad (\ddot{\theta}_i = \dot{\boldsymbol{\nu}}_{6+i})$

$^{li}\mathbf{f}_{p,i} = m_i{}^{li}\mathbf{a}_{li} + {}^{li}\dot{\boldsymbol{\omega}}_{li} \times {}^{li}\mathbf{ms}_i + {}^{li}\boldsymbol{\omega}_{li} \times ({}^{li}\boldsymbol{\omega}_{li} \times {}^{li}\mathbf{ms}_i)$

$^{li}\mathbf{m}_{p,i} = {}^{li}\mathbf{I}_{O_i}{}^{li}\dot{\boldsymbol{\omega}}_{li} + {}^{li}\mathbf{ms}_i \times {}^{li}\mathbf{a}_{li} + {}^{li}\boldsymbol{\omega}_{li} \times ({}^{li}\mathbf{I}_{O_i}{}^{li}\boldsymbol{\omega}_{li})$

$\boldsymbol{\tau}_{6+i} = \boldsymbol{\delta}_i^T{}^{li}\mathbf{m}_{p,i}$

$^{p}\mathbf{f}_p = {}^{p}\mathbf{f}_p + {}^{p}\mathbf{R}_{li}{}^{li}\mathbf{f}_{p,i}$

$^{p}\mathbf{m}_p = {}^{p}\mathbf{m}_p + {}^{p}\mathbf{R}_{li}{}^{li}\mathbf{m}_{p,i} + {}^{p}\mathbf{r}_{PA_i} \times ({}^{p}\mathbf{R}_{li}{}^{li}\mathbf{f}_{p,i})$

end

$\boldsymbol{\tau}_{1:3} = \mathbf{R}_p{}^{p}\mathbf{f}_p; \quad \boldsymbol{\tau}_{4:6} = {}^{p}\mathbf{m}_p$

end

2.5.3 Dynamic Modelling using Spatial Vector Notation

Featherstone's algorithms are used to model the dynamics of a rigid-body system based on spatial vector notation [Featherstone, 2008]. As presented in Section 2.5.2, a rigid body in 3-dimensional space has 6-DoF motion, with force and moment expressed by 3-dimensional vectors. In spatial vector notation, 6-dimensional wrenches or twists are introduced combining the linear and angular aspects of the rigid body's motion to simplify the equations and computations. For instance, linear and angular velocity/acceleration are combined to form a 6-dimensional spatial velocity/acceleration vector; force and moment

are combined to form a spatial force vector. Using these 6-dimensional vectors, the equation of motion for a rigid body j can be written as

$$\tilde{\mathbf{f}}_j = \tilde{\mathbf{I}}_j \tilde{\mathbf{a}}_j + \tilde{\mathbf{v}}_j \times^* \tilde{\mathbf{I}}_j \tilde{\mathbf{v}}_j \tag{2.46}$$

where $\tilde{\mathbf{f}}_j = \begin{bmatrix} \mathbf{m}_j \\ \mathbf{f}_j \end{bmatrix} \in \mathbb{F}$ is the spatial force acting on the body j, $\tilde{\mathbf{v}}_j = \begin{bmatrix} \boldsymbol{\omega}_j \\ \mathbf{v}_j \end{bmatrix} \in \mathbb{M}$ and $\tilde{\mathbf{a}}_j = \begin{bmatrix} \dot{\boldsymbol{\omega}}_j \\ \dot{\mathbf{v}}_j \end{bmatrix} \in \mathbb{M}$ are the spatial velocity and acceleration of the body j, with $\mathbb{F}$ and $\mathbb{M}$ denoting respectively spatial force vector space and motion vector space. $\tilde{\mathbf{I}}_j$ is the body j's spatial inertia tensor detailed in (2.47). The symbol $\times^*$ denotes the cross product of a spatial motion vector with a spatial force vector, i.e. $\mathbb{M} \times^* \mathbb{F}$. Note that for spatial vectors, additional tildes are placed above the symbols to distinguish them from a classical 3-dimensional vector, and the vectors of the angular/rotational aspects are usually placed above those of the linear/translational ones. For simplicity, all the spatial quantities are written in body coordinates without explicit expressions.

$$\tilde{\mathbf{I}}_j = \begin{bmatrix} \mathbf{I}_{O_j} & [\mathbf{ms}_j]_\times \\ [\mathbf{ms}_j]_\times^T & m_j \mathbf{1}_{3\times 3} \end{bmatrix} \tag{2.47}$$

Recall that $[.]_\times$ represents the skew-symmetric matrix for the cross-product operation of a 3-dimensional vector. m_j, $\mathbf{ms}_j$ and $\mathbf{I}_{O_j}$ are respectively the mass, vector of the first moment of inertia, and inertia parameters as defined in (2.31). The cross-product operation of a spatial motion vector with the symbol $\times^*$ is given by

$$\tilde{\mathbf{v}}_j \times^* = \begin{bmatrix} \boldsymbol{\omega}_j \\ \mathbf{v}_j \end{bmatrix} \times^* = \begin{bmatrix} [\boldsymbol{\omega}_j]_\times & [\mathbf{v}_j]_\times \\ \mathbf{0}_{3\times 3} & [\boldsymbol{\omega}_j]_\times \end{bmatrix} \tag{2.48}$$

The cross-product operation between two spatial motion vectors denoted by $\times$, i.e. $\mathbb{M}^6 \times \mathbb{M}^6$, is regarded as the dual of $\times^*$ and given by

$$\tilde{\mathbf{v}}_j \times = \begin{bmatrix} \boldsymbol{\omega}_j \\ \mathbf{v}_j \end{bmatrix} \times = \begin{bmatrix} [\boldsymbol{\omega}_j]_\times & \mathbf{0}_{3\times 3} \\ [\mathbf{v}_j]_\times & [\boldsymbol{\omega}_j]_\times \end{bmatrix} \tag{2.49}$$

Note that $\dot{\mathbf{v}}_j$ in the spatial acceleration vector represents the body-frame linear acceleration, which is obtained directly by the time derivative of the linear velocity $\mathbf{v}_j$, while the linear acceleration $\mathbf{a}_j$ in the dynamic modelling using Khalil's method is relative to the reference frame $\mathfrak{F}_0$, with the relationship given by

$$\mathbf{a}_j = \dot{\mathbf{v}}_j + \boldsymbol{\omega}_j \times \mathbf{v}_j \tag{2.50}$$

It is therefore remarked that the equation of motion depicted by (2.46) is equivalent to the Newton-Euler equations in (2.31), which can be applied to a recursive computation algorithm for the dynamic modelling as presented in Section 2.5.2. In this section, more compact algorithms for dynamic modelling are introduced taking advantage of the spatial vector notation, including

- ❖ a composite-rigid-body algorithm for computing the inertia matrix;
- ❖ a numerical algorithm for computing the Coriolis matrix.

Composite-Rigid-Body Algorithm

The numerical computation of the inertia matrix detailed in the recursive Newton-Euler algorithm has the advantage of being simple and straightforward. It is however not the most efficient way to calculate $\mathbf{M}$, since the algorithm runs the same computations n_{dof} times which is the total number of the DoFs of the robot. The composite-rigid-body algorithm [Featherstone, 2008], as the fastest algorithm to compute the generalised inertia matrix, considers the robot as a single composite rigid body (from which the algorithm gets its name). The algorithm starts from the computation of the spatial inertia tensor of each body from the base, i.e. $\tilde{\mathbf{I}}_p$ of the platform (considered as the floating base) in the FPR as

$$\tilde{\mathbf{I}}_p = \begin{bmatrix} {}^p\mathbf{I}_{O_p} & [{}^p\mathbf{ms}_p]_\times \\ [{}^p\mathbf{ms}_p]_\times^T & m_p\mathbf{1}_{3\times 3} \end{bmatrix} \tag{2.51}$$

with m_p, ${}^p\mathbf{ms}_p$ and ${}^p\mathbf{I}_{O_p}$ the mass, vector of the first moment of inertia and the moment of inertia of the platform, expressed at the origin of the platform frame $\mathfrak{F}_p$. Similarly, the inertia tensor of each leg i can be calculated by

$$\tilde{\mathbf{I}}_i = \begin{bmatrix} {}^{li}\mathbf{I}_{O_i} & [{}^{li}\mathbf{ms}_i]_\times \\ [{}^{li}\mathbf{ms}_i]_\times^T & m_i\mathbf{1}_{3\times 3} \end{bmatrix} \tag{2.52}$$

with m_i, ${}^{li}\mathbf{ms}_i$ and ${}^{li}\mathbf{I}_{O_i}$ being the composed mass, first moment of inertia and moment of inertia of the leg i expressed at the origin of the leg's frame $\mathfrak{F}_{li}$ and computed by equations (2.36) and (2.41) taking into account the attached multirotor's mass.

Then a new quantity is introduced, $\tilde{\mathbf{I}}_j^C$, denoted as the composite inertia of the subtree of the robot rooted at an arbitrary body j, which is treated as a single composite rigid body. $\tilde{\mathbf{I}}_j^C$ is given by

$$\tilde{\mathbf{I}}_j^C = \tilde{\mathbf{I}}_j + \sum_{k\in\mu(j)} {}^j\tilde{\mathbf{X}}_k^* \, \tilde{\mathbf{I}}_k^C \, {}^k\tilde{\mathbf{X}}_j \tag{2.53}$$

where $\mu(j)$ represents the set of all child bodies of the body j (the downstream bodies supported directly by the body j), ${}^j\mathbf{X}_k^*$, ${}^k\mathbf{X}_j \in \mathbb{R}^{6\times 6}$ are spatial matrices for the joint transform between two frames. Given two arbitrary frames $\mathfrak{F}_j$ and $\mathfrak{F}_k$, they are defined

by

$$ {}^{j}\tilde{\mathbf{X}}_k = \begin{bmatrix} {}^{j}\mathbf{R}_k & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^{j}\mathbf{R}_k \end{bmatrix} \begin{bmatrix} \mathbf{1}_{3\times3} & \mathbf{0}_{3\times3} \\ [{}^{j}\mathbf{r}_k]_\times^T & \mathbf{1}_{3\times3} \end{bmatrix}, \quad {}^{j}\tilde{\mathbf{X}}_k^* = \begin{bmatrix} {}^{j}\mathbf{R}_k & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^{j}\mathbf{R}_k \end{bmatrix} \begin{bmatrix} \mathbf{1}_{3\times3} & [{}^{j}\mathbf{r}_k]_\times^T \\ \mathbf{0}_{3\times3} & \mathbf{1}_{3\times3} \end{bmatrix} \tag{2.54} $$

with ${}^{j}\mathbf{R}_k$ representing the rotation matrix from $\mathfrak{F}_k$ to $\mathfrak{F}_j$, and ${}^{j}\mathbf{r}_k$ being the translation vector between the origins of two frames expressed in $\mathfrak{F}_j$. The inverse of these two transforms can be given by

$$ {}^{k}\tilde{\mathbf{X}}_j = \begin{bmatrix} \mathbf{1}_{3\times3} & \mathbf{0}_{3\times3} \\ [{}^{j}\mathbf{r}_k]_\times & \mathbf{1}_{3\times3} \end{bmatrix} \begin{bmatrix} {}^{j}\mathbf{R}_k^T & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^{j}\mathbf{R}_k^T \end{bmatrix}, \quad {}^{k}\tilde{\mathbf{X}}_j^* = \begin{bmatrix} \mathbf{1}_{3\times3} & [{}^{j}\mathbf{r}_k]_\times \\ \mathbf{0}_{3\times3} & \mathbf{1}_{3\times3} \end{bmatrix} \begin{bmatrix} {}^{j}\mathbf{R}_k^T & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^{j}\mathbf{R}_k^T \end{bmatrix}. \tag{2.55} $$

It can be noted that ${}^{k}\tilde{\mathbf{X}}_j^* = {}^{j}\tilde{\mathbf{X}}_k^T$.

Finally, each component of the inertia matrix $\mathbf{M}$ can be calculated from the composite inertia by the following relations

$$ \mathbf{M}_{jk} = \begin{cases} \tilde{\mathbf{S}}_j^T \; \tilde{\mathbf{I}}_j^C \; {}^{j}\tilde{\mathbf{X}}_k \; \tilde{\mathbf{S}}_k & \text{if } j \in \upsilon(k) \\ \tilde{\mathbf{S}}_j^T \; {}^{j}\tilde{\mathbf{X}}_k^* \; \tilde{\mathbf{I}}_k^C \; \tilde{\mathbf{S}}_k & \text{if } k \in \upsilon(j) \\ 0 & \text{otherwise} \end{cases} \tag{2.56} $$

where $\tilde{\mathbf{S}}_j$, $\tilde{\mathbf{S}}_k$ are the spatial vectors/matrices defining the joint's motion subspace, which are dependent on the joint type. For instance, $\tilde{\mathbf{S}} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T$ for the revolute joint's motion; $\tilde{\mathbf{S}} = \mathbf{1}_{6\times6}$ for a 6-DoF joint motion (free motion). $\upsilon(j)$, $\upsilon(k)$ are the sets of bodies in the subtree starting from the body j or k, meaning that body j is within the downstream bodies of body k if $j \in \upsilon(k)$, or vice visa. If $j = k$, it refers to the same body and $\mathbf{M}_{jj}$ (or $\mathbf{M}_{kk}$) becomes $\tilde{\mathbf{S}}_j^T \; \tilde{\mathbf{I}}_j^C \; \tilde{\mathbf{S}}_j$. $\mathbf{M}_{jk} = 0$ if body j and body k are not located in the same chain from the robot base to the tips. However, to devise an efficient calculation for $\mathbf{M}_{jk}$ in (2.56), a new quantity ${}^{k}\tilde{\mathbf{F}}_j$ is introduced, which is the value of $\tilde{\mathbf{I}}_j^C \; \tilde{\mathbf{S}}_j$ expressed in body k's coordinates, i.e. ${}^{k}\tilde{\mathbf{F}}_j = {}^{k}\tilde{\mathbf{X}}_j^* \; \tilde{\mathbf{I}}_j^C \; \tilde{\mathbf{S}}_j$. This quantity can be calculated recursively by the relationship

$$ {}^{\lambda(k)}\tilde{\mathbf{F}}_j = {}^{\lambda(k)}\tilde{\mathbf{X}}_k^* \; {}^{k}\tilde{\mathbf{F}}_j \;\; \text{with} \;\; {}^{j}\tilde{\mathbf{F}}_j = \tilde{\mathbf{I}}_j^C \; \tilde{\mathbf{S}}_j \tag{2.57} $$

where $\lambda(k)$ represents the parent bodies supporting the body k. The computation of this quantity is therefore achieved from the tips back to the base. The inertia matrix is then given by

$$ \mathbf{M}_{jk} = \begin{cases} \tilde{\mathbf{S}}_j^T \; {}^{j}\tilde{\mathbf{F}}_k & \text{if } k \in \upsilon(j) \\ \mathbf{M}_{kj}^T & \text{if } j \in \upsilon(k) \\ 0 & \text{otherwise} \end{cases} \tag{2.58} $$

Using the above-mentioned computation method, the inertia matrix of the FPR dynamics can be calculated, which is detailed as follows. The composite inertia of each body in the FPR is firstly calculated by a recursive computation from the downstream bodies (i.e. legs) to the base (i.e. platform), in which each leg's composite inertia is equal to the inertia tensor defined in (2.52), i.e. $\tilde{\mathbf{I}}_i^C = \tilde{\mathbf{I}}_i$. The composite inertia of the platform is then computed according to the relation (2.53) by

$$\tilde{\mathbf{I}}_p^C = \tilde{\mathbf{I}}_p + \sum_{i=1}^{n} {}^p\tilde{\mathbf{X}}_i^* \; \tilde{\mathbf{I}}_i^C \; {}^i\tilde{\mathbf{X}}_p \tag{2.59}$$

given the joint transform matrices defined by

$${}^p\tilde{\mathbf{X}}_i^* = \begin{bmatrix} {}^p\mathbf{R}_{li} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^p\mathbf{R}_{li} \end{bmatrix} \begin{bmatrix} \mathbf{1}_{3\times3} & [{}^p\mathbf{r}_{PA_i}]_\times^T \\ \mathbf{0}_{3\times3} & \mathbf{1}_{3\times3} \end{bmatrix}, \quad {}^i\tilde{\mathbf{X}}_p = \begin{bmatrix} \mathbf{1}_{3\times3} & \mathbf{0}_{3\times3} \\ [{}^p\mathbf{r}_{PA_i}]_\times & \mathbf{1}_{3\times3} \end{bmatrix} \begin{bmatrix} {}^p\mathbf{R}_{li}^T & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^p\mathbf{R}_{li}^T \end{bmatrix} \tag{2.60}$$

recalling that ${}^p\mathbf{R}_{li}$ is the rotation matrix from $\mathfrak{F}_{li}$ to $\mathfrak{F}_p$ defined in (2.8), ${}^p\mathbf{r}_{PA_i} = r \cdot {}^p\mathbf{r}_i$ is the vector of $\overrightarrow{PA_i}$ expressed in $\mathfrak{F}_p$ with the unit vector ${}^p\mathbf{r}_i$ defined in (2.7).

Then for the definition of joint's motion subspace matrices, as the translational motion of the platform and its corresponding actuated force are defined in a global frame $\mathfrak{F}_0$, the joint motion matrix $\tilde{\mathbf{S}}_p \in \mathbb{R}^{6\times6}$ is particularly defined by

$$\tilde{\mathbf{S}}_p = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{1}_{3\times3} \\ \mathbf{R}_p^T & \mathbf{0}_{3\times3} \end{bmatrix} \tag{2.61}$$

Note that the platform is considered as a floating base with 6-DoF free motion, and the joint motion matrix is thus similar to that of a 6-DoF joint motion (identity matrix), but the translational motion is further converted from $\mathfrak{F}_0$ to $\mathfrak{F}_p$ with the order adjusted to be coherent with the one defined in 3-dimensional vectors. The joint motion matrix for each leg i can be given by (2.62), since one-DoF revolute joints are applied

$$\tilde{\mathbf{S}}_i = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T \tag{2.62}$$

Based on the equations (2.56) and (2.58), the inertia matrix of the FPR computed by the composite-rigid-body algorithm is finally given by

$$\begin{cases} \mathbf{M}_{1:6,1:6} = \tilde{\mathbf{S}}_p^T \; \tilde{\mathbf{I}}_p^C \; \tilde{\mathbf{S}}_p \\ \mathbf{M}_{6+i,6+i} = \tilde{\mathbf{S}}_i^T \; \tilde{\mathbf{I}}_i^C \; \tilde{\mathbf{S}}_i \\ \mathbf{M}_{1:6,6+i} = \tilde{\mathbf{S}}_p^T \; {}^p\tilde{\mathbf{F}}_i \quad (\text{with } {}^p\tilde{\mathbf{F}}_i = {}^p\tilde{\mathbf{X}}_i^* \; \tilde{\mathbf{I}}_i^C \; \tilde{\mathbf{S}}_i^T) \\ \mathbf{M}_{6+i,1:6} = \mathbf{M}_{1:6,6+i}^T \end{cases} \tag{2.63}$$

where $\mathbf{M}_{1:6,1:6}$ is the top left (6×6) sub-matrix, $\mathbf{M}_{6+i,6+i}$ is the component at the index $(6+i,\ 6+i)$ and $\mathbf{M}_{1:6,6+i}$, $\mathbf{M}_{6+i,1:6}$ represent respectively a (6×1) column vector and a (1×6) row vector in the $(6+i)^{\text{th}}$ column/row of the inertial matrix.

Numerical Algorithm for Coriolis Matrix Computation

The Coriolis Matrix $\mathbf{C}$ of the dynamic model written in form of (2.26) allows to compute the Coriolis and centrifugal terms factorized by the velocity of the robot. It is well known that many possible choices of $\mathbf{C}$ exist, which provide the correct dynamics satisfying the relation of $\boldsymbol{\nu}^T\Big(\dot{\mathbf{M}}(\mathbf{q}) - 2\mathbf{C}(\mathbf{q},\boldsymbol{\nu})\Big)\boldsymbol{\nu} = 0$ [Siciliano, 2010], or more generally

$$\mathbf{u}^T\Big(\dot{\mathbf{M}}(\mathbf{q}) - 2\mathbf{C}(\mathbf{q},\boldsymbol{\nu})\Big)\mathbf{u} = 0, \quad \forall\ \mathbf{u} \in \mathbb{R}^{n_{dof}} \tag{2.64}$$

This relation is equivalent to the condition that $\dot{\mathbf{M}}(\mathbf{q}) - 2\mathbf{C}(\mathbf{q},\boldsymbol{\nu})$ is skew symmetric or that

$$\dot{\mathbf{M}}(\mathbf{q}) = \mathbf{C}(\mathbf{q},\boldsymbol{\nu}) + \mathbf{C}(\mathbf{q},\boldsymbol{\nu})^T \tag{2.65}$$

One special computation for $\mathbf{C}$ can be achieved by

$$\mathbf{C}_{ij} = \sum_k \Gamma_{ijk} \cdot \nu_k, \quad \text{where } \Gamma_{ijk} = \frac{1}{2}\left(\frac{\partial \mathbf{M}_{ij}}{\partial q_k} + \frac{\partial \mathbf{M}_{ik}}{\partial q_j} - \frac{\partial \mathbf{M}_{jk}}{\partial q_i}\right) \tag{2.66}$$

are referred to as the Christoffel symbols [Siciliano, 2010]. Note that ν_k is the velocity of the joint k, $\mathbf{M}$ is the inertia matrix of the system expressed in an analytical form. The Coriolis matrix computed by Christoffel symbols can be denoted by $\mathbf{C}^\star(\mathbf{q},\boldsymbol{\nu})$, which is also called the Christoffel-consistent Coriolis factorization. However, this method needs to compute the partial differentiation of the inertia matrix with respect to the robot generalised coordinates $\mathbf{q}$, requiring a relatively complex analytical derivation. Therefore, the other Coriolis factorization methods for the computation of $\mathbf{C}$ can be applied, among which an admissible Coriolis factorization is defined as follows [Echeandia, 2021].

Definition (Admissible Coriolis Factorization)**.** Consider a system with the inertia matrix given by $\mathbf{M}(\mathbf{q})$. A matrix-valued function $\mathbf{C}(\mathbf{q},\boldsymbol{\nu})$ is said to be an admissible factorization if

1. $\mathbf{C}(\mathbf{q},\boldsymbol{\nu})\boldsymbol{\nu} = \mathbf{C}^\star(\mathbf{q},\boldsymbol{\nu})\boldsymbol{\nu}$ with $\mathbf{C}^\star(\mathbf{q},\boldsymbol{\nu})$ being the Christoffel-consistent Coriolis factorization.
2. $\forall \mathbf{q},\boldsymbol{\nu}$, $\dot{\mathbf{M}}(\mathbf{q}) - 2\mathbf{C}(\mathbf{q},\boldsymbol{\nu})$ is skew-symmetric.

The Coriolis computation can be achieved by factorization of the spatial equation (2.46) in which the velocity-product term $\tilde{\mathbf{v}}_j \times^* \tilde{\mathbf{I}}_j\tilde{\mathbf{v}}_j$ is firstly factorized in a way such

that

$$\tilde{\mathbf{v}}_j \times^* \tilde{\mathbf{I}}_j \tilde{\mathbf{v}}_j = \tilde{\mathbf{B}}(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j)\tilde{\mathbf{v}}_j \tag{2.67}$$

from which one immediate factorization can be taken as $\tilde{\mathbf{B}}(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j) = (\tilde{\mathbf{v}}_j \times^* \tilde{\mathbf{I}}_j)$, while another factorization adopted by [Echeandia, 2021] is

$$\tilde{\mathbf{B}}(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j) = \frac{1}{2}\Big((\tilde{\mathbf{v}}_j \times^*)\tilde{\mathbf{I}}_j + (\tilde{\mathbf{I}}_j \tilde{\mathbf{v}}_j \bar{\times}^*) - \tilde{\mathbf{I}}_j(\tilde{\mathbf{v}}_j \times)\Big) \tag{2.68}$$

with the operation $\bar{\times}^*$ defined as an equivalence of the cross product $\times^*$ between a spatial motion vector $\tilde{\mathbf{v}}_j$ and a spatial force vector $\tilde{\mathbf{f}}_j$, i.e.

$$(\tilde{\mathbf{f}}_j \bar{\times}^*)\tilde{\mathbf{v}}_j = (\tilde{\mathbf{v}}_j \times^*)\tilde{\mathbf{f}}_j \ . \tag{2.69}$$

It can be noticed from (2.48) that this operation is derived as

$$\tilde{\mathbf{f}}_j \bar{\times}^* = \begin{bmatrix} \mathbf{m}_j \\ \mathbf{f}_j \end{bmatrix} \bar{\times}^* = \begin{bmatrix} [\mathbf{m}_j]_\times^T & [\mathbf{f}_j]_\times^T \\ [\mathbf{f}_j]_\times^T & \mathbf{0}_{3\times 3} \end{bmatrix}. \tag{2.70}$$

The factorization of the velocity-product terms given by (2.68) satisfies the properties for an admissible factorization of the Coriolis matrix $\mathbf{C}(\mathbf{q}, \boldsymbol{\nu})$ in the dynamic model, which is proven as follows.

Proof. Consider the spatial inertia tensor of a body j denoted by $\tilde{\mathbf{I}}_j$ and its spatial velocity vector $\tilde{\mathbf{v}}_j$. Knowing the time derivative of spatial inertia that is given by [Featherstone, 2008]

$$\dot{\tilde{\mathbf{I}}}_j = (\tilde{\mathbf{v}}_j \times^*)\tilde{\mathbf{I}}_j - \tilde{\mathbf{I}}_j(\tilde{\mathbf{v}}_j \times) \tag{2.71}$$

it can be demonstrated that the factorization of $\tilde{\mathbf{B}}(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j)$ in (2.68) satisfies

$$\tilde{\mathbf{v}}^T\Big(\dot{\tilde{\mathbf{I}}}_j - 2\tilde{\mathbf{B}}(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j)\Big)\tilde{\mathbf{v}} = 0, \ \ \forall\ \tilde{\mathbf{v}} \in \mathbb{M} \tag{2.72}$$

which is equivalent to that $\dot{\tilde{\mathbf{I}}}_j - 2\tilde{\mathbf{B}}(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j)$ is skew-symmetric when expressed in coordinates or $\dot{\tilde{\mathbf{I}}}_j = \tilde{\mathbf{B}}(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j) + \tilde{\mathbf{B}}(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j)^T$. Such factorization is said to be an admissible body-level factorization which will naturally lead to system-level Coriolis matrix $\mathbf{C}$ satisfying the conditions for an admissible Coriolis factorization.

Then, the factorization of $\tilde{\mathbf{I}}_j \tilde{\mathbf{a}}_j$ by $\tilde{\mathbf{v}}_j$ is additionally required as the spatial acceleration of each body j involves the joint velocity. The spatial velocity and acceleration of a body can be computed recursively as presented in the Newton-Euler recursive algorithm but

written in spatial vector notation as

$$\tilde{\mathbf{v}}_k = \sum_{j \in \lambda(k)} \tilde{\mathbf{S}}_j \tilde{\mathbf{v}}_j \tag{2.73}$$

$$\tilde{\mathbf{a}}_k = \sum_{j \in \lambda(k)} (\tilde{\mathbf{S}}_j \tilde{\mathbf{a}}_j + \dot{\tilde{\mathbf{S}}}_j \tilde{\mathbf{v}}_j) \tag{2.74}$$

with $\lambda(k)$ representing the sets of the parent bodies of the body k (upstream bodies from the body k to the base), and $\tilde{\mathbf{S}}_k$ being the joint's motion subspace of the body k. $\dot{\tilde{\mathbf{S}}}_k$ is the derivative of the joint's motion subspace matrix due to the joint moving and given by

$$\dot{\tilde{\mathbf{S}}}_k = (\tilde{\mathbf{v}}_k \times) \tilde{\mathbf{S}}_k \tag{2.75}$$

Therefore, it can be noticed from (2.74) that the additional terms corresponding to $\tilde{\mathbf{I}}_j \sum_{k \in \mu(j)} (\dot{\tilde{\mathbf{S}}}_k \tilde{\mathbf{v}}_k)$ of each body should be included in the Coriolis terms, which necessitates the computation of the body's composite inertia as given by (2.53). The body-level computation of $\tilde{\mathbf{B}}(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j)$ can be achieved using the same principle of the composite-rigid-body algorithm, i.e.

$$\tilde{\mathbf{B}}^C(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j) = \tilde{\mathbf{B}}(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j) + \sum_{k \in \mu(j)} \tilde{\mathbf{B}}^C(\tilde{\mathbf{v}}_k, \tilde{\mathbf{I}}_k) \tag{2.76}$$

By grouping the factorization terms $\tilde{\mathbf{B}}(\tilde{\mathbf{v}}_j, \tilde{\mathbf{I}}_j)$ and $\tilde{\mathbf{I}}_j \sum_{k \in \mu(j)} (\dot{\tilde{\mathbf{S}}}_k)$ and expressing in body coordinates, the component of Coriolis matrix can be calculated as

$$\begin{aligned} \mathbf{C}_{jk} &= \tilde{\mathbf{S}}_j^T \, {}^j\tilde{\mathbf{X}}_k^* \, (\tilde{\mathbf{I}}_k^C \, \dot{\tilde{\mathbf{S}}}_k + \tilde{\mathbf{B}}_k^C \, \tilde{\mathbf{S}}_k) \\ \mathbf{C}_{kj} &= \left(\dot{\tilde{\mathbf{S}}}_j^T \, {}^j\tilde{\mathbf{X}}_k^* \, \tilde{\mathbf{I}}_k^C \, \tilde{\mathbf{S}}_k + \tilde{\mathbf{S}}_j^T \, {}^j\tilde{\mathbf{X}}_k^* \, (\tilde{\mathbf{B}}_k^C)^T \, \tilde{\mathbf{S}}_k \right)^T \end{aligned} \tag{2.77}$$

when $j \in \upsilon(k)$, i.e. body j is in the path from body k to the robot base. Similarly as in (2.57), additional quantities can be defined to facilitate the calculation

$$\begin{cases} {}^{\lambda(k)}\tilde{\mathbf{F}}_{1,j} = {}^{\lambda(k)}\tilde{\mathbf{X}}_k^* \, {}^k(\tilde{\mathbf{I}}_j^C \, \dot{\tilde{\mathbf{S}}}_j + \tilde{\mathbf{B}}_j^C \, \tilde{\mathbf{S}}_j) \\ {}^{\lambda(k)}\tilde{\mathbf{F}}_{2,j} = {}^{\lambda(k)}\tilde{\mathbf{X}}_k^* \, {}^k(\tilde{\mathbf{I}}_j^C \, \tilde{\mathbf{S}}_j) \\ {}^{\lambda(k)}\tilde{\mathbf{F}}_{3,j} = {}^{\lambda(k)}\tilde{\mathbf{X}}_k^* \, {}^k((\tilde{\mathbf{B}}_j^C)^T \, \tilde{\mathbf{S}}_j) \end{cases} \tag{2.78}$$

which can then be calculated recursively from the downstream bodies to the robot base. The Coriolis matrix calculation is finally given by

$$\begin{cases} \mathbf{C}_{jk} = \tilde{\mathbf{S}}_j^T \, {}^j\tilde{\mathbf{F}}_{1,k} \\ \mathbf{C}_{kj} = (\dot{\tilde{\mathbf{S}}}_j^T \, {}^j\tilde{\mathbf{F}}_{2,k} + \tilde{\mathbf{S}}_j^T \, {}^j\tilde{\mathbf{F}}_{3,k})^T \end{cases} \tag{2.79}$$

For the FPR, the computation of the Coriolis matrix is achieved following the above-mentioned process, which is initiated by computing the composite factorization term $\tilde{\mathbf{B}}^C(\tilde{\mathbf{v}}_i, \tilde{\mathbf{I}}_i) = \tilde{\mathbf{B}}(\tilde{\mathbf{v}}_i, \tilde{\mathbf{I}}_i)$ of each leg. Then the composite factorization term of the platform is calculated according to (2.76) by

$$\mathbf{B}^C(\tilde{\mathbf{v}}_p, \tilde{\mathbf{I}}_p) = \mathbf{B}(\tilde{\mathbf{v}}_p, \tilde{\mathbf{I}}_p) + \sum_{i=1}^{n} \mathbf{B}^C(\tilde{\mathbf{v}}_i, \tilde{\mathbf{I}}_i) \tag{2.80}$$

where $\mathbf{B}(\tilde{\mathbf{v}}_p, \tilde{\mathbf{I}}_p)$ and each $\mathbf{B}(\tilde{\mathbf{v}}_i, \tilde{\mathbf{I}}_i)$ are computed using the factorization of (2.68). For the additional terms corresponding to the derivative of the joint's motion subspace matrix $\dot{\tilde{\mathbf{S}}}_k$, it is noted that for each leg

$$\dot{\tilde{\mathbf{S}}}_i = (\tilde{\mathbf{v}}_i \times)\tilde{\mathbf{S}}_i \tag{2.81}$$

with $\tilde{\mathbf{v}}_i$ the spatial velocity of the leg i that can be calculated recursively using the relation of (2.73) and $\tilde{\mathbf{S}}_i$ being the revolute-joint motion vector given by (2.62). The derivative of the platform's joint motion matrix, denoted by $\dot{\tilde{\mathbf{S}}}_p$, is however different from the original definition in (2.75), since the platform is a floating base and does not have the notion of joint motion. The joint motion matrix defined in (2.61) is only used for the purpose of transforming frames. From the definition of $\tilde{\mathbf{S}}_p$, its derivative can therefore be defined by

$$\dot{\tilde{\mathbf{S}}}_p = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ [{}^p\boldsymbol{\omega}_p]_\times^T & \mathbf{0}_{3\times3} \end{bmatrix} \tag{2.82}$$

Knowing that $\dot{\mathbf{R}}_p = \mathbf{R}_p[{}^p\boldsymbol{\omega}_p]_\times$ and as the linear velocity in $\tilde{\mathbf{v}}_p$ is defined in $\mathfrak{F}_0$, the left-down corner of $\dot{\tilde{\mathbf{S}}}_p$ becomes $[{}^p\boldsymbol{\omega}_p]_\times^T$. Note that the acceleration of the platform in the generalised acceleration vector of (2.3) is defined with respect to $\mathfrak{F}_0$, while the spatial acceleration is referred to as body-frame acceleration, with the relationship presented in (2.50). The joint motion derivative matrix of the platform $\dot{\tilde{\mathbf{S}}}_p$ is applied to satisfy this relationship as

$$\tilde{\mathbf{a}}_p = \begin{bmatrix} {}^p\dot{\boldsymbol{\omega}}_p \\ {}^p\dot{\mathbf{v}}_p \end{bmatrix} = \tilde{\mathbf{S}}_p \begin{bmatrix} {}^p\dot{\boldsymbol{\omega}}_p \\ \mathbf{a}_p \end{bmatrix} + \dot{\tilde{\mathbf{S}}}_p \tilde{\mathbf{v}}_p \tag{2.83}$$

where $\tilde{\mathbf{S}}_p$ is the platform's motion subspace matrix defined in (2.61).

Based on equations (2.78) and (2.79), the Coriolis matrix of the FPR can finally be calculated by

$$\begin{cases} \mathbf{C}_{1:6,1:6} = \tilde{\mathbf{S}}_p^T \, {}^p\tilde{\mathbf{F}}_{1,p} \\ \mathbf{C}_{6+i,6+i} = \tilde{\mathbf{S}}_i^T \, {}^i\tilde{\mathbf{F}}_{1,i} \\ \mathbf{C}_{1:6,6+i} = \tilde{\mathbf{S}}_p^T \, {}^p\tilde{\mathbf{F}}_{1,i} \\ \mathbf{C}_{6+i,1:6} = (\dot{\tilde{\mathbf{S}}}_p^T \, {}^p\tilde{\mathbf{F}}_{2,i} + \tilde{\mathbf{S}}_p^T \, {}^p\tilde{\mathbf{F}}_{3,i})^T \end{cases} \tag{2.84}$$

where $\mathbf{C}_{1:6,1:6}$ is the top left (6×6) sub-matrix of $\mathbf{C}$, $\mathbf{C}_{6+i,6+i}$ is the component at the index $(6{+}i,\ 6{+}i)$, and $\mathbf{C}_{1:6,6+i}$, $\mathbf{C}_{6+i,1:6}$ represent respectively a (6×1) column vector and a (1×6) row vector in the $(6+i)^{\text{th}}$ column/row of the Coriolis matrix. The additional quantities are given by

$$\begin{cases} {}^{p}\tilde{\mathbf{F}}_{1,p} = \tilde{\mathbf{I}}_p^C\ \dot{\tilde{\mathbf{S}}}_p + \tilde{\mathbf{B}}_p^C\ \tilde{\mathbf{S}}_p \\ {}^{i}\tilde{\mathbf{F}}_{1,i} = \tilde{\mathbf{I}}_i^C\ \dot{\tilde{\mathbf{S}}}_i + \tilde{\mathbf{B}}_i^C\ \tilde{\mathbf{S}}_i \\ {}^{p}\tilde{\mathbf{F}}_{1,i} = {}^{p}\tilde{\mathbf{X}}_i^*\ (\tilde{\mathbf{I}}_i^C\ \dot{\tilde{\mathbf{S}}}_i + \tilde{\mathbf{B}}_i^C\ \tilde{\mathbf{S}}_i) \\ {}^{p}\tilde{\mathbf{F}}_{2,i} = {}^{p}\tilde{\mathbf{X}}_i^*\ (\tilde{\mathbf{I}}_i^C\ \tilde{\mathbf{S}}_i) \\ {}^{p}\tilde{\mathbf{F}}_{3,i} = {}^{p}\tilde{\mathbf{X}}_i^*\ ((\tilde{\mathbf{B}}_i^C)^T\ \tilde{\mathbf{S}}_i)\ . \end{cases} \tag{2.85}$$

The algorithm for computing the inertia matrix and the Coriolis matrix of the FPR based on spatial vector notation can be summarised by the following algorithm. The overall algorithm has a time complexity of $O(n)$, with n being the total number of legs.

Algorithm. Computation of Inertia and Coriolis Matrix based on Spatial Vectors

Input: $\mathbf{q}$, $\boldsymbol{\nu}$ *Output:* $\mathbf{M}(\mathbf{q})$, $\mathbf{C}(\mathbf{q}, \boldsymbol{\nu})$

$\tilde{\mathbf{v}}_p = \begin{bmatrix} {}^{p}\boldsymbol{\omega}_p^T & \mathbf{v}_p^T \end{bmatrix}^T \quad (\mathbf{v}_p = \boldsymbol{\nu}_{1:3},\ {}^{p}\boldsymbol{\omega}_p = \boldsymbol{\nu}_{4:6})$

$\tilde{\mathbf{I}}_p^C = \tilde{\mathbf{I}}_p$

$\tilde{\mathbf{B}}_p^C = \frac{1}{2}\Big((\tilde{\mathbf{v}}_p\times^*)\tilde{\mathbf{I}}_p + (\tilde{\mathbf{I}}_p\tilde{\mathbf{v}}_p\bar{\times}^*) - \tilde{\mathbf{I}}_p(\tilde{\mathbf{v}}_p\times)\Big)$

for $i = 1$ ***to*** n

- $\tilde{\mathbf{v}}_i = {}^{i}\tilde{\mathbf{X}}_p\tilde{\mathbf{v}}_p + \tilde{\mathbf{S}}_i\dot{\theta}_i \quad (\dot{\theta}_i = \boldsymbol{\nu}_{6+i})$
- $\dot{\tilde{\mathbf{S}}}_i = (\tilde{\mathbf{v}}_i\times)\tilde{\mathbf{S}}_i$
- $\tilde{\mathbf{I}}_i^C = \tilde{\mathbf{I}}_i$
- $\tilde{\mathbf{B}}_i^C = \frac{1}{2}\Big((\tilde{\mathbf{v}}_i\times^*)\tilde{\mathbf{I}}_i + (\tilde{\mathbf{I}}_i\tilde{\mathbf{v}}_i\bar{\times}^*) - \tilde{\mathbf{I}}_i(\tilde{\mathbf{v}}_i\times)\Big)$
- $\tilde{\mathbf{F}}_{1,i} = \tilde{\mathbf{I}}_i^C\ \dot{\tilde{\mathbf{S}}}_i + \tilde{\mathbf{B}}_i^C\ \tilde{\mathbf{S}}_i$
- $\tilde{\mathbf{F}}_{2,i} = \tilde{\mathbf{I}}_i^C\ \tilde{\mathbf{S}}_i$
- $\tilde{\mathbf{F}}_{3,i} = (\tilde{\mathbf{B}}_i^C)^T\tilde{\mathbf{S}}_i$
- $\mathbf{C}_{6+i,6+i} = \tilde{\mathbf{S}}_i^T\ \tilde{\mathbf{F}}_{1,i};\ \ \mathbf{M}_{6+i,6+i} = \tilde{\mathbf{S}}_i^T\ \tilde{\mathbf{F}}_{2,i}$
- ${}^{p}\tilde{\mathbf{F}}_{1,i} = {}^{p}\tilde{\mathbf{X}}_i^*\ \tilde{\mathbf{F}}_{1,i};\ \ {}^{p}\tilde{\mathbf{F}}_{2,i} = {}^{p}\tilde{\mathbf{X}}_i^*\ \tilde{\mathbf{F}}_{2,i};\ \ {}^{p}\tilde{\mathbf{F}}_{3,i} = {}^{p}\tilde{\mathbf{X}}_i^*\ \tilde{\mathbf{F}}_{3,i}$
- $\mathbf{C}_{1:6,6+i} = \tilde{\mathbf{S}}_p^T\ {}^{p}\tilde{\mathbf{F}}_{1,i}$
- $\mathbf{C}_{6+i,1:6} = \dot{\tilde{\mathbf{S}}}_p^T\ {}^{p}\tilde{\mathbf{F}}_{2,i} + \tilde{\mathbf{S}}_p^T\ {}^{p}\tilde{\mathbf{F}}_{3,i}$
- $\mathbf{M}_{1:6,6+i} = \tilde{\mathbf{S}}_p^T\ {}^{p}\tilde{\mathbf{F}}_{2,i}$
- $\mathbf{M}_{6+i,1:6} = \mathbf{M}_{1:6,6+i}^T$
- $\tilde{\mathbf{I}}_p^C = \tilde{\mathbf{I}}_p^C + {}^{p}\tilde{\mathbf{X}}_i^*\ \tilde{\mathbf{I}}_i^C\ {}^{i}\tilde{\mathbf{X}}_p$
- $\tilde{\mathbf{B}}_p^C = \tilde{\mathbf{B}}_p^C + {}^{p}\tilde{\mathbf{X}}_i^*\ \tilde{\mathbf{B}}_i^C\ {}^{i}\tilde{\mathbf{X}}_p$

end

$\tilde{\mathbf{F}}_{1,p} = \tilde{\mathbf{I}}_p^C\ \dot{\tilde{\mathbf{S}}}_p + \tilde{\mathbf{B}}_p^C\ \tilde{\mathbf{S}}_p$

$\tilde{\mathbf{F}}_{2,p} = \tilde{\mathbf{I}}_p^C \, \tilde{\mathbf{S}}_p$
$\tilde{\mathbf{F}}_{3,p} = (\tilde{\mathbf{B}}_p^C)^T \tilde{\mathbf{S}}_p$
$\mathbf{C}_{1:6,1:6} = \tilde{\mathbf{S}}_p^T \, \tilde{\mathbf{F}}_{1,p}$; $\quad \mathbf{M}_{1:6,1:6} = \tilde{\mathbf{S}}_p^T \, \tilde{\mathbf{F}}_{2,p}$
return **M**, **C**

2.5.4 Computation of Gravity Wrench

The vector of the generalised gravity wrench $\mathbf{g}(\mathbf{q})$ in the dynamic model (2.26) can be computed using the recursive Newton-Euler algorithm by setting $\boldsymbol{\nu} = \mathbf{0}$, $\dot{\boldsymbol{\nu}} = \mathbf{0}$ (see Section 2.5.2). However, this term refers to the forces and moments due to the gravitational effects and can be computed more straightforward by deriving the static model written as

$$\boldsymbol{\tau}_g = \mathbf{g}(\mathbf{q}) \tag{2.86}$$

where $\boldsymbol{\tau}_g$ refers to the actuation wrench needed to compensate for the gravitational effects. It is remarked that the static model (2.86) indicates the robot working in (quasi-)static conditions where the velocity $\boldsymbol{\nu}$ and acceleration $\dot{\boldsymbol{\nu}}$ are assumed to be zero.

The elements of $\boldsymbol{\tau}_g$ are defined in the same space of $\boldsymbol{\tau}$ in (2.27) and given by

$$\boldsymbol{\tau}_g = \begin{bmatrix} \mathbf{f}_{p,g} \\ {}^p\mathbf{m}_{p,g} \\ \mathbf{m}_{l,g} \end{bmatrix} \tag{2.87}$$

with $\mathbf{m}_{l,g} = \begin{bmatrix} \mathrm{m}_{1,g} & \mathrm{m}_{2,g} & \dots & \mathrm{m}_{n,g} \end{bmatrix}^T$ being the moments exerted about the revolute-joint axis of each leg due to the gravitation effects. It is obvious that the force acting on the platform $\mathbf{f}_{p,g}$ to compensate for the gravitational effect can be given by

$$\mathbf{f}_{p,g} = -m_{tot}\mathfrak{g} \tag{2.88}$$

where $\mathfrak{g} = \begin{bmatrix} 0 & 0 & -g \end{bmatrix}^T$ is the gravity vector expressed in $\mathfrak{F}_0$ with $g = 9.81 \ m/s^2$, m_{tot} is the total mass of the robot

$$m_{tot} = m_p + \sum_{i=1}^{n} m_i \tag{2.89}$$

Note that m_i is the total mass of the leg i and the attached multirotor given in (2.36). The gravitational effect on the moment of the platform is computed recursively by

$${}^p\mathbf{m}_{p,g} = -\Big({}^p\mathbf{ms}_p \times {}^p\mathfrak{g} + {}^p\mathbf{R}_{li} \sum_{i=1}^{n} ({}^{li}\mathbf{ms}_i \times {}^{li}\mathfrak{g})\Big) \tag{2.90}$$

with ${}^{p}\mathfrak{g} = \mathbf{R}_p^T \mathfrak{g}$ and ${}^{li}\mathfrak{g} = {}^{p}\mathbf{R}_{li}^T \mathbf{R}_p^T \mathfrak{g}$.

Finally, the moment of each leg to compensate the gravity can be written by

$$\mathrm{m}_{i,g} = -\boldsymbol{\delta}_i^T ({}^{li}\mathbf{ms}_i \times {}^{li}\mathfrak{g}) \tag{2.91}$$

with $\boldsymbol{\delta}_i = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ being the axis of the revolute joint.

2.5.5 Multirotor Rotational Dynamics

As presented in Section 2.5, the translational dynamics of the multirotors are systematically considered in the dynamic model of the passive architecture, as they are rigidly connected to the passive architecture and their effects are taken into account by adding additional mass at each leg's tip. The rotational dynamics is however decoupled for each multirotor. Equations of motion for the rotational movement of the multirotor i can be written as [Brescianini, 2013]

$$\begin{aligned} \dot{\mathfrak{q}}_i &= \frac{1}{2}\mathfrak{q}_i \circ \begin{bmatrix} 0 \\ {}^{bi}\boldsymbol{\omega}_i \end{bmatrix} \\ {}^{bi}\mathbf{m}_i &= {}^{bi}\mathbf{I}_i \, {}^{bi}\dot{\boldsymbol{\omega}}_i + {}^{bi}\boldsymbol{\omega}_i \times {}^{bi}\mathbf{I}_i \, {}^{bi}\boldsymbol{\omega}_i \end{aligned} \tag{2.92}$$

where $\mathfrak{q}_i \in \mathbb{H}$, ${}^{bi}\boldsymbol{\omega}_i \in \mathbb{R}^3$ and ${}^{bi}\dot{\boldsymbol{\omega}}_i \in \mathbb{R}^3$ are the quaternion-represented orientation, angular velocity and acceleration of the multirotor i. ${}^{bi}\mathbf{m}_i$ is the body-frame torque, and ${}^{bi}\mathbf{I}_i$ is the rotational inertia of the multirotor. The operation $\circ$ represents the quaternion multiplication detailed in (A.48) of Appendix A.1.

The decoupling property of the multirotor's rotational dynamics from the dynamics of the passive architecture is ensured by the fact that the moment cannot be transmitted through a spherical joint, which necessitates that the CoM position of the multirotor B_i is aligned with the location of the spherical joint S_i, i.e. $d = 0$ as presented in Section 2.3.3. When $d \neq 0$, the interaction force between the multirotor and the leg will additionally generate a moment in the multirotor's body-fixed frame. Given the interaction force $\mathbf{f}_{l,i}$ exerted by the leg on the multirotor, the additional moment due to the spherical-joint location $\mathbf{d}_i = \overrightarrow{S_i B_i}$ away from the multirotor's CoM position can be calculated by

$$\mathbf{m}_{l,i} = \mathbf{d}_i \times \mathbf{f}_{l,i} \tag{2.93}$$

written in global coordinates (expressed in $\mathfrak{F}_0$) as shown in Fig. 2.4.

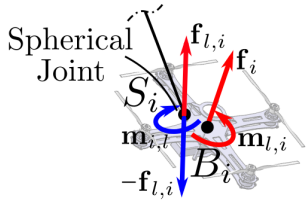


Figure 2.4 – Forces and moment exerted on the multirotor (red arrows), and reaction force and moment exerted on the leg (blue arrows).

Similarly, the actuated force $\mathbf{f}_i$ produced by the multirotor is transmitted to the passive architecture via the leg, generating an additional moment $\mathbf{m}_{i,l}$ due to the spherical joint location that is away from the CoM location of the multirotor. However, this moment will not affect the dynamics of the passive architecture since the moment between two bodies connected via the spherical joint cannot be transmitted. Therefore, the condition $d = 0$ can be assumed even if it is not necessarily the case, with the moment $\mathbf{m}_{l,i}$ considered as a disturbance exerted on the multirotor degenerating its rotational dynamics. The disturbance of $\mathbf{m}_{l,i}$ is bounded and can be neglected in general cases as the value of d is small enough.

2.6 Numerical Validation

The modelling methods presented in the above sections are usually applied to construct model-based control algorithms or estimation techniques. In these methods, the modelling accuracy is an important issue affecting the validity and stability of the estimation and control system. The numerical validation of the models should therefore be conducted. This can be done by cross-validating the different modelling algorithms, comparing the results of the dynamic models respectively based on Khalil's algorithm [Khalil, 2002b] and Featherstone's Spatial Vector algorithms [Featherstone, 2008].

The principle of the numerical validation on the dynamic modelling is that by randomly choosing a set of values for the robot states (i.e. generalised coordinates $\mathbf{q}$ and velocity vector $\boldsymbol{\nu}$), the results using two methods should be identical. As the expressions of the dynamic model between (2.25) and (2.26) are slightly different using two algorithms, a fair comparison is done by the following steps:

- $\mathbf{M}(\mathbf{q})$ the inertia matrix computed respectively using recursive Newton-Euler and Composite-Rigid-Body algorithms.
- $\mathbf{c}(\mathbf{q}, \boldsymbol{\nu})$ the Coriolis vector computed by the recursive Newton-Euler algorithm considering only the Coriolis and centrifugal effects with the gravity g set as zero, compared with the term $\mathbf{C}(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu}$ in which $\mathbf{C}(\mathbf{q}, \boldsymbol{\nu})$ is calculated using the Coriolis

Matrix Computation algorithm.

- $\mathbf{g}(\mathbf{q})$ the vector of the gravity wrench computed using two methods: one computed by the recursive Newton-Euler algorithm excluding the Coriolis and centrifugal terms by setting $\boldsymbol{\nu} = \mathbf{0}$, the other directly obtained by the static model detailed in Section 2.5.4.

The numerical implementation of the dynamic models on a specific FPR composed of three legs attached with multirotors was done in C++ and then the results were numerically validated using the G-Test tool in ROS2 environment [Open-Robotics, 2022]. The repeatable results were found during the numerical validation by taking different sets of random values in the robot coordinates and one result is presented below. While the values on the constant parameters can be found in Table B.1 of Appendix B, the generalised coordinates and velocity of the robot are generated by random numbers as follows

$$\begin{aligned}
\mathbf{q} &= [p_{p,x},\ p_{p,y},\ p_{p,z},\ \mathfrak{q}_{p,0},\ \mathfrak{q}_{p,1},\ \mathfrak{q}_{p,2},\ \mathfrak{q}_{p,3},\ \theta_1,\ \theta_2,\ \theta_3]^T \\
&= [0.815,\ 0.906,\ 0.127,\ 0.795,\ 0.550,\ 0.085,\ 0.242,\ 0.547,\ 0.958,\ 0.965]^T \\
\boldsymbol{\nu} &= [v_{p,x},\ v_{p,y},\ v_{p,z},\ \omega_{p,x},\ \omega_{p,y},\ \omega_{p,z},\ \dot{\theta}_1,\ \dot{\theta}_2,\ \dot{\theta}_3]^T \\
&= [0.158,\ 0.971,\ 0.957,\ 0.485,\ 0.800,\ 0.142,\ 0.422,\ 0.916,\ 0.792]^T
\end{aligned} \tag{2.94}$$

Note that the units are respectively metres for $\mathbf{p}_p$, radians for $\boldsymbol{\theta}_l$, metres per second for $\mathbf{v}_p$, and radians per second for angular velocities such as $\boldsymbol{\nu}_p$ and $\dot{\boldsymbol{\theta}}_l$. The values of the unit quaternion $\mathfrak{q}_p = [\mathfrak{q}_{p,0}, \mathfrak{q}_{p,1}, \mathfrak{q}_{p,2}, \mathfrak{q}_{p,3}]^T = [0.795, 0.550, 0.085, 0.242]^T$ have been normalised. The results on the numerical comparison are shown in Table 2.1.

Comparison Term	Computation Method		Maximum Difference
	Khalil's method	Featherstone's method	
Inertia matrix	$\mathbf{M}_{RNE}(\mathbf{q})$	$\mathbf{M}_{Composite}(\mathbf{q})$	4.4e-16
Coriolis vector	$\mathbf{c}_{RNE}(\mathbf{q}, \boldsymbol{\nu}, g = 0)$	$\mathbf{C}_{Coriolis}(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu}$	8.9e-16
Gravity term	$\mathbf{c}_{RNE}(\mathbf{q}, \boldsymbol{\nu} = \mathbf{0}, g = 9.81)$	$\mathbf{g}^*(\mathbf{q})$	0

Table 2.1 – Numerical validation of the modelling algorithms based on Khalil's method and Featherstone's Spatial Vector notation. Note that *RNE* refers to the recursive Newton-Euler algorithm, *Composite* stands for the Composite-Rigid-Body algorithm and *Coriolis* for the Coriolis Matrix Computation algorithm.

It can be noticed that results from the two modelling methods are well coherent with each other, with the maximum values of the absolute differences on every element between two comparison terms all near zero. Note that the gravity wrench $\mathbf{g}^*(\mathbf{q})$ under Featherstone's method is actually not based on Spatial Vector notation, but directly computed via the static model. It can be done by developing Featherstone's algorithm with the velocity $\boldsymbol{\nu}$ set as zero, similarly to what has been done in Khalil's method. However, due to the simplicity of the static model, it is not preferred to restart the algorithm from the beginning

for computing the gravity term.

The numerical validation of the other models such as the geometric relations and the Jacobian matrix in the kinematic model has also been conducted. As the analytical expressions of such models are well known and carefully verified, the only concern remains on the programming mistakes. To avoid this, cross-validation between different programming platforms was done, in which the results from the C++ codes were compared with those obtained by the codes written in MATLAB. The acceptance precision was set to 1e-7 as the success criterion for G-Test and all the tests were successfully passed. Another numerical test for the cross-validation between the IGM and the inverse kinematics (IKM) has been done, in which the idea is that by setting arbitrarily initial values for the generalised coordinates $\mathbf{q}$ and some small variations $\delta\mathbf{q}$, the initial and final multirotor positions can be computed using the IGM as $\mathbf{p}_{init} = \texttt{IGM}(\mathbf{q})$, $\mathbf{p}_{fin} = \texttt{IGM}(\mathbf{q} + \delta\mathbf{q})$, while the final positions can also be computed approximately using kinematic model by the Jacobian matrix as $\mathbf{p}'_{fin} = \mathbf{p}_{init} + \mathbf{J}(\mathbf{q})\boldsymbol{\nu}'$. Note that $\boldsymbol{\nu}'$ is obtained from $\delta\mathbf{q}$, in which the transformation from the quaternion derivative $\delta\mathfrak{q}_p$ to the angular velocity ${}^p\boldsymbol{\omega}_p$ is given by the relationship detailed in (A.83) (see Appendix A.3). During this test, $\mathbf{q}$ was initialised by the values given in (2.94) and several set of random values on $\delta\mathbf{q}$ were generated by a normal distribution $\mathcal{N} \sim (0, 0.001)$. The results on the norm of the difference $\mathbf{p}_{fin} - \mathbf{p}'_{fin}$ were all limited within 1e-5.

All the numerical validations have been successfully passed, allowing the validation of the modelling algorithms presented in this chapter.

2.7 Conclusion

In this chapter, a specific design of the Fling Parallel Robot is presented, in which a number of multirotors support collectively a moving platform with passive rigid legs. The robot coordinates vector is composed of the 6-dimensional platform pose and the additional degrees of freedom of the revolute joints by which the legs are attached to the platform. The geometric, kinematics and dynamics of this particular FPR are detailed, allowing to respectively compute the internal configuration (i.e. leg angles) of the robot, relate the actuators' velocities (i.e. linear velocities of multirotors) to the generalised velocity of the robot, and develop the dynamic model written in generic forms using several numerical algorithms. All the models have been numerically validated to ensure the correctness of the modelling and programming progress. The models derived in this chapter are used to construct the model-based estimation and control algorithms which will be discussed in the next chapters.

Chapter 3

Wrench Estimation and Interaction Control

This chapter presents the external wrench estimation and interaction control techniques applied to the Flying Parallel Robot (FPR) for physically interacting with the environment. An introduction to the robot-environment interaction of aerial robots, as well as a literature review on different approaches, are given in the first section. Then the momentum-based observers are presented in the second section, which can be used to construct an estimation technique of the external wrench acting on the robot. The third section presents an impedance-based interaction control method with the desired wrench tracking capacity, including a high-level impedance controller of the FPR and low-level multirotors' attitude controllers. A discussion of the overall estimation and control framework is given in the fourth section. Extensive experimental results are finally presented in the fifth section, validating the presented estimation and control techniques for cancelling modelling uncertainties, rejecting disturbances and accomplishing contact-based interaction tasks.

3.1 Introduction

As a potential aerial manipulator dedicated to industrial applications, it is important to show or enhance the ability of the Flying Parallel Robot in conducting real-world tasks, such as pushing/pulling an object, twisting on a surface, etc. The topic of the FPR physically interacting with the environment must be investigated. It is obviously a challenging topic especially for aerial robots as they can easily suffer from aerodynamic disturbances and uncertainties. Unlike the classical robot manipulators, flying robots are unstable and dynamically faster systems, which makes it more difficult to preserve the stability of the system when dealing with external disturbances or even interacting actively with the environment. A such problem commonly encountered by single UAVs or more sophisticated aerial manipulators is generally addressed by two classes of approaches in the literature:

- via **robust control** algorithms such as [Islam, 2015; Nguyen, 2018; Zhang, 2019; Sanalitro, 2020; Chen, 2020];

- ❖ by **estimation technique** to construct the external effects as in [Bellens, 2012; Ruggiero, 2014; Ruggiero, 2015b; Tomić, 2017; Bodie, 2019].

In the first strategy, it is common to assume that the unknown external disturbances are bounded, which can be overcome by a robust controller with Lyapunov-based stability. However, this kind of control strategy might not be adequate to handle significant robot-environment interactions, during which the external forces or moments might easily exceed the presumed limits, nor applicable to the scenarios where it is expected to control the interaction behaviour of the robot. In contrast, using the estimation techniques to construct knowledge of the external wrench acting on the aerial robot, a reaction behaviour of the robot can then be designed and thus controlled. This can usually be achieved by force control strategies [Bruno, 2000], including hybrid force/position control applied in [Bellens, 2012; Marconi, 2012; Scholten, 2013; Nguyen, 2013; Tzoumanikas, 2020], admittance control such as [Barawkar, 2017; Ryll, 2019; Smrcka, 2021], and impedance control that can be found in [Forte, 2012; Ruggiero, 2014; Cataldi, 2016; Car, 2018; Bodie, 2019].

In hybrid force/position control, a separated control paradigm is selected, in which a set of robot's degrees of freedom are controlled in position while the others are selected to be force controlled. This often requires prior knowledge of the environment model and a properly defined position/force trajectory, which is however not easy to be satisfied. Admittance and impedance controllers are dual approaches to dynamic control relating the position and force. They are often used in applications where a manipulator interacts with the environment and the position/force relation is of concern. While an admittance control law defines the motions that result from a force input, impedance control considers the ratio of force output to motion input. Like the hybrid position/force control, the admittance control additionally requires a considerably precise position (or motion) loop to achieve the overall control. This is however not evident for multi-UAV parallel robots such as the FPR, as the accuracy of the motion controller is extremely affected by the aerodynamic disturbances and any degenerated motion tracking will cause the controller failure. Therefore, impedance control seems to be a more preferable method among force control strategies, because the motion control loop is entirely replaced by a regulation of the desired robot interaction behaviour as a virtual impedance system. The impedance-controlled aerial robots reacting passively to the external effects would therefore be more robust, especially for significant robot-environment interactions.

As mentioned beforehand, the FPR under impedance control enhanced by an estimation of the external wrench using momentum-based observers might be more appropriate for cancelling the modelling errors, rejecting disturbances and potentially interacting with the environment, which will be presented in the next sections.

3.2 External Wrench Estimation

In this section, an estimation technique based on momentum computation is presented, which is capable of estimating the wrench exerted on the robot due to the external effects without the need of knowing the acceleration of the robot as mainly required using acceleration-based techniques (applying directly the IDM). The external effects might consist of modelling errors, disturbances and potential interaction with the environment. The momentum-based observers show its efficiency in estimating all these effects as the external wrench acting on the full DoFs of the robot by a mismatch between the predicted and actual dynamics of the robot.

In the following subsections, the principle of the momentum-based approach is firstly presented, which allows for designing external wrench observers using the momentum dynamics. Three observers applying different techniques are then presented, including:

- First-Order Wrench Observer (FOWO);
- Second-Order Wrench Observer (SOWO);
- Sliding-Mode Wrench/Momentum Observer (SMWO).

As mentioned above, the external wrench acting on the robot can be estimated by these momentum-based observers using accessible measurements of the robot states, without the requirement of any acceleration measurements which might potentially be too noisy to perform a good estimation. The discussion on the momentum-based approach for estimating the external effects will be given at the end of this section.

3.2.1 Principle of Momentum-based Approach

The momentum-based estimation technique is constructed on the basis of the robot's dynamic model as the one presented in (2.26). An additional term $\boldsymbol{\tau}_e$ corresponding to the external wrench is furthermore taken into account in the dynamic model as

$$\mathbf{M}(\mathbf{q})\dot{\boldsymbol{\nu}} + \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} + \boldsymbol{\tau}_e \tag{3.1}$$

where $\boldsymbol{\tau}_e$ is composed of

$$\boldsymbol{\tau}_e = \begin{bmatrix} \mathbf{f}_{p,e} \\ {}^{p}\mathbf{m}_{p,e} \\ \mathbf{m}_{l,e} \end{bmatrix} \in \mathbb{R}^{6+n} \tag{3.2}$$

with $\mathbf{f}_{p,e}$, ${}^{p}\mathbf{m}_{p,e}$ representing the external force and moment exerted on the platform, expressed in $\mathfrak{F}_0$ and $\mathfrak{F}_p$ respectively, and $\mathbf{m}_{l,e}$ being the external moments acting on the revolute-joint axis of each leg.

Once the dynamic expression is obtained, the momentum of the robot can be computed. Let $\boldsymbol{\mathcal{P}}$ be the generalised momentum of the robot that can be determined by

$$\boldsymbol{\mathcal{P}} = \mathbf{M}(\mathbf{q})\boldsymbol{\nu} \in \mathbb{R}^{6+n} \tag{3.3}$$

where $\mathbf{M}(\mathbf{q})$ is the generalised inertia matrix given in the dynamic model of (3.1), and $\boldsymbol{\nu}$ is the generalised velocity vector. Then the time derivative of the generalised momentum is given by

$$\dot{\boldsymbol{\mathcal{P}}} = \dot{\mathbf{M}}(\mathbf{q})\boldsymbol{\nu} + \mathbf{M}(\mathbf{q})\dot{\boldsymbol{\nu}} \tag{3.4}$$

The skew-symmetric property of the generalised inertia matrix as in (2.65) and [Siciliano, 2010] indicates that

$$\dot{\mathbf{M}}(\mathbf{q}) = \mathbf{C}(\mathbf{q}, \boldsymbol{\nu}) + \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})^T \tag{3.5}$$

Based on equations (3.1) and (3.5), the time derivative of generalised momentum in (3.4) is finally determined by

$$\begin{aligned} \dot{\boldsymbol{\mathcal{P}}} &= \Big(\mathbf{C}(\mathbf{q}, \boldsymbol{\nu}) + \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})^T\Big)\boldsymbol{\nu} + \Big(\boldsymbol{\tau} + \boldsymbol{\tau}_e - \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu} - \mathbf{g}(\mathbf{q})\Big) \\ &= \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})^T\boldsymbol{\nu} - \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau} + \boldsymbol{\tau}_e \end{aligned} \tag{3.6}$$

which can be used to directly compute the external wrench by

$$\boldsymbol{\tau}_e = \dot{\boldsymbol{\mathcal{P}}} - \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})^T\boldsymbol{\nu} + \mathbf{g}(\mathbf{q}) - \boldsymbol{\tau} \tag{3.7}$$

Note that $\boldsymbol{\tau}$ is the actuation wrench exerted by the thrust forces of the multirotors as calculated in (2.28), $\mathbf{C}(\mathbf{q}, \boldsymbol{\nu})$ and $\mathbf{g}(\mathbf{q})$ are the Coriolis matrix and gravity wrench computed from the accessible robot generalised coordinates and velocity vectors (i.e. $\mathbf{q}$ and $\boldsymbol{\nu}$). Therefore, one can reconstruct an estimate of the external wrench exerted on the robot using momentum computation, with only knowledge of robot states (generalised coordinates and velocity) and the commands as the actuation wrench $\boldsymbol{\tau}$ sent to the robot. However, as noises in the measurements of the robot states might affect the computation result of the external wrench term, additional filtering techniques are often applied on the basis of the direct computation in (3.7), which will be presented in the following sections.

3.2.2 First-Order Wrench Observer (FOWO)

As mentioned above, an additional filter should be applied to the computation of the external wrench given in (3.7) to leverage the measurement noises. A first-order filter can be adopted in the first place due to its simplicity. Let $\hat{\boldsymbol{\tau}}_e$ denotes an estimate of the external wrench. By applying a first-order filter, the time-domain expression of the

external wrench estimate can be determined as follows

$$\begin{aligned}\hat{\boldsymbol{\tau}}_e(t) =&\mathbf{K}_O\Big[\boldsymbol{\mathcal{P}}(t) - \int_{t_0}^{t}\Big(\mathbf{C}\big(\mathbf{q}(t),\boldsymbol{\nu}(t)\big)^T\boldsymbol{\nu}(t) - \mathbf{g}\big(\mathbf{q}(t)\big) + \boldsymbol{\tau}(t)\\ &+ \hat{\boldsymbol{\tau}}_e(t-\Delta t)\Big)dt - \boldsymbol{\mathcal{P}}(t_0)\Big]\end{aligned} \tag{3.8}$$

where $\mathbf{K}_O \in \mathbb{R}^{(6+n)\times(6+n)}$ is a positive-definite diagonal matrix as the observer gains, $\hat{\boldsymbol{\tau}}_e(t-\Delta t)$ denotes an external wrench estimate at the previous timestamp with Δt referring to a single time step of the digital computation, $\boldsymbol{\mathcal{P}}(t)$ is the robot's momentum variable at timestamp t with $\boldsymbol{\mathcal{P}}(t_0)$ being the momentum at the initial time which is usually set to zero supposing $\boldsymbol{\nu}(t_0) = \mathbf{0}$, $\boldsymbol{\tau}(t)$ refers to the actuation wrench sent to the robot at the current time which can be calculated using relation of (2.28). Remark that the current estimate is computed by (3.8) based on the previous estimate and the terms computed by the dynamic model.

By differentiating (3.8) with respect to time, the observer dynamics can be obtained from (3.6) as

$$\dot{\hat{\boldsymbol{\tau}}}_e(t) + \mathbf{K}_O\big(\hat{\boldsymbol{\tau}}_e(t-\Delta t) - \boldsymbol{\tau}_e\big) = \mathbf{0} \tag{3.9}$$

Performing furthermore an approximation between two consecutive timestamps (sufficiently small time steps between every two estimation iterations), the estimation error dynamics can be finally depicted by the following differential equation

$$\dot{\tilde{\boldsymbol{\tau}}}_e + \mathbf{K}_O\tilde{\boldsymbol{\tau}}_e = \mathbf{0} \tag{3.10}$$

with $\tilde{\boldsymbol{\tau}}_e = \hat{\boldsymbol{\tau}}_e - \boldsymbol{\tau}_e$ being the estimation error. This relationship can be expressed element-wise in the Laplace domain by

$$\mathcal{G}_j(s) = \frac{\hat{\tau}_{e,j}(s)}{\tau_{e,j}(s)} = \frac{K_{O,j}}{s + K_{O,j}}, \quad j = 1, \dots, N \tag{3.11}$$

with N (equal to $6+n$) decoupled transfer functions $\mathcal{G}_j(s)$ for each element of the external wrench estimates. In ideal conditions, the following relation holds [De Luca, 2006]

$$\mathbf{K}_O \to \infty \quad \Rightarrow \quad \tilde{\boldsymbol{\tau}}_e \approx \mathbf{0} \quad \Rightarrow \quad \hat{\boldsymbol{\tau}}_e \approx \boldsymbol{\tau}_e \tag{3.12}$$

which means that the estimation gains should be taken as large as possible in the practice. However, larger values for $\mathbf{K}_O$ also result in higher noise amplification in the wrench estimates which might deteriorate the observer performance in terms of sensitivity, and the limitation given by the digital implementation should also be taken into account. Therefore, a good compromise has to be reached to properly tune the estimation gains.

3.2.3 Second-Order Wrench Observer (SOWO)

Based on the direct computation of the external wrench by (3.7), a second-order filter can also be applied, which might better weaken the effects of high-frequency noises and thus have the performance superior to that of the first-order observer as a simple low-pass filter. Using the second-order filtering technique, the estimated external wrench in time domain can be given by

$$\begin{aligned}\hat{\boldsymbol{\tau}}_e(t) = \mathbf{K}_{O_1}\Bigg[\int_{t_0}^{t} -\hat{\boldsymbol{\tau}}_e(t-\Delta t) + \mathbf{K}_{O_2}\Big[\boldsymbol{\mathcal{P}}(t) - \int_{t_0}^{t}\Big(\mathbf{C}\Big(\mathbf{q}(t),\boldsymbol{\nu}(t)\Big)^T\boldsymbol{\nu}(t) \\ - \mathbf{g}\Big(\mathbf{q}(t)\Big) + \boldsymbol{\tau}(t) + \hat{\boldsymbol{\tau}}_e(t-\Delta t)\Big)dt\Big]dt\Bigg]\end{aligned} \tag{3.13}$$

supposing that the momentum and the external wrench at initial timestamp are all zero. $\mathbf{K}_{O_1}$, $\mathbf{K}_{O_2} \in \mathbb{R}^{(6+n)\times(6+n)}$ are the positive-definite diagonal matrices for the second-order observer gains.

By differentiating two times (3.13) with respect to time and neglecting the time difference between any two consecutive timestamps (i.e. $(t-\Delta t)$ and t) supposing Δt is arbitrarily small, the estimation error dynamics can be obtained as

$$\ddot{\tilde{\boldsymbol{\tau}}}_e + \mathbf{D}_O\dot{\tilde{\boldsymbol{\tau}}}_e + \mathbf{K}_O\tilde{\boldsymbol{\tau}}_e = \mathbf{0} \tag{3.14}$$

with $\tilde{\boldsymbol{\tau}}_e = \hat{\boldsymbol{\tau}}_e - \boldsymbol{\tau}_e$ being the estimation error and the gains related to the estimation gains defined in (3.13) as

$$\begin{cases}\mathbf{D}_O = \mathbf{K}_{O_1} \\ \mathbf{K}_O = \mathbf{K}_{O_1}\mathbf{K}_{O_2}\end{cases} \tag{3.15}$$

Similarly, (3.14) can be written in the Laplace domain with element-wise transfer functions defined as

$$\mathcal{G}_j(s) = \frac{K_{O,j}}{s^2 + D_{O,j}s + K_{O,j}}, \quad j = 1, \dots, N \tag{3.16}$$

From the definition of a second-order transfer function, the unitary gains for each element can be determined by

$$\begin{cases}D_{O,j} = K_{O_1,j} = 2\zeta_j\omega_{n,j} \\ K_{O,j} = K_{O_1,j}K_{O_2,j} = \omega_{n,j}^2\end{cases} \tag{3.17}$$

with $\omega_{n,j}$ and ζ_j being respectively the natural frequency and damping factor of the system decoupled on each element. In an ideal case, notice that [Ruggiero, 2014]

$$\begin{cases}\zeta_j \to 1 \\ \omega_{n,j} \to \infty\end{cases} \quad \Rightarrow \quad \tilde{\boldsymbol{\tau}}_e \approx \mathbf{0} \quad \Rightarrow \quad \hat{\boldsymbol{\tau}}_e \approx \boldsymbol{\tau}_e \tag{3.18}$$

which means that the gains in second-order wrench observer should also be taken as large as possible. Due to the same issues considering the sensibility and real-world implementation of the wrench observer, the values of these gains should be properly tuned.

3.2.4 Sliding-Mode Wrench/Momentum Observer (SMWO)

The momentum-based wrench estimator can be further enhanced by the sliding mode technique according to the expression of $\dot{\boldsymbol{\mathcal{P}}}$ in (3.6), which is reformulated to separate the accessible and unknown terms as

$$\mathbf{C}(\mathbf{q},\boldsymbol{\nu})^T\boldsymbol{\nu} - \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau} = \dot{\boldsymbol{\mathcal{P}}} - \boldsymbol{\tau}_e \tag{3.19}$$

where the accessible left-side terms can be used in place of unknown right-side terms to design an observer that provides an estimation of both the generalised momentum and the external wrench in finite time [Garofalo, 2019]. In fact, applying the sliding mode technique, an estimation on the generalised momentum is firstly constructed and then the external wrench is estimated by the mismatch between the estimated momentum dynamics and the real one, which is consequently performed in finite time. The observer structure can be designed according to the Super Twist Algorithm (STA) [Moreno, 2008] for observation using second-order sliding modes as follows

$$\begin{aligned}\dot{\hat{\boldsymbol{\mathcal{P}}}} &= \mathbf{C}(\mathbf{q},\boldsymbol{\nu})^T\boldsymbol{\nu} - \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau} - \mathbf{K}_{O_T}\|\tilde{\boldsymbol{\mathcal{P}}}\|^{\frac{1}{2}}\ \mathtt{sgn}^*(\tilde{\boldsymbol{\mathcal{P}}}) + \boldsymbol{\sigma}\\ \dot{\boldsymbol{\sigma}} &= -\mathbf{K}_{O_S}\ \mathtt{sgn}^*(\tilde{\boldsymbol{\mathcal{P}}})\end{aligned} \tag{3.20}$$

where $\tilde{\boldsymbol{\mathcal{P}}} = \hat{\boldsymbol{\mathcal{P}}} - \boldsymbol{\mathcal{P}}$ is the momentum estimation error, with $\hat{\boldsymbol{\mathcal{P}}}$ being the estimated momentum, and $\mathbf{K}_{O_S}$, $\mathbf{K}_{O_T} \in \mathbb{R}^{(6+n)\times(6+n)}$ are positive-definite diagonal matrices for the observer gains. Note that all the operators in (3.20) should be applied element-wise. $\mathtt{sgn}^*(.)$ denotes an adapted sign function given by (3.21), which has continuous output to avoid chattering issues that might occur using the sliding mode technique.

$$\mathtt{sgn}^*(s) = \begin{cases} 1 & \text{if } \|s\| > \epsilon \text{ and } s > 0;\\ \dfrac{s}{\epsilon} & \text{if } \|s\| \le \epsilon;\\ -1 & \text{if } \|s\| > \epsilon \text{ and } s < 0 \end{cases} \tag{3.21}$$

ϵ is an arbitrarily small and non-zero value, which is used to perform a linear approximation to avoid the discontinuity in the output of the sign function when input is near zero.

Using (3.4) and defining $\mathbf{s} = \boldsymbol{\sigma} - \boldsymbol{\tau}_e$, the error dynamics of the sliding mode observer

can be obtained by

$$\begin{aligned}\dot{\tilde{\boldsymbol{\mathcal{P}}}} &= -\mathbf{K}_{O_T}\|\tilde{\boldsymbol{\mathcal{P}}}\|^{\frac{1}{2}}\ \mathtt{sgn}^*(\tilde{\boldsymbol{\mathcal{P}}}) + \mathbf{s} \\ \dot{\mathbf{s}} &= -\mathbf{K}_{O_S}\ \mathtt{sgn}^*(\tilde{\boldsymbol{\mathcal{P}}}) - \dot{\boldsymbol{\tau}}_e\end{aligned} \tag{3.22}$$

which refers to N independent and decoupled super twist algorithms (corresponding to the element-wise estimates of the momentum and external wrench). As global finite-time stability of the equilibrium point defined by $(\tilde{\boldsymbol{\mathcal{P}}}, \mathbf{s}) = (\mathbf{0}, \mathbf{0})$ can be guaranteed for the STA being robust [Moreno, 2008; Garofalo, 2019], $\boldsymbol{\sigma}$ is an estimation of the external wrench $\boldsymbol{\tau}_e$ in finite time. For a successful implementation and application of such sliding mode wrench observer, it is important to properly tune the observer gains $\mathbf{K}_{O_S}$ and $\mathbf{K}_{O_T}$ which affect both the stability and the dynamic behaviour of the observer [Garofalo, 2019]. To obtain an intuition on tuning the observer gains, the error dynamics in (3.22) can be rewritten as a second-order differential equation

$$\dot{\tilde{\boldsymbol{\mathcal{P}}}} = -\mathbf{K}_{O_T}\|\tilde{\boldsymbol{\mathcal{P}}}\|^{\frac{1}{2}}\ \mathtt{sgn}^*(\tilde{\boldsymbol{\mathcal{P}}}) - \mathbf{K}_{O_S}\int_{t_0}^{t}\mathtt{sgn}^*(\tilde{\boldsymbol{\mathcal{P}}})dt - \boldsymbol{\tau}_e \tag{3.23}$$

where the observer gain $\mathbf{K}_{O_T}$ can be interpreted as equivalent to the proportional gain, while $\mathbf{K}_{O_S}$ is equivalent to the integral gain. Therefore, too high values for $\mathbf{K}_{O_T}$ will lead to overshoots, and too small values result in poor convergence of the estimation. In addition, values for $\mathbf{K}_{O_S}$ are tuned to reduce the static remaining errors. Empirically, the desired behaviour of the sliding mode observer is obtained by tuning the observer gains with initial values taking the relation of $K_{O_T,j} = 1.6\sqrt{K_{O_S,j}}$ $(j = 1, ..., N)$. Then the estimated external wrench can be written in the time domain by

$$\begin{aligned}\hat{\boldsymbol{\mathcal{P}}}(t) &= \int_{t_0}^{t}\Big(\mathbf{C}\big(\mathbf{q}(t), \boldsymbol{\nu}(t)\big)^T\boldsymbol{\nu}(t) - \mathbf{g}\big(\mathbf{q}(t)\big) + \boldsymbol{\tau}(t) \\ &\qquad - \mathbf{K}_{O_T}\|\tilde{\boldsymbol{\mathcal{P}}}(t-\Delta t)\|^{\frac{1}{2}}\ \mathtt{sgn}^*\big(\tilde{\boldsymbol{\mathcal{P}}}(t-\Delta t)\big) + \hat{\boldsymbol{\tau}}_e(t-\Delta t)\Big)dt \\ \tilde{\boldsymbol{\mathcal{P}}}(t) &= \hat{\boldsymbol{\mathcal{P}}}(t) - \boldsymbol{\mathcal{P}}(t) \\ \hat{\boldsymbol{\tau}}_e(t) &= \int_{t_0}^{t} -\mathbf{K}_{O_S}\ \mathtt{sgn}^*\big(\tilde{\boldsymbol{\mathcal{P}}}(t)\big)dt\end{aligned} \tag{3.24}$$

where $\tilde{\boldsymbol{\mathcal{P}}}(t-\Delta t)$, $\tilde{\boldsymbol{\mathcal{P}}}(t)$ and $\hat{\boldsymbol{\tau}}_e(t-\Delta t)$, $\hat{\boldsymbol{\tau}}_e(t)$ are respectively the momentum estimation error and the estimated external wrench at the previous and current timestamp. Note that for all quantities, the initial values are assumed to be zero such that the constant terms of the integration in (3.24) can be omitted.

3.2.5 Discussion on Wrench Observers

The above-mentioned momentum-based wrench observers are derived on the basis of the dynamic model depicted by (3.1) and its property, which are capable of estimating the

unknown external wrench with only knowledge of the accessible robot states, i.e. robot configuration $\mathbf{q}$ and velocity $\boldsymbol{\nu}$. In terms of dynamic modelling, the Spatial Vector algorithms presented in Section 2.5.3 can be adopted, allowing an online computation of generalised inertia matrix and Coriolis matrix needed to construct the estimation. It is worth mentioning that although three types of wrench observers all use the momentum dynamics to obtain an estimation of the external wrench, the ways in which the relation (3.7) is used are different. In first-order and second-order observers, the momentum dynamics is used as an integrand in order to construct a low-pass filter to leverage high-frequency measurement noises, while the states of the sliding mode observer are directly an estimation of the generalised momentum and the external wrench, in which the momentum dynamics is internally considered in constructing the sliding mode.

Additionally, all these observers guarantee theoretically an estimation of the external wrench that converges to $\boldsymbol{\tau}_e$ in finite time. This is performed by the mismatch between the expected robot dynamics and its real behaviour which assumes that the command $\boldsymbol{\tau}$ sent to the robot has been well achieved. Knowing the relationship of (2.28), this means that the thrust force commands $\mathbf{f}$ should be sufficiently well tracked by the multirotors, which however cannot be assured in real case scenarios. Therefore, the estimation of the external wrench will also contain uncertainties on thrust forces generated by the multirotors. In addition to this factor, any other modelling uncertainties, measurement noises and errors will inevitably lead to a noisy and/or biased estimation. As a consequence, the estimated external wrench will not be exactly zero even when no external effects are present. A threshold has to be empirically set to avoid false positives if the observers are used to detect the contacts or collisions.

3.3 Impedance-based Interaction Control

After having discussed the external wrench estimation based on momentum computation, the next step is to investigate the interaction control algorithm for interacting with the environment. As discussed in Section 3.1, the impedance control method is an indirect force control algorithm which regulates passively the robot's behaviour with respect to the external wrench. The desired behaviour is determined by a virtual impedance system, i.e. a mass-spring-damper system, taking the wrench exerted on the environment by the robot as the system's input. This control paradigm is particularly suitable for flying robots which can easily be suffered from aerodynamic uncertainties and disturbances, since the impedance controller can be applied to both reject the disturbance and control the interaction behaviour, making the robot itself robust enough for performing the interaction tasks. Additionally, as the motion control loop is totally replaced by the impedance

law, the controller can handle both free-space flights and potential interaction tasks without the need of switching controllers as required in most of other force control approaches.

In this section, an impedance-based control of the flying parallel robot with wrench tracking capacity is presented. The high-level control law is firstly computed by designing the desired impedance behaviour of the robot. Then the desired thrust magnitude and attitude setpoints of each multirotor are derived in order to properly actuate the system. The low-level multirotor's attitude control is then discussed, which is based on a proportional regulation of the attitude and PID regulation of the angular rate. Additional discussions on the overall control structure and controller tuning will be given in the last subsection.

3.3.1 High-level Impedance-based Controller

Recall that $\mathbf{q}$, $\boldsymbol{\nu}$ and $\dot{\boldsymbol{\nu}}$ denote as the robot's generalised position, velocity and acceleration vectors, defined in Section 2.2. Let $\mathbf{q}^d$, $\boldsymbol{\nu}^d$ and $\dot{\boldsymbol{\nu}}^d$ be the desired position, desired velocity and desired acceleration of the robot, which can be defined by an off-line trajectory generation process usually based on an interpolation technique with a list of prescribed waypoints. A generalised form of the desired impedance behaviour of the robot interacting with the environment can be chosen as

$$\mathbf{M}^d(\dot{\boldsymbol{\nu}}^d - \dot{\boldsymbol{\nu}}) + \mathbf{B}^d(\boldsymbol{\nu}^d - \boldsymbol{\nu}) + \mathbf{K}^d\boldsymbol{\varepsilon}_q(\mathbf{q}^d, \mathbf{q}) = \boldsymbol{\varepsilon}_\tau \tag{3.25}$$

where $\mathbf{M}^d$, $\mathbf{B}^d$ and $\mathbf{K}^d \in \mathbb{R}^{(6+n)\times(6+n)}$ are the positive-definite diagonal matrices respectively for the desired mass, damping and stiffness coefficients of the system, which are decoupled on each robot state. The input of the desired impedance system is defined to be the wrench tracking error $\boldsymbol{\varepsilon}_\tau$ as in [Seraji, 1997], which is calculated by

$$\boldsymbol{\varepsilon}_\tau = -\hat{\boldsymbol{\tau}}_e - \boldsymbol{\tau}_e^d \in \mathbb{R}^{6+n} \tag{3.26}$$

with $\hat{\boldsymbol{\tau}}_e$ being the estimated external wrench that can be constructed by the momentum-based observers discussed in Section 3.2, and $\boldsymbol{\tau}_e^d$ representing the desired interaction wrench the robot is expected to exert on the environment. Note that $-\hat{\boldsymbol{\tau}}_e$ refers to the estimated interaction wrench the robot exerted on the environment. $\boldsymbol{\varepsilon}_q(\mathbf{q}^d, \mathbf{q})$ is the tracking error of the robot configuration, which takes the form of

$$\boldsymbol{\varepsilon}_q(\mathbf{q}^d, \mathbf{q}) = \begin{bmatrix} \mathbf{p}_p^d - \mathbf{p}_p \\ \boldsymbol{\varepsilon}_o(\mathbf{q}_p^d, \mathbf{q}_p) \\ \boldsymbol{\theta}_l^d - \boldsymbol{\theta}_l \end{bmatrix} \in \mathbb{R}^{6+n} \tag{3.27}$$

with $\boldsymbol{\varepsilon}_o(\mathfrak{q}_p^d, \mathfrak{q}_p) \in \mathbb{R}^3$ being the platform orientation error computed by the quaternion error between the desired and actual orientations [Brescianini, 2013] as

$$\begin{aligned} \boldsymbol{\varepsilon}_o(\mathfrak{q}_p^d, \mathfrak{q}_p) &= \mathtt{sgn}(\mathfrak{q}_{p,0}^{err})\mathfrak{q}_{p,1:3}^{err} \\ \mathfrak{q}_p^{err} &= \mathfrak{q}_p^{-1} \circ \mathfrak{q}_p^d \end{aligned} \tag{3.28}$$

with $\circ$ being the quaternion multiplication as detailed in (A.48) of Appendix A.1, $\mathfrak{q}_0$ and $\mathfrak{q}_{1:3}$ are respectively the scalar part and the vector part of a quaternion. $\mathtt{sgn}(\mathfrak{q}_{p,0}^{err})$ being the classical sign function defined as

$$\mathtt{sgn}(\mathfrak{q}_0^{err}) = \begin{cases} 1, & \mathfrak{q}_0^{err} \geq 0 \ ; \\ -1, & \mathfrak{q}_0^{err} < 0 \ . \end{cases} \tag{3.29}$$

Based on the desired impedance in (3.25), an auxiliary input corresponding to the acceleration of the system can be determined by

$$\mathbf{u} = \dot{\boldsymbol{\nu}} = \dot{\boldsymbol{\nu}}^d + (\mathbf{M}^d)^{-1}\Big(\mathbf{B}^d(\boldsymbol{\nu}^d - \boldsymbol{\nu}) + \mathbf{K}^d\boldsymbol{\varepsilon}_q(\mathbf{q}^d, \mathbf{q}) - \boldsymbol{\varepsilon}_\tau\Big) \tag{3.30}$$

The control input of the high-level impedance controller can then be calculated by feedback linearization using the dynamic model of (3.1) as

$$\begin{aligned} \boldsymbol{\tau} &= \mathbf{M}(\mathbf{q})\mathbf{u} + \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\mathbf{q}) - \hat{\boldsymbol{\tau}}_e \\ &= \mathbf{M}(\mathbf{q})\dot{\boldsymbol{\nu}}^d + \mathbf{M}(\mathbf{q})(\mathbf{M}^d)^{-1}\Big(\mathbf{B}^d(\boldsymbol{\nu}^d - \boldsymbol{\nu}) + \mathbf{K}^d\boldsymbol{\varepsilon}_q(\mathbf{q}^d, \mathbf{q}) - \boldsymbol{\varepsilon}_\tau\Big) \\ &\quad + \mathbf{C}(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\mathbf{q}) - \hat{\boldsymbol{\tau}}_e \end{aligned} \tag{3.31}$$

which is the actuation wrench that should be applied to the FPR to properly achieve the desired impedance behaviour of the system. By taking the relationship defined in (2.28), the required thrust forces of the multirotors actuating the robot can be calculated by

$$\mathbf{f} = \Big[\mathbf{J}(\mathbf{q})^T\Big]^\dagger \boldsymbol{\tau} \tag{3.32}$$

with $\mathbf{J}(\mathbf{q})$ being the Jacobian matrix derived in (2.20) and $\Big[\mathbf{J}(\mathbf{q})^T\Big]^\dagger$ denoting the pseudo-inverse of the transpose of the Jacobian matrix. In case where the number of legs $n = 3$, it can be simplified by $\mathbf{J}(\mathbf{q})^{-T}$ as the inverse of $\mathbf{J}(\mathbf{q})^T$. However, one should assume that the Jacobian matrix is invertible, i.e. $\det(\mathbf{J}) \neq 0$. When $\det(\mathbf{J}) = 0$, it means that the specific robot configuration is singular and the determination of required thrust forces is impossible under such configuration. According to the singularity analysis conducted in [Six, 2018b], the singularity-free configuration can be assured when the leg angles are all restricted within the range of $[0, \frac{\pi}{2}]$.

The stability of the impedance controller with external wrench estimates can be proven by considering the stability of perturbed systems studied in [Khalil, 2002a]. The error dynamics applying the control law of (3.31) can be written as

$$\mathbf{M}(\dot{\boldsymbol{\nu}} - \dot{\boldsymbol{\nu}}^d) + \mathbf{M}_v\mathbf{B}^d(\boldsymbol{\nu} - \boldsymbol{\nu}^d) + \mathbf{M}_v\mathbf{K}^d\boldsymbol{\varepsilon}_q(\mathbf{q}, \mathbf{q}^d) + \mathbf{M}_v\boldsymbol{\varepsilon}_\tau = -\tilde{\boldsymbol{\tau}}_e \tag{3.33}$$

with $\tilde{\boldsymbol{\tau}}_e = \hat{\boldsymbol{\tau}}_e - \boldsymbol{\tau}_e$ recalled as the estimation error and $\mathbf{M}_v = \mathbf{M}(\mathbf{M}^d)^{-1}$. In case of perfect compensation of external wrench, i.e. $\tilde{\boldsymbol{\tau}}_e = \mathbf{0}$, it can be proven that a global asymptotically stable equilibrium point is achieved by taking the following conditions [Khalil, 2002a]

$$\boldsymbol{\varepsilon}_q(\mathbf{q}, \mathbf{q}^d) = \mathbf{0}, \;\; \boldsymbol{\nu} - \boldsymbol{\nu}^d = \mathbf{0}, \;\; \dot{\boldsymbol{\nu}} - \dot{\boldsymbol{\nu}}^d = \mathbf{0}, \;\; \boldsymbol{\varepsilon}_\tau = \mathbf{0} \; . \tag{3.34}$$

In most cases where the condition of $\tilde{\boldsymbol{\tau}}_e = \mathbf{0}$ is not necessarily ensured, the stability proof of the system has to be accomplished by taking into account non-vanishing perturbations, in which a stable equilibrium point of the system is no longer existing. However, with a reasonable hypothesis of the perturbation term being bounded, i.e.

$$\|\boldsymbol{\tau}_e - \hat{\boldsymbol{\tau}}_e\| \leq \delta \tag{3.35}$$

with δ an arbitrarily small and positive value, it is possible to prove that the nominal system has an exponentially stable equilibrium at the origin (that of conditions in (3.34)) because it shows that arbitrarily small and uniformly bounded perturbations will not result in large steady-state deviations of the system from the origin [Khalil, 2002a]. In other words, the stability of the impedance control law of (3.31) with external wrench estimation can be ensured if the estimation error, i.e. $\boldsymbol{\tau}_e - \hat{\boldsymbol{\tau}}_e$, is bounded. Furthermore, the time-scale separation argument commonly found in several robotics applications is exploited in dealing with feedback control with estimation. The estimator dynamics are supposed fast enough such that its transient behaviour can be considered as a bounded perturbation with respect to the robot motion, which further proves the assumption of bounded perturbation term in (3.35) to the system.

3.3.2 Computation of Multirotor Thrust/Attitude Setpoints

Based on $\mathbf{f}_i$ resulted from (3.32) corresponding to the multirotor i's thrust force vector in 3-dimensional space, the desired thrust magnitude and desired attitude of multirotor i can be determined. As the classical multirotor (with untilted rotors) can only provide thrust force vertically in its body frame, the total thrust magnitude and the z axis of the

multirotor's body frame $\mathfrak{F}_{bi}$ for the desired attitude can be systematically defined by

$$f_{t,i}^d = \|\mathbf{f}_i\|, \quad \mathbf{z}_i^d = \frac{\mathbf{f}_i}{\|\mathbf{f}_i\|} \tag{3.36}$$

Any orientation satisfying $\mathbf{z}_i^d$ with an arbitrary yaw around $\mathbf{z}_i^d$ can be chosen to provide the correct desired thrust force $\mathbf{f}_i$. To fully determine the desired attitude, a desired yaw can be chosen by defining the desired x axis (or y axis) of the body frame. A constant angle β_i (as presented in Section 2.3.3) between the leg's direction and the x axis of the multirotor's frame can be chosen, for the purpose of avoiding potential collisions between the rigid legs and the propellers. The definition of the desired x axis can be done in two steps:

1. defining firstly an auxiliary axis $\mathbf{x}_i'^d$ which is perpendicular to both leg's direction $\overrightarrow{A_iB_i}$ (with its unit vector expressed by $\mathbf{l}_i$) and the axis $\mathbf{z}_i^d$;
2. then rotating an angle of $(\beta - \frac{\pi}{2})$ around the axis $\mathbf{z}_i^d$.

The computation of the desired x axis of each multirotor's body frame is shown in Fig. 3.1 and detailed in (3.37).

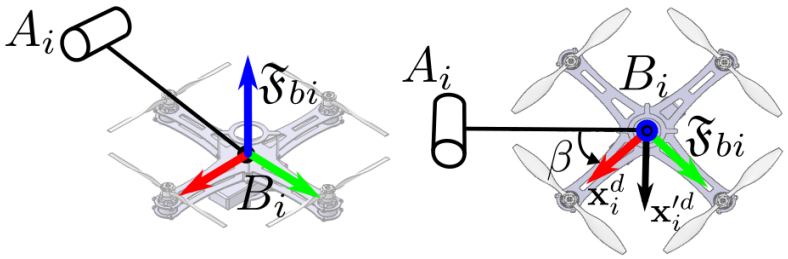


Figure 3.1 – Definition of the desired x axis of the multirotor's body frame.

$$\begin{aligned} \mathbf{x}_i'^d &= \mathbf{l}_i \times \mathbf{z}_i^d, \\ \mathbf{x}_i^d &= \texttt{AxisAngle}(\mathbf{z}_i^d, \beta - \frac{\pi}{2})\ \mathbf{x}_i'^d \end{aligned} \tag{3.37}$$

with $\mathbf{u}_i = \frac{\overrightarrow{A_iB_i}}{\|\overrightarrow{A_iB_i}\|}$ and $\overrightarrow{A_iB_i}$ computed via the relation of (2.12). Note that `AxisAngle`(**axis**, *angle*) refers to the axis-angle representation of an orientation (see Appendix A.1). Finally, the rotation matrix associated with the desired attitude of the multirotor i can be fully determined as

$$\mathbf{R}_i^d = \begin{bmatrix} \mathbf{x}_i^d & \mathbf{z}_i^d \times \mathbf{x}_i^d & \mathbf{z}_i^d \end{bmatrix} \tag{3.38}$$

which is then converted to the unit quaternion $\mathbf{q}_i^d$ corresponding to the desired attitude of the multirotor i.

3.3.3 Low-level Multirotor Attitude Controller

Once the commands of $(f_{t,i}^d, \mathfrak{q}_i^d)$ are obtained corresponding to multirotor i's desired thrust magnitude and desired attitude, a thrust/attitude controller commonly seen in the literature can be chosen for the low-level multirotor control. For the cases where hexarotors or octorotors are used as the aerial vehicle platforms in the FPR design, the knowledge of desired thrust magnitude and attitude setpoints is not enough to determine the actuator inputs (i.e. motor speeds of the rotors). To solve this, a control allocation or optimisation method is often adopted, satisfying the achievement of thrust/attitude commands and optimising energy consumptions as investigated in [Kotarski, 2018; Werink, 2019; Liu, 2013]. In the works conducted within this thesis, the quadrotors are chosen to actuate the robot with no over-actuation needed to be handled as it is sufficient to fully determine the four motor inputs given the desired thrust magnitude and desired attitude of a quadrotor. A quaternion-based attitude control of quadrotor such as [Brescianini, 2013; Fresk, 2013] can then be applied, which is based on the quadrotor's kinematics and detailed as follows.

The objective of the low-level quadrotor control is to design a feedback law which stabilizes the quadrotor at the desired attitude while generating the desired thrust. From the desired and actual attitude of the quadrotor, the angular rate commands can be calculated considering the following control law

$$\boldsymbol{\omega}_i^d(\mathfrak{q}_i) = \frac{2}{\tau}\mathtt{sgn}(\mathfrak{q}_{i,0}^{err})\mathfrak{q}_{i,1:3}^{err} \tag{3.39}$$

where τ is the first-order system time constant corresponding to the attitude (proportional) control gain, $\mathfrak{q}_i^{err} = \mathfrak{q}_i^{-1} \circ \mathfrak{q}_i^d$ is the quaternion error, and $\mathtt{sgn}(\mathfrak{q}_{i,0}^{err})$ is the sign function of the scalar part of the quaternion error, which has been defined in (3.29).

The computation of the desired angular rate commands by (3.39) satisfies $\boldsymbol{\omega}_i^d(\mathfrak{q}_i) = \boldsymbol{\omega}_i^d(-\mathfrak{q}_i)$, required to solve undesired unwinding phenomena when using quaternion-based controllers as $\mathfrak{q}_i$ and $-\mathfrak{q}_i$ represent the same physical attitude [Mayhew, 2011]. It is evident since any physical attitude in $SO(3)$ corresponds to two antipode quaternions, i.e. $\mathfrak{q}_i$ and $-\mathfrak{q}_i$, and the controller outputs must therefore be consistent in these two equivalent attitude inputs. Then a PID-based controller can be designed to achieve the angular rate commands, as

$$\mathbf{m}_i^d = \mathbf{K}_P\boldsymbol{\varepsilon}_\omega + \mathbf{K}_D\dot{\boldsymbol{\varepsilon}}_\omega + \mathbf{K}_I\int_0^t \boldsymbol{\varepsilon}_\omega dt \tag{3.40}$$

with $\boldsymbol{\varepsilon}_\omega = \boldsymbol{\omega}_i^d - \boldsymbol{\omega}_i$ being the error between the angular rate setpoints and the measured values, $\mathbf{K}_P$, $\mathbf{K}_D$ and $\mathbf{K}_I$ are the positive-definitive diagonal matrices for the PID gains.

It is additionally noted that the angular rate commands for each multirotor (i.e. $\boldsymbol{\omega}_i^d$) should theoretically be depending on the rates of high-level control outputs (i.e. the rate of thrust commands computed from the high-level loop). This term or even higher-order terms affecting the low-level control are however neglected, supposing that the desired attitude setpoints are achieved instantaneously, a hypothesis ensured by the higher control rate and faster convergence of the low-level attitude controller compared to the high-level one for the relatively slow dynamics of the passive architecture.

After having obtained the commands on the quadrotor's thrust magnitude and rotational moments, the rotor speeds can be directly computed without any ambiguity. It is well known that for a quadrotor the thrust force applied to the airframe is the sum of thrusts produced by four rotors, while the reaction moments acting on the airframe are due to the rotor drag forces. Their relations between the squares of the motor speeds for a quadrotor can be written in matrix form [Mahony, 2012]

$$\begin{bmatrix} f_t \\ \mathrm{m}_x \\ \mathrm{m}_y \\ \mathrm{m}_z \end{bmatrix} = \begin{bmatrix} c_T & c_T & c_T & c_T \\ 0 & dc_T & 0 & -dc_T \\ -dc_T & 0 & dc_T & 0 \\ -c_Q & c_Q & -c_Q & c_Q \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \tag{3.41}$$

where Ω_j for $j \in \{1, 2, 3, 4\}$ are the motor speeds, c_T and c_Q are the thrust force and drag force coefficients, and d is the distance from the central axis of the quadrotor to the rotor axis. This constant matrix is then inverted to compute the motor speeds from the commands of $f_{t,i}^d$ and $\mathbf{m}_i^d$, which is usually referred to as the output mixing process.

3.3.4 Discussion on Controller Structure and Tuning

The overall control structure is summarized by the diagram shown in Fig. 3.2. The high-level impedance controller deals with the tracking of the desired trajectory $(\mathbf{q}^d, \boldsymbol{\nu}^d, \dot{\boldsymbol{\nu}}^d)$ as well as the desired interaction wrench $\boldsymbol{\tau}_e^d$, via regulation on the impedance behaviour of the system. The output of the impedance controller, the vector of 3-dimensional thrust forces $\mathbf{f}$, is then used to compute the thrust and attitude setpoints for the low-level multirotor attitude controller, which should be run at a frequency higher than the high-level controller to ensure the fast convergence towards the attitude setpoints.

Regarding the control gains, the multirotor (quadrotor) controller is experimentally tuned with aggressive attitude and angular rate gains to achieve fast convergence and good robustness to the perturbations. This can be accomplished by setting high proportional gains for the attitude control and properly tuning the PID gains for the angular rate

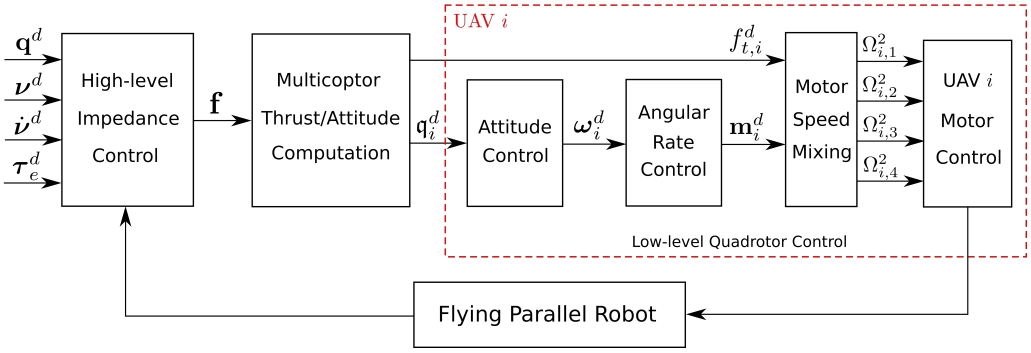


Figure 3.2 – Diagram of the impedance-based interaction control along with the low-level quadrotor control.

control with good behaviour in convergence time and damped oscillations. The impedance gains in (3.25) are determined on the basis of the desired impedance behaviour for the robot-environment interaction, with values being tuned according to the specific task. For instance, a high stiffness can be chosen to reject disturbances during a steady flight, while a lower value will be more appropriate to ensure the compliance behaviour of the robot interacting with the external environment. On the other hand, the mass and damping coefficients must be properly defined to avoid overshooting and oscillations. Furthermore, the damping ratio of the desired impedance system can be taken into account, which is calculated by

$$\zeta = \frac{\mathrm{B}^d}{2\sqrt{\mathrm{K}^d\mathrm{M}^d}} \tag{3.42}$$

The system is under-damped when $\zeta < 1$, critically damped when $\zeta = 1$, and over-damped when $\zeta > 1$. The damping ratio is usually selected between 0.4 and 0.7 in general case [Shinners, 1998].

3.4 Overall Estimation and Control Framework

The overall framework of the external wrench estimation and impedance-based control applied to the FPR is summarized in Fig. 3.3. An off-line trajectory generation procedure is firstly conducted, in which the desired waypoints of the generalised position coordinates $\mathbf{q}^d$ are defined in a list, and the desired velocity and acceleration are calculated generally using the 5-order or 7-order polynomial interpolation techniques [Kermorgant, 2022]. Then the desired wrench can be planned similarly in an off-line process or sent to the FPR online by an operator, providing the possibility of deriving from the tracking of the desired trajectory in case of interacting with the environment to achieve potential manipulation tasks. The determination of the desired wrench $\boldsymbol{\tau}_e^d$ the robot exerts to the environment depends on the specific tasks performed by the robot. In the interaction experiments

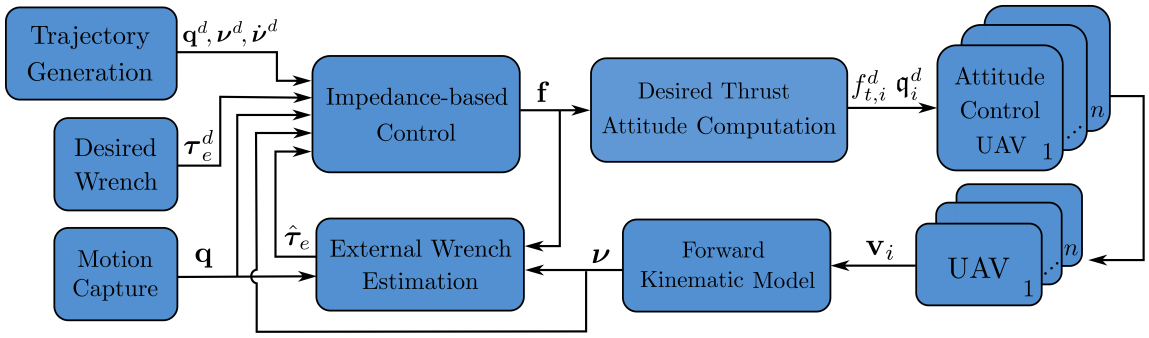


Figure 3.3 – Block diagram of the overall external wrench estimation and impedance-based control framework.

shown in the next section, the FPR is controlled to exert a certain amount of the contact force along the platform's normal direction (i.e. z axis of the platform) while interacting with the environment. The desired force exerted by the platform on the environment is thus chosen to be

$$ {}^p\mathbf{f}_{p,e}^d = \begin{bmatrix} 0 & 0 & f_c^d \end{bmatrix}^T \tag{3.43} $$

with f_c^d representing the desired contact force between the platform and the environment along the normal direction of the platform plane. This desired force is converted to be expressed in $\mathfrak{F}_0$ by multiplying the rotation matrix associated with the platform's actual orientation as

$$ \mathbf{f}_{p,e}^d = \mathbf{R}_p \, {}^p\mathbf{f}_{p,e}^d \tag{3.44} $$

Similarly, the contact force can be estimated by the wrench observers by inverting the rotation matrix as

$$ {}^p\hat{\mathbf{f}}_{p,e} = \mathbf{R}_p^T \hat{\mathbf{f}}_{p,e} \tag{3.45} $$

from which the estimated contact force exerted by the robot is taken as $-{}^p\hat{f}_{pz,e}$ (i.e. negative of the z component of ${}^p\hat{\mathbf{f}}_{p,e}$). The desired and estimated forces on the platform are then used to calculate the force tracking error (i.e. the first three components of $\boldsymbol{\varepsilon}_\tau$). The rest of the elements in $\boldsymbol{\tau}_e^d$ are set to zero during the contact-based interaction experiments, resulting in a compliance behaviour of the robot to the modelling errors and external disturbances acting on those axes.

In terms of state measurements, the generalised position $\mathbf{q}$ of the FPR is measured directly by the Motion Capture (MOCAP) system, while the velocity of the robot $\boldsymbol{\nu}$ is computed using the Forward Kinematics detailed in (2.21) from the measurements of linear velocities onboard the quadrotors by their IMU sensors. These easily accessible measurements of the robot states are taken as input for both external wrench estimation and the impedance-based interaction controller. Remark that for the velocity measure-

ments, the derivatives from MOCAP data are not preferable due to the noise after the numerical differentiation, while the onboard linear velocity measurements of each UAV are usually well estimated by the additional filtering techniques.

3.5 Experimental Validation

In this section, the implementation and experimental validation of the wrench observers and the impedance-based control are presented, for which a video of experiments can be found at https://youtu.be/ryffKG-VG68 and results in [Liu, 2022]. In the FPR prototype applied to the experimental works detailed in Appendix B, the quadrotors are used to actuate the system and the number of the legs (and attached quadrotors) is selected to be 3, which is the minimum number requested to be able to control the full DoFs of the robot. Experiments have been conducted to show the performance of the wrench observers and impedance-based controller applied to the FPR dealing with modelling uncertainties, external disturbances and physical interactions with the environment.

The high-level impedance-based controller as well as the wrench observers are implemented on a ground computer and run at 50 Hz. The commands are sent to each quadrotor by a 5 GHz Wifi network and tracked by the onboard autopilot of attitude control run at 250 Hz. The experiments of the FPR were done in an enclosed $6 \times 4 \times 3.5$ m flight arena equipped with an 8-camera Qualisys Motion Capture (MOCAP) system [Qualisys, 2022]. The MOCAP system can provide the measurements of the poses of all the bodies recorded in the arena, and stream the data over the Wifi network at 100 Hz.

3.5.1 Experiment I: Hovering in Free Space

In the first experiment, the FPR is flown in free space for the purpose of tuning the observer gains and the impedance gains. As the modelling errors can be estimated by the wrench observers, the free-space flight is also performed to analyse the effects of modelling errors and uncertainties. For tuning the wrench observers, the gains are particularly chosen to filter the high-frequency noises, while limiting the sensibility of the observer and the estimation oscillations at the same time. The gains of different wrench observers are determined to have equivalent cut-off frequencies. From the aspect of controller tuning, the impedance gains are chosen experimentally taking into account the damping ratio of the desired impedance system given by (3.42). The values of the impedance gains can be varied according to the specific application scenarios. In the general case, the gains are decoupled along different robot coordinates (i.e. platform position, orientation and leg angles of the FPR). A relatively small damping ratio on the platform position coordinates is selected

to have a rapid response to the external effects during the flight. A slightly under-damped behaviour of the platform's rotational movement is chosen to reach the best trade-off between the response time and the acceptable overshoot on the platform orientation. In contrast, over-damped impedance gains are preferable on the leg angles to limit the overshooting and sensibility to external disturbances, which are beneficial to both avoid the singular configurations and maintain the wrench capability during the interaction with the environment. The pre-tuned gains of the wrench observers and impedance controller for achieving a stable free-space flight are summarized in Table 3.1.

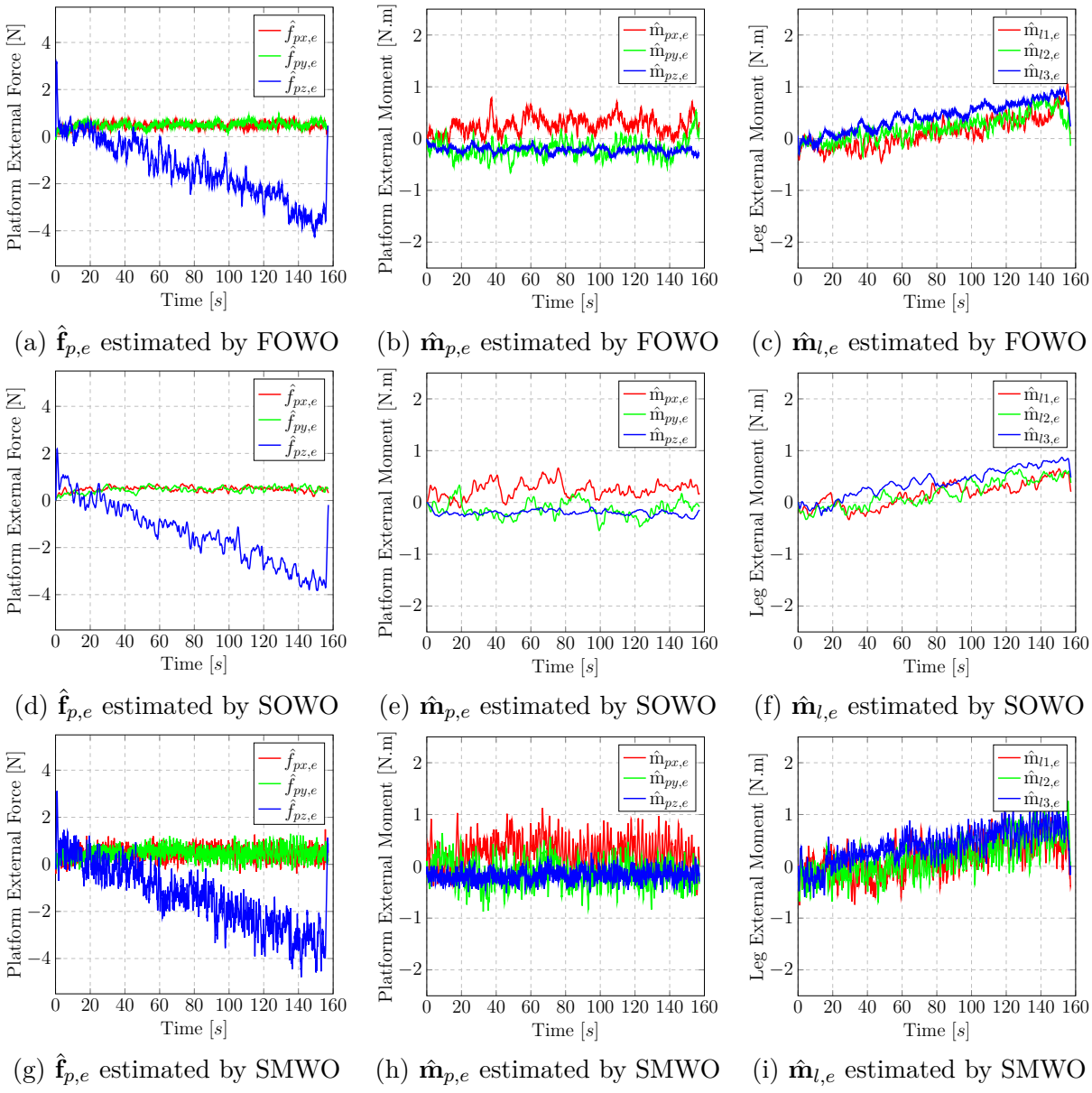
Robot State	Wrench Observer Gains					Impedance Gains			
	K_O	K_{O_1}	K_{O_2}	K_{O_S}	K_{O_T}	M^d	B^d	K^d	ζ
Platform position	2	4	1	4	3.2	5	10	25	0.45
Platform orientation	1	2	0.5	1	1.6	8	25	25	0.88
Leg angles	1	2	0.5	1	1.6	5	20	15	1.15

Table 3.1 – Wrench observer and impedance control gains along different robot states. The corresponding damping ratio of the desired impedance system is calculated by (3.42).

One of the first analyses conducted on the results of this experiment is the performance of the implemented momentum-based observers. Fig. 3.4 shows the results of the external wrench estimation performed by three different momentum-based wrench observers during a hovering flight of 150 s. The range and evolution of the external wrenches estimated by three different observers are similar, which can be seen as cross-validation on the estimation techniques based on momentum computation. However, their performance is slightly different: compared to the first-order wrench observer (FOWO), the estimated values of the second-order wrench observer (SOWO) are less noisy but less reactive to the external wrench changes as well; in contrast, the sliding-mode wrench observer (SMWO) reacts very fast to the evolution of external wrench, the noise amplitude of which is however much higher.

Furthermore, the effects of modelling errors and uncertainties estimated by the observers can be analysed. One may notice that the z-axis external force exerted on the platform decreases linearly as time goes on. This is caused by the drops in battery levels as the flight lasts which lead to a decrease in thrusts generated by quadrotors. Therefore, the mismatch between the desired thrusts and the actual thrusts produced by quadrotors has been estimated in the external wrench term, in particular the z-component of $\hat{\mathbf{f}}_{p,e}$. One may also find an increase in external moments on the legs due to the same reason since a decrease in quadrotor thrust will cause an external moment around the revolute-joint axis considered to be additionally exerted on each leg. According to Fig. 3.5(a), the tracking of the platform's z position was affected equivalently by applying three different wrench observers, caused by the mismatch of the actual thrusts produced by quadrotors.

The estimates of the external wrench on the other states are small, proving that the rest of the modelling errors except the uncertainties on quadrotor thrusts are negligible.



(a) $\hat{\mathbf{f}}_{p,e}$ estimated by FOWO (b) $\hat{\mathbf{m}}_{p,e}$ estimated by FOWO (c) $\hat{\mathbf{m}}_{l,e}$ estimated by FOWO

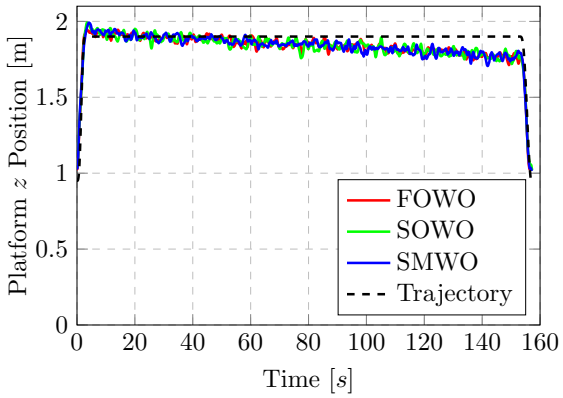
(d) $\hat{\mathbf{f}}_{p,e}$ estimated by SOWO (e) $\hat{\mathbf{m}}_{p,e}$ estimated by SOWO (f) $\hat{\mathbf{m}}_{l,e}$ estimated by SOWO

(g) $\hat{\mathbf{f}}_{p,e}$ estimated by SMWO (h) $\hat{\mathbf{m}}_{p,e}$ estimated by SMWO (i) $\hat{\mathbf{m}}_{l,e}$ estimated by SMWO

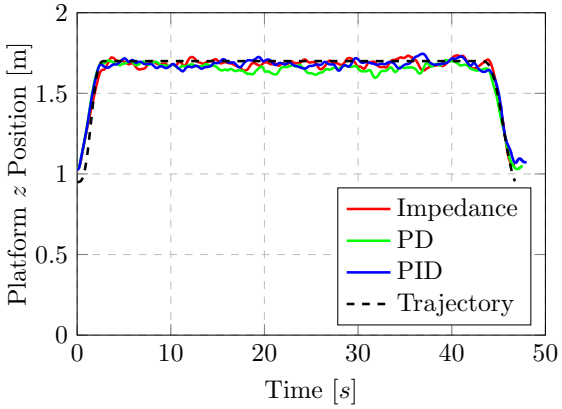
Figure 3.4 – External wrench estimation performed respectively by: (a)-(c) the first-order wrench observer (FOWO), (d)-(f) the second-order wrench observer (SOWO), (g)-(i) the sliding-mode wrench observer (SMWO) during the hovering flight in free space.

Additionally, the regulation on the thrusts produced by quadrotors can be achieved in order to cancel the effects due to the battery level changes, which is detailed in Appendix C. After applying the regulation of thrust based on the actual battery levels used onboard, a free-space hovering flight of the FPR was performed again. The results on the evolution of the platform's z position after the thrust regulation are demonstrated in Fig. 3.5, in which the comparison with the PD and PID-based controllers initially pro-

posed in [Six, 2018b; Liu, 2021] is also conducted. It can be seen that after the thrust regulation, the impedance controller can well perform the tracking of desired trajectories in free space, with the results comparable to that of the PD and PID-based controllers which are previously well-tuned to have the best performance such controllers can achieve. In the following experiments, the thrust regulation is a priori applied to reduce as much as possible the effects of modelling uncertainties. The quantitative results presented in Table 3.2 also show that even with well-tuned gains, the PD controller is still inevitably affected by static errors, especially seen in the positioning errors, while the impedance controller and PID controller can both work fine in tracking desired trajectories during the hovering flights in free space.



(a) Platform z position without thrust regulation, applying three different wrench observers.



(b) Platform z position with thrust regulation, comparing impedance controller with PD and PID-based controllers.

Figure 3.5 – Evolution of platform's z position without/with the thrust regulation based on battery levels during the hovering flight in Experiment I.

Controller	$\mathbf{p}_p$ [cm]	ϕ_p [°]	ϑ_p [°]	$\boldsymbol{\theta}_l$ [°]
Impedance	6.8	4.6	3.0	5.0
PD	12.5	5.1	6.6	3.9
PID	7.1	3.3	2.9	3.7

Table 3.2 – Root-mean-square error (RMSE) of the trajectory tracking during a hovering flight in free space. Note that RMSE for $\mathbf{p}_p$ refers to the platform positioning error, ϕ_p and ϑ_p represent the roll and pitch Euler angles of the platform, and RMSE for $\boldsymbol{\theta}_l$ is the mean value of RMSE of three leg angles.

3.5.2 Experiment II: Hovering with Additional Payload

In the second experiment, the FPR was flown with an additional payload attached to the centre of the platform (as shown in Fig. 3.6), which is unconsidered in the model

and thus acts as an external disturbance on the platform. A hovering flight with a payload weighting 200 g was firstly conducted to test the performance of different observers and their effects on the control. The results of the external wrench estimation and the trajectory tracking on the platform's x and z position as well as leg angles are shown respectively in Fig. 3.7 and Fig. 3.8.

It can be remarked that all three observers can well perform the estimation, with estimated values converged to 2 N as the ground truth and the RMSE of the estimation all limited within 1.1 N as shown in Table 3.3. The tracking of the platform's z position is slightly degenerated to stably control the robot suffering from the external force along z direction, which is the compliance behaviour ensured by the desired impedance settings. In addition, the performance of the observers discovered in Experiment I can also be well identified: the FOWO has an average convergence time, and the high-frequency noises are well filtered; the SOWO can largely reduce the estimation noises, but it converges relatively slow to stable values; the SMWO is more reactive to the external wrench and convergences faster, however it further amplifies the external wrench such that the estimated values are very unstable compared to the other observers. The undesired behaviour found in SOWO and SMWO affects the stability of the control system, which can be seen in Fig. 3.7(b) that the tracking of the platform's x axis is poor using SMWO due to its noise amplification and in Fig. 3.8 that the slow convergence of SOWO affects the tracking of the leg angle. The quantitative results presented in Table 3.3 also demonstrate that the stability of different estimation techniques affects positioning accuracy and the tracking of the platform's roll and pitch angles. Consequently, the FOWO is kept in the following experiments because of its best compromise between the estimation noise and convergence time.

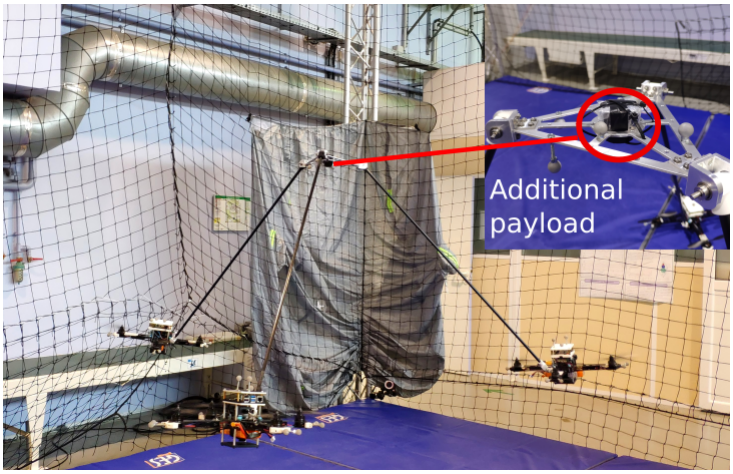
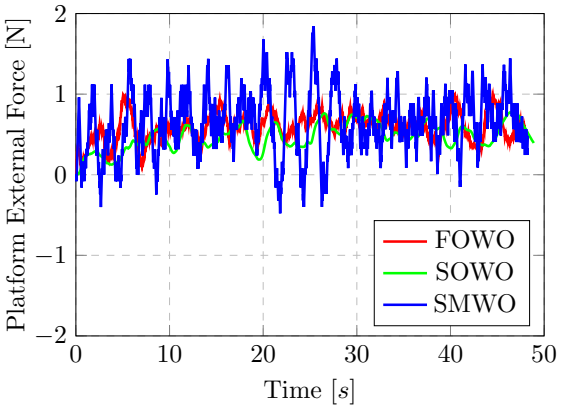
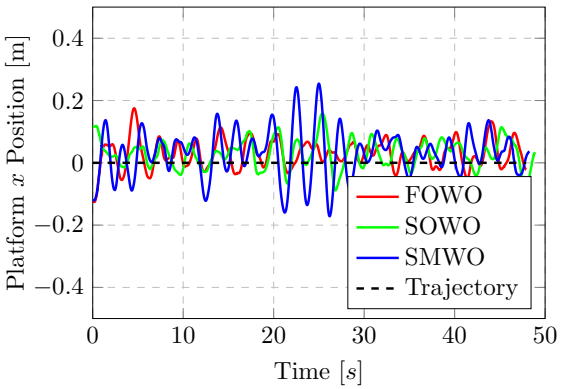


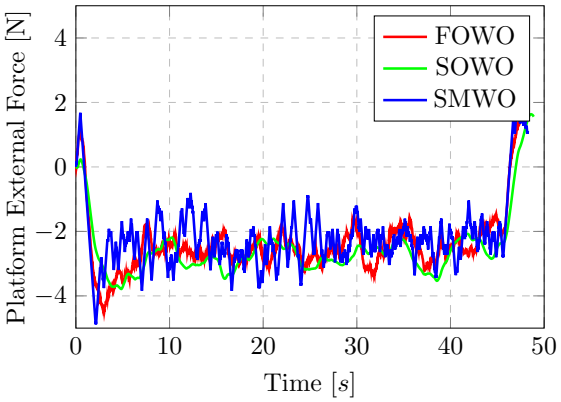
Figure 3.6 – FPR hovering with additional payload in Experiment II.



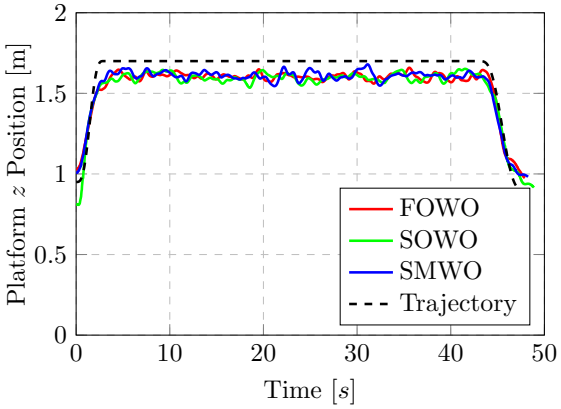
(a) x-axis platform external force



(b) x position of the platform

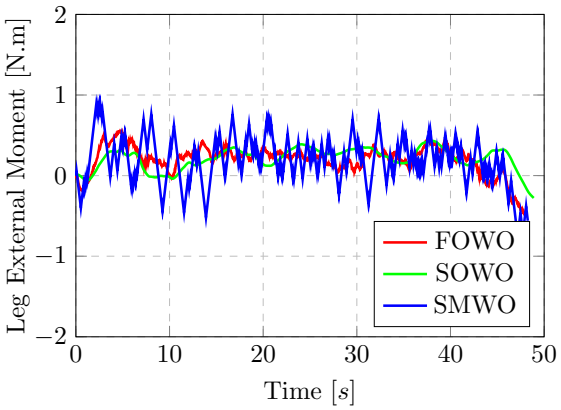


(c) z-axis platform external force

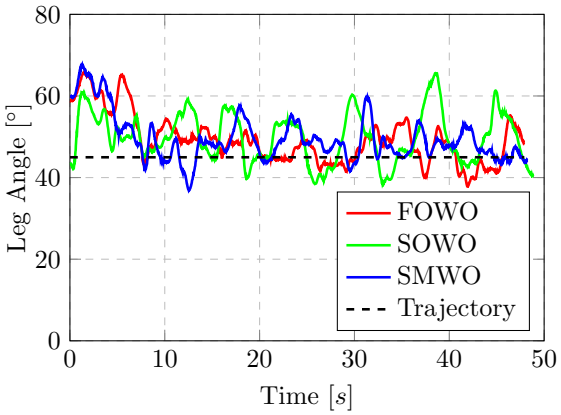


(d) z position of the platform

Figure 3.7 – Evolution of the x and z-axis external force estimates and the platform's x and z position during the hovering flight with an additional payload of 200 g.



(a) External moment around leg's joint axis



(b) Leg's revolute-joint angle

Figure 3.8 – Evolution of the external moment estimates on $\hat{\mathrm{m}}_{l3,e}$ and the revolute-joint angle of the leg 3 during the hovering flight with an additional payload of 200 g.

In the next step, the hovering flight in addition to an external payload of 300 g was flown with the external wrench estimated by FOWO. For purpose of comparing with

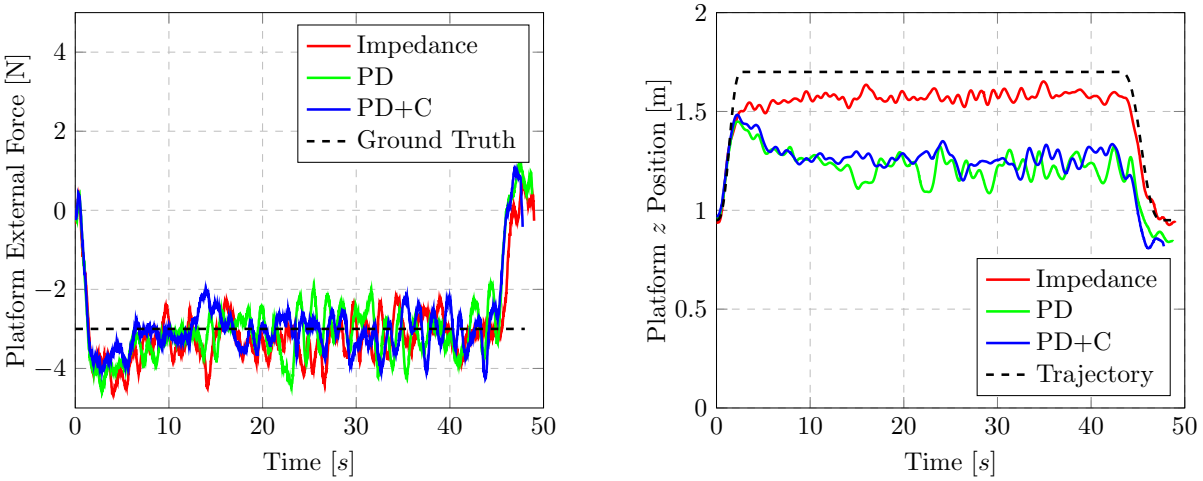
Observer	$\mathbf{p}_{p,xy}$ [cm]	ϕ_p [°]	ϑ_p [°]	$\boldsymbol{\theta}_l$ [°]	$\hat{\mathrm{f}}_{pz,e}$ [N]
FOWO	7.3	4.2	3.7	6.7	1.1
SOWO	8.5	7.2	5.0	6.5	1.1
SMWO	11.9	5.5	5.8	6.4	1.0

Table 3.3 – Root-mean-square error (RMSE) of the trajectory tracking and the z-axis external force estimation during a hovering flight with 200 g payload using different observers. Note that RMSE for $\mathbf{p}_{p,xy}$ refers to the platform positioning error on its x and y axes, and RMSE for $\hat{\mathrm{f}}_{pz,e}$ is computed relative to the ground truth.

existing controllers, the same testing flights were conducted using the following controllers:

1. Impedance-based controller presented in this chapter;
2. PD-based controller initially presented in [Six, 2018b];
3. PD-based controller with feed-forward compensation term corresponding to the external wrench estimation (named as PD+C controller).

All controllers are experimentally tuned to have good tracking results in free-space flights and then tested to handle the additional payload as the external disturbance. It has to be mentioned that PD controller is not dedicated to the interaction with the environment, whose performance is normal to be worse. However, the flights with this controller were still performed in order to set up a reference of the worst case. In addition, the PD+C controller is also compared in which the PD controller is enhanced by the knowledge of external wrench. The addition of an integral term in the PID-based controller was however shown to be unable to handle the external payload and inevitably result in crashes, which is therefore excluded in the comparison.



(a) z-axis platform external force

(b) z position of the platform

Figure 3.9 – Evolution of the external force estimates on $\hat{\mathrm{f}}_{pz,e}$ and the platform's z position during the hovering flight with an additional payload of 300 g.

Fig. 3.9 shows the results of the tracking of the platform's z position, and the estimated external force along z axis in $\mathfrak{F}_0$. The root-mean-square errors (RMSE) of the trajectory tracking and the external force estimation are presented in Table 3.4. From the results, it can be seen that the impedance controller can better deal with unknown disturbances, with good compliance behaviour shown in the platform's z position and better tracking performance on the other states. In addition, the effectiveness of the wrench estimator is validated as the RMSE of the z-component external force estimates are all about 1 N, which is an acceptable value, especially taking into account the large estimation errors in take-off and landing periods due to the ground effects (during the time interval within [0,2] s and [45,47] s shown in Fig. 3.9).

Controller	$\mathbf{p}_{p,xy}$ [cm]	ϕ_p [°]	ϑ_p [°]	$\boldsymbol{\theta}_l$ [°]	$\hat{\mathrm{f}}_{pz,e}$ [N]
Impedance	6.2	5.3	4.9	5.1	1.0
PD	9.3	10.7	8.0	13.6	1.1
PD+C	8.9	9.2	6.9	12.0	0.9

Table 3.4 – Root-mean-square error (RMSE) of the trajectory tracking and the z-axis external force estimation during a hovering flight with additional payload of 300 g. Refer to Table 3.2 and Table 3.3 for the meaning of each state.

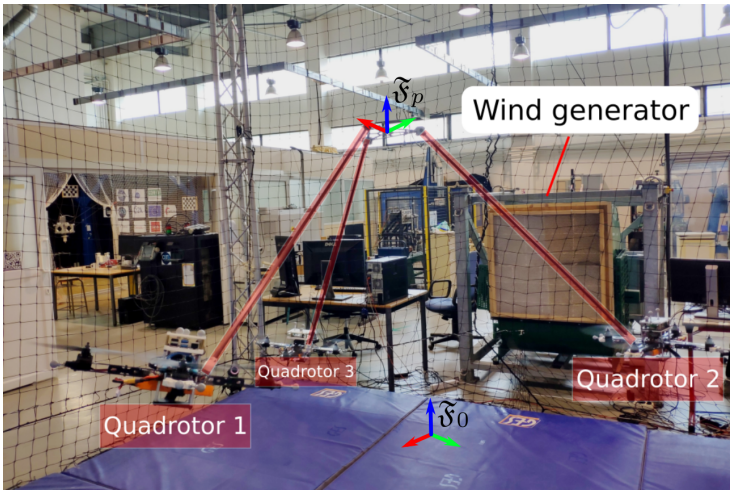


Figure 3.10 – FPR hovering in presence of external wind perturbations in Experiment III.

3.5.3 Experiment III: Hovering with External Wind Perturbations

After having validated the effectiveness of the estimation and impedance control techniques, this experiment has an objective of comparing the behaviour of the impedance system under different impedance gains. The FPR was flown to hover in front of a wind generator with a wind speed measured about 26 km/h from the centroid of the three

quadrotors at their hovering height (as shown in Fig. 3.10). The winds are generated along the x axis of the global frame, which is approximately aligned with the platform's x axis since zero yaw of the platform relative to the global frame is set in the desired trajectory. Therefore, the external winds will perturb the control of the leg angles and generate an external wrench on the platform mainly composed of:

- an x-axis force expressed in the global frame;
- a moment around the negative y axis of the platform frame (direction of the negative pitch angle of the platform).

Then the hovering flights in presence of wind perturbations during 60 s were performed, with different impedance gains on the platform's position and orientation states, while leaving the same gains for the additional DoFs on the leg angles. The mass and stiffness coefficients are varied with values higher or lower than the originally tuned ones to analyse the influence of impedance tuning, and the damping coefficients are calculated accordingly to maintain the same damping ratio as given in Table 3.1. Table 3.5 shows the variation in the impedance gains studied during the experiments, and the results on the tracking error distribution of the platform's x position and pitch angle are shown in Fig. 3.11, with quantitative results on the tracking performance of different impedance systems presented in Table 3.6.

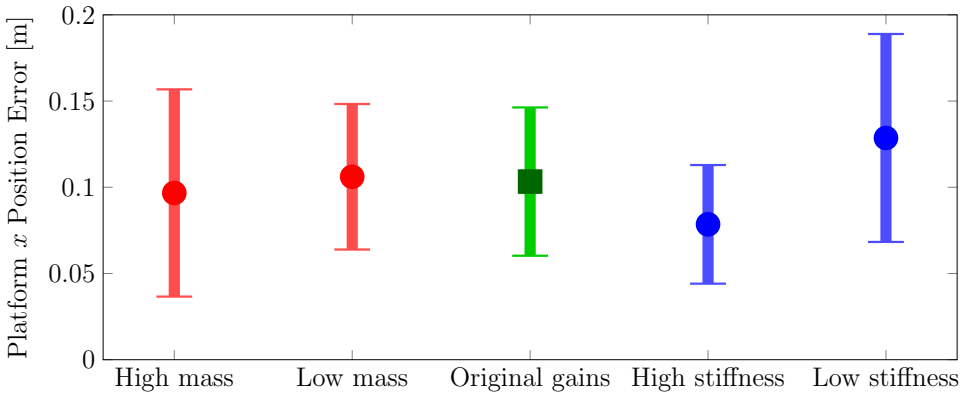
Impedance Gains	M_p^d	B_p^d	K_p^d	M_o^d	B_o^d	K_o^d
Originally tuned	5	10	25	8	25	25
High mass	8	25	12.65	12	25	30.62
Low mass	3	25	7.74	5	25	19.77
High stiffness	5	35	11.83	8	50	35.36
Low stiffness	5	20	8.94	8	15	19.36

Table 3.5 – Impedance gains chosen in Experiment III. Note that the subscripts p and o represent respectively the platform's position and orientation states.

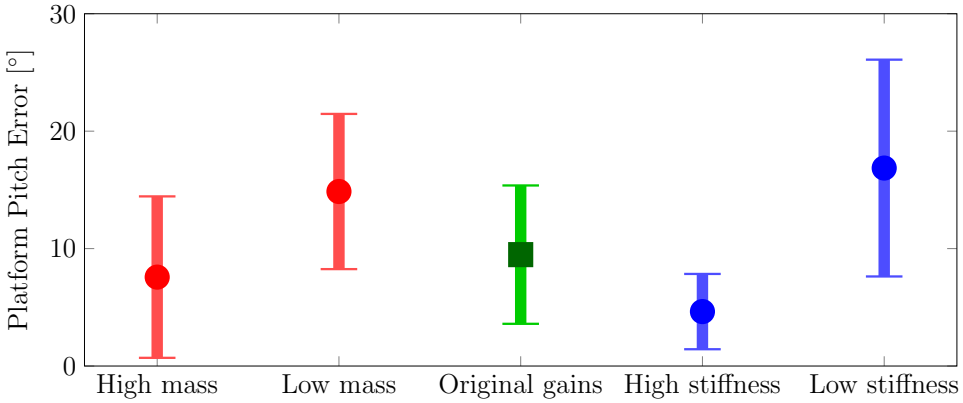
Impedance Gains	$\mathbf{p}_p$ [cm]	ϕ_p [°]	ϑ_p [°]	$\boldsymbol{\theta}_l$ [°]
Originally tuned	12.3	3.4	11.2	5.0
High mass	14.1	5.1	10.2	5.4
Low mass	12.4	3.5	16.3	5.4
High stiffness	9.4	2.1	5.6	4.6
Low stiffness	15.8	6.1	19.2	5.8

Table 3.6 – Root-mean-square errors (RMSE) of the trajectory tracking during the hovering flight in presence of external wind perturbations corresponding to different impedance gains. Refer to Table 3.2 for the meaning of each state.

From the results, different compliance behaviour according to the variations in mass and stiffness coefficients can be well identified. A higher value in mass coefficient might be



(a) Error distribution on the platform's x position.



(b) Error distribution on the platform's pitch angle.

Figure 3.11 – Error distribution on the platform's x position and pitch angle during the hovering flight in presence of external wind perturbations with different impedance gains. Note that the error distributions are calculated using the norm errors on the tracking of desired trajectories.

beneficial to resist the external wrench (especially shown in the platform pitch response). However, the robot becomes less reactive to external effects, which is unexpected during the interaction tasks. A proper selection for the mass coefficient such as the values selected in the original gains which are coherent with the mechanical and structural properties of the robot is preferable. On the other hand, a stiffer impedance system is always advantageous in rejecting the disturbances as shown in both cases, in which the tracking of the platform orientation is much enhanced by increasing the stiffness of the impedance system. The concern of increasing stiffness would be the loss of compliance with the external environment and the problems of overshooting and oscillation that might occur. Based on this analysis, proper values for the impedance gains can be chosen according to the specific tasks the robot is controlled to achieve. For instance, to achieve contact-based interaction tasks, the original gains are chosen for the platform's position and leg angle states, while a stiffer behaviour of the platform orientation state is desirable. In this way, the platform's compliance during the interaction is maintained, while there is more resilience against perturbations on the platform orientation.

3.5.4 Experiment IV: Contact-based Interaction Tasks

In this experiment, the FPR was flown to work in more challenging scenarios where the contact-based interaction tasks were performed in order to mimic real-world applications using this kind of aerial manipulators. A simply designed cylinder-form end-effector is attached at the centre of the platform, whose frame is supposed to be aligned with the platform frame. During the interaction tasks, the platform of the FPR (with the attached end-effector) was controlled to interact with a 30×30 cm wooden board fixed in the middle of the flight arena by exerting force along its normal direction (i.e. z axis of the platform frame). The wooden board is considered as an unknown object in the environment, such as the surface of a ceiling. It is hung by several tightly stretched ropes and attached with a heavy payload above, and thus assumed to be stiff enough to resist the interaction force between the platform during the experiment. It has to be mentioned that the exact position of the external object is not necessarily known. A predefined trajectory is sent to the FPR for taking off and reaching the potential interaction pose measured by the MOCAP. Then a human operator sends the desired contact force commands by a joystick to perform the interactions with the board by the end-effector attached to the platform. Fig. 3.12 shows the configuration of several contact-based interaction tasks performed by the FPR in this experiment, demonstrating different application scenarios that can be achieved by the FPR.

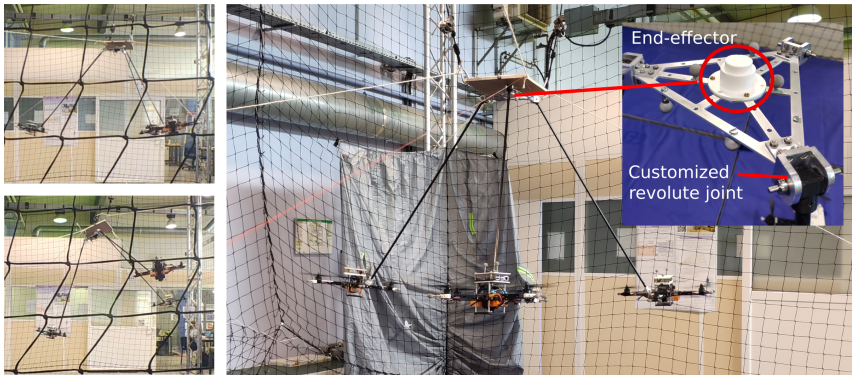


Figure 3.12 – FPR performing contact-based interaction tasks in Experiment IV.

Experiment	$\mathbf{p}_p$ [cm]	ϕ_p [°]	ϑ_p [°]	$\boldsymbol{\theta}_l$ [°]	f_c^d [N]
Task 1 - Case 1	6.9	5.5	5.3	5.0	1.7
Task 1 - Case 2	7.9	6.5	6.7	4.3	1.5
Task 2	7.4	3.4	3.1	3.6	1.3
Task 3	10.3	2.7	6.1	4.7	1.0

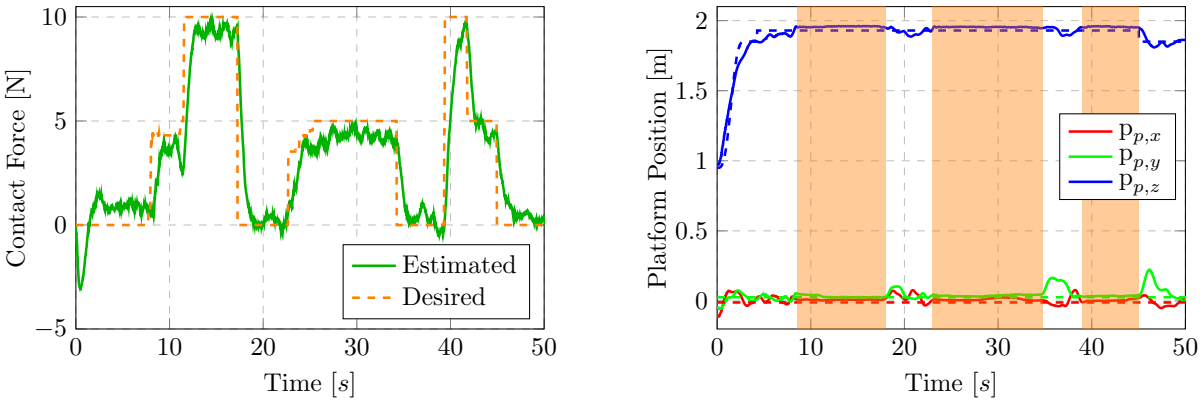
Table 3.7 – RMSE of the tracking of desired trajectories and contact force in Experiment IV. Note that RMSE for f_c^d refers to the tracking error of the desired contact force, and the rest of the states are defined in Table 3.2.

Task 1: pushing with different orientations

In the first task, the FPR was controlled to exert force onto the board by the platform in two different orientations: one in flat orientation and the other at an angle of about 10° in roll and 25° in pitch. The desired contact force is sent by the operator when the platform approaches the potential interaction pose with the board. Fig. 3.13 and Fig. 3.14 show the evolution of the desired and estimated contact force and the platform's position in two case studies with different orientations. In both cases, the estimated values of the contact force converge to desired values, with the range of the contact force exerted up to 10 N, validating the force-tracking performance of the proposed impedance controller. From the results of the tracking of the platform's position, the steady contacts between the platform and the board can be well identified, during which the platform position is further stabilized (indicated by the shaded areas in orange). The RMSE on the tracking of the other states in Table 3.7 also demonstrates that the robot configuration is stably regulated by the impedance controller during the interaction tasks.

Task 2: pushing and twisting

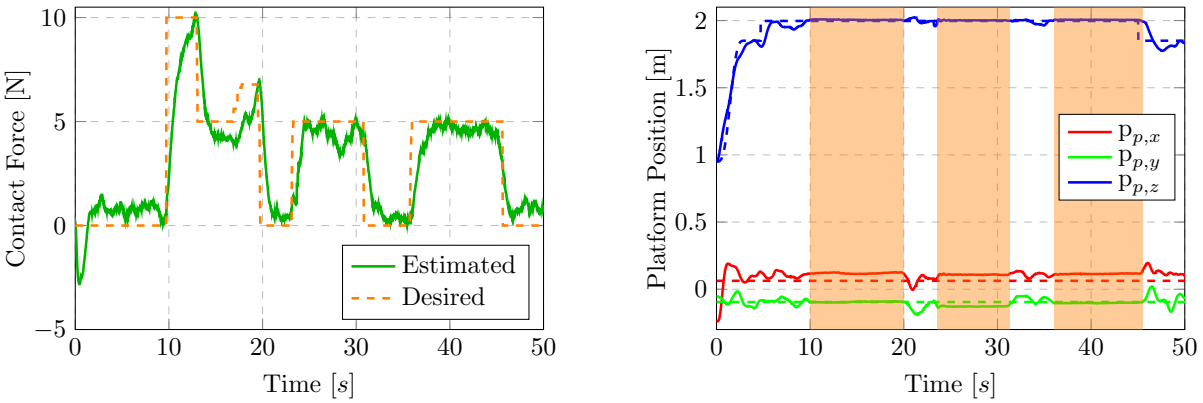
In this task, the platform was controlled to interact with the board in a flat orientation while twisting itself by yaw movement. When the contact is possible, the operator sends both the desired contact force and yaw rate commands to establish the interaction with the board and twist the platform at the same time. Fig. 3.15 shows the tracking of the desired contact force and platform's position, while the evolution of the platform's yaw angle is presented in Fig. 3.16. From these results, it can be remarked that the pushing & twisting task was successfully accomplished, with the contact force exerted by the platform up to 6 N. This task shows the FPR capable of accomplishing more complex tasks such as screwing or equipment replacement at remote locations.



(a) Evolution of the contact force

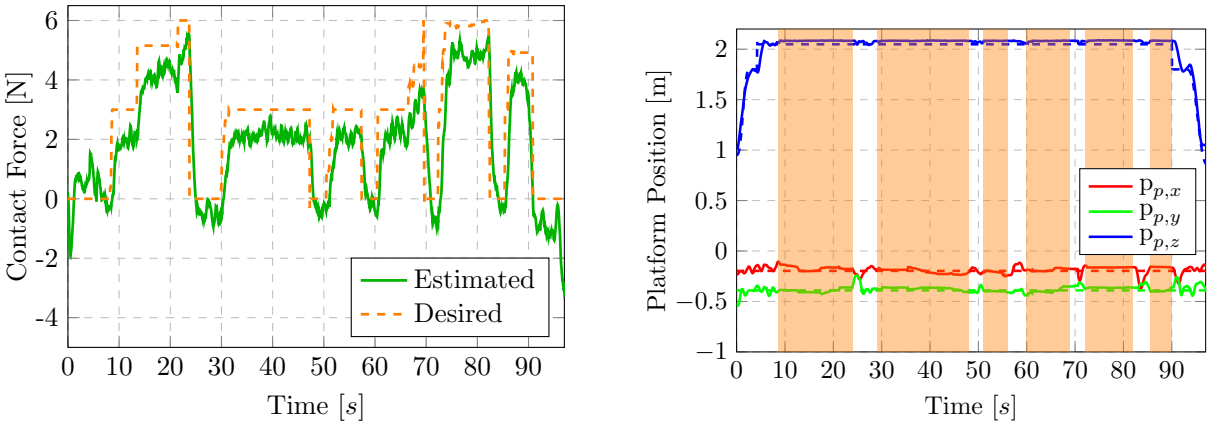
(b) Evolution of the platform's position

Figure 3.13 – Evolution of the desired and estimated contact force and the platform's position during the interaction task in a flat orientation. Note that the dashed lines are the desired values and the shaded areas in orange indicate the performed steady contacts.



(a) Evolution of the contact force (b) Evolution of the platform's position

Figure 3.14 – Evolution of the desired and estimated contact force and the platform's position during the interaction task with inclined orientation.



(a) Evolution of the contact force (b) Evolution of the platform's position

Figure 3.15 – Evolution of the desired and estimated contact force and the platform's position during the pushing and twisting task.

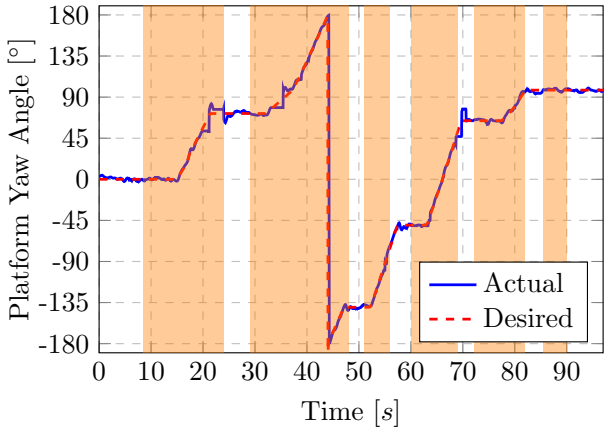
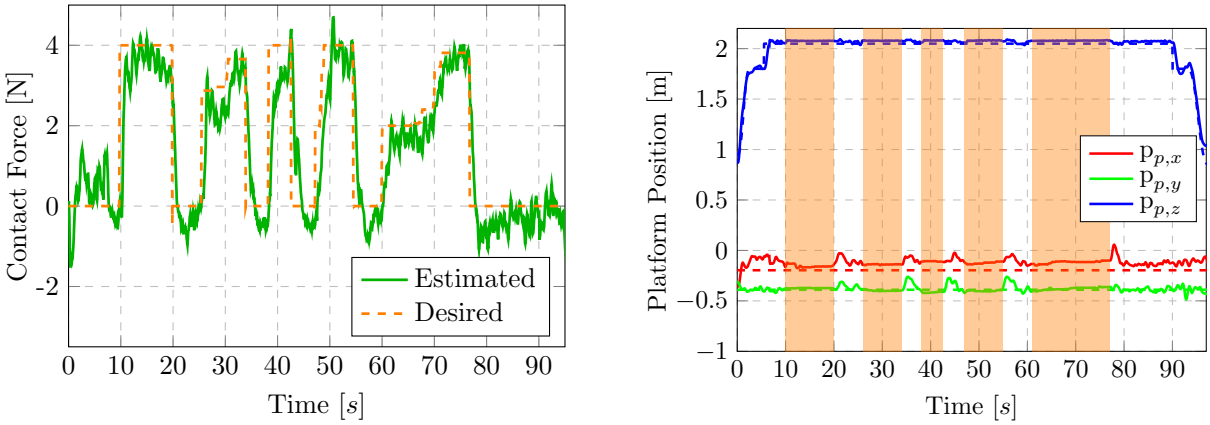


Figure 3.16 – Evolution of the platform's yaw angle during the pushing and twisting task.

Task 3: pushing in presence of external winds

This task involves a more challenging scenario where the FPR was performing the pushing task in presence of wind perturbations at a speed of about 13 km/h. The tracking results shown in Fig. 3.17 demonstrate that the interactions with the board were successfully achieved under the wind perturbations. The effects of the external winds are well estimated by the observer as shown in Fig. 3.18, demonstrating a x-axis force of about 2 N and a negative pitch moment of about -2 N.m on the platform. The tracking of the platform's x position and the pitch angle was perturbed, however the errors remain in a reasonable range to not perturb the steady contacts performed by the platform.



(a) Evolution of the contact force

(b) Evolution of the platform's position

Figure 3.17 – Evolution of the desired and estimated contact force and the platform's position during the pushing task with external wind perturbations.

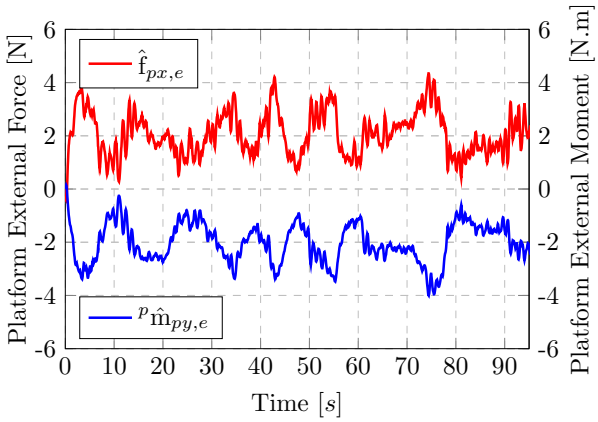


Figure 3.18 – Evolution of the platform's external force and moment estimates.

It can be concluded that all the interaction tasks were successfully accomplished, with acceptable RMSE on the tracking of the desired values given in Table 3.7. The desired contact forces are well tracked in all tasks, with errors limited within 1.7 N. These results have shown the effectiveness of the impedance controller with the external wrench

estimated by the first-order wrench observer applied to the FPR for performing contact-based interaction tasks in relatively complex working scenarios.

3.6 Conclusion

In this chapter, a methodology of controlling the FPR interacting with the environment has been studied, which involves firstly an estimation technique based on momentum computation to estimate the external wrench acting on the robot due to the modelling uncertainties, external disturbances and potential interactions with the environment. Several momentum-based observers are studied, including first-order wrench observer (FOWO), second-order wrench observer (SOWO) and sliding-mode wrench observer (SMWO). It has then been shown that these observers present different performances in terms of response time and noise amplitude in estimation. Then, based on the estimated external wrench, an impedance-based controller can be applied, which is designed by the desired impedance system and has the ability to track the desired interaction wrench set by a human operator. Extensive experimental validation has been done, in which the implemented estimation and control techniques have been tested and shown to be applicable to the FPR to cancel the modelling uncertainties, deal with the external disturbances and perform contact-based interaction tasks.

The method studied within this chapter has shown good potential in accomplishing a variety of industrial applications by the FPR. However, until now, this method still requires precise and high-rate exteroceptive measurements of the robot states and is usually implemented in a centralized way relying on the stable communication between each computation unit, which might not be realistic for real-world applications. In the next chapter, the presented estimation and control methods are shown to be applicable in a decentralized manner with the state estimation deployed onboard each multirotor using only intrinsic measurements.

Chapter 4

Decentralized Estimation and Control

This chapter presents the decentralized control algorithms applied to the Flying Parallel Robot (FPR). An introduction to the topics of the decentralization of multi-vehicle aerial robots is given in the first section. While most of the previous works on such robots often depend on the use of exteroceptive state measurements by the Motion Capture (MOCAP) systems, a method for reconstructing the robot state distributed on each multirotor using only intrinsic measurements is presented in the second section. Then the decentralized controllers based on previously studied control methods are introduced in the third section, including two PID-based motion controllers and an impedance-based interaction controller decentralized on each UAV. These decentralized control methods are complemented by integrating the teleoperation from a human operator as no global reference frame is considered available for positioning the platform. Finally, the proposed decentralization methods are validated by extensive experiments detailed in the fourth section.

4.1 Introduction

In most of the previous works on multi-UAV manipulators discussed in Section 1.1.2 such as the FPR, the controller is often deployed in a centralized manner, implying that a central computation unit receives all the measurements and computes the control for all the UAVs. This control scheme is usually straightforward to be developed, however it relies on a highly reliable bidirectional communication channel for transmitting all the control and measurement messages, and the crashes will almost inevitably occur from any communication interruption of more than a few control iterations (this means several tenths of a second for an aerial robot). Therefore, the deployment of a decentralized controller allowing each multirotor to perform its own control seems to be more robust to communication interruptions.

Another issue along with the centralized controller of an aerial robot is that this scheme often requires reliable and high-rate measurements of the robot pose in a global reference

frame, which is handled in most of the works by the Motion Capture (MOCAP) systems. These systems are usually bulky, expensive and require unobstructed lines of sight from multiple angles to each rigid body for the pose measurements. Along with high costs, the obvious impracticality in outdoor, cluttered or unknown environments is a disappointing limitation for an aerial robot. In some of the recent works, the MOCAP system is replaced by onboard visual and inertial sensing, such as in [Tagliabue, 2017] for leader-follower control of a suspended payload, [Loianno, 2018a] for transporting a bar-like object collaboratively by two UAVs, and [Li, 2021a] for transporting a payload with quadrotors suspended by cables. However, these works were for the cooperative transportation of a payload by multiple UAVs, a relatively simple system. Multi-UAV manipulators with rigidly-connected structure working 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 to avoid potential collisions.

The decentralization and the lack of global localisation in a common inertial frame are the issues that are extensively treated by the multi-robot control communities for swarming and formation flight [Coppola, 2020]. In these fields, a centralized control scheme is often undesirable not only because stable and reliable communication channels between every pair of a large number of agents seem infeasible, but also due to the complexity of a high-dimensional system resulting in centralized computation almost impossible. Therefore, it is much preferable 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 [Chung, 2018]. The application of such decentralized algorithms can be seen in the formation control of a team of aerial robots such as [Turpin, 2012; Schiano, 2016], leader-follower consensus of multiple UAVs [Hou, 2018; Guerrero-Castellanos, 2019], and outdoor flocking of a large number of UAVs [Vásárhelyi, 2014]. In these works, decentralization is often made possible by the individual robots finding some virtual common frame through consensus or other algorithms which moves with the group of robots [Coppola, 2020; Schiano, 2016]. For the systems with rigidly-connected links, the classical decentralized strategies are not applicable since the multiple UAVs are physically interfered with by the rigid links and have effects on each other. One may however remark that all the UAVs are connected to a common body (the payload or the moving platform), and therefore a common frame can be assigned with respect to which the individual UAVs can localise and control themselves based on intrinsic and onboard measurements.

In the following sections, the robot state estimation distributed on each multirotor is firstly investigated, which is performed using only the onboard visual and inertial cues.

Then, the decentralized schemes of the previously studied control law will be presented, which are then validated in real-world experiments.

4.2 Distributed State Estimation

As mentioned above, one of the main drawbacks of the previous works on the FPR has been that the controller often required reliable and high-frequency measurements of the robot pose expressed in a global reference frame. This is also one of the current limitations of other multi-UAV parallel robots (with rigidly connected links), which has meant until now the necessity of using MOCAP systems, rendering such robots far away from practical applications in the real world. In this section, a solution for recovering the robot state using only intrinsic measurements is proposed, which can be distributed on each multirotor and allows to sufficiently reconstruct a partial set of robot states required by the control.

The vector of the generalised coordinates given by (2.1) that needs to be reconstructed can be divided into two parts: the platform pose (i.e. $\mathbf{p}_p$ and $\mathfrak{q}_p$), and the internal configuration (i.e. passive leg angles $\boldsymbol{\theta}_l$). In the previous works, the platform pose was measured by the MOCAP system and the leg angles were computed using the geometric relation given the platform pose and the positions of the multirotors as presented previously in Section 2.3.2. Considering for now there is no longer external localisation available, two options can be proposed for measuring the leg angles:

- ❖ Potentiometers (or encoders) at the passive leg joints;
- ❖ Monocular pinhole cameras mounted on each multirotor.

The measurements of the leg angles by potentiometers could be straightforward. However, along with the much added weight due to the necessary stiffening of the revolute joint to maintain an accurate alignment, the additional arrangements such as the wiring, tuning and potential filtering to the noisy and biased measurements would reduce the practicality of using potentiometers. In contrast, the monocular vision seems to be a better solution, as cameras are now very lightweight and a branch of vision-based detection algorithms are available in the literature, such as an ArUco marker detection system [Romero-Ramirez, 2018; Romero-Ramirez, 2021], or model-fitting visual techniques that are rapidly becoming more computationally feasible [Tekin, 2017; Rad, 2017]. In addition, the vision onboard the multirotors can provide more information not only for estimating the leg angles by detecting the relative position, but also for reconstructing the orientation of the platform by the knowledge of relative orientation. As a result, the solution using pinhole cameras is chosen in this work.

In the following parts, the methodology of measuring the relative pose (i.e. position and orientation) between the platform and each multirotor is firstly presented, which is based on a marker detection system. A filtering technique based on Extended Kalman Filter (EKF) on the relative pose is implemented, for the purpose of maintaining a high frequency of the relative pose estimates and reducing the rate of false detections. This filtered relative pose is then used to reconstruct a partial set of robot coordinates, in which a notion of flat frames is introduced to represent the states such as the orientations in the case where no inertial reference frame is available.

4.2.1 Relative Pose Measurement

To measure the relative pose, a camera is assumed to be mounted on each multirotor and has its own frame $\mathfrak{F}_{Ci}$. An identified ArUco marker is supposed to be attached below the platform and has the marker frame $\mathfrak{F}_M$. Fig. 4.1 shows the definition of the frames for the onboard pose measurement on each multirotor i. Note that an ArUco marker is a synthetic square fiducial marker composed of a wide black border and an inner binary matrix which determines its identifier. A specific library is chosen to generate an ArUco marker [Garrido-Jurado, 2016], which can be detected for pose estimation using the ArUco detection algorithm detailed in [Romero-Ramirez, 2021] given the marker ID and size. The relative pose between the marker and the camera is therefore supposed to be estimated and represented by a homogeneous transformation matrix ${}^{Ci}\mathbf{T}_M$. Note that the homogeneous transformation matrix between two arbitrary frames $\mathfrak{F}_A$, $\mathfrak{F}_B$ is given by

$$ {}^{A}\mathbf{T}_B = \begin{bmatrix} {}^{A}\mathbf{R}_B & {}^{A}\mathbf{p}_B \\ \mathbf{0}_3^T & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4} \tag{4.1} $$

where ${}^{A}\mathbf{R}_B$ and ${}^{A}\mathbf{p}_B$ are respectively the rotation matrix and displacement vector of $\mathfrak{F}_B$ expressed in $\mathfrak{F}_A$, representing the relative pose between two frames.

The fixed transformations between the frames can be known by a prior calibration process, which are

- ${}^{bi}\mathbf{T}_{Ci}$ the transformation of the camera frame relative to the multirotor frame;
- ${}^{p}\mathbf{T}_M$ the transformation of the marker frame relative to the platform frame.

Then, if the camera is able to detect the ArUco marker and estimate the relative pose of ${}^{Ci}\mathbf{T}_M$, each multirotor will have knowledge of its pose relative to the platform frame $\mathfrak{F}_p$. This can be done by multiplying the transformation matrices as

$$ {}^{p}\mathbf{T}_{bi} = {}^{p}\mathbf{T}_M {}^{M}\mathbf{T}_{Ci} {}^{Ci}\mathbf{T}_{bi} \tag{4.2} $$

with ${}^{p}\mathbf{T}_{bi} = \begin{bmatrix} {}^{p}\mathbf{R}_{i} & {}^{p}\mathbf{p}_{i} \\ \mathbf{0}_{3}^{T} & 1 \end{bmatrix}$ the transformation matrix between $\mathfrak{F}_{bi}$ and $\mathfrak{F}_{p}$, which is composed of the relative position and orientation of the multirotor i expressed in $\mathfrak{F}_{p}$ (represented by the translation vector ${}^{p}\mathbf{p}_{i}$ and the rotation matrix ${}^{p}\mathbf{R}_{i}$). The fixed transformation matrices are computed as ${}^{M}\mathbf{T}_{Ci} = {}^{Ci}\mathbf{T}_{M}^{-1}$, ${}^{Ci}\mathbf{T}_{bi} = {}^{bi}\mathbf{T}_{Ci}^{-1}$ with the inverse transformation defined by

$$ {}^{B}\mathbf{T}_{A} = {}^{A}\mathbf{T}_{B}^{-1} = \begin{bmatrix} {}^{A}\mathbf{R}_{B}^{T} & -{}^{A}\mathbf{R}_{B}^{T}{}^{A}\mathbf{p}_{B} \\ \mathbf{0}_{3}^{T} & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \tag{4.3} $$

given two arbitrary frames $\mathfrak{F}_{A}$ and $\mathfrak{F}_{B}$.

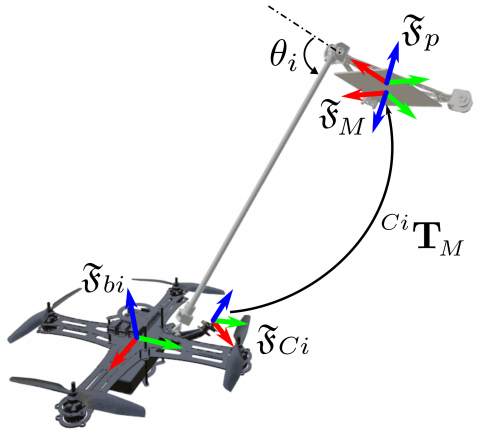


Figure 4.1 – Definition of the frames in the FPR for the pose estimation on each multirotor.

4.2.2 EKF-based Pose Estimation

The relative pose between the platform and each multirotor represented by ${}^{p}\mathbf{T}_{bi}$ is supposed to be estimated by the ArUco marker detection system. This may present problems in practical cases like uneven detection rate, false detection and noise 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. Therefore, the detected relative pose ${}^{p}\mathbf{T}_{bi}$ is not directly used in the next step for the robot pose reconstruction (presented in the next subsection), but rather filtered to output at a constant frequency with reduced noises and errors. To achieve this, an Extended Kalman Filter (EKF)-based algorithm is adopted, a well-known technique for data fusion and filtering in non-linear and discrete systems [Welch, 1995].

The relative pose to be filtered is rearranged by a vector defined as the system state used in the Kalman filter

$$ \mathbf{x} = \begin{bmatrix} {}^{p}\mathbf{p}_{i} \\ {}^{p}\mathbf{q}_{i} \end{bmatrix} \tag{4.4} $$

with ${}^{p}\mathbf{q}_i$ a unit quaternion converted from the rotation matrix ${}^{p}\mathbf{R}_i$ (see the conversion detailed in Appendix A.2). The evolution of the system state is then defined by

$$\dot{\mathbf{x}} = \begin{bmatrix} {}^{p}\dot{\mathbf{p}}_i \\ {}^{p}\dot{\mathbf{q}}_i \end{bmatrix} \tag{4.5}$$

To develop the prediction model in the EKF, two main hypotheses are made:

- ${}^{p}\dot{\mathbf{p}}_i$ is supposed to be zero at each instant, due to the fact that each multirotor is rigidly attached to the passive structure and its relative position in the platform frame should be constant if the corresponding leg angle is not changing.
- ${}^{p}\dot{\mathbf{q}}_i$ is only affected by the angular velocity of each multirotor, as the rotational movements of multirotors are fully decoupled with the passive architecture and the angular velocity of the platform is negligible compared to the fast rotational dynamics of UAVs.

Therefore, the expression of $\dot{\mathbf{x}}$ can be given by

$$\dot{\mathbf{x}} = \begin{bmatrix} \mathbf{0}_3 \\ \frac{1}{2}{}^{p}\mathbf{q}_i \circ \begin{bmatrix} 0 \\ {}^{bi}\boldsymbol{\omega}_i \end{bmatrix} \end{bmatrix} \tag{4.6}$$

with ${}^{bi}\boldsymbol{\omega}_i$ the angular velocity of the multirotor i, and $\circ$ representing the quaternion multiplication. The evolution of the quaternion can be simplified by the matrix multiplication

$${}^{p}\dot{\mathbf{q}}_i = \boldsymbol{\Omega}({}^{bi}\boldsymbol{\omega}_i){}^{p}\mathbf{q}_i \tag{4.7}$$

with $\boldsymbol{\Omega}(\boldsymbol{\omega})$ is a (4×4) skew-symmetric matrix associated to a vector of the angular velocity $\boldsymbol{\omega}$, which can be defined as [Schwab, 2002]

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) = \frac{1}{2}\begin{bmatrix} 0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & -[\boldsymbol{\omega}]_\times \end{bmatrix} \tag{4.8}$$

Remind that $[.]_\times$ represents the skew-symmetric matrix associated with a 3-dimensional vector for the cross product operation (referred to (A.50) in Appendix A.1).

It should be noticed that if the leg angles are changing or the platform is rotating (especially in roll and pitch angles), $\dot{\mathbf{x}}$ will also be affected by such movements. However, the slow dynamic response of the FPR structure limits the agility of the robot, which makes the evolution of the relative pose caused by the unconsidered movements as bounded modelling errors in the transition phase. They would slightly degenerate the prediction process, but will rapidly be rectified once the robot gets back to the quasi-static condition

or during the next correction step.

According to [Welch, 1995], the equations of the EKF can be written as follows for the state prediction and correction models

$$\begin{aligned} \mathbf{x}_k &= \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_{k-1} \\ \mathbf{z}_k &= \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \end{aligned} \tag{4.9}$$

with $\mathbf{u}_k$ as the input of the system at each instant k, and additive vectors $\mathbf{w}_{k-1}$, $\mathbf{v}_k$ respectively for the prediction and measurement noises. The prediction equation corresponding to (4.6) can be written in discrete time, where the discrete-time expression of the evolution of the unit quaternion corresponding to (4.7) can be given by [Sabatini, 2006]

$${}^{p}\mathfrak{q}_{i,k} = \exp\left(\mathbf{\Omega}_k({}^{bi}\boldsymbol{\omega}_{i,k})T_s\right){}^{p}\mathfrak{q}_{i,k-1} + \mathbf{w}_{\mathfrak{q},k-1} \tag{4.10}$$

where $\mathbf{u}_k = {}^{bi}\boldsymbol{\omega}_{i,k}$ is considered as the system input at time k, T_s is the time step of the prediction process, $\mathbf{w}_{\mathfrak{q},k-1}$ is the vector of prediction noises in quaternion elements, and ${}^{p}\mathfrak{q}_{i,k-1}$ and ${}^{p}\mathfrak{q}_{i,k}$ are the quaternions at two consecutive time steps within an interval of T_s. When T_s is small enough, $\mathbf{\Omega}_k({}^{bi}\boldsymbol{\omega}_{i,k})$ is assumed to be constant. Therefore, one can rewrite $\exp\left(\mathbf{\Omega}_k({}^{bi}\boldsymbol{\omega}_{i,k})T_s\right)$ approximately using its first-order and second-order items of Taylor series expansion as [Feng, 2017]

$${}^{p}\mathfrak{q}_{i,k} = \left(\mathbf{1}_{4\times4} + \mathbf{\Omega}({}^{bi}\boldsymbol{\omega}_{i,k})T_s\right){}^{p}\mathfrak{q}_{i,k-1} + \mathbf{w}_{\mathfrak{q},k-1} \tag{4.11}$$

which yields the prediction model and the state transition matrix finally given by

$$\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k) = \mathbf{x}_{k-1} + \begin{bmatrix} \mathbf{0} \\ \mathbf{\Omega}({}^{bi}\boldsymbol{\omega}_{i,k})T_s \end{bmatrix} \tag{4.12}$$

and

$$\mathbf{F}_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_{k-1},\mathbf{u}_k} = \begin{bmatrix} \mathbf{1}_{3\times3} & \mathbf{0}_{3\times4} \\ \mathbf{0}_{4\times3} & \mathbf{1}_{4\times4} + \mathbf{\Omega}({}^{bi}\boldsymbol{\omega}_{i,k})T_s \end{bmatrix} \tag{4.13}$$

For the observation model, the system state defined in (4.4) is considered to be directly measured via the relationship of (4.2), which gives

$$\mathbf{h}(\mathbf{x}_k) = \mathbf{x}_k; \quad \mathbf{H}_k = \left.\frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right|_{\mathbf{x}_k} = \mathbf{1}_{7\times7} \tag{4.14}$$

where $\mathbf{H}_k$ is the observation matrix set by a (7×7) identity matrix.

After having constructed the state prediction and observation models, the prediction

and correction equations of the EKF algorithm can be written as follows [Welch, 1995]
Prediction:

$$\begin{aligned} \mathbf{x}'_k &= \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k) \\ \mathbf{P}'_k &= \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T + \mathbf{Q}_k \end{aligned} \tag{4.15}$$

Correction:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{x}'_k + \mathbf{K}_k \Big(\mathbf{z}_k - \mathbf{h}(\mathbf{x}'_k) \Big) \\ \mathbf{K}_k &= \mathbf{P}'_k \mathbf{H}_k'^T (\mathbf{H}'_k \mathbf{P}'_k \mathbf{H}_k'^T + \mathbf{R}_k)^{-1} \\ \mathbf{P}_k &= (\mathbf{1} - \mathbf{K}_k \mathbf{H}'_k) \mathbf{P}'_k \end{aligned} \tag{4.16}$$

Note that $\mathbf{P}_k$ is the covariance matrix of the estimation at each instant k, with the covariance $\mathbf{P}_0$ initialized with the initial state $\mathbf{x}_0$. $\mathbf{K}_k$ is the matrix for the Kalman gains. $\mathbf{Q}_k$ and $\mathbf{R}_k$ are the covariance matrices respectively for the prediction and measurement noises, defined by $\mathbf{Q}_k = E[\mathbf{w}_k \mathbf{w}_k^T]$ and $\mathbf{R}_k = E[\mathbf{v}_k \mathbf{v}_k^T]$. The terms with the prime symbol are those predicted but not yet updated.

The tuning of the EKF system stays at the determination of prediction and measurement noise covariance, i.e. $\mathbf{Q}_k$ and $\mathbf{R}_k$. The former depends on the correctness of the prediction model, while the latter is usually defined based on the measurement uncertainties according to the noise covariance of sensor outputs. The prediction and update process are usually performed at the same rate (i.e. time interval of T_s). However, as the correction model does not depend on the time step, the update process can be done asynchronously each time when the measurements are available.

One major advantage of using EKF for the pose estimation is therefore that by a proper and regular prediction process, the EKF system can output the filtered relative pose at a constant rate. This is beneficial to recovering the robot state and the control in the following steps. The algorithm of EKF-based pose estimation can potentially be extended by adding another source of measurements, for which multiple sensor data can be fused to increase the redundancy of the pose estimation to avoid false detection issues. However, the Extended Kalman Filter is in general not an optimal estimator and only ensures local convergences due to the linearisation of the models (it is optimal when the prediction and correction models are both linear, the case where an EKF is identical to a regular Kalman filter). Additionally, the linear prediction and correction models in the EKF might not satisfy the normalisation constraint of the unit quaternion, which is ensured by a manual normalisation of the resulted quaternion after every prediction and correction process. Therefore, it would be better to adopt another filter more appropriate for the rotation group and quaternion parameterisation, such as Multiplicative Extended Kalman Filter (MEKF) [Markley, 2003; Li, 2020a], Invariant Extended Kalman Filter (IEKF) [Bonnabel,

2009; Barrau, 2017] or Equivariant Filter (EqF) [Goor, 2022]. This may be the future improvements of the works presented in this section.

4.2.3 Robot State Reconstruction

Supposing now the relative pose of the platform expressed in $\mathfrak{F}_{bi}$ is well estimated by each multirotor, the current focus remains on how to reconstruct the robot states (i.e. the generalised coordinates $\mathbf{q}$ and the velocity $\boldsymbol{\nu}$) that are necessary for the control. This is done by firstly computing the leg angles knowing the position of multirotor i expressed in the platform frame, i.e. ${}^{p}\mathbf{p}_i$. With the fixed revolution joint position of each leg ${}^{p}\mathbf{p}_{A_i}$ expressed in $\mathfrak{F}_p$ that can be given by the expression of ${}^{p}\mathbf{r}_i$ in (2.7), the vector describing the leg's direction expressed in $\mathfrak{F}_p$ can be computed by

$$ {}^{p}\mathbf{r}_{A_iB_i} = {}^{p}\mathbf{p}_i - {}^{p}\mathbf{p}_{A_i} \tag{4.17} $$

Note that the relative position ${}^{p}\mathbf{p}_i$ is equivalent to the vector of ${}^{p}\mathbf{p}_{Bi}$, i.e. the CoM position B_i of the multirotor i expressed in $\mathfrak{F}_p$.

Then, the revolute joint angle of each leg can be calculated using the equation equivalent to (2.11), but expressed in $\mathfrak{F}_p$

$$ \theta_i = \operatorname{acos}\left(\frac{{}^{p}\mathbf{r}_{A_iB_i} \cdot {}^{p}\mathbf{p}_{A_i}}{\|{}^{p}\mathbf{r}_{A_iB_i}\| \cdot \|{}^{p}\mathbf{p}_{A_i}\|}\right) \tag{4.18} $$

While this geometric model for computing the leg angles holds true for the FPR, the method may be generalised to other aerial manipulators by substituting their geometric models to compute the internal configuration with the knowledge of ${}^{p}\mathbf{T}_{bi}$.

The platform pose was previously measured by the MOCAP systems and expressed with respect to a reference frame $\mathfrak{F}_0$, which has lost much of its meaning under the situation that there is no more external localisation system, nor a global reference frame. To deal with the lack of the reference frame, one could of course use $\mathfrak{F}_p$ as a common frame for expressing all the states, and instead of controlling directly the position and orientation of the platform (previously in a global frame), one could regulate its linear and angular velocity expressed in $\mathfrak{F}_p$ through the first-order kinematic model described in Section 2.4.1. However, this might invoke additional problems: for teleoperation purposes, the operator must stabilise all six DoFs of the platform via the control of 6-dimensional velocity, which is difficult with a standard 4-axis joystick; excessive roll and pitch during the control can rapidly lead to instability of the robot; while this may work for eye-in-hand visual servoing, the additional tuning of the sensor-space control will potentially be too

complicated. Therefore, a better choice of the common reference frame can be determined and a more ideal arrangement for the control of the platform's orientation should be made.

Actually, it is remarked that each multirotor can necessarily estimate its own attitude using onboard IMU measurements with the state estimation techniques such that it can provide additional information on the global attitude. Recall the Euler angles of ZYX convention $\boldsymbol{\eta}_i = \begin{bmatrix}\phi_i & \vartheta_i & \psi_i\end{bmatrix}^T \in \mathbb{R}^3$ for each multirotor i, denoted by ψ_i (yaw), ϑ_i (pitch) and ϕ_i (roll) angles. The roll and pitch are commonly defined about the axes orthogonal to the gravity vector and are necessarily well estimated by the onboard sensing and estimation process for a stable flight. However, for the yaw estimates, the multirotors are usually not sharing a common frame, as in practice this measurement suffers from noises and biases especially in presence of magnetic field perturbations close to the metallic structures and electrical circuits onboard the multirotor. It can then be reasonably assumed that each multirotor's attitude is estimated with respect to an unknown reference frame $\mathfrak{F}_{0i}$, which is constrained such that its z axis is aligned with the z axis of $\mathfrak{F}_0$ (i.e. $\mathbf{z}_{0i} = \mathbf{z}_0$), but with an unknown yaw ${}^0\psi_{0i}$ relative to $\mathfrak{F}_0$. Therefore, a notion of flat frame $\mathfrak{F}_{Fj}$ attached to an arbitrary body j is introduced to represent an orientation with zero roll and pitch relative to a global reference frame, and an unknown yaw with respect to the unknown global frame $\mathfrak{F}_0$ (as shown in Fig. 4.2). The orientation of a body can thus be expressed in the flat frame, in which the roll and pitch are expressed similarly as previously defined in a global inertial frame, with an unknown yaw considered to be always zero. The orientation of multirotor i expressed in its flat frame can be expressed as

$$ {}^{Fi}\mathbf{R}_i = \mathbf{R}_y(\vartheta_i)\mathbf{R}_x(\phi_i) \tag{4.19} $$

with $\mathbf{R}_j(.)$ (for $j = \{x, y\}$) the unit coordinate rotations detailed in Appendix A.1.

As the relative orientation of the platform with respect to each multirotor i is known by ${}^i\mathbf{R}_p = {}^p\mathbf{R}_i^T$, with ${}^p\mathbf{R}_i$ computed by its associated unit quaternion ${}^p\mathfrak{q}_i$, the orientation of the platform can be further expressed in $\mathfrak{F}_{Fi}$ by

$$ {}^{Fi}\mathbf{R}_p = {}^{Fi}\mathbf{R}_i\,{}^i\mathbf{R}_p \tag{4.20} $$

which allows to obtain the roll and pitch of the platform expressed in $\mathfrak{F}_{Fi}$ independently to any yaw of the platform. Remark that the measurements and computations of ${}^{Fi}\mathbf{R}_p$ on each multirotor must be identical. To simplify the derivation, a flat platform frame $\mathfrak{F}_{Fp}$ is introduced, which shares a common z axis with $\mathfrak{F}_{Fi}$ and has an unknown yaw relative to any $\mathfrak{F}_{Fi}$. According to the expression of the rotation matrix written by ZYX-convention

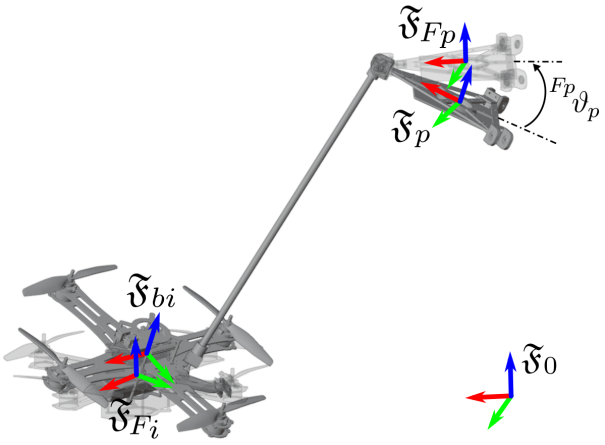


Figure 4.2 – Introduction of flat frames for expressing the orientation defined by the roll and pitch angles on one leg of the FPR. The real platform and multirotor poses are solid, and their poses in the flat frames are transparent. Note that the origins of real and flat frames should be collinear, which is not the case in the figure just for better visualisation.

Euler angles, the roll and pitch of the platform expressed in $\mathfrak{F}_{Fp}$ are computed by

$$\begin{aligned} {}^{Fp}\phi_p &= \text{atan2}\Big({}^{Fi}\mathbf{R}_p(3,2), {}^{Fi}\mathbf{R}_p(3,3)\Big) \\ {}^{Fp}\vartheta_p &= -\text{asin}\Big({}^{Fi}\mathbf{R}_p(3,1)\Big) \end{aligned} \tag{4.21}$$

where $\mathbf{R}(r,c)$ refers to the element at the r-th row and c-th column of the matrix. This allows to reconstruct the quaternion of the platform relative to $\mathfrak{F}_{Fp}$, which is then used in the control. The flat frame attached to the platform can therefore be served as a common frame to express all the other states of each multirotor i, for which the transformation matrix between $\mathfrak{F}_{bi}$ and $\mathfrak{F}_{Fp}$ is computed by

$${}^{Fp}\mathbf{T}_{bi} = {}^{Fp}\mathbf{T}_p \, {}^{p}\mathbf{T}_{bi} \tag{4.22}$$

This allows to further express the linear velocities of all the multirotors expressed commonly in $\mathfrak{F}_{Fp}$ by

$${}^{Fp}\mathbf{v}_i = {}^{Fp}\mathbf{R}_i \, {}^{bi}\mathbf{v}_i \tag{4.23}$$

with ${}^{Fp}\mathbf{R}_i$ representing the relative orientation of the multirotor i expressed in $\mathfrak{F}_{Fp}$, extracted from the transformation matrix ${}^{Fp}\mathbf{T}_{bi}$, and ${}^{bi}\mathbf{v}_i$ being the body-frame linear velocity of each multirotor i measured onboard.

Then the computation of the generalised velocity of the FPR can be done similarly by using the forward kinematics of (2.21) as previously discussed in Section 3.4 for a centralized scheme. As the linear velocities of multirotors are all expressed in $\mathfrak{F}_{Fp}$, the linear velocity of the platform (i.e. ${}^{Fp}\mathbf{v}_p$) is now expressed in $\mathfrak{F}_{Fp}$, while the body-frame angular

velocity of the platform ${}^{p}\boldsymbol{\omega}_{p}$ and the leg angles rates $\dot{\boldsymbol{\theta}}_{l}$ are the same as computed in the centralized scheme.

The geometric and kinematic models as well as the introduction of the flat frames have allowed to reconstruct the full vector of the robot velocity and a partial set of the robot pose as follows

$$\mathbf{q} = \begin{bmatrix} \mathbf{0}_3^T & {}^{Fp}\mathbf{q}_p^T & \boldsymbol{\theta}_l^T \end{bmatrix}^T \tag{4.24}$$

with the platform orientation ${}^{Fp}\mathbf{q}_p$ defined by its roll and pitch expressed in $\mathfrak{F}_{Fp}$ (i.e. ${}^{Fp}\phi_p$ and ${}^{Fp}\vartheta_p$) and the leg angles $\boldsymbol{\theta}_l$, up to an unknown position $\mathbf{p}_p$ (considered as zero in the first three elements of $\mathbf{q}$) and unknown yaw ψ_p of the platform with respect to any inertial frame.

4.3 Decentralized Controllers

Having shown in the previous section that sufficient information on the robot states can be recovered from intrinsic measurements and estimation techniques, this section addresses decentralized controllers which allow each multirotor to compute its own control law based on the available robot states estimated onboard. For the control law itself, it is based on the same controller presented in Section 1.2.2 for motion control or the interaction controller developed in Section 3.3. Compared to a centralized scheme which depends on highly reliable communication between each multirotor and the centralized computer, the deployment of the control law in a decentralized manner onboard each multirotor requires only partial information shared among all the agents, which would be more robust to communication delays or interruptions.

In the following parts, the control problem is firstly formulated in line with the robot states reconstructed by each multirotor. Then the decentralization of the motion control and interaction control law are presented, within which different schemes are discussed depending on whether the sharing of information between multirotors is available/necessary or not. For each multirotor, the computation of desired attitude setpoints is furthermore adjusted, considering the fact that there is no more global reference frame to express commonly the attitude setpoints and measurements.

4.3.1 Control Problem Formulation

The reconstruction of the robot states (i.e. robot pose and velocity) described in Section 4.2 has built knowledge of the platform orientation ${}^{Fp}\mathbf{q}_p$ expressed in $\mathfrak{F}_{Fp}$, leg angles $\boldsymbol{\theta}_l$ and the robot velocity. The unknown coordinates are the position $\mathbf{p}_p$ and yaw

ψ_p of the platform relative to any inertial reference frame. One may remark that these unknown coordinates are the same as the flat outputs of a quadrotor [Mellinger, 2011a], which are easily controlled with a joystick. Therefore, the control of the platform is similar to the manual pilot of a single multirotor, with the roll ϕ_p and pitch ϑ_p of the platform decoupled from its yaw rate $\dot{\psi}_p$, and translation movements of the platform controlled by the linear velocity $\mathbf{v}_p$. Based on this arrangement, a new set of robot states that can be reconstructed is defined as

$$\boldsymbol{\chi} = \begin{bmatrix} {}^{Fp}\mathbf{v}_p^T & {}^{Fp}\dot{\psi}_p & {}^{Fp}\vartheta_p & {}^{Fp}\phi_p & \boldsymbol{\theta}_l^T \end{bmatrix}^T \tag{4.25}$$

which can be known without relying on any external localisation system and thus defined as the set of controllable variables.

This is furthermore an easily teleoperable system, as the leg angles, roll and pitch of the platform may be regulated by values (instead of by rate), which are the states susceptible to causing physical or numerical instabilities when the values are too small or too large (i.e. near unstable configurations). An operator is then able to set the desired values for those sensible states, and focus on the positioning and heading of the platform by controlling the linear velocity and yaw rate of the platform in $\mathfrak{F}_{Fp}$, independent of changes in roll and pitch of the platform. It is therefore assumed that an operator sends the desired values for $\boldsymbol{\chi}^d$ during the teleoperation of the FPR. To further ensure the smoothness of the evolution of the robot states, these teleoperation commands sent by the operator are further filtered, with the derivatives $\dot{\boldsymbol{\chi}}^d$ output from a basic low-pass filter provided to the controller as well. Therefore, the decentralized controller deployed on each multirotor will listen to the teleoperation commands $\boldsymbol{\chi}^d$ and $\dot{\boldsymbol{\chi}}^d$ sent by an operator, and regulate the robot states towards the desired setpoints, while the human operator is involved in the highest control loop for regulating the platform's position and yaw.

From the controller side, once the teleoperation commands of $\boldsymbol{\chi}^d$ and $\dot{\boldsymbol{\chi}}^d$ are received, the desired trajectory corresponding to the desired generalised coordinates, velocity and acceleration of the robot (i.e. $\mathbf{q}^d$, $\boldsymbol{\nu}^d$ and $\dot{\boldsymbol{\nu}}^d$) needs to be further computed to construct the tracking error in the control law. The position of the platform $\mathbf{p}_p$ is now unknown, and thus its desired value is always set to zero. As the orientation of the platform is now defined by its roll and pitch angles in $\mathfrak{F}_{Fp}$, the unit quaternion ${}^{Fp}\mathfrak{q}_p^d$ associated to the rotation matrix for desired angles ${}^{Fp}\mathbf{R}_p^d = \mathbf{R}_y({}^{Fp}\vartheta_p^d)\mathbf{R}_x({}^{Fp}\phi_p^d)$ is defined as the desired orientation. For the leg angles, the definition of the tracking error remains the same because their values are known by the geometric model.

The commands on the velocities given from $\boldsymbol{\chi}^d$ and $\dot{\boldsymbol{\chi}}^d$ are furthermore converted to compute the desired velocity and acceleration of the FPR as follows:

- The desired linear velocity and acceleration of the platform is set by ${}^{Fp}\mathbf{v}_p^d$ and ${}^{Fp}\dot{\mathbf{v}}_p^d$ expressed in $\mathfrak{F}_{Fp}$;
- The desired angular velocity and acceleration of the platform can be computed by

$$\begin{aligned} {}^{p}\boldsymbol{\omega}_p^d &= \mathbf{D}({}^{Fp}\phi_p, {}^{Fp}\vartheta_p)\ {}^{Fp}\dot{\boldsymbol{\eta}}_p^d \\ {}^{p}\dot{\boldsymbol{\omega}}_p^d &= \dot{\mathbf{D}}({}^{Fp}\phi_p, {}^{Fp}\vartheta_p)\ {}^{Fp}\dot{\boldsymbol{\eta}}_p^d + \mathbf{D}({}^{Fp}\phi_p, {}^{Fp}\vartheta_p)\ {}^{Fp}\ddot{\boldsymbol{\eta}}_p^d \end{aligned} \tag{4.26}$$

  where the desired Euler angle rates are extracted from the teleoperation commands $\boldsymbol{\chi}^d$ and $\dot{\boldsymbol{\chi}}^d$, i.e. ${}^{Fp}\dot{\boldsymbol{\eta}}_p^d = \begin{bmatrix} {}^{Fp}\dot{\phi}_p^d & {}^{Fp}\dot{\vartheta}_p^d & {}^{Fp}\dot{\psi}_p^d \end{bmatrix}^T$ and ${}^{Fp}\ddot{\boldsymbol{\eta}}_p^d = \begin{bmatrix} 0 & 0 & {}^{Fp}\ddot{\psi}_p^d \end{bmatrix}^T$, $\mathbf{D}$ and $\dot{\mathbf{D}}$ are the matrices defined in (A.78) and (A.82) (see Appendix A.3).
- The desired leg angle rates $\dot{\boldsymbol{\theta}}_l^d$ are directly obtained from $\dot{\boldsymbol{\chi}}^d$, with their values on desired accelerations $\ddot{\boldsymbol{\theta}}_l^d$ set by zero.

Note that the desired accelerations are zero for the roll and pitch of the platform and the leg angles such that smoothness for those states susceptible to instability is ensured.

4.3.2 Decentralized Motion Controllers

In order to achieve the controller in a decentralized manner, meaning that each multi-rotor performs its own control based on the onboard measurements and estimations, the motion control law previously proposed in [Six, 2018a] and introduced in Section 1.2.2 is firstly adopted. The high-level motion control law is summarised as follows in line with the definition of the dynamic model and the robot states in this manuscript, but the method is equivalent to the one presented in the literature.

Considering the dynamic model of the FPR represented in form of (2.25) and the relationship between the thrust forces and the actuation wrench of (2.28), the vector of thrust forces is defined by an inversion of the dynamics as

$$\mathbf{f} = \left[\mathbf{J}(\mathbf{q})^T\right]^{\dagger}\Big(\mathbf{M}(\mathbf{q})\mathbf{u} + \mathbf{c}(\mathbf{q}, \boldsymbol{\nu})\Big) \tag{4.27}$$

Note that $\left[\mathbf{J}(\mathbf{q})^T\right]^{\dagger}$ represents the pseudo-inverse of the transpose of the Jacobian matrix $\mathbf{J}$; $\mathbf{q}$ and $\boldsymbol{\nu}$ are the generalised coordinates and velocity of the robot; $\mathbf{u}$ is an auxiliary input defined to minimise the tracking error by a PID regulation as

$$\mathbf{u} = \dot{\boldsymbol{\nu}}^d + \mathbf{K}_d(\boldsymbol{\nu}^d - \boldsymbol{\nu}) + \mathbf{K}_p\boldsymbol{\varepsilon} + \mathbf{K}_i \int \boldsymbol{\varepsilon}\ dt \tag{4.28}$$

where $\mathbf{K}_p$, $\mathbf{K}_i$, $\mathbf{K}_d$ are positive-definite diagonal matrices respectively for the PID gains, $\boldsymbol{\nu}^d$, $\dot{\boldsymbol{\nu}}^d$ are the desired velocity and acceleration, and $\boldsymbol{\varepsilon}$ is the tracking error that can be equivalently defined by (3.27). As discussed in Section 4.3.1, it is now given by

$$\boldsymbol{\varepsilon}(\mathbf{q}^d, \mathbf{q}) = \begin{bmatrix} \mathbf{0}_3 \\ \boldsymbol{\varepsilon}_o({}^{Fp}\mathfrak{q}_p^d, {}^{Fp}\mathfrak{q}_p) \\ \boldsymbol{\theta}_l^d - \boldsymbol{\theta}_l \end{bmatrix} \in \mathbb{R}^{6+n} \tag{4.29}$$

with ${}^{Fp}\mathfrak{q}_p$ the platform's orientation converted from ${}^{Fp}\mathbf{R}_p$ and ${}^{Fp}\mathfrak{q}_p^d$ its desired orientation computed from the desired roll and pitch ${}^{Fp}\phi_p^d$, ${}^{Fp}\vartheta_p^d$, all expressed in $\mathfrak{F}_{Fp}$. $\boldsymbol{\varepsilon}_o(.)$ is the orientation error defined in (3.28).

Then, the control input of the high-level control loop is further converted to determine the desired thrust magnitude and attitude for each multirotor similarly to the method presented in Section 3.3.2. The final commands sent to the low-level control on each multirotor can be defined by a vector named *control vector* and denoted by

$$\mathbf{u}_i = \begin{bmatrix} f_{t,i}^d \\ \mathfrak{q}_i^d \end{bmatrix} \in \mathbb{R}^5 \tag{4.30}$$

with $f_{t,i}^d$ the desired thrust magnitude and $\mathfrak{q}_i^d$ defined as the unit quaternion for the desired attitude of each multirotor i, which are afterwards tracked by the low-level attitude controllers.

As the low-level control is already performed onboard each multirotor independently, the only need is to decentralize the high-level computation of desired commands to the multirotors. One may notice that the motion control law of (4.27) can be done individually on all the multirotors. In this case, each one of them computes the full vector of the required thrust forces for all the multirotors but only uses the thrust vector belonging to itself to perform the next low-level control. The overall control on each multirotor is therefore deployed in a decentralized manner and receives feedback from decentralized sources, while listening to the common teleoperation commands remotely sent by an operator.

However, for each multirotor, the construction of the control law by (4.27) means the necessity of knowing some of the states reconstructed on the other multirotors, which are specifically the other leg angles and the linear velocities of the other multirotors. It is assumed that each multirotor has an estimation of the leg angle and its linear velocity

expressed in $\mathfrak{F}_{Fp}$, denoted by a vector named *information vector* as

$$\boldsymbol{\zeta}_i = \begin{bmatrix} \theta_i \\ {}^{Fp}\mathbf{v}_i \end{bmatrix} \tag{4.31}$$

and can apply the control law if it has knowledge of the states of the other agents, i.e. $\boldsymbol{\zeta}_j = \begin{bmatrix} \theta_j & {}^{Fp}\mathbf{v}_j^T \end{bmatrix}^T$, $\forall j \neq i$. Based on how these unknown states are obtained on the multirotor, two different decentralized controllers are proposed with the control diagrams illustrated in Fig. 4.3:

- Communicating controller (**C-controller**): sharing of information among all the multirotors to exchange the most recent measurements or estimates of $\boldsymbol{\zeta}_j$, $\forall j \neq i$ for each multirotor, supposing that the communication between every two of all the multirotors is reliable.
- Non-communicating controller (**NC-controller**): a communication-less method for which each multirotor assumes that each of the other multirotors perfectly tracks the desired states, and thus the unknown states are given by their desired values, i.e. $\boldsymbol{\zeta}_j^d = \begin{bmatrix} \theta_j^d & {}^{Fp}\mathbf{v}_j^{dT} \end{bmatrix}^T$, $\forall j \neq i$.

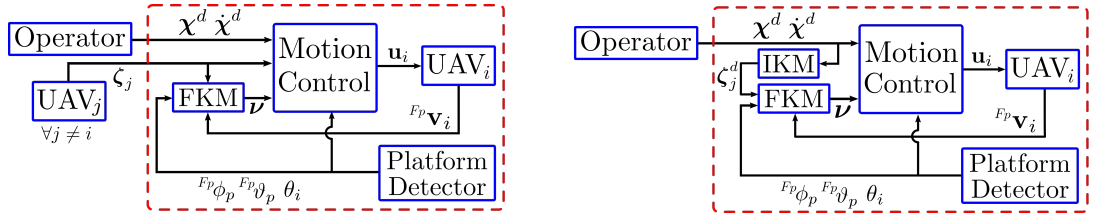


Figure 4.3 – The communicating (left) and non-communicating (right) decentralized control diagrams. IKM and FKM refer to the Inverse Kinematic Model and the Forward Kinematic Model, and the red dashed lines indicate the modules that are embedded on each multirotor.

Note that for NC-controller, the desired leg angles θ_i^d are directly obtained from the teleoperation commands, while the desired linear velocity of the other multirotors can be computed from the desired velocity commands $\boldsymbol{\nu}^d$ using the IKM detailed in (2.20), but expressed now in $\mathfrak{F}_{Fp}$ as

$${}^{Fp}\mathbf{v}^d = \mathbf{J}(\mathbf{q})\boldsymbol{\nu}^d \tag{4.32}$$

where ${}^{Fp}\mathbf{v}^d = \begin{bmatrix} {}^{Fp}\mathbf{v}_1^d & {}^{Fp}\mathbf{v}_2^d & \cdots & {}^{Fp}\mathbf{v}_n^d \end{bmatrix} \in \mathbb{R}^{3n}$ is the vector of desired linear velocities of all the multirotors, $\mathbf{J}(\mathbf{q})$ is the Jacobian matrix defined in (2.20) taking the robot state reconstructed now by ${}^{Fp}\mathbf{q}_p$ and $\boldsymbol{\theta}_l$ as input, and $\boldsymbol{\nu}^d$ is the desired velocity computed from the teleoperation commands presented in Section 4.3.1.

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 states, however it might be less accurate when tracking varying trajectories, in particular sudden changes in trajectory (such as a step signal) will cause differences between the desired and actual states large enough to affect the controller stability.

4.3.3 Decentralized Interaction Controller

As previously studied decentralized motion controllers, the impedance-based interaction control presented in Section 3.3 beforehand can be deployed in a decentralized manner as well. Similarly as presented in Section 4.3.2, the need is to decentralize the computation of the desired thrust magnitude and desired attitude for each multirotor by the high-level control. For this, the interaction control law is adopted from (3.31) and (3.32), which is summarised by

$$\mathbf{f} = \Big[\mathbf{J}(\mathbf{q})^T\Big]^\dagger \Big(\mathbf{M}(\mathbf{q})\mathbf{u} + \mathbf{C}(\mathbf{q},\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\mathbf{q}) - \hat{\boldsymbol{\tau}}_e\Big) \tag{4.33}$$

with $\mathbf{M}(\mathbf{q})$, $\mathbf{C}(\mathbf{q},\boldsymbol{\nu})$ and $\mathbf{g}(\mathbf{q})$ being respectively the generalised inertia matrix, the Coriolis matrix and the gravity wrench of the robot. $\hat{\boldsymbol{\tau}}_e$ is the external wrench estimate and $\mathbf{u}$ is an auxiliary input defined by the impedance system of (3.25)

$$\mathbf{u} = \dot{\boldsymbol{\nu}}^d + (\mathbf{M}^d)^{-1}\Big(\mathbf{B}^d(\boldsymbol{\nu}^d - \boldsymbol{\nu}) + \mathbf{K}^d\boldsymbol{\varepsilon}_q - \boldsymbol{\varepsilon}_\tau\Big) \tag{4.34}$$

where $\mathbf{M}^d$, $\mathbf{K}^d$, $\mathbf{B}^d$ are the desired mass, spring and damping matrices, the same as defined in (3.25), $\boldsymbol{\nu}^d$, $\dot{\boldsymbol{\nu}}^d$ are the desired velocity and acceleration computed from the teleoperation commands, $\boldsymbol{\varepsilon}_q$ is the tracking error equivalently given by (4.29) considering the available states computed from decentralized sources, and $\boldsymbol{\varepsilon}_\tau$ is the tracking error of the desired interaction wrench defined in (3.26). Note that the desired interaction wrench $\boldsymbol{\tau}_e^d$ can be defined by an off-line procedure or sent by an operator along with the teleoperation commands using a joystick.

Similarly as illustrated in Section 4.3.2, the vector of thrust forces resulted from the high-level control loop of (4.33) is then converted into the desired thrust magnitude and desired attitude, represented by the control vector $\mathbf{u}_i = \begin{bmatrix} f_{t,i}^d & \mathfrak{q}_i^{dT} \end{bmatrix}^T$ for each multirotor i, which can then be regulated by the low-level onboard controller.

It is furthermore noted that an estimation of the external wrench $\hat{\boldsymbol{\tau}}_e$ is necessary to

complete the impedance control law. To achieve this, the first-order momentum observer is chosen thanks to its advantageous behaviour regarding the convergence time and the stability previously validated in Section 3.5, with the estimates rewritten as

$$\hat{\boldsymbol{\tau}}_e(t) = \mathbf{K}_O\Big[\boldsymbol{\mathcal{P}}(t) - \int_{t_0}^{t}\Big(\mathbf{C}\big(\mathbf{q}(t),\boldsymbol{\nu}(t)\big)^T\boldsymbol{\nu}(t) - \mathbf{g}\big(\mathbf{q}(t)\big) + \boldsymbol{\tau}(t) + \hat{\boldsymbol{\tau}}_e(t-\Delta t)\Big)dt\Big] \quad (4.35)$$

with $\mathbf{K}_O$ the observation gains, $\boldsymbol{\mathcal{P}} = \mathbf{M}(\mathbf{q})\boldsymbol{\nu}$ the momentum of the robot, and $\boldsymbol{\tau}$ the actuation wrench that can be computed by the relationship of (2.28), knowing the vector of the thrust force commands for all the multirotors, i.e. $\mathbf{f} = \begin{bmatrix}\mathbf{f}_1^T & \mathbf{f}_2^T & \cdots & \mathbf{f}_n^T\end{bmatrix}^T \in \mathbb{R}^{3n}$.

The decentralized achievement of the control law by (4.33) and estimation by (4.35) requires however the knowledge of not only some of the states (i.e. leg angles and linear velocities), but also the commands of the other multirotors. This information should be accurate enough to ensure the stability of the entire system. It is therefore assumed that inter-agent communications are possible as the C-controller in Section 4.3.2 and each multirotor shares the most recent values of its states and thrust magnitude commands with others, of which the *information vector* is now defined as

$$\boldsymbol{\zeta}_i = \begin{bmatrix} \theta_i \\ {}^{Fp}\mathbf{v}_i \\ {}^{Fp}\mathbf{q}_i \\ f_{t,i}^d \end{bmatrix} \quad (4.36)$$

with θ_i being the leg i's angle computed by the multirotor i based on the geometric model, ${}^{Fp}\mathbf{v}_i$ the linear velocity and ${}^{Fp}\mathbf{q}_i$ the attitude estimated onboard with respect to $\mathfrak{F}_{Fp}$, and $f_{t,i}^d$ the desired thrust magnitude from the last commands computed.

Based on the information received on each multirotor i, i.e. $\boldsymbol{\zeta}_j,\ \forall j \neq i$, the robot coordinates and velocity vectors are known and the control law can be completed. For the control input required by the momentum observer, the thrust forces for all the multirotors can be further computed to be expressed in $\mathfrak{F}_{Fp}$

$${}^{Fp}\mathbf{f}_i = {}^{Fp}\mathbf{R}_i \begin{bmatrix} 0 \\ 0 \\ f_{t,i}^d \end{bmatrix}, \quad i = 1, 2, ..., n \quad (4.37)$$

with ${}^{Fp}\mathbf{R}_i$ representing the rotation matrix associated to the unit quaternion ${}^{Fp}\mathbf{q}_i$ that has been reconstructed on each multirotor and $f_{t,i}^d$ the most recent thrust magnitude command sent to the low-level controller. On a given multirotor i, the other states of

$f^d_{t,j}$ and ${}^{Fp}\mathfrak{q}_j$ ($\forall j \neq i$) are obtained by the sharing of information. The overall control diagram can be summarised by Fig. 4.4, in which the interaction control module includes the momentum observer and the impedance-based controller.

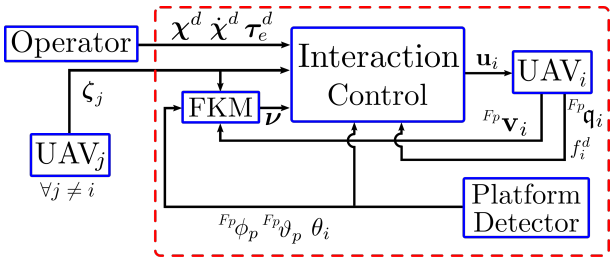


Figure 4.4 – Decentralized interaction control diagram. FKM refers to the Forward Kinematic Model, and the red dashed lines indicate the modules that are embedded in each multirotor.

This allows for each multirotor to construct the vector of thrust forces expressed in $\mathfrak{F}_{Fp}$, i.e. ${}^{Fp}\mathbf{f} = \begin{bmatrix} {}^{Fp}\mathbf{f}_1^T & {}^{Fp}\mathbf{f}_2^T & \dots & {}^{Fp}\mathbf{f}_n^T \end{bmatrix}^T \in \mathbb{R}^{3n}$, and then to perform the estimation of the external wrench, in which the platform's external force is now expressed in $\mathfrak{F}_{Fp}$, while the external moment of the platform and external moments of legs are still expressed relative to their own frames as initially defined in (3.2). Note that for constructing the thrust force vectors, the actual values of each multirotor's attitude are used rather than the desired ones. This is because each multirotor is not necessarily performing the estimation at the same time (which is however the case in centralized estimation) and the use of actual attitudes would allow to construct the external wrench estimates closer to the real values, better satisfying the controller stability requirements as discussed in Section 3.3.1.

In summary, the decentralized interaction controller requires stable communication between the multirotors for a non-biased external wrench estimation from decentralized sources of information on each UAV. The impedance-based controller decentralized on multiple UAVs can maintain its capability of tracking the desired trajectory and the desired interaction wrench sent by a human operator, as long as the inter-UAV communication is reliable. The overall decentralized scheme shows a teleoperation system allowing the operator to pilot the FPR potentially interacting with the environment.

4.3.4 Adjustment on Low-level Attitude Commands

After having obtained the desired commands $\mathbf{u_i}^d$ for each multirotor from the decentralized high-level controllers (no matter for motion control or interaction control), the desired attitude commands ${}^{Fp}\mathfrak{q}_i^d$ computed in $\mathfrak{F}_{Fp}$ should be adjusted to be expressed in the coherent local frame $\mathfrak{F}_{0i}$ of each multirotor, which is however an unknown inertial frame (referred to Section 4.2.3). It is known that the common reference frame $\mathfrak{F}_{Fp}$ can

express equivalent roll and pitch angles compared to any inertial frame, but has a different and unknown yaw. Therefore, the further adjustment on the attitude commands is necessary to provide coherent yaw angles to the low-level attitude controllers.

This can be done by computing the desired Euler angles in $\mathfrak{F}_{Fp}$ converted from ${}^{Fp}\mathbf{q}_i^d$, denoted by ${}^{Fp}\phi_i^d$, ${}^{Fp}\vartheta_i^d$ and ${}^{Fp}\psi_i^d$, where the desired roll and pitch angles of each multirotor in $\mathfrak{F}_{0i}$ should be the same as defined in the flat frame $\mathfrak{F}_{Fp}$ (i.e. ${}^{0i}\phi_i^d = {}^{Fp}\phi_i^d$, ${}^{0i}\vartheta_i^d = {}^{Fp}\vartheta_i^d$). The expression of the desired yaw is however different as there is unknown relative yaw between the two frames. It can be calculated knowing an onboard IMU measurement of the multirotor's actual yaw ${}^{0i}\psi_i$ and an estimate of the relative yaw ${}^{p}\psi_i$ extracted from ${}^{p}\mathbf{R}_i$ by

$$ {}^{0i}\psi_i^d = {}^{0i}\psi_i + ({}^{Fp}\psi_i^d - {}^{p}\psi_i) \tag{4.38} $$

Note that ${}^{p}\psi_i = {}^{Fp}\psi_i$ as the flat frame $\mathfrak{F}_{Fp}$ and the platform's body frame $\mathfrak{F}_p$ share a common yaw. This relationship allows to correct the desired yaw commands to be coherent with the multirotor's yaw measurements relative to the unknown inertial frame $\mathfrak{F}_{0i}$. Actually, instead of converting the desired yaw directly from $\mathfrak{F}_{Fp}$ to $\mathfrak{F}_{0i}$, a difference between the desired and actual yaw angle is computed, i.e. $({}^{Fp}\psi_i^d - {}^{p}\psi_i)$, which is then set to correct the desired yaw ${}^{0i}\psi_i^d$ based on its onboard measurements ${}^{0i}\psi_i$.

The desired attitude of each multirotor in its own inertial frame $\mathfrak{F}_{0i}$ can finally be determined by the quaternion ${}^{0i}\mathbf{q}_i^d$ which is converted from the Euler angles ${}^{0i}\phi_i^d$, ${}^{0i}\vartheta_i^d$ and ${}^{0i}\psi_i^d$ and tracked by the onboard quaternion-based attitude controller.

4.4 Experimental Validation

In this section, the proposed decentralized controllers are validated by real-world experiments on an FPR prototype detailed in Appendix B. The decentralized motion controllers detailed in Section 4.3.2 were firstly validated using emulated camera detection data by the Motion Capture (MOCAP) system, with a video of experiments available at https://youtu.be/Rmmq0-b2zws. Then, real-world implementation of pose estimation with ArUco marker system based on monocular vision has been done, which is shown in the experimental validation of the decentralized interaction controller presented in Section 4.3.3. A video of experiments can be found at https://youtu.be/T0-fycf1V28.

During the experiments demonstrated in this section, the FPR was flown by teleoperation using a 4-axis gamepad controller with desired values set by axes and buttons as shown in Fig. 4.5. Note that the values for all the leg angles are set to be identical in the

experiments. All the flights were performed inside a $4 \times 6 \times 3.5$ m flight arena equipped with an 8-camera Qualisys MOCAP system, streaming data over a 5 GHz wifi network. The decentralized controllers have been implemented onboard the companion computer of each quadrotor at 50 Hz. The onboard ArUco detection rate is 20 Hz for the experiments when the onboard vision is used, with the RGB images of size 640×480 pixels captured by Raspberry Pi Camera V2. It is additionally noted that the communication between the onboard Raspberry Pi and Pixhawk autopilot for each quadrotor was handled using Ethernet (referred to Appendix B), in order to reduce the communication burden when the onboard camera is enabled.

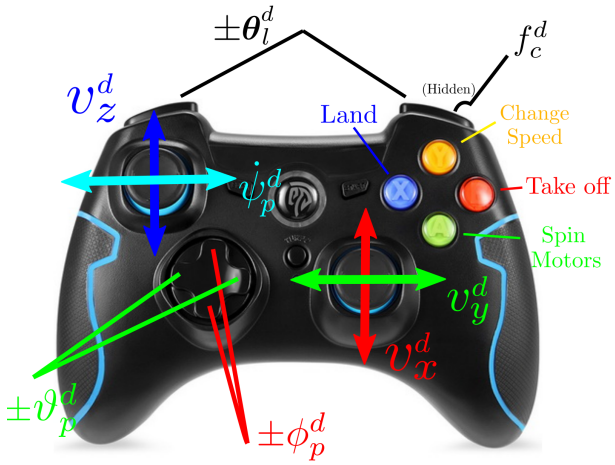


Figure 4.5 – Demonstration of 4-axis gamepad for teleoperating the FPR.

4.4.1 Experiments on Decentralized Motion Controllers

In the experiments validating the decentralized motion controllers presented in Section 4.3.2, the MOCAP system is used to emulate the relative pose that can be estimated by the camera and vision-based detection techniques. In addition, the onboard velocity estimation of each multirotor is enhanced by the MOCAP system in these preliminary experiments. The integration of optical sensors can however be done and is only related to technological concerns. More precisely, the MOCAP system is used for:

- Extracting the pose of the platform in each multirotor's local frame (i.e. extracting ${}^p\mathbf{T}_{bi}$). It could therefore be replaced by a monocular camera observing the ArUco marker as presented in Section 4.2.1. To represent the measurement noise, Gaussian noises with a standard deviation of 2 cm for translations and 2° for rotations are added along/around each axis of the relative pose.
- Sensor fusion onboard the UAVs to improve the estimates of their body-frame linear velocities. It could thus be replaced by onboard optical flow sensors.

- Providing ground truth in post-flight analysis.

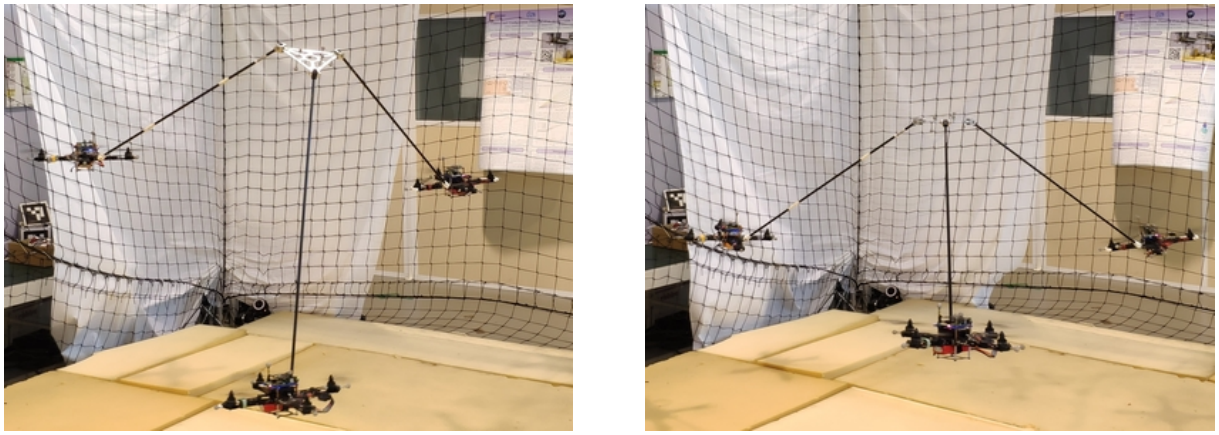
Two experimenting scenarios have been conducted to verify the effectiveness of the decentralized controllers, respectively the C-controller and NC-controller based on the PID regulation for a free-space flight of the FPR, which are:

1. Control of Orientation and Internal Configuration
2. Precise Positioning with Teleoperation

In the first scenario, the decentralized controller is tested to control the platform orientation and the internal configuration (i.e. leg angles). Then the precise positioning of the platform performed by a human pilot with teleoperation is shown in the second scenario.

Scenario I: Control of Orientation and Internal Configuration

The first experiment demonstrating the validity of the control method is simply to fly the FPR without any specific task, and track the desired configuration ϕ_p^d, ϑ_p^d, and $\boldsymbol{\theta}^d$ given by the operator. Fig. 4.6(a) shows a configuration with a large angle on the platform roll, while Fig. 4.6(b) shows a wide leg configuration (with small values on the leg angles). The results with the centralized controller previously presented in the literature making full use of MOCAP and computing all control in $\mathfrak{F}_0$ are also provided for comparison. The PID control gains of the centralized and decentralized controllers are given in Table 4.1. Note that the saturation on the integral term is enabled in the PID regulation to avoid integral wind-up [Azar, 2015].



(a) Configuration with large platform roll

(b) Configuration with small leg angles

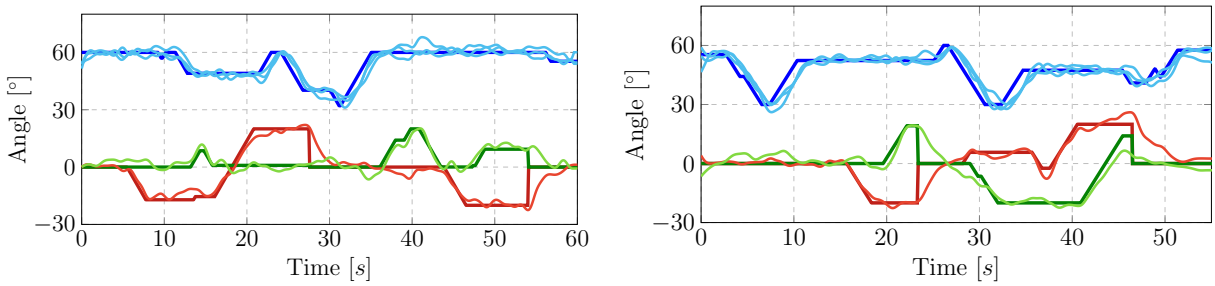
Figure 4.6 – Control of the platform orientation and leg angles of the FPR using decentralized C- and NC-controller.

The desired and measured states for the control of the platform orientation and leg angles using the C- and NC-controller are plotted in Fig. 4.7. The result of the controlled states using both decentralized controllers as well as that of the centralized controller

Robot State	Centralized Controller			Decentralized Controller		
	K_p	K_d	K_i	K_p	K_d	K_i
$\mathbf{p}_{p,xy}$	9	6	25	*	9	*
$\mathbf{p}_{p,z}$	16	8	50	*	8	*
$\mathbf{o}_{p,xy}$	9	6	1	9	6	1
$\mathbf{o}_{p,z}$	9	6	0.5	*	6	*
$\boldsymbol{\theta}$	9	6	10	9	6	0.5

Table 4.1 – Numerical values for the PID control gains. The control gains are decoupled along/around each axis of the robot state. $\mathbf{p}_{p,.}$, $\mathbf{o}_{p,.}$ stand for position and orientation axes of the platform decoupled between x or y and z axes. $\boldsymbol{\theta}$ represents the leg angles on which the gains are the same. (*) indicates that the corresponding term is not available in the decentralized controller.

tracking a trajectory with similar ranges of values and continuous accelerations are summarised in Table 4.2.



(a) Results of C-controller (b) Results of NC-controller

Figure 4.7 – Results of the tracking on the platform roll and pitch, as well as the leg angles using C- and NC-controller. The top curves show the desired value for all the legs $\boldsymbol{\theta}_l^d$ (blue line) and measured ones $\boldsymbol{\theta}_l$ (light blue lines). The bottom curves show the tracking of the platform orientation, i.e. desired roll ϕ_p^d (dark red line), actual roll ϕ_p (light red line), desired pitch ϑ_p^d (dark green line) and actual pitch ϑ_p (light green line).

Robot State	C-controller	NC-controller	Centralized
ϕ_p [°]	4.1	5.0	2.5
ϑ_p [°]	2.9	4.2	3.3
mean $\boldsymbol{\theta}_l$ [°]	2.8	3.8	2.1

Table 4.2 – Results on the root-mean-square error (RMSE) for the experiment of configuring the platform orientation and leg angles. Note that mean $\boldsymbol{\theta}_l$ refers to the mean RMSE of three leg angles.

It can be seen in the plots that the platform orientation and the internal configuration track quite well their desired values. 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 vastly superior to the two decentralized methods. In fact, as the centralized controller doesn't have any measurement noise (outside that of MOCAP which is negligible) 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.

Scenario II: Precise Positioning with Teleoperation

The second experimental scenario shows that fine positioning using eye-to-platform teleoperation is achievable. Three targets (the tennis balls) are suspended in the flight arena as shown in Fig. 4.8. An operator flies the FPR with the task of positioning the platform at each of the three targets. 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 centre of the platform, which can be confirmed by the stabilisation of the positioning error around a small value near zero corresponding to the radius of the tennis ball. After the time is elapsed, a new target 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 investigated in the experiments. The evolution of the platform positioning error towards the targets using the C- and NC-controller over the course of the full experiment in 165 s is shown in Fig. 4.9. Note that the successful examples in Fig. 4.9(b) and (e) show that the positioning error stabilises below 0.05 m, corresponding to the radius of the ball, which indicates the ball has been successfully picked up. The unsuccessful examples are shown in Fig. 4.9(c) and (f) as the positioning error is never stabilised. The large oscillations in positioning errors for the failed tasks correspond to the ball swinging due to a collision with the platform early in the task.

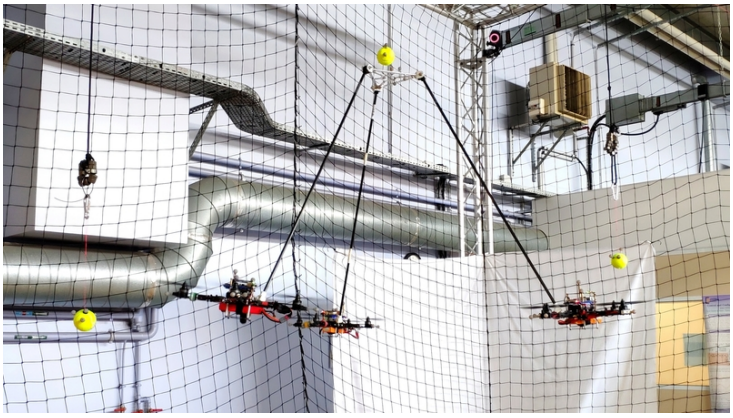
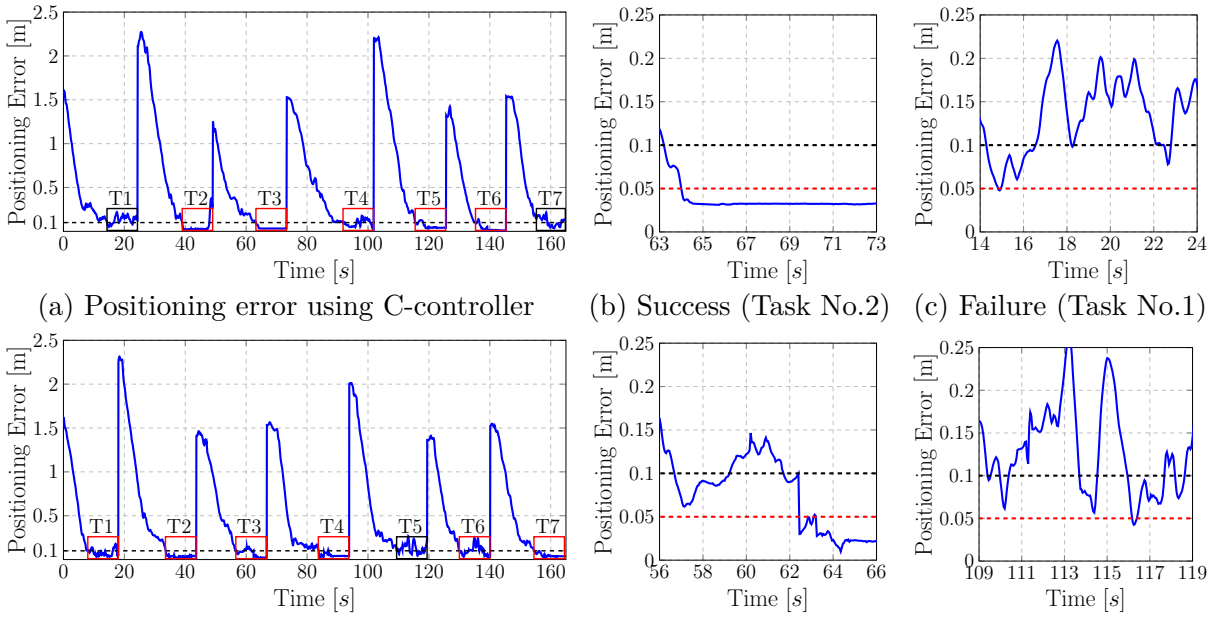


Figure 4.8 – Fine positioning of the platform with teleoperation towards the tennis balls hanging in the air.

There was little difference in performing the tasks between the C-controller and NC-



(a) Positioning error using C-controller (b) Success (Task No.2) (c) Failure (Task No.1)

(d) Positioning error using NC-controller (e) Success (Task No.3) (f) Failure (Task No.5)

Figure 4.9 – Evolution of positioning error of the platform during the experiment using respectively C-controller (top) and NC-controller (bottom). Examples of successful and failed tasks are shown in (b) and (c) for C-controller, (e) and (f) for NC-controller. The black dashed lines represent the criteria for stating the precise positioning task, while the red dashed lines show the criteria for a successful task. The rectangle boxes in (a) and (d) refer to individual positioning tasks, with successful tasks in red and failed ones in black.

controller. From the operator's perspective, both controllers were equivalent in terms of stability and positioning accuracy. 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 not straightforward to standardize between tests.

In addition, the configuration states of the FPR were well tracked in both cases as shown in Table 4.3, with RMSE values almost half of what they were in Scenario I (since the configuration states had constant reference trajectories). However, a significant remark is that in Scenario I, the C-controller was better across all configuration states, while in Scenario II 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 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 occurred in the C-controller but not necessarily the NC-controller, probably due to communication delays.

Robot State	C-controller	NC-controller
ϕ_p [°]	2.4	2.1
ϑ_p [°]	2.1	1.8
mean $\boldsymbol{\theta}_l$ [°]	1.8	1.6
Success rate	5/7	6/7

Table 4.3 – Results on the root-mean-square error (RMSE) of the robot configuration during the precise positioning of the platform, comparing the configuration tracking error as in Table 4.2. The last row is the number of successful positioning tasks accomplished in the experiment.

While these results have demonstrated centimetre-level precise for teleoperation using both C- and NC-controller, two factors likely contributed to the difficulty in accomplishing manipulation tasks:

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

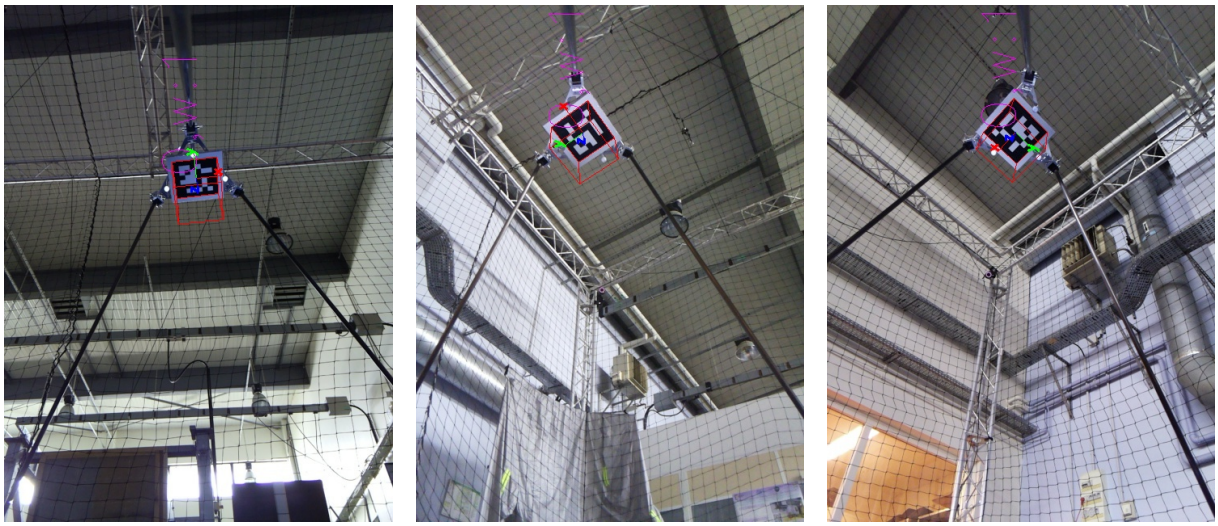
Despite these difficulties, the decentralized motion controllers using either a communicating or non-communicating scheme have been verified and shown to be able to regulate the robot configuration (platform roll and pitch, and leg angles) as well as to achieve precise positioning through eye-to-platform teleoperation.

4.4.2 Experiments on Decentralized Interaction Controller

As the experimental results in the previous section have validated the decentralized controllers for free-space flight with emulated camera measurements by MOCAP, a more realistic experimental setup is considered in this section, involving the relative pose estimation by ArUco marker system in real-time and the controller handling the physical interaction with the environment.

A fiducial marker is attached beneath the platform, with the size of 0.109 m and ID 0 selected among the ArUco marker dictionary "ARUCO_MIP_36h12" [Garrido-Jurado, 2016]. Fig. 4.10 shows the images captured by the onboard Raspberry Pi cameras and the detection results by ArUco detection library on three multirotors. Note that these images are captured when the UAVs are motionless and staying on the ground (images are however rotated 90° counter-clockwise), but the detection algorithm is capable of tracking the moving marker to ensure its robustness to the image noise and blur due to the camera motion [Romero-Ramirez, 2021]. Based on the detected relative pose between the marker and a camera frame represented by ${}^{Ci}\mathbf{T}_M$ and with knowledge of the global poses of the

frames by the MOCAP system, the fixed transformations ${}^{bi}\mathbf{T}_{Ci}$, ${}^{p}\mathbf{T}_{M}$ required to compute the relative pose of ${}^{bi}\mathbf{T}_{p}$ on each multirotor can be calibrated, with the calibration process and results detailed in Appendix D.



(a) Detection on multirotor 1 (b) Detection on multirotor 2 (c) Detection on multirotor 3

Figure 4.10 – Visualisation of the ArUco marker detected onboard each of three multirotors of the FPR from different angles of view. The cubic boxes and the three-axis arrows represent the estimated pose of the detected marker relative to the camera frame. On each image, the number 0 indicates the detected marker ID, and the values following the keyword "w:" are the percentage of confidence between 0 and 1.

Even though the ArUco detection library seems robust enough to perform the pose estimation (with the estimation noise and error on translation and rotation states in comparison to the MOCAP data given in Table 4.4), the real-time results of this detection algorithm during the experiments still suffer from other issues, such as the sensitivity to lightening conditions (too bright or too dark), the marker loss in camera's field of view, etc. Therefore, the emulated data is still used to make up for these defective situations. Additionally, the MOCAP system is used in the following experiments for:

- Emulating the relative pose ${}^{p}\mathbf{T}_{bi}$ with the same noises presented in Section 4.4.1 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°). This is required for safety reasons as the detection failure will inevitably cause instabilities and crashes, it may however be handled by redundant detection from other sources of sensors in the future.
- Emulating a high-level perception of the environment to obtain the velocity commands for x, y and yaw axes in experimenting scenario II. It can be achieved by adding an additional camera on the platform to estimate the pose to the contact surface.

- Sensor fusion onboard the UAVs to improve the linear velocity estimates of each UAV. It could thus be replaced by onboard optical flow sensors.
- Providing ground truth in post-flight analysis.

Estimation State	Estimation Noise (STD)		Absolute
	ArUco	MOCAP	Difference
Translation [mm]	0.22	0.03	2.61
Rotation [°]	0.02	0.02	0.46

Table 4.4 – Estimation noise of the ArUco detection algorithm [Romero-Ramirez, 2021] represented by standard deviation (STD) in comparison to the MOCAP noise. The data was recorded with a static marker whose pose was being estimated using both ArUco and MOCAP (with additional settings for the detection tags) systems over 80 s. The absolute differences between ArUco and MOCAP results are seen as the estimation error.

Several experimenting scenarios are set up to validate the decentralized interaction controller based on the impedance-based control law along with the first-order wrench observer (FOWO) estimating the external wrench, including:

- Precise positioning of the platform towards a target and pick-up of the load
- Contact-based interactions with an object in the environment

As the impedance-based controller has been proven to be able to handle both free-space flights and the physical interactions with the environment, the estimation and control gains are set by the same values for all the experiments, which are summarised in Table 4.5.

Robot State	K_O	M^d	B^d	K^d
Platform position	2	5	10	*
Platform orientation	1	8	25	25*
Leg angles	1	5	20	15

Table 4.5 – Estimation and control gains applied in the experiments. *The stiffness coefficient for the platform position and yaw is meaningless in the decentralized controller.

Scenario I: Precise Positioning and Pick-up of Load with Teleoperation

In this experimenting scenario, the platform is teleoperated by a human operator using a gamepad joystick shown in Fig. 4.5 for the precise positioning towards a target (tennis ball) hanging in the air, with a load suspended below corresponding to a total weight of about 400 g, as shown in Fig. 4.11. 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 UAVs. Then the human operator pilots the platform by sending linear velocity and yaw rate commands, similarly to the precise positioning experiment shown in Section 4.4.1, but

with the task of picking up the load. The impedance-based controller thus regulates the robot configuration while tracking the desired velocity commands sent by the operator, involving a free-space control before touching the object and then the interaction with the load as its mass is no longer negligible.

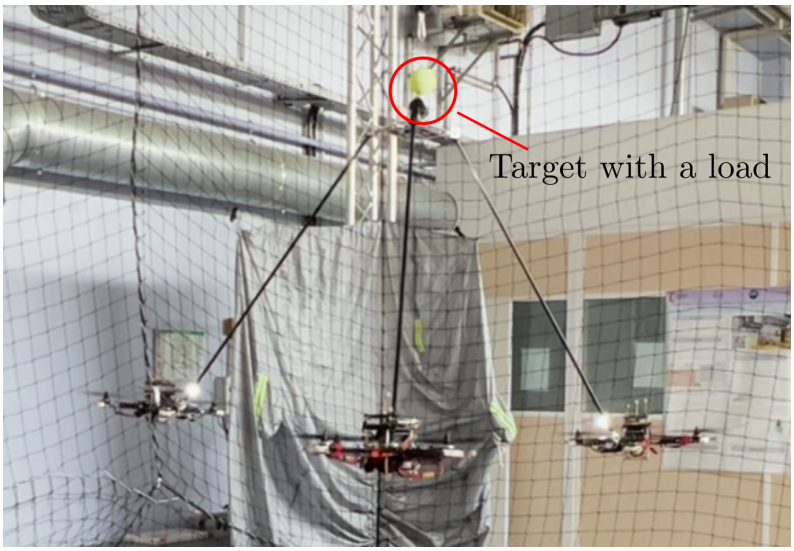
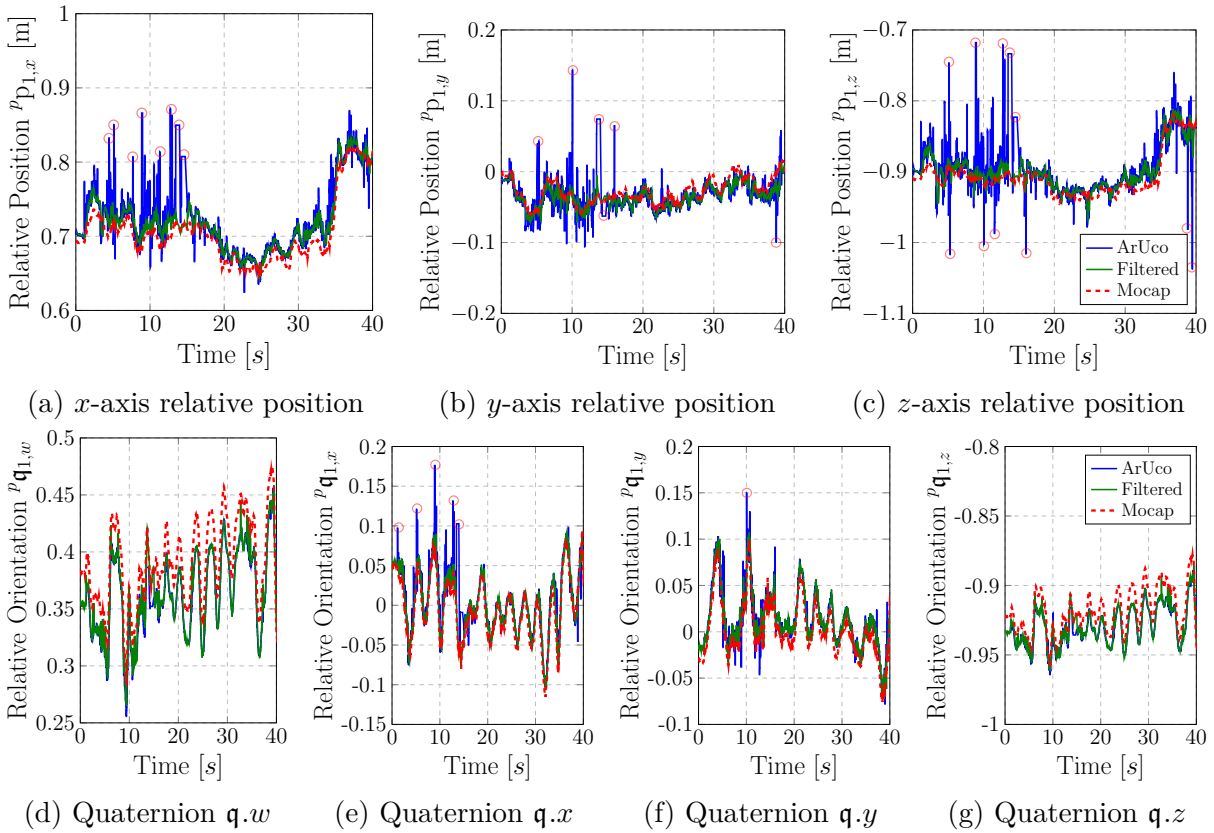


Figure 4.11 – Precise positioning of the platform towards a target (tennis ball) suspended with a load and pick-up of the load.

The real-time pose estimation between the platform and the UAVs by ArUco detection algorithm and EKF-based filtering is firstly evaluated. The results of the multirotor 1 are plotted in Fig. 4.12, consisting in the estimated relative pose, the filtered output from the EFK and the ground truth known by MOCAP. Note that each axis of the relative position ${}^{p}\mathbf{p}_1$ and the orientation represented by the quaternion ${}^{p}\mathbf{q}_1$ are separately analysed.

It can be noticed that except for several outliers, the ArUco detection results are very close to the ground truth, and the measurement noise is well filtered from the outputs of the EKF. Table 4.6 shows the RMSE of the detection and filtering results for all the UAVs using onboard ArUco detection and EKF techniques. The RMSE of the EKF outputs are all limited within 3 cm for relative position and 3° for relative orientation, which has validated the effectiveness of the relative pose estimates used afterwards in the control. Although the detection failures and interruptions illustrated by the outliers in Fig. 4.12 had to be compensated by MOCAP data, the percentage of successful detections presented in Table 4.8 shows a relatively good detection rate in the real-world scenario.



(a) x-axis relative position (b) y-axis relative position (c) z-axis relative position

(d) Quaternion $\mathbf{q}.w$ (e) Quaternion $\mathbf{q}.x$ (f) Quaternion $\mathbf{q}.y$ (g) Quaternion $\mathbf{q}.z$

Figure 4.12 – Relative pose of the multirotor 1 expressed in the platform frame. Blue curves represent the pose estimated by the ArUco detection algorithm [Romero-Ramirez, 2021], green curves are the filtered results from the EKF and the red dashed curves are the ground truth measured by the MOCAP. The outliers corresponding to the detection failures or interruptions are pointed out by red circles.

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 4.6 – Root-mean-square error (RMSE) of the ArUco detection and EKF-based filtering results on the relative pose between each UAV and the platform during one positioning task. Note that the roll, pitch and yaw angles for the relative orientation are converted from the quaternions ${}^{p}\mathbf{q}_{i}$.

Since the control of the FPR by teleoperation involves a human-in-the-loop situation, the operator was asked to perform the same task 6 times to reduce the human factor affecting the results, during which the operator was given 45 s to position the platform

towards the target and to pick up the load attached below. From the pilot's point of view, the system is quite easy to manipulate, similar to the manual pilot of a single quadrotor without many difficulties. The results on the evolution of positioning errors for the 6 tasks (5 success and 1 failure) are shown in Fig. 4.13, with successful tasks well identified by the positioning error being stabilised below 15 cm. The z-axis external force of the platform estimated by UAV 1 during the successful task ① is plotted in Fig. 4.14, from which the pick-up of the load is illustrated with the estimated values converged to the ground truth of around -4 N. It is also remarked that there is very little difference in the external wrench estimates performed on different multirotors, which is reasonable as the decentralized estimation of the external wrench might only be affected by communication delays (negligible in the experiment). The overall success rate (5/6) of the pick-up tasks is acceptable with RMSE on the robot configuration presented in Table 4.7 , showing that the fine positioning of the platform by teleoperation considering the interaction with the environment is achievable.

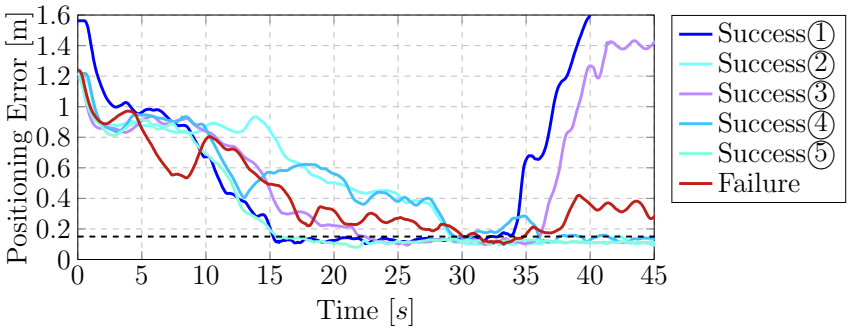


Figure 4.13 – Evolution of positioning errors of the platform with respect to the target in 6 performed tasks. Note that the black dashed line is the criterion for a successful task, corresponding to the distance from the target's centre to the attached load (≈ 15 cm). The blue and purple curves (Success ① and ③) are two successful cases where the load was released before the end of the given time (the reason why their positioning errors increase at the end).

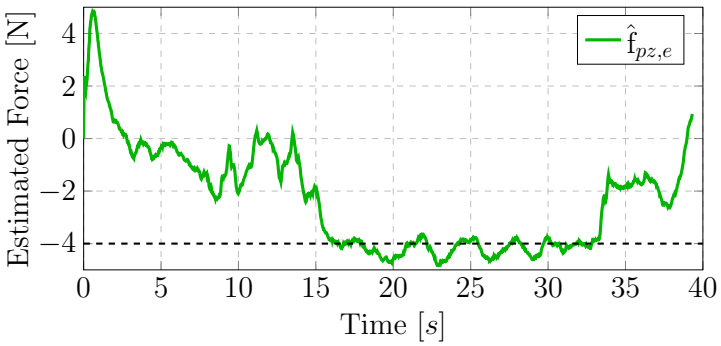


Figure 4.14 – Evolution of the z-axis external force estimates of the platform during the successful experiment (Success ①). Note that only the force estimated by UAV 1 is plotted, but the estimates on the other UAVs are identical to this curve.

Scenario II: Contact-based Interactions with the Environment

This experiment has been done in a more complex scenario where the FPR is controlled with teleoperation to interact with an object in the environment and exert a certain amount of force on it. It can therefore be seen as an extension of the contact-based interaction experiments presented in Section 3.5.4, but with onboard vision handling the robot pose estimation, and the estimation and control methods decentralized on each UAV. Similarly to the experimental setup presented in Section 3.5.4, a wooden board of dimension 30×30 cm is hanging in the air with tightly stretched ropes and additional masses to sufficiently consider it as a stiff object in the environment. A region of dimension 10×10 cm is pointed out, with its centre considered as the target for contact-based interactions. Fig. 4.15 illustrates the interaction experiment performed by the FPR.

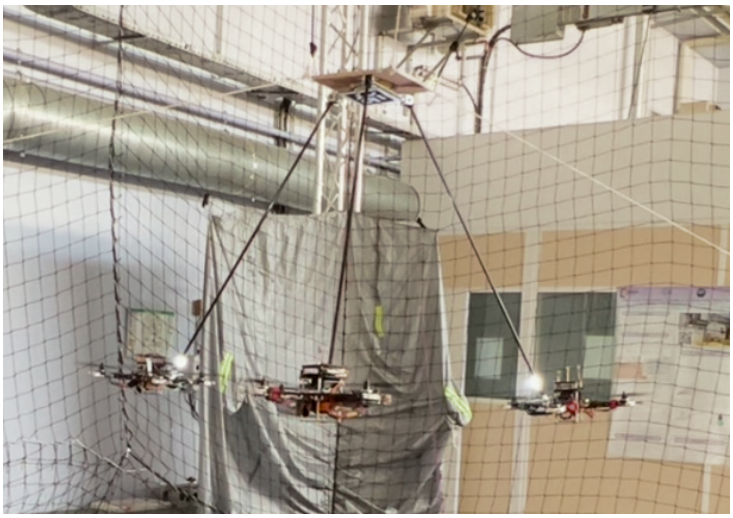


Figure 4.15 – Contact-based interaction experiment of the FPR during which the platform is controlled to exert forces in normal direction on a board fixed in the environment.

Since the teleoperation still involves the human in the loop, which is undesired for validating the methodology independent of the proficiency of the human operator. 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 desired yaw rate ${}^{Fp}\dot{\psi}_p$ using the MOCAP data. The desired velocity can be computed based on the proportional law given as

$$\begin{aligned} {}^{Fp}\mathbf{v}_{p,x/y} &= \lambda_p {}^{Fp}\mathbf{p}_{t,x/y} \\ {}^{Fp}\dot{\psi}_p &= \lambda_\psi {}^{Fp}\psi_t \end{aligned} \tag{4.39}$$

where ${}^{Fp}\mathbf{p}_{t,x/y}$ is the relative position (in x and y axes) of the target with respect to $\mathfrak{F}_{Fp}$, ${}^{Fp}\psi_t$ is the relative yaw of the target with respect to $\mathfrak{F}_{Fp}$, both known by the MOCAP, and the proportional gains for controlling the platform to approach the target with

aligned yaw are set by $\lambda_p = 0.4$, $\lambda_\psi = 0.25$. Note that the knowledge of ${}^{Fp}\mathbf{p}_{t,x/y}$ and ${}^{Fp}\psi_t$ can be done by adding an additional camera on the platform pointing towards the target and integrating an online pose estimation achievable by ArUco detection algorithm [Romero-Ramirez, 2021] or other model-based/learning-based algorithms. A visual servoing procedure for minimising some image features to attain the contact position may also be done [Chaumette, 2004; Lippiello, 2016]. Therefore, the experiment was conducted in a hybrid scheme: the platform's linear velocity ${}^{Fp}\mathbf{v}_{p,x/y}$ and yaw rate ${}^{Fp}\dot{\psi}_p$ are controlled by high-level control law (4.39) with emulated pose estimation (using MOCAP), the z-axis linear velocity of the platform and the desired contact force are controlled by the human operator, while the leg configuration and the platform orientation are regulated with desired values set by the teleoperation commands.

Fig. 4.16 shows the evolution of the positioning error between the platform and the target, and Fig. 4.17 are the desired and estimated values of the contact force between the platform and the board. It has taken about 27 s for the FPR to establish contact with the board, and during the steady contact (in the time interval of [27, 55] s) the positioning error is stabilised at a value lower than 6 cm. From Fig. 4.17, the estimated contact force well tracks the desired values of 5 N during the interaction, except the peak between [27, 30] s probably because an impact occurs at the beginning of the contact.

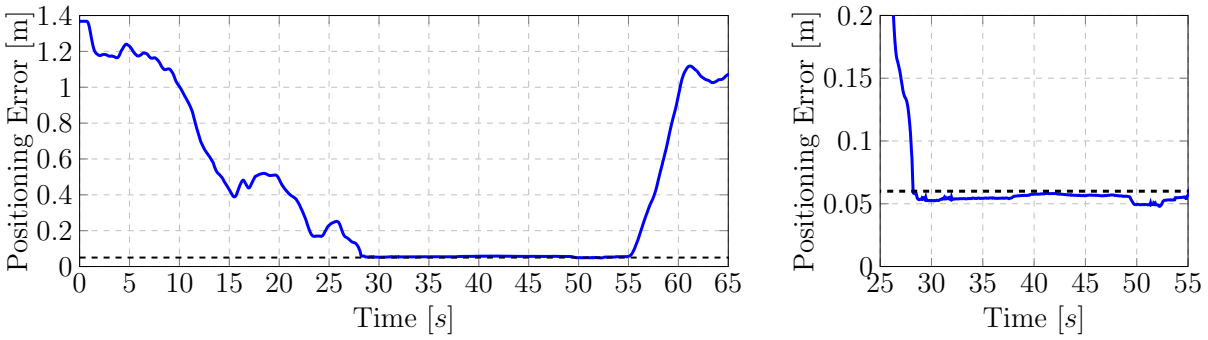


Figure 4.16 – Evolution of the positioning error between the platform and the target during the interaction experiment.

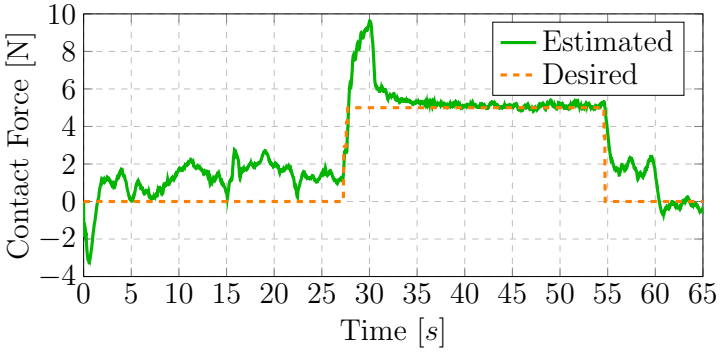


Figure 4.17 – Evolution of the desired and estimated values of the contact force between the platform and the board during the interaction experiment, recorded on UAV 2.

These results have shown that the proposed decentralization of the impedance-based controller on each UAV is capable of positioning the platform with good precision and exerting a certain range of forces along the normal direction of the platform. The success rates of the online ArUco detection are summarised in Table 4.8 and the RMSE of the robot configuration (i.e. platform roll and pitch and the leg angles) are presented in Table 4.7, which demonstrates a considerably favourable result in a real-world scenario. This experiment has shown the ability of the FPR to accomplish real manipulation tasks, such as contact-based inspection or repair on the surface of industrial infrastructures.

Robot State	Scenario I	Scenario II
ϕ_p [°]	5.4	4.6
ϑ_p [°]	5.9	5.9
mean $\boldsymbol{\theta}_l$ [°]	5.4	5.3

Table 4.7 – Root-mean-square error (RMSE) on the tracking of the platform orientation and leg angles. Note that mean $\boldsymbol{\theta}_l$ indicates the mean value of the RMSE of three leg angles. The results on Scenario I are computed by the average values of 6 performed positioning tasks.

UAV	Scenario I	Scenario II
1	93.1%	95.3%
2	88.3%	89.4%
3	88.7%	91.2%

Table 4.8 – Successful detection rates for all the UAVs in two experimenting scenarios. Note that a successful detection is that the detection errors are less than 10 cm for position and 10° for rotation.

The conducted experiments have therefore validated the interaction controller decentralized on each UAV to achieve interaction tasks such as the pick-up of a load and contacts with an object in the environment. The overall success rate of the onboard pose estimation by monocular cameras based on ArUco detection and EKF algorithms is from 88.3% to 95%. Compared to the previous results using emulated camera data, the results of the robot configuration tracking given by Table 4.7 seem very acceptable when switching to a real-time pose estimation technique based on onboard vision. In addition, it has been observed in the experiment that the decentralized controller shows its advantageous feature compared to the centralized one especially in the emergency case where the communications among the UAVs were lost, while they are still capable of maintaining their own control and landing properly without crashing (by using desired states for the other UAVs as investigated in NC-controller in Section 4.3.2). This work is thus an encouraging step towards the control and teleoperation of the FPR with only intrinsic measurements

to accomplish manipulation tasks.

It is however found in the experiment that the platform angles and the leg configuration should be regulated within a small interval to ensure the marker visibility from the camera attached to each UAV. The experiments involving the control of the platform orientation in free space were conducted (as shown in Fig. 4.18), whose results were unfavourable because of the loss of detection during longer series of control iterations (the successful detection rate has dropped down to 61% for one UAV in the worst case). This is therefore a limitation of the proposed pose estimation technique based on vision. Although the marker visibility issue may potentially be solved by attaching to each UAV a specific marker on the platform with a better angle of view, the failure cases related to environmental conditions such as brightness still present the frustrating aspect of the vision-based method.

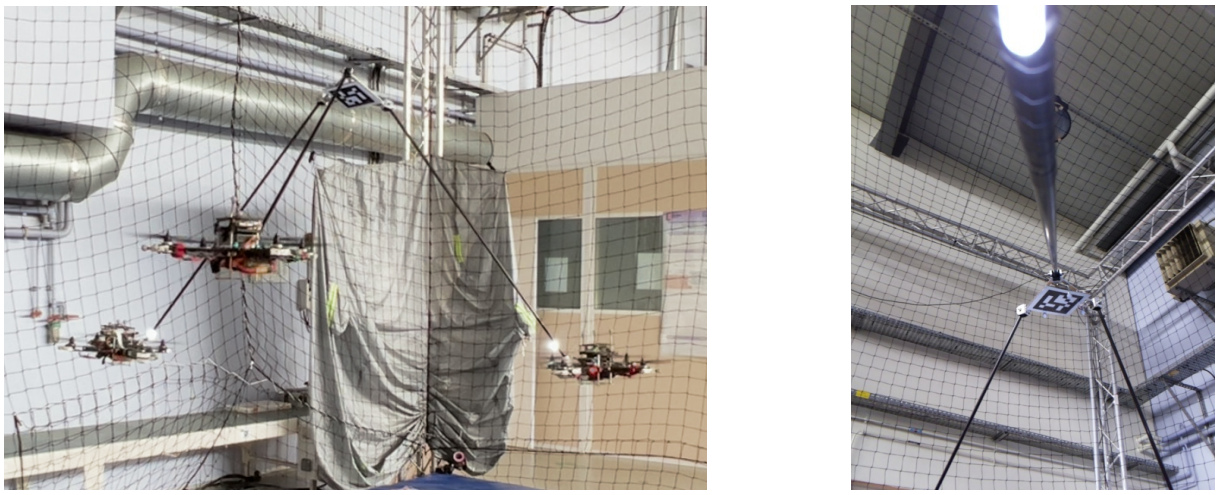


Figure 4.18 – Unfavourable configuration with large platform angle for onboard detection of the ArUco marker. Right figure shows the image captured on UAV 3 on which the marker tracking is lost.

4.5 Conclusion

In this chapter, the methodology related to the decentralized estimation and control of the Flying Parallel Robot (FPR) has been investigated. A vision-based method for estimating and reconstructing the robot pose has been presented, which is constructed on basis of ArUco marker detection system and EKF-based filtering technique. Then several decentralized controllers have been proposed according to the application scenarios: for motion control, C- and NC-controllers depending on whether the sharing of information between the UAVs is possible, and a communicating impedance-based controller considering the interaction with the environment. Extensive experimental results for all

the proposed controllers have shown the effectiveness of the method dealing with precise positioning of the platform, and contact-based interactions with the environment. These methods are applicable to other multi-UAV parallel manipulators as the FPR for applications such as the pick-and-transport of objects and contact-based inspections.

Even though the overall success rate of the onboard detection is quite high in the experiments using the vision-based technique for the pose estimation, the requisites on the lightening condition and the robot configuration (i.e. narrow intervals of platform orientation and leg angles for marker visibility) are quite strict for real-world applications. Therefore, future works may focus on the improvement of the onboard detection algorithm to be robust to the environmental conditions, or integrate other sources of sensors for the pose estimation such as the IMU on the platform for the orientation, and encoders for measuring the leg angles. The exploration of the outdoor scenario for the FPR can also be done, where the global positions of UAVs can be measured by low-cost GPS and possibly enhanced by time-of-flight-based sensors measuring the internal distances between the UAVs.

Chapter 5

Wrench Feasibility Analysis

In this chapter, a wrench feasibility analysis on the Flying Parallel Robot (FPR) is investigated. The objective of this analysis is to build knowledge of available wrench the FPR can exert/support in quasi-static equilibrium circumstances. The content of this chapter is structured as follows: a general introduction to the wrench feasibility analysis is firstly presented, with some basic concepts introduced in the second section. Then a detailed definition of the Available Thrust Set (ATS) and the computation of the Available Wrench Set (AWS) on the full degrees of freedom of the FPR are detailed in the third section. A comprehensive static feasibility analysis is illustrated in the fourth section, which is then applied in order to determine the optimal leg configurations maximising the wrench feasibility of the platform in different platform orientations in the fifth section.

5.1 Introduction

The wrench feasibility analysis is a useful technique commonly found in a variety of mechanics and robotic applications. In parallel mechanisms or parallel robotics, the topic of the Wrench-Feasible Workspace (WFW) has been extensively studied to determine the robot workspace. A WFW usually refers to the set of poses for which the robot can balance any wrench of a specified set of wrenches, such that the actuation wrench in each actuator remains within a prescribed range [Gouttefarde, 2007]. It is therefore one of the most important criteria that can be applied to robot design, planning and control.

The WFW analysis approaches are probably the topics the most investigated in Cable-Driven Parallel Robots (CDPRs), as the tensile-only nature of cables (i.e. unilateral and non-negative cable tension) reduces dramatically the feasible workspace of such robots. For instance, [Riechel, 2004; Bosscher, 2006] studied the analytical generation of the boundaries of the WFW for point-mass cable robots. [Loloei, 2009] proposed analysis on the WFW of a redundant CDPR formulated by Linear Matrix Inequalities (LMI); A more general case of CDPRs with variable configurations and degrees of freedom was considered in [Bouchard, 2010], in which the fundamental nature of Available Wrench Set (AWS) and Task Wrench Set (TWS) was studied. A numerical approach based on interval analysis was proposed in [Gouttefarde, 2007; Gouttefarde, 2011], aiming at the determination

of 6D workspace with consideration of more practical uncertainties and application to optimal design. While most of the previous works were based on a quasi-static equilibrium situation, the determination of the dynamic feasible workspace has been proposed in [Gagliardini, 2018]. The determination of WFW for CDPRs is in general achieved by taking into account both the requirement of non-negative cable tensions and that of maximum admissible values. It has allowed an enhanced knowledge of the robot's capacities, which is thus applicable to robot planning and optimal design.

The wrench analysis is also an important topic in legged robots for the contact/motion planning based on the computation of Actuation Wrench Polytope (AWP) and Feasible Wrench Polytope (FWP) as in [Carpentier, 2017; Fernbach, 2018; Orsolino, 2018; Carpentier, 2018; Aceituno-Cabezas, 2018; Orsolino, 2020], because the determination of footprints that remains in the feasible wrench workspace is crucial for stable locomotion of such robots. Therefore, the computed FWP can be employed to an optimization process for footholds (contact points) selection [Fernbach, 2018; Aceituno-Cabezas, 2018; Orsolino, 2020], trajectory/motion planning [Carpentier, 2017; Orsolino, 2018; Aceituno-Cabezas, 2018] and optimal control [Carpentier, 2018]. Similar problems have equivalently been investigated in research related to robotic grasping, a domain that has flourished much earlier [Miller, 2004; Borst, 2004; Krug, 2016]. In these works, the notion of Grasp Wrench Space (GWS) is applied, indicating the set of wrenches that can be applied to an object by a grasp given limits on the contact normal forces [Miller, 2004]. A GWS can be created by the combination of friction cones on the contacts, called Cone Wrench Space (CWS), within which a force in each contact can be applied to the object to perform a stable grasp. The GWS can therefore be employed to analyse the grasping stability.

More recently, the wrench analysis has been employed in multi-UAV parallel robots to determine the available wrench that can be generated on a moving platform (and the attached end-effector) actuated by multiple UAVs. This analysis can be employed in aerial cable-towed systems [Erskine, 2018; Erskine, 2019a; Jamshidifar, 2020], aerial manipulator architectures [Park, 2018; Nguyen, 2018; Li, 2022] for various applications such as the optimal design of the novel architectures, determination/optimization of the robot workspace and the optimal control by maximising the wrench feasibility.

Inspired by these previous works, an analysis of the wrench feasibility of the Flying Parallel Robot (FPR) is conducted, which can be applied to determine the feasible task the FPR can exert and potentially to an optimal determination of the internal configuration (i.e. leg angles) having the optimal wrench feasibility according to the task specifications. Before detailing the methodology for wrench feasibility analysis of the FPR, some basic

concepts regarding the wrench feasibility are introduced in the next section.

5.2 Basic Concept of Wrench Feasibility

The wrench feasibility analysis is generally initiated by finding the relationship between the actuated forces and the wrench exerted on the platform/end-effector of the robot, which can be depicted by a general form as in [Bouchard, 2010]

$$\mathbf{W}\mathbf{t} = \mathbf{w} \tag{5.1}$$

where $\mathbf{t}$ is the vector of actuated forces, $\mathbf{w}$ is the wrench (i.e. force and moment) applied on the platform, and $\mathbf{W}$ is the *wrench matrix* being pose-dependent [1]. In cable-driven robots, $\mathbf{t} = \begin{bmatrix} t_1 & ... & t_m \end{bmatrix}^T$ denoting the vector of cable tensions. However, for aerial manipulators actuated by multiple UAVs, the vector of the actuated forces $\mathbf{t}$ is given by $\mathbf{f} = \begin{bmatrix} \mathbf{f}_1^T & ... & \mathbf{f}_m^T \end{bmatrix}^T$ composed of vectors of 3-dimensional thrust forces produced by the multirotors. The wrench matrix $\mathbf{W}$ can often be characterised by the Jacobian matrix as the relationship between the actuated wrench and thrust forces illustrated in Section 2.5.1, resulting in the wrench matrix equal to the transpose of the Jacobian matrix, i.e. $\mathbf{W} = \mathbf{J}^T$.

Knowing the relationship of (5.1), a Wrench-Feasible Workspace (WFW) is defined as

> **Definition** (Wrench-Feasible Workspace)**.** Consider a set of actuated forces, denoted as $\mathcal{T}$, which includes all possible $\mathbf{t}$ within the actuation limits, and a prescribed set of wrenches $\mathbf{w}$ on the mobile platform (end-effector), denoted as $\mathcal{W}_s$. The Wrench-Feasible Workspace is **the set of mobile platform poses** that are wrench-feasible, i.e. the poses for any wrench $\mathbf{w}$ in $\mathcal{W}_s$, there exists a vector of actuated forces $\mathbf{t}$ in $\mathcal{T}$ such that $\mathbf{W}\mathbf{t} = \mathbf{w}$.

The Available Wrench Set (AWS) is a dual concept of the WFW that can be defined as

> **Definition** (Available Wrench Set)**.** Consider a set of actuated forces, donated as $\mathcal{T}$, which inludes all possible $\mathbf{t}$ within the actuation limits. The Available Wrench Set (AWS), denoted as $\mathcal{W}$, is **the set of wrenches $\mathbf{w}$** that can be generated on the mobile platform (end-effector) at a given pose, i.e. for all actuated forces $\mathbf{t}$ in $\mathcal{T}$, $\mathcal{W}$ is the set of wrenches $\mathbf{w}$ such that $\mathbf{W}\mathbf{t} = \mathbf{w}$.

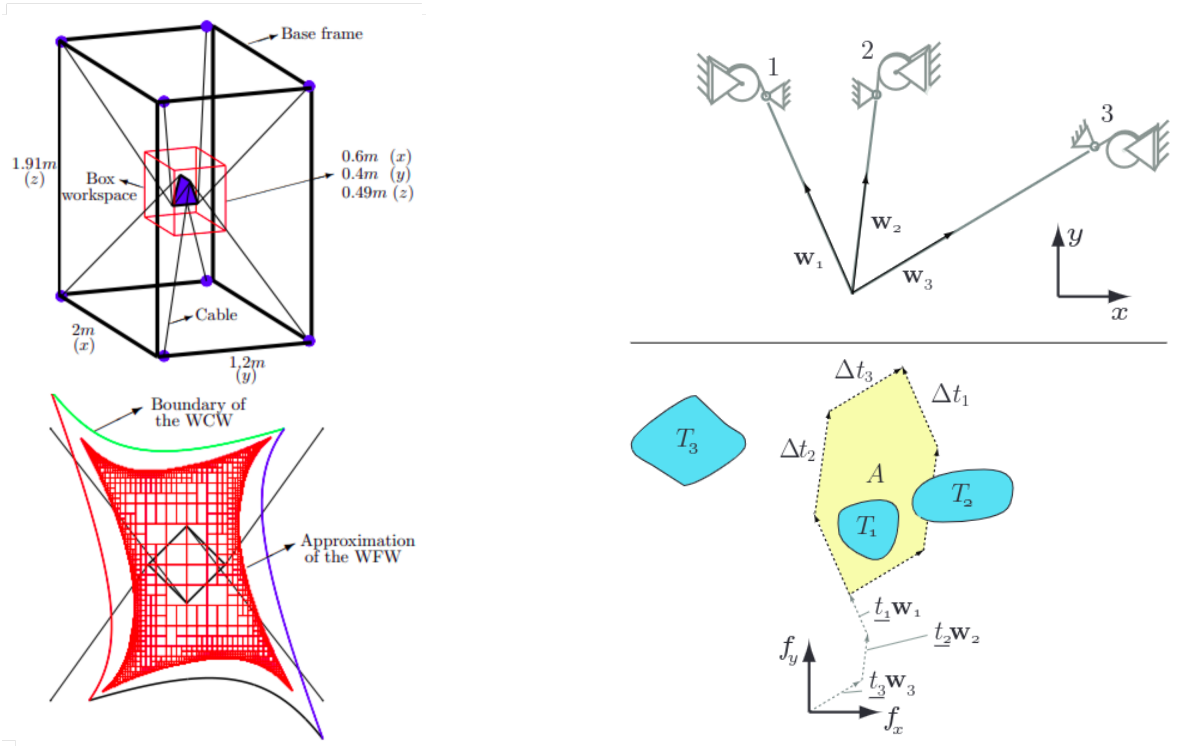
The above-mentioned notions of WFW and AWS can be employed to analyse the feasibility of tasks in a certain condition. Given a prescribed task defined by a set of the

1. Pose-dependent means that the matrix is dependent on the robot pose, i.e. $\mathbf{W}(\mathbf{q})$ with $\mathbf{q}$ usually including the end-effector pose and the robot internal configurations.

required wrenches on the platform (end-effector), denoted by $\mathcal{W}_t$, the task is said to be feasible

- If the platform pose remains in the Wrench-Feasible Workspace during the generation of required task wrenches in $\mathcal{W}_t$; or
- If the set of task wrenches $\mathcal{W}_t$ is a subset of the Available Wrench Set, i.e. $\mathcal{W}_t \subset \mathcal{W}$.

One may remark that the second criterion is easier for analysing the task feasibility, since the determination of WFW itself is a complex problem. As shown in Fig. 5.1, the determination of WFW in Fig. 5.1(a) is tedious and a numerical method for approximating the WFW has been adopted, while the calculation of AWS in Fig. 5.1(b) for a planner CDPR allows to verify if a task set (illustrated by regions T_1, T_2 and T_3) is feasible, i.e. if it is a subset of the Available Wrench Set (region A).



(a) Determination of WFW for a 6-DoF CDPR [Gouttefarde, 2007]

(b) Task feasibility analysed by the definition of AWS [Bouchard, 2010]

Figure 5.1 – Wrench-Feasible Workspace analysis (left) and task feasibility analysis based on the Available Wrench Set (right).

Apart from analysing the feasible workspace or the feasibility for a given task, another common investigation on the feasibility focuses on the static equilibrium of the robot pose, for which the Static Feasible Workspace can be defined as a special case of WFW where only the gravity wrench induced by the platform weight has to be balanced considering the actuation limits [Gouttefarde, 2011]. In other words, the static wrench feasibility states

that the wrench required to balance the gravity wrench $\mathbf{w}_g$ induced by the robot weight should be belonging to the AWS. A detailed definition is given as follows:

Definition (Static Wrench Feasibility)**.** Consider a set of actuated forces $\mathcal{T}$ taking into account the actuation limits. A moving platform pose is said to be static feasible if the gravity wrench $\mathbf{w}_g$ that corresponds to the moving platform (and the payload)'s weight can be balanced without violating the actuation limits, i.e.

$$\exists\, \mathbf{t} \in \mathcal{T} \quad \text{such that} \quad \mathbf{W}\mathbf{t} + \mathbf{w}_g = \mathbf{0} \tag{5.2}$$

or

$$-\mathbf{w}_g \in \mathcal{W} \tag{5.3}$$

where $\mathcal{W}$ is the set of available wrench $\mathbf{w}$ for which $\mathbf{W}\mathbf{t} = \mathbf{w}$, $\forall\, \mathbf{t} \in \mathcal{T}$.

The dynamic feasibility criterion can also be analysed as in [Loloei, 2009; Erskine, 2019b], in which the moving platform (or the payload)'s dynamic movements are considered beyond the quasi-static conditions. However, for aerial manipulators such as the FPR studied within the scope of this thesis, the dynamic response of the overall structure is relatively slow and the robot behaviour is furthermore close to a quasi-static motion when performing the interaction tasks. Therefore, the dynamic feasibility is not studied in this manuscript even though it is possible and not very complex to consider it.

Having discussed the basic concepts related to the wrench feasibility analysis, a detailed analysis of the Available Wrench Set of the FPR will be given in the next section.

5.3 Analysis of Available Wrench Set

The wrench analysis of the FPR has the objective to construct knowledge of the available wrench the robot is capable of resisting or exerting. Unlike the classical parallel mechanism such as CDPRs, the actuation by aerial vehicles and the additional DoFs provided by the mobility of internal configuration (i.e. leg angles) make the study of Wrench-Feasible Workspace (WFW) meaningless. The moving platform of the FPR has an unlimited workspace (in position and orientation) in 3-dimensional space as long as the internal configuration is well regulated for all wrench-feasible orientations. However, the calculation of Available Wrench Set (AWS) is useful to determine whether an interaction task is feasible or to optimise the robot configuration to ensure better wrench feasibility.

Taking the basic concepts of feasibility analysis presented in Section 5.2, the Available Wrench Set of the FPR is particularly defined as an augmented high-dimensional space

in $\mathbb{R}^{6+n}$ with n being equal to the number of legs, which depicts not only the 6D wrench (force and moment) actuated on the platform that is achievable, but also the available moments actuated on the revolute-joint axes of the legs, i.e.

$$\mathcal{W} = \left\{ \mathbf{w} \in \mathbb{R}^{6+n} \;\middle|\; \mathbf{w} = \begin{bmatrix} \mathbf{f}_p^T & \mathbf{m}_p^T & \mathbf{m}_l^T \end{bmatrix}^T \right\} \tag{5.4}$$

where $\mathbf{f}_p$ is the force actuated on the platform expressed in inertial reference frame $\mathfrak{F}_0$, $\mathbf{m}_p$ is the body-frame moment actuated on the platform, and $\mathbf{m}_l$ concatenates the moments actuated on the revolute joints of the legs. Note that from its definition, the wrench set of the FPR shares the same space with that of the actuation wrench defined in (2.27).

To analytically describe a high-dimensional space, the mathematical techniques of *convex hull* and *convex polytopes* are usually adopted. If a space is convex [1], the intersection of any family of convex sets is again convex, and any affine transformation of the convex space is also convex [Grünbaum, 2003]. This interesting property of convex polytopes facilitates the definition and computation of available wrench sets. A more detailed mathematical definition of polytopes and convex hulls will be given in the next subsection. Considering the actuation limitation of a single UAV, a notation of Available Thrust Set is presented and computed in the second subsection. Then a detailed methodology for the computation of Available Wrench Set of the FPR will be presented using two different representations. Several case studies and results will finally be demonstrated at the end of this section.

5.3.1 Definition of Convex Polytopes

As previously mentioned, the definition of a available set by means of convex polytopes is advantageous for further computation by a linear mapping or affine transformation. The precise mathematical definition of several common notations to be referred to throughout the calculation of Available Wrench Set is given as follows [Herceg, 2013]:

Definition (Convex Set)**.** A set $\mathcal{S} \subseteq \mathbb{R}^d$ is convex if the line segment connecting any pair of points of $\mathcal{S}$ lies entirely in $\mathcal{S}$, i.e. if for any $s_1, s_2 \in \mathcal{S}$ and any α with $0 \leq \alpha \leq 1$, $\alpha s_1 + (1-\alpha) s_2 \in \mathcal{S}$.

Definition (Convex Hull)**.** The convex hull of a subset $\mathcal{S} \subseteq \mathbb{R}^d$, denoted by $\texttt{conv}(\mathcal{S})$, is the intersection of all the convex sets in $\mathbb{R}^d$ which contain $\mathcal{S}$.

1. A space is called *convex* if any points on the segment between two arbitrary points within the space still belong to the space.

Definition (Polytope)**.** A polytope is a bounded convex set given as the intersection of a finite number of hyperplanes and half-spaces or as a convex combination of a finite number of vertices.

Based on these definitions, a polytope can be mathematically written using two different representations, respectively named H-representation and V-representation defined as

Definition (H-representation)**.** A polytope $\mathcal{P}$ is formed by the intersection of hyperplanes and half-spaces characterised by m_i inequalities and m_e equalities, i.e.

$$\mathcal{P} = \left\{ \mathbf{x} \in \mathbb{R}^d \;\middle|\; \mathbf{A}\mathbf{x} \leq \mathbf{b},\ \mathbf{A}_e\mathbf{x} = \mathbf{b}_e \right\} \tag{5.5}$$

where $\mathbf{A} \in \mathbb{R}^{m_i \times d}$, $\mathbf{b} \in \mathbb{R}^{m_i}$, $\mathbf{A}_e \in \mathbb{R}^{m_e \times d}$, $\mathbf{b}_e \in \mathbb{R}^{m_e}$ are the matrices representing the half-spaces and hyperplane, respectively.

Definition (V-representation)**.** A polytope $\mathcal{P}$ is formed by the convex combination (convex hull) of n_v vertices, i.e.

$$\mathcal{P} = \left\{ \mathbf{x} \in \mathbb{R}^d \;\middle|\; \mathbf{x} = \sum_{i=1}^{n_v} \lambda_i \mathbf{v}_i,\ \ \lambda_i > 0,\ \sum_{i=1}^{n_v} \lambda_i = 1 \right\} \tag{5.6}$$

where $\mathbf{v}_i$ $(i = 1, 2, ..., n_v) \in \mathbb{R}^d$ are the vertices.

These two representations are dual concepts for defining a polytope, as the H-representation (by the intersection of hyperplanes and half-spaces) can be converted to the V-representation, and vice versa, from which the represented polytopes are equivalent.

5.3.2 Computation of Available Thrust Set

With the knowledge of convex polytopes as the mathematical tool to define the space, the first step is to determine the Available Thrust Set (ATS) for a single UAV. The limitation of the actuated forces in systems involving UAVs is principally due to the limited thrust outputs of the propellers of a UAV [Park, 2018]. Additionally, the multirotors physically connected to a rigid structure in these designs are usually constrained by the mechanical stops in the joints, further limiting their rotational movements, as investigated in [Nguyen, 2018]. These constraints can be all described with the thrust force $\mathbf{f}_i$ generated by multirotor i expressed in $\mathfrak{F}_0$, which allows to determine the Available Thrust Set (ATS) that a single multirotor i can achieve in the FPR as follows

$$\mathcal{T}_i = \left\{ \mathbf{f}_i \in \mathbb{R}^3 \;\middle|\; \|\mathbf{f}_i\| \leq f_{max},\ \|\mathbf{f}_i\| \cdot \cos\gamma \leq \mathbf{z}^T\mathbf{f}_i,\ \mathbf{z} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \right\} \tag{5.7}$$

with two parameters characterising the actuation limitation, $\gamma \in \mathbb{R}, 0 < \gamma < \pi/2$ the maximum angle of a multirotor's rotational motion in roll and pitch (i.e. $\phi_i \leq \gamma$, $\vartheta_i \leq \gamma$), and f_{max} the maximum thrust magnitude (total thrust of all the propellers). An instinctive interpretation of $\mathcal{T}_i$ is that it is defined by the intersection between a sphere (constrained by $\|\mathbf{f}_i\| \leq f_{max}$ representing the maximum total thrust of a multirotor) and a cone with an apex located at the sphere centre (representing the constrained rotational movements). The expression of the cone is given by $\|\mathbf{f}_i\| \cdot \cos\gamma \leq \mathbf{z}^T\mathbf{f}_i$ and can be interpreted as

$$\mathbf{z}^T \frac{\mathbf{f}_i}{\|\mathbf{f}_i\|} = \mathbf{z}^T\mathbf{z}_i = \|\mathbf{z}\| \cdot \|\mathbf{z}_i\| \cdot \cos(\phi_i \text{ or } \vartheta_i) \geq \cos\gamma \tag{5.8}$$

from which the inclination angle of the multirotor is constrained by a constant angle γ, i.e. ϕ_i or $\vartheta_i \leq \gamma$, with $\mathbf{z}_i$ representing the z axis of the multirotor i's body frame expressed in $\mathfrak{F}_0$. A visual representation of the ATS defined by (5.7) is given as follows.

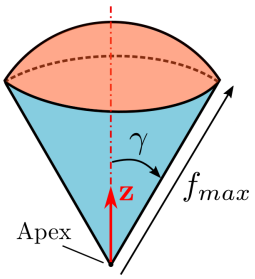


Figure 5.2 – Continuous ATS representing the actuation limit of a single UAV, with the sphere surface rendered in orange and the cone surface rendered in light blue.

As the ATS of a single multirotor defined by (5.7) and illustrated in Fig. 5.2 has continuous surfaces and represents an infinite number of vertices, a common way to handle the infinity is to linearise (or discretise) the expression of $\mathcal{T}_i$ by a finite number of vertices. This is done by evenly choosing the points (i.e. vertices) on the surface of the cone space and the sphere space, respectively. The selection of a finite number of vertices on the cone surface is equivalent to the approximation of a circle by n-side regular polygon shown in Fig. 5.3, which is the intersection circle on the cone surface bounded by the maximum thrust limit. Along with the apex point of the cone, the selected vertices for a bounded cone surface are written by (5.9).

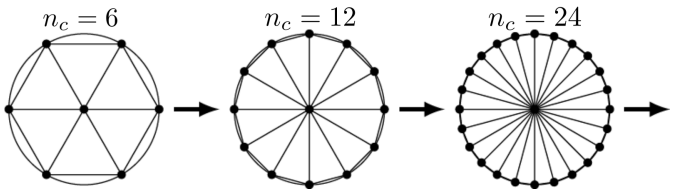


Figure 5.3 – Approximation of a circle by a regular polygon with n_c sides [Brilliant, 2022], from top view of Fig. 5.2.

$$\mathbf{v}_{c,0} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T, \quad \mathbf{v}_{c,j} = f_{max} \begin{bmatrix} \sin\gamma \sin\alpha \\ \sin\gamma \cos\alpha \\ \cos\gamma \end{bmatrix}, \text{ with } \alpha = \frac{2\pi j}{n_c}, j = 1, 2, ..., n_c. \tag{5.9}$$

Note that n_c is the number of points to approximate a circle, and $\mathbf{v}_{c,0}$ is the apex of the cone corresponding to the point that represents minimum zero thrust produced by the multirotor (when the motors are off). One may also remark that the expression of $\mathbf{v}_{c,j}$ is written in spherical coordinates, in which a point is addressed by two angular coordinates, the polar angle γ and the azimuthal angle α, with the polar angle being fixed to a constant value. This representation is beneficial to the definition of points on a sphere.

Consider a point located on the surface of a sphere depicting the maximum thrust limits, and characterised by the polar angle $\vartheta \in [0, \pi]$ and the azimuthal angle $\varphi \in [0, 2\pi]$. Given the radius of the sphere r, being equal to the maximum thrust f_{max}, the Cartesian coordinates of a point on the surface of a sphere are given by

$$\mathbf{v}_s = r \begin{bmatrix} \sin\vartheta \cos\varphi \\ \sin\vartheta \sin\varphi \\ \cos\vartheta \end{bmatrix} \tag{5.10}$$

The discretisation of the sphere surface can then be done by choosing a finite number of vertices. For this, an algorithm for equi-distributing points on the surface of a sphere is adopted, in which the points are regularly placed such that their distances in two orthogonal directions are locally always the same [Deserno, 2004]. It can be summarised by the following algorithm.

Algorithm. Regular Placement of Points on the Surface of a Sphere

Input: N *the total number of points*
Initialise $n_s = 0$
$a = 4\pi r^2/N$ *and* $d = \sqrt{a}$
$M_\vartheta = round(\pi/d)$
$d_\vartheta = \pi/M_\vartheta$ *and* $d_\varphi = a/d_\vartheta$
for $j = 0$ ***to*** $M_\vartheta - 1$
    $\vartheta = \pi(j + 0.5)/M_\vartheta$
    $M_\varphi = round(2\pi \sin\vartheta/d_\varphi)$
    for $k = 0$ ***to*** $M_\varphi - 1$
        $\varphi = 2\pi k/M_\varphi$
        Create point for (ϑ, φ) *using (5.10)*

```
            n_s = n_s + 1
        end
    end
```

This regular equi-distribution algorithm is actually achieved by choosing circles of latitude at constant intervals d_ϑ and then points on these circles with distance d_φ, such that $d_\vartheta \simeq d_\varphi$ and that $d_{\vartheta_s} d_\varphi$ equals the average area a per point [Deserno, 2004]. Note that at the end of the algorithm, n_s points have been placed on the surface of a complete sphere, with n_s very close to the desired number of points N. To further select the points that are within the cone space limited by the rotational motion of multirotor, one can make a selection by a simple condition on the polar angle ϑ which should be in the interval $[\pi/2 - \gamma; \pi/2]$, with γ the maximum inclination angle.

Finally, a finite number of vertices describing the actuation constraints of a single UAV are generated, which allows to determine a discretised ATS by a convex combination of all the vertices

$$\mathcal{T}_i = \left\{ \mathbf{f}_i \in \mathbb{R}^3 \;\middle|\; \mathbf{f}_i = \sum_{k=1}^{n_v} \lambda_k \mathbf{v}_k, \;\; \lambda_k > 0, \; \sum_{k=1}^{n_v} \lambda_k = 1 \right\} \tag{5.11}$$

with $\mathbf{v}_k$ a vertex from the collection of $\mathbf{v}_{c,j}$, $(j = 0, 1, ..., n_c)$ and $\mathbf{v}_{s,j}$, $(j = 1, ..., n_s)$, and the total number of vertices $n_v = n_s + n_c + 1$. Recall that n_s is the number of sphere surface points, n_c the number of circle points on the cone surface, plus the apex point of the cone.

5.3.3 Computation of Available Wrench Set

Knowing the ATS, the Available Wrench Set (AWS) can then be computed, combining the actuation constraints of all the multirotors depicted each by the ATS defined in the previous section and mapping into the wrench set. It is reasonably assumed that the FPR has a homogeneous multirotor platform design and the ATS of all the multirotor are the same. The computation of AWS in both V-representation and H-representation is detailed as follows.

Available Wrench Set in V-representation

The computation of AWS using V-representation is straightforward, since every vertex in the thrust set $\mathcal{T}_i$ can be directly mapped into a vertex in the wrench set $\mathcal{W}$ by the

relationship of (5.1), which can be further decomposed and written as

$$\begin{aligned}\mathbf{w} &= \mathbf{W}\mathbf{f} \\ &= \begin{bmatrix}\mathbf{W}_1 & \mathbf{W}_2 & \cdots & \mathbf{W}_n\end{bmatrix} \begin{bmatrix}\mathbf{f}_1 \\ \mathbf{f}_2 \\ \vdots \\ \mathbf{f}_n\end{bmatrix}\end{aligned} \tag{5.12}$$

with $\mathbf{W}_i \in \mathbb{R}^{(6+n)\times 3}$ a sub-matrix mapping the multirotor i's thrust vector $\mathbf{f}_i \in \mathbb{R}^3$ to the wrench vector $\mathbf{w} \in \mathbb{R}^{6+n}$.

This relationship allows to compute the Available Wrench Set corresponding to the actuation limitation of each single UAV, i.e. $\mathcal{W}_i$, with its definition given by

$$\mathcal{W}_i = \left\{ \mathbf{w} \in \mathbb{R}^{6+n} \;\middle|\; \mathbf{w} = \mathbf{W}_i \mathbf{f}_i, \;\; \text{with} \;\; \mathbf{f}_i \in \mathcal{T}_i \right\} \tag{5.13}$$

or equivalently by

$$\mathcal{W}_i = \left\{ \mathbf{w} \in \mathbb{R}^{6+n} \;\middle|\; \mathbf{w} = \sum_{k=1}^{n_v} \lambda_k \mathbf{v}_k^i \;\; \lambda_k > 0, \; \sum_{k=1}^{n_v} \lambda_k = 1 \right\} \tag{5.14}$$

with $\mathbf{v}_k^i = \mathbf{W}_i \mathbf{v}_k$ $(i = 1, 2, ..., n, k = 1, 2, ..., n_v)$ being a vertex of each space $\mathcal{W}_i$ obtained from $\mathcal{T}_i$. Remark that this mapping (from $\mathbf{f}_i \in \mathbb{R}^3$ to $\mathbf{w} \in \mathbb{R}^{6+n}$) indicates a lifting in dimensions, resulting in $\mathcal{W}_i$ a lower-dimension notation represented in high-dimensional space (examples as a line located in a 2D surface, or a 2D plane in 3-dimensional space).

After having computed all the sub-spaces $\mathcal{W}_i$ by a linear mapping of the thrust sets for all the individual multirotors, the overall Available Wrench Set represented by V-representation can be determined by the Minkowski sum of all the sub-spaces or the convex hull of all the vertices in $\mathcal{W}$, i.e.

$$\mathcal{W} = \mathcal{W}_1 \oplus \mathcal{W}_2 \oplus \cdots \oplus \mathcal{W}_n \tag{5.15}$$

with $\oplus$ being the Minkowski sum, or equivalently expressed by

$$\mathcal{W} = \left\{ \mathbf{w} \in \mathbb{R}^{6+n} \;\middle|\; \mathbf{w} = \sum_{k=1}^{n'_v} \lambda'_k \mathbf{v}'_k, \;\; \lambda'_k > 0, \; \sum_{k=1}^{n'_v} \lambda'_k = 1 \right\} \tag{5.16}$$

where $\mathbf{v}'_k$ $(k = 1, 2, ..., n'_v)$ are the vertices of the wrench set $\mathcal{W}$. Remark that the Minkowski sum of any two polytopes defined by their vertices is defined as follows.

Definition (Minkowski Sum of Polytopes by Vertices)**.** Consider two convex polytopes $\mathcal{A}$, $\mathcal{B}$ and the sets of their vertices $\mathcal{V}_A$, $\mathcal{V}_B$. Let $\mathcal{C}$ be the Minkowski sum of two polytopes $\mathcal{C} = \mathcal{A} \oplus \mathcal{B}$ and $\mathcal{V}_C$ denotes the set of vertices, in which the vertex is defined by:

For all $a_i \in \mathcal{V}_A$ and $b_j \in \mathcal{V}_B$, check whether there exists a pair $(a', b') \neq (a_i, b_j)$ with $a' \in \mathcal{V}_A$ and $b' \in \mathcal{V}_B$ such that $(a' + b') = (a_i + b_j)$. If it is the case, then $(a_i + b_j) \notin \mathcal{V}_C$, otherwise $(a_i + b_j) \in \mathcal{V}_C$.

It is often tedious to determine the Minkowski sum of two polytopes by their vertices, not to mention the need for computing the AWS by (5.15), which indicates $n - 1$ times of the Minkowski sum. Therefore, a computationally efficient algorithm can be adopted, which is based on the formulation of linear programming problems that can be solved in polynomial time [Delos, 2015]. The time complexity of computing the AWS by vertices for the FPR is $O({n_v}^n)$, with n_v the number of vertices in ATS and n the number of multirotors.

Available Wrench Set in H-representation

The computation of AWS in H-representation is useful because the definition of hyperplanes and half-spaces by a set of inequality or equality equations can facilitate the determination of wrench feasibility afterwards. This requires the conversion of the expression of a space from V-representation to H-representation, which is however not an easy problem. Therefore, instead of converting the wrench set from its V-representation defined by (5.16) to H-representation, which is dependent on the robot pose, it is more convenient to convert the ATS expressed in V-representation to its H-representation counterpart, as the ATS is not pose-dependent and considered constant.

The conversion from V-representation to H-representation can be done using a generic method based on the Fourier-Motzkin elimination [Matoušek, 2007]. Recall the ATS in V-representation of (5.11), and let $\mathbf{V} \in \mathbb{R}^{3 \times n_v}$ be a vertex matrix whose columns are the set of vertices. The convex combination of all the vertices, i.e. the discretised ATS can be rewritten as

$$\mathcal{T}_i = \left\{ \mathbf{f}_i \in \mathbb{R}^3 \;\middle|\; \mathbf{f}_i = \mathbf{V}\boldsymbol{\lambda},\;\; \boldsymbol{\lambda} > 0,\; \mathbf{1}^T\boldsymbol{\lambda} = 1 \right\} \tag{5.17}$$

where $\boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 & \lambda_2 & \cdots & \lambda_{n_v} \end{bmatrix}^T$ is the vector of coefficients. The expression of (5.17) is already close to the H-representation. By eliminating all the coefficients λ_m ($m = 1, 2, ..., n_v$) of $\boldsymbol{\lambda}$ from the system of linear inequalities by Fourier-Motzkin yields an H-representation (refer to [Matoušek, 2007] for more details), which can be written by

$$\mathcal{T}_i = \left\{ \mathbf{f}_i \in \mathbb{R}^3 \;\middle|\; \mathbf{A}_\mathrm{f}\mathbf{f}_i \leq \mathbf{b}_\mathrm{f} \right\} \tag{5.18}$$

with $\mathbf{A}_{\mathrm{f}} \in \mathbb{R}^{n_h \times 3}$, $\mathbf{b}_{\mathrm{f}} \in \mathbb{R}^{n_h}$ representing the half-spaces corresponding to the vertices in $\mathbf{V}$, n_h being the total number of half-spaces (i.e. inequality equations). Remark that the elimination of vertex coefficients only results in the inequality constraints, and not always ensures non-redundant solutions.

Based on the expression of ATS in H-representation for a single multirotor, a notion of Augmented Thrust Set is introduced, in which the actuation limits for all the multirotors are included. This space can be constructed using H-representation by

$$\mathcal{T}_{aug} = \left\{ \mathbf{f} = \begin{bmatrix} \mathbf{f}_1 & \mathbf{f}_2 & \cdots & \mathbf{f}_n \end{bmatrix}^T \in \mathbb{R}^{3n} \;\middle|\; \mathbf{A}_{aug}\mathbf{f} \leq \mathbf{b}_{aug} \right\} \tag{5.19}$$

with $\mathbf{A}_{aug} \in \mathbb{R}^{n \cdot n_h \times 3n}$ and $\mathbf{b}_{aug} \in \mathbb{R}^{n \cdot n_h}$ respectively given by

$$\mathbf{A}_{aug} = \begin{bmatrix} \mathbf{A}_{\mathrm{f}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{\mathrm{f}} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_{\mathrm{f}} \end{bmatrix}, \quad \mathbf{b}_{aug} = \begin{bmatrix} \mathbf{b}_{\mathrm{f}} \\ \mathbf{b}_{\mathrm{f}} \\ \vdots \\ \mathbf{b}_{\mathrm{f}} \end{bmatrix} \tag{5.20}$$

The AWS in H-representation is finally obtained by the linear mapping from the augmented thrust set via the wrench matrix as

$$\mathcal{W} = \left\{ \mathbf{w} \in \mathbb{R}^{6+n} \;\middle|\; \mathbf{A}_{\mathrm{w}}\mathbf{w} \leq \mathbf{b}_{\mathrm{w}} \right\} \tag{5.21}$$

where the matrix $\mathbf{A}_{\mathrm{w}} \in \mathbb{R}^{n \cdot n_h \times (6+n)}$ and the vector $\mathbf{b}_{\mathrm{w}} \in \mathbb{R}^{n \cdot n_h}$ for the inequality constraints are respectively given by

$$\mathbf{A}_{\mathrm{w}} = \mathbf{A}_{aug}\mathbf{W}^{-1}, \quad \mathbf{b}_{\mathrm{w}} = \mathbf{b}_{aug} \tag{5.22}$$

with $\mathbf{W}^{-1}$ being the simple inverse of the wrench matrix if its dimension is square. Note that if $\mathbf{W}$ is not a square matrix, the inversion is usually performed by the pseudo-inverse, which is however not guaranteed not to break the inequality constraints. In such cases, solving an inequality-constrained optimisation problem is a viable solution in place of the pseudo-inverse of the wrench matrix [Orsolino, 2020].

5.3.4 Case Study

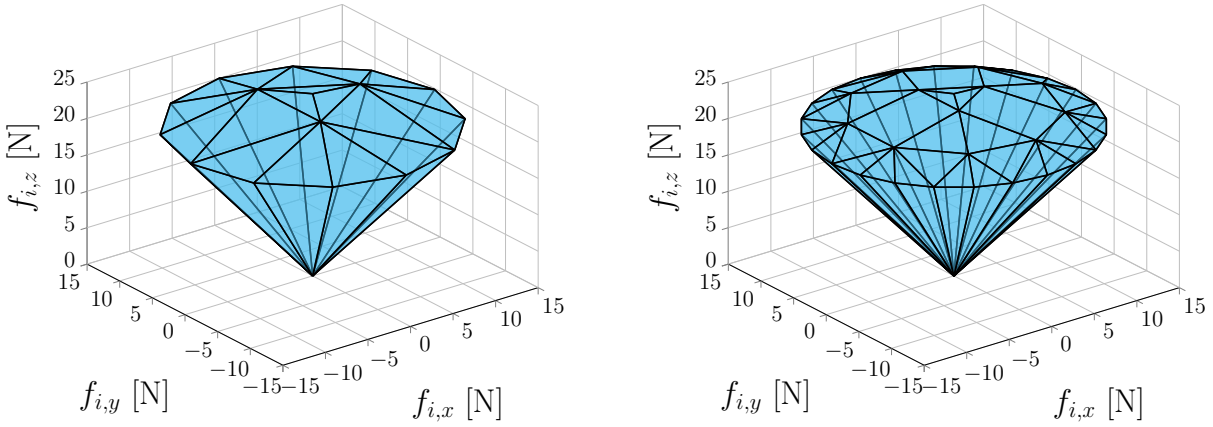
The analysis of the available wrench set of a specific FPR (with n the number of legs/multirotors being equal to 3) is conducted as a case study. The definition and computation of polytopes by V-representation and H-representation are implemented using Matlab. While there exists a lot of open-source toolboxes and libraries for solving the

problems regarding the convex polytopes such as [Lofberg, 2004; Herceg, 2013], it is still worthwhile to develop the algorithms presented beforehand by generic codes in order to easily migrate the algorithms to other programming platforms.

For the definition of Available Thrust Set corresponding to the actuation limitation of a single multirotor, the following constant values on the constraints are reasonably considered in line with the mechanical stops of the joints and the specifications of the brushless motors used to actuate the propellers.

$$f_{max} = 25N, \quad \gamma = 35^\circ \tag{5.23}$$

The discretisation of the ATS is done by choosing a number of vertices approximating the spherical cone surface. With more points selected, the space is closer to theoretically the real one, but the computation time especially for the mapping to the wrench set is much greater. Two discretised ATS are therefore obtained by selecting respectively 17 ($n_c = 12$, $n_s = 4$) and 38 ($n_c = 24$, $n_s = 13$) vertices, which can be visualised as follows.



(a) ATS discretised by 17 vertices

(b) ATS discretised by 38 vertices

Figure 5.4 – Visualisation of discretised Available Thrust Set (ATS) by different number of vertices chosen.

The V-representation of the ATS is systematically obtained by the convex hull of these discretised vertices, while the conversion to H-representation can be done using the toolbox [Herceg, 2013]. The theoretical volume of the original and continuous spherical-cone space is known by

$$V = \frac{2}{3}\pi f_{max}^2 h \tag{5.24}$$

with $h = f_{max}(1 - \cos\gamma)$. The volume of a convex polytope can be computed based on the algorithm presented in [Büeler, 2000]. In this case study, it is computed by the toolbox [Herceg, 2013]. The precision of the discretisation on an ATS can therefore be depicted by the percentage of its coverage volume with respect to the volume of the

theoretically actual ATS. The precision of discretised thrust sets and the number of half-spaces (i.e. inequalities) for the H-representation are given in Table 5.1. Remark that as the discretised ATS by 17 vertices can cover more than 90% of the theoretical ATS, this thrust set is chosen in the following analysis considering the fact that the computation time and complexity explode exponentially when the number of vertices increases.

Discretised ATS	Fig. 5.4(a)	Fig. 5.4(b)
V-representation	17 vertices	38 vertices
H-representation	30 half-spaces	69 half-spaces
Precision	90.3%	96.4%

Table 5.1 – Information of the discretised ATS by different number of vertices (V-representation) or half-spaces (H-representation) and their precisions with respect to the theoretical ATS.

Then, the discretised ATS can be mapped to the wrench set via two different pathways depending on the representation as discussed in Section 5.3.3:

1. Using V-representation: mapping all the vertices in discretised ATS to the vertices in the wrench set by a sub-wrench matrix → computing the Minkowski sum of the vertices in the wrench set.
2. Using H-representation: converting the discretised ATS defined by vertices to its H-representation counterpart → constructing the augmented ATS considering all the multirotors → mapping to the wrench set.

The mapping from the thrust set to the wrench set requires the computation of the Jacobian matrix (i.e. $\mathbf{W} = \mathbf{J}^T$), which is dependent on the robot pose. In this case study, the robot generalised coordinates are set by the following values

$$\begin{aligned}\mathbf{q} &= [p_{p,x},\ p_{p,y},\ p_{p,z},\ \mathfrak{q}_{p,0},\ \mathfrak{q}_{p,1},\ \mathfrak{q}_{p,2},\ \mathfrak{q}_{p,3},\ \theta_1,\ \theta_2,\ \theta_3]^T \\ &= [0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0.785,\ 0.785,\ 0.785]^T\end{aligned} \tag{5.25}$$

corresponding to a configuration with a flat platform orientation and leg angles of 45°.

Note that the Jacobian matrix is particularly square for this specific FPR with 3 multirotors (i.e. $6 + n = 3n$, when $n = 3$), and thus the mapping from augmented thrust set $\mathcal{T}_{aug} \in \mathbb{R}^9$ to the wrench set $\mathcal{W} \in \mathbb{R}^9$ requires simply an inverse of the wrench matrix. This makes the construction of wrench sets in (5.21) by H-representation much easier compared to the V-representation method which needs to solve n_v^3 linear programming problems. The pathway of using H-representation is valid and efficient as long as the Jacobian matrix is invertible, which can be ensured by singular-free robot configurations.

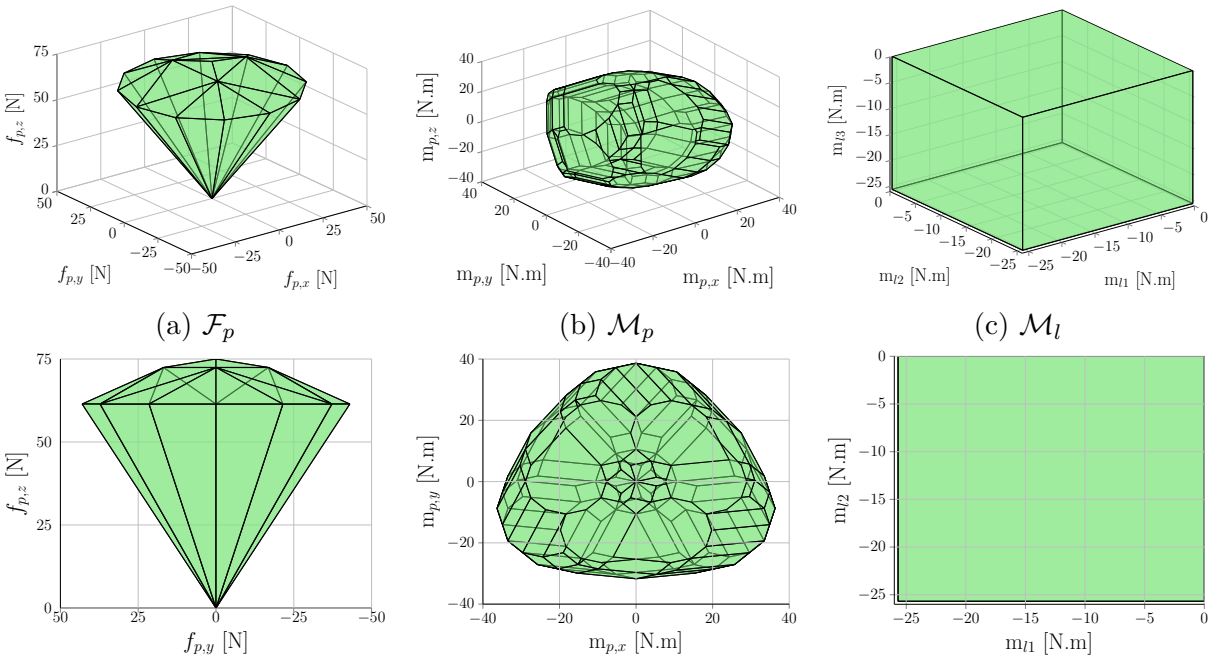
For the FPR, this means that the leg angles should always be constrained between 0° to 90° [Six, 2018b]. However, apart from the simplicity of computing wrench sets using H-representation, the visualisation of high-dimensional space (i.e. $\mathcal{W} \in \mathbb{R}^9$) defined by the intersection of half-spaces is not easy to deal with. In general, the high-dimensional space (hyperplanes) is projected into a low-dimensional space that can be visualised (such as 3D or 2D space). In contrast, the definition of space by vertices is more advantageous for visualisation purposes, since the vertices in wrench set mapped from the ATS can be decomposed to construct three independent spaces, i.e.

$$\mathcal{W} \in \mathbb{R}^9 \xrightarrow{\text{decomposed}} \mathcal{F}_p \in \mathbb{R}^3, \mathcal{M}_p \in \mathbb{R}^3, \mathcal{M}_l \in \mathbb{R}^3 \tag{5.26}$$

where $\mathcal{F}_p$, $\mathcal{M}_p$ and $\mathcal{M}_l$ are the available sets respectively for the 3-dimensional force and moment actuated on the platform and the moments actuated about the revolute joints of the legs. Based on the V-representation, they are systematically defined by the vertices in (5.16), i.e obtained by decomposition on each vertices $\mathbf{v}'_k = \begin{bmatrix}\mathbf{f}_{p,k}^T & {}^p\mathbf{m}_{p,k}^T & \mathbf{m}_{l,k}^T\end{bmatrix}^T$. It is however remarked that these vertices decomposed from those of the overall AWS are probably redundant for the representation of the decomposed spaces. The redundancy of vertices can be removed by checking the uniqueness of vertex using the algorithm detailed in [Delos, 2015] when the Minkowski sums of polytopes are performed. This can also be handled by constructing irredundant decomposed spaces $\mathcal{F}_{p,i}$, $\mathcal{M}_{p,i}$ and $\mathcal{M}_{p,i}$ for each multirotor i from its thrust set $\mathcal{T}_i$ and performing afterwards the Minkowski sums respectively for $\mathcal{F}_p$, $\mathcal{M}_p$ and $\mathcal{M}_l$.

The visualisation of $\mathcal{F}_p$, $\mathcal{M}_p$ and $\mathcal{M}_l$ using V-representation according to the robot configuration given by (5.25) is shown in Fig. 5.5, which has been validated to be equivalent to the spaces represented by half-spaces. Table 5.2 shows the details on the number of irredundant vertices and half-spaces defining the decomposed spaces. It can be noted that $\mathcal{F}_p$ has a similar form of $\mathcal{T}_i$ since the corresponding sub-wrench matrix mapping the thrust sets to $\mathcal{F}_p$ is identity (i.e. first three columns of $\mathbf{W}_i$), indicating a simple Minkowski sum of the ATS vertices. $\mathcal{F}_p$ is therefore not dependent on the robot pose as it depicts the available force actuated on the platform expressed in $\mathfrak{F}_0$. $\mathcal{M}_p$ is a more complex space in which all the surfaces are mapped from the thrust sets by the rotational parts of the Jacobian matrix (i.e. $\mathbf{J}_{\boldsymbol{\omega}_1}$ defined in (2.20)), while the moments generated on the legs are equivalently constrained from zero to about 25 N.m in the clockwise direction (for which the sign is negative) when the leg angles are 45°.

Based on the definition of ATS and the computation of AWS, the static wrench feasibility can be analysed and will be presented in the next section.



(a) $\mathcal{F}_p$ (b) $\mathcal{M}_p$ (c) $\mathcal{M}_l$

(d) Side view of $\mathcal{F}_p$ (e) Top view of $\mathcal{M}_p$ (f) Top view of $\mathcal{M}_l$

Figure 5.5 – Visualisation of decomposed Available Wrench Sets.

Available Wrench Set	$\mathcal{F}_p$	$\mathcal{M}_p$	$\mathcal{M}_l$
V-representation	17 vertices	170 vertices	8 vertices
H-representation	30 half-spaces	150 half-spaces	6 half-spaces

Table 5.2 – Number of vertices (V-representation) or half-spaces (H-representation) defined in the decomposed AWS.

5.4 Analysis of Static Wrench Feasibility

The construction of Available Wrench Set has the objective to analyse the feasibility of a specific task, if the wrench set required to accomplish the given task is known. However, for more general cases, the static wrench feasibility is of great interest, as it can provide information about the feasibility in quasi-static equilibrium conditions. Without the need of determining the wrench set of a known task, the static wrench feasibility can be analysed to determine quantitatively the available wrench that the robot can exert according to certain robot configurations after having compensated the gravity wrench. This analysis is particularly significant for evaluating the ability of the FPR such as the potential wrench capacity during a manipulation task, where the movements of the robot are considered slow enough to neglect its dynamic response.

In the following subsections, the notion of feasibility margin is firstly presented, which serves as quantitative metrics to evaluate the wrench feasibility in static conditions. Then, an extended analysis of the static wrench feasibility of the FPR based on the computation of AWS detailed in Section 5.3 is given. Finally, several case studies on different robot configurations, as well as the corresponding wrench set and feasibility will be given.

5.4.1 Definition of Static Feasibility Margin

As introduced in Section 5.2, the analysis of the static feasibility necessitates the computation of the wrench required to compensate for the gravitational effects (i.e. $\mathbf{w}_g$). From the computation of gravity wrench detailed in Section 2.5.4, this term can be known by the static model written as

$$\mathbf{w} - \mathbf{w}_g = \mathbf{0} \tag{5.27}$$

which is pose dependent, i.e. the computation of $\mathbf{w}_g$ takes the generalised coordinates $\mathbf{q}$ as input. According to the definition in (5.4), the $\mathbf{w}_g$ can be decomposed and written as

$$\mathbf{w}_g = \begin{bmatrix} \mathbf{f}_{p,g}^T & \mathbf{m}_{p,g}^T & \mathbf{m}_{l,g}^T \end{bmatrix}^T \tag{5.28}$$

The static feasibility is therefore evaluated by the condition if $\mathbf{w}_g \in \mathbb{R}^{6+n}$ is belonging to the Available Wrench Set $\mathcal{W}$. It is remarked however that the fulfilment of this condition only verifies if the robot in a certain configuration is capable of resisting the wrench due to the gravity in static equilibrium, without any quantitative metrics to possibly compare the feasibility in different configurations for example. Therefore, a feasibility metric can be adopted which is based on the capacity margin initially proposed in [Gagliardini, 2016] for CDPRs, and applied to the study of ACTS in [Erskine, 2019b].

In this manuscript, the feasibility metric in quasi-static conditions is named as *Static Feasibility Margin*, which is defined as the shortest distance from the $\mathbf{w}_g$ to the nearest facet (i.e. hyperplanes) of $\mathcal{W}$. It is therefore an index used to evaluate the robustness of the equilibrium in a specific robot configuration. The mathematical definition of the static feasibility margin can be given as follows.

$$\begin{aligned} &\texttt{min} \;\; \|\mathbf{w}' - \mathbf{w}_g\|, \;\; \forall\, \mathbf{w}' \in \mathcal{W} \\ &\text{s.t.} \;\; \mathbf{A}_\text{w}\mathbf{w}' = \mathbf{b}_\text{w} \end{aligned} \tag{5.29}$$

where $\mathbf{A}_\text{w}$ and $\mathbf{b}_\text{w}$ are respectively the matrix and vector defining the half-spaces of the AWS given by (5.21). The definition of feasibility margin allows to compute for each configuration a quantitative result evaluating the wrench feasibility in static equilibrium.

5.4.2 Computation of Static Feasibility

As it can be seen in the definition of static feasibility margin (named as feasibility margin in the rest of this chapter), the expression of the AWS by $\mathbf{A}_{\mathrm{w}}$ and $\mathbf{b}_{\mathrm{w}}$ (i.e. half-spaces) is needed for the computation of this variable. This makes the H-representation more favourable than the V-representation for the analysis of static wrench feasibility, especially because additional conditions corresponding to the intersection with hyper-planes (i.e. equality constraints) can easily be added to the expression of wrench sets in H-representation.

Taking advantage of using H-representation, the static Available Wrench Set can be further evaluated separately for the force and moment spaces decomposed from the wrench sets of the platform, i.e. $\mathcal{F}_p \in \mathbb{R}^3, \mathcal{M}_p \in \mathbb{R}^3$. These spaces describe the wrench capacity of the FPR in static equilibrium when the gravity wrench is well compensated. It is remarked that the feasibility analysis on the available moment set of the leg angles (denoted by the $\mathcal{M}_l \in \mathbb{R}^n$) can also be done similarly, which is however omitted because it is not supposed that the manipulation task will be handled by the legs. Although it is possible to evaluate the range of available moments each leg can exert apart from resisting the moment due to the gravity, the moments actuated on the legs in the analysis presented in this section are only considered to compensate the gravity to stably maintain the internal configuration of the robot.

A detailed expression of the Available Wrench Set for analysing respectively the force and moment feasibility of the platform in static equilibrium can be written as

$$\mathcal{W}^s_{\mathbf{f}_p} = \left\{ \mathbf{w} \in \mathbb{R}^{6+n} \;\middle|\; \mathbf{A}_{\mathrm{w}}\mathbf{w} \leq \mathbf{b}_{\mathrm{w}},\ \mathbf{A}_{\mathbf{f}_p}\mathbf{w} = \mathbf{b}_{\mathbf{f}_p} \right\} \tag{5.30}$$

$$\mathcal{W}^s_{\mathbf{m}_p} = \left\{ \mathbf{w} \in \mathbb{R}^{6+n} \;\middle|\; \mathbf{A}_{\mathrm{w}}\mathbf{w} \leq \mathbf{b}_{\mathrm{w}},\ \mathbf{A}_{\mathbf{m}_p}\mathbf{w} = \mathbf{b}_{\mathbf{m}_p} \right\} \tag{5.31}$$

where the additional equality constraints are characterised by the matrices and vectors given by

$$\begin{aligned} \mathbf{A}_{\mathbf{f}_p} &= \begin{bmatrix} \mathbf{0}_{(3+n)\times 3} & \mathbf{1}_{(3+n)\times(3+n)} \end{bmatrix}, \quad \mathbf{b}_{\mathbf{f}_p} = \begin{bmatrix} \mathbf{m}_{p,g}^T & \mathbf{m}_{l,g}^T \end{bmatrix}^T \\ \mathbf{A}_{\mathbf{m}_p} &= \begin{bmatrix} \mathbf{1}_{3\times 3} & \mathbf{0}_{3\times 3} & \mathbf{0}_{n\times n} \\ \mathbf{0}_{n\times 3} & \mathbf{0}_{n\times 3} & \mathbf{1}_{n\times n} \end{bmatrix}, \quad \mathbf{b}_{\mathbf{m}_p} = \begin{bmatrix} \mathbf{f}_{p,g}^T & \mathbf{m}_{l,g}^T \end{bmatrix}^T \end{aligned} \tag{5.32}$$

Note that the static AWS for force $\mathcal{W}^s_{\mathbf{f}_p}$ or moment $\mathcal{W}^s_{\mathbf{m}_p}$ are two different sets parameterising the force or moment feasibility of the platform in different static conditions. The static available force and moment sets $\mathcal{F}^s_p, \mathcal{M}^s_p \in \mathbb{R}^3$ are then obtained by the projection of the respective wrench sets onto a 3-dimensional space, i.e. $\mathcal{F}^s_p = \mathcal{W}^s_{\mathbf{f}_p}.\texttt{project}(1:3)$, $\mathcal{M}^s_p = \mathcal{W}^s_{\mathbf{m}_p}.\texttt{project}(4:6)$. Remark that this projection onto the 3-dimensional space

is done for visualisation purpose, which does not loose any information of the initially computed AWS. The expression of $\mathcal{F}_p$ and $\mathcal{M}_p$ after the projection can be further written using H-representation as

$$\mathcal{F}_p^s = \left\{ \mathbf{f}_p \in \mathbb{R}^3 \;\middle|\; \mathbf{A}_{\mathbf{f}_p}^s \mathbf{f}_p \leq \mathbf{b}_{\mathbf{f}_p}^s \right\} \tag{5.33}$$

$$\mathcal{M}_p^s = \left\{ \mathbf{m}_p \in \mathbb{R}^3 \;\middle|\; \mathbf{A}_{\mathbf{m}_p}^s \mathbf{m}_p \leq \mathbf{b}_{\mathbf{m}_p}^s \right\} \tag{5.34}$$

Note that the matrices $\mathbf{A}_{\mathbf{f}_p}^s, \mathbf{A}_{\mathbf{m}_p}^s$ and vectors $\mathbf{b}_{\mathbf{f}_p}^s, \mathbf{b}_{\mathbf{m}_p}^s$ defining the half-spaces in 3-dimensional space are obtained by the intersection of initial half-spaces represented by $\mathbf{A}_{\mathrm{w}}\mathbf{w} \leq \mathbf{b}_{\mathrm{w}}$ and the hyperplanes respectively given by $\mathbf{A}_{\mathbf{f}_p}\mathbf{w} = \mathbf{b}_{\mathbf{f}_p}$ and $\mathbf{A}_{\mathbf{m}_p}\mathbf{w} = \mathbf{b}_{\mathbf{m}_p}$, which are then projected into the corresponding force or moment space in $\mathbb{R}^3$.

After having obtained the respective available force and moment sets of the platform, the feasibility margin can be computed to evaluate quantitatively the feasibility in certain robot configurations, especially for different leg angles. The feasibility margin for $\mathcal{F}_p$ and $\mathcal{M}_p$ is therefore determined by the shortest distance respectively from the $\mathbf{f}_{p,g}$ and $\mathbf{m}_{p,g}$ to all the facets of the available set.

Based on the definition of feasibility margin, the finding of minimum distance can be formulated as a Quadratic Programming (QP) problem that can be written as follows.

$$\begin{aligned} \texttt{min} \quad & \mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{b}^T\mathbf{x} + \mathbf{c} \\ \text{s.t.} \quad & \mathbf{A}_e\mathbf{x} = \mathbf{b}_e \end{aligned} \tag{5.35}$$

According to the computation of feasibility margin for the force or moment spaces, the arguments for solving the QP problems are respectively defined by Table 5.3.

Available Set	$\mathbf{H}$	$\mathbf{b}$	$\mathbf{c}$	$\mathbf{A}_e$	$\mathbf{b}_e$
$\mathcal{F}_p^s\ (\mathbf{x} = \mathbf{f}_p)$	$\mathbf{1}_{3\times3}$	$-2 \cdot \mathbf{f}_{p,g}$	$\mathbf{f}_{p,g}^T\mathbf{f}_{p,g}$	$\mathbf{A}_{\mathbf{f}_p}^s$	$\mathbf{b}_{\mathbf{f}_p}^s$
$\mathcal{M}_p^s\ (\mathbf{x} = \mathbf{m}_p)$	$\mathbf{1}_{3\times3}$	$-2 \cdot \mathbf{m}_{p,g}$	$\mathbf{m}_{p,g}^T\mathbf{m}_{p,g}$	$\mathbf{A}_{\mathbf{m}_p}^s$	$\mathbf{b}_{\mathbf{m}_p}^s$

Table 5.3 – Definition of arguments for QP problem solving the feasibility margins.

The computation of static Available Wrench Sets (respectively on the available force and moment of the platform), as well as the determination of feasibility margin have allowed the completion of the feasibility analysis by a quantitative evaluation. The case study on this static feasibility analysis in different robot configurations is given in the following subsection.

5.4.3 Case Study

The case study is performed on a specific FPR design with 3 multirotors and the ATS discretised by 17 vertices presented in Section 5.3.4. The computation of the static AWS, i.e. $\mathcal{F}_p^s$ and $\mathcal{M}_p^s$, is achieved by the toolbox [Herceg, 2013] in Matlab, with the definition of H-representation spaces given by (5.30) and (5.31). The QP problem for determining the feasibility margin can be solved by a generic solver such as the one presented in [Herceg, 2013].

As the static AWS are pose-dependent, implying that changes in platform orientation and leg angles will induce evolution in the wrench sets as well as the feasibility margin. Therefore, a first investigation can be done to fix a specific platform orientation, while evaluating the feasibility with varied values in the leg angles. To maintain balanced feasibility along different axes and avoid potential collisions, the values on different leg angles are always set to be equal.

The available wrench sets $\mathcal{F}_p^s$ and $\mathcal{M}_p^s$ and their feasibility margins corresponding to three different leg angles are studied, with the values on the angles chosen to be (see Fig. 5.6)

- 15° for wide configuration (rendered in red);
- 45° for normal configuration (rendered in green);
- 75° for narrow configuration (rendered in blue).

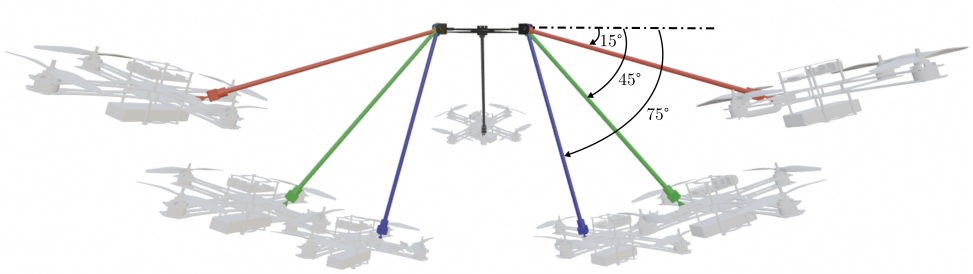
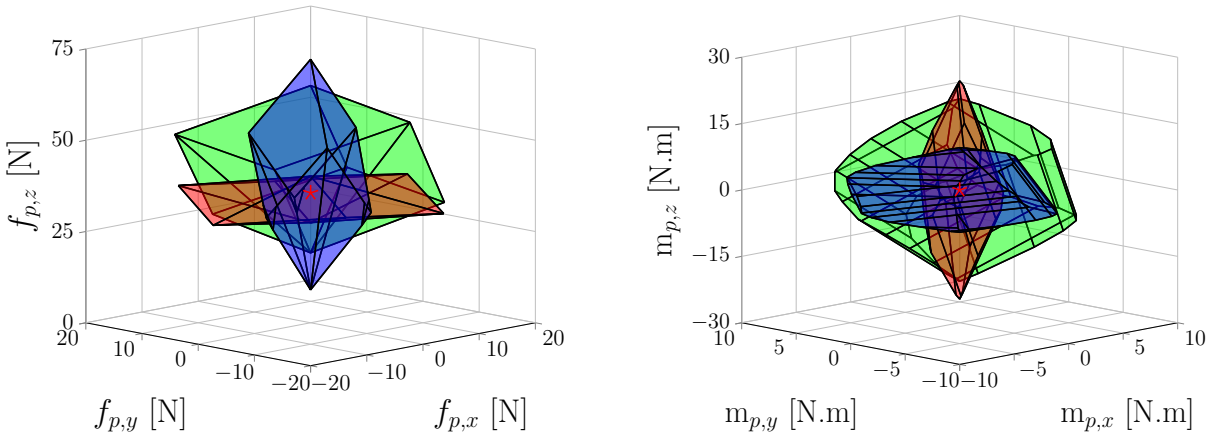


Figure 5.6 – Illustration of configurations with different leg angles.

Two cases on the platform orientation are furthermore considered, with the first one in the condition of flat orientation (roll and pitch are all zero) and the second one in an inclined orientation (roll=10°, pitch=25°). The results on the Available Wrench Sets of each case can be visualised in Fig. 5.7 and Fig. 5.8. The feasibility margins of the corresponding force and moment spaces $\mathcal{F}_p^s$, $\mathcal{M}_p^s$ in static equilibrium are summarised in Table 5.4.

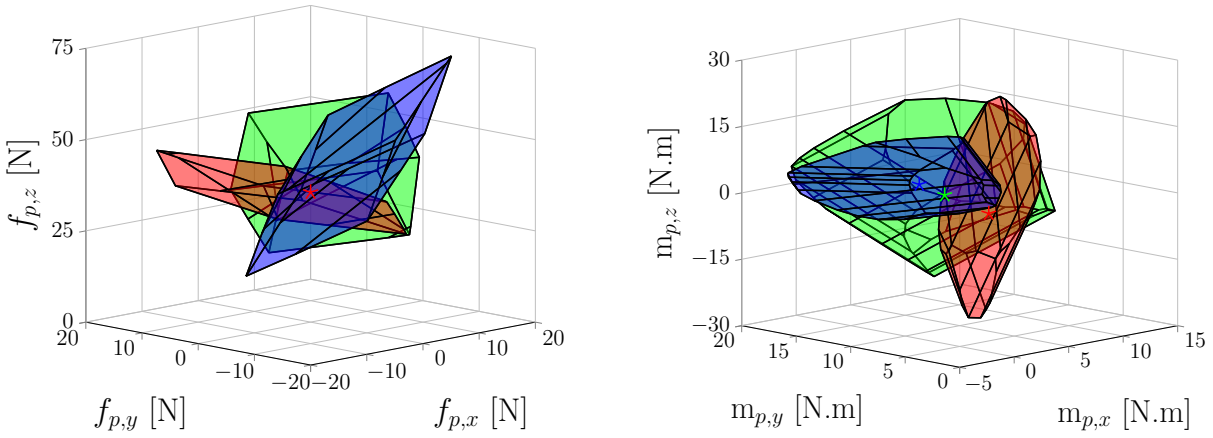
Case 1: Flat Platform Orientation (roll=0°, pitch=0°)



(a) Available force sets $\mathcal{F}_p^s$ (b) Available moment sets $\mathcal{M}_p^s$

Figure 5.7 – Available Wrench Sets evaluated at static equilibrium of $\mathbf{A}_{\mathbf{f}_p}\mathbf{w} = \mathbf{b}_{\mathbf{f}_p}$ (left) and $\mathbf{A}_{\mathbf{m}_p}\mathbf{w} = \mathbf{b}_{\mathbf{m}_p}$ (right) for the flat platform orientation (roll = 0°, pitch = 0°) and the leg angles being respectively 15° (red regions), 45° (green regions) and 75° (blue regions). The red star corresponds to the static equilibrium point at $\mathbf{f}_{p,g}$ (left) and $\mathbf{m}_{p,g}$ (right).

Case 2: Inclined Platform Orientation (roll=10°, pitch=25°)



(a) Available force sets $\mathcal{F}_p^s$ (b) Available moment sets $\mathcal{M}_p^s$

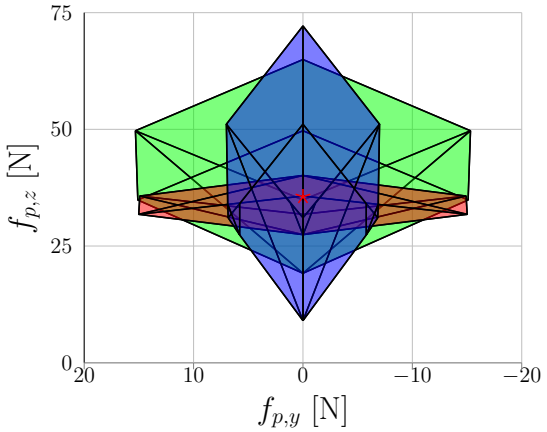
Figure 5.8 – Available Wrench Sets evaluated at static equilibrium of $\mathbf{A}_{\mathbf{f}_p}\mathbf{w} = \mathbf{b}_{\mathbf{f}_p}$ (left) and $\mathbf{A}_{\mathbf{m}_p}\mathbf{w} = \mathbf{b}_{\mathbf{m}_p}$ (right) for the inclined platform orientation (roll = 10°, pitch = 25°) and the leg angles being respectively 15° (red regions), 45° (green regions) and 75° (blue regions). Note that the static equilibrium point for available force sets is given by the star point in red (left), while for available moment sets, they are respectively represented by the star points in red, green and blue (right) for each case of leg angles.

It can be seen that in both cases, the spaces with leg angles being 45° (green spaces) cover more feasibility areas. The narrow configuration (blue spaces) has better feasibility in the forces exerted towards the normal direction of the platform (peaks of the $\mathcal{F}_p^s$ in blue), while the wide configuration is capable of exerting more moments around the z axis of the platform (peaks in red $\mathcal{M}_p^s$). Furthermore, the Available Wrench Sets have been

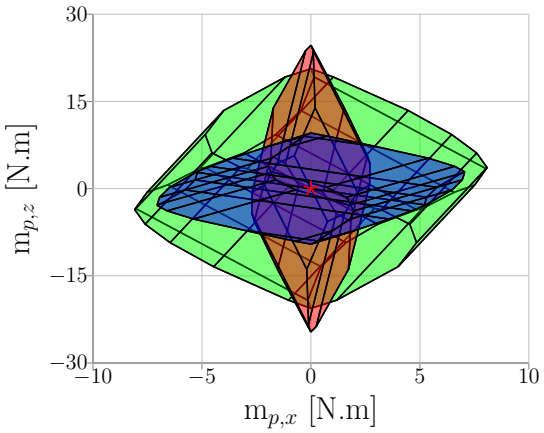
Configurations		Flat Orientation		Inclined Orientation	
		$\mathcal{F}_p^s$ [N]	$\mathcal{M}_p^s$ [N.m]	$\mathcal{F}_p^s$ [N]	$\mathcal{M}_p^s$ [N.m]
Leg Angles	15°	4.10	2.42	3.02	1.78
	45°	8.22	6.90	7.07	6.03
	75°	5.01	4.99	2.96	3.04

Table 5.4 – Feasibility margin of $\mathcal{F}_p^s$ and $\mathcal{M}_p^s$ with different robot configurations.

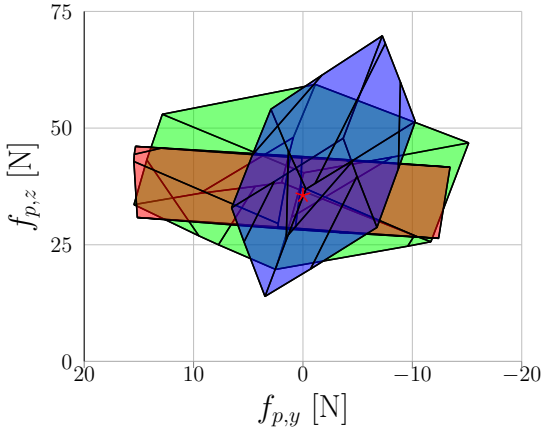
shifted when the platform is inclined, implying that the robustness is no longer symmetric relative to the static equilibrium point. This can be better visualised by the side views shown in Fig. 5.9, from which the shifting of the available sets can be well identified. Note that the force $\mathbf{f}_{p,g}$ actuated on the platform due to the gravity in the available force sets remains constant as the platform force is expressed in the global frame.



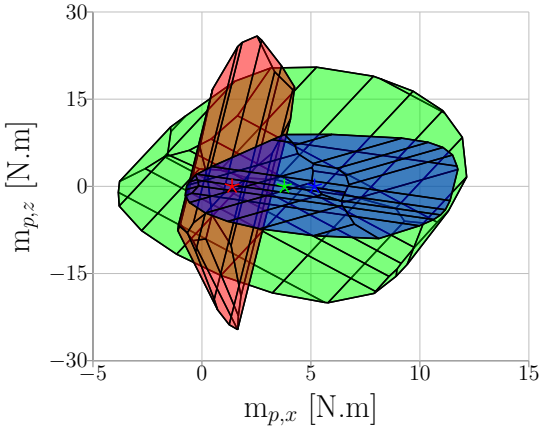
(a) Side view of $\mathcal{F}_p^s$ in flat orientation



(b) Side view of $\mathcal{M}_p^s$ in flat orientation



(c) Side view of $\mathcal{F}_p^s$ in inclined orientation



(d) Side view of $\mathcal{M}_p^s$ in inclined orientation

Figure 5.9 – Side views of the Available Wrench Sets of the platform evaluated at static equilibrium respectively for the flat and inclined platform orientations.

In addition, one may remark from the quantitative results shown in Table 5.4 that in both cases for different orientations, the configuration with leg angles of 45° has presented better feasibility in force and moment margins, indicating that the robustness of

the equilibrium in this normal configuration is better compared to a wide or narrow configuration. This has justified the selection of the leg angles preferable to be around 45° in the previous works, although the other configurations might be suitable for specific cases such as in certain applications where the task requires a higher force or moment feasibility along/about the z axis.

The analysis of static feasibility can be applied to the determination of the best leg angles for every platform orientation required to accomplish a specific task, which will be presented in the next section.

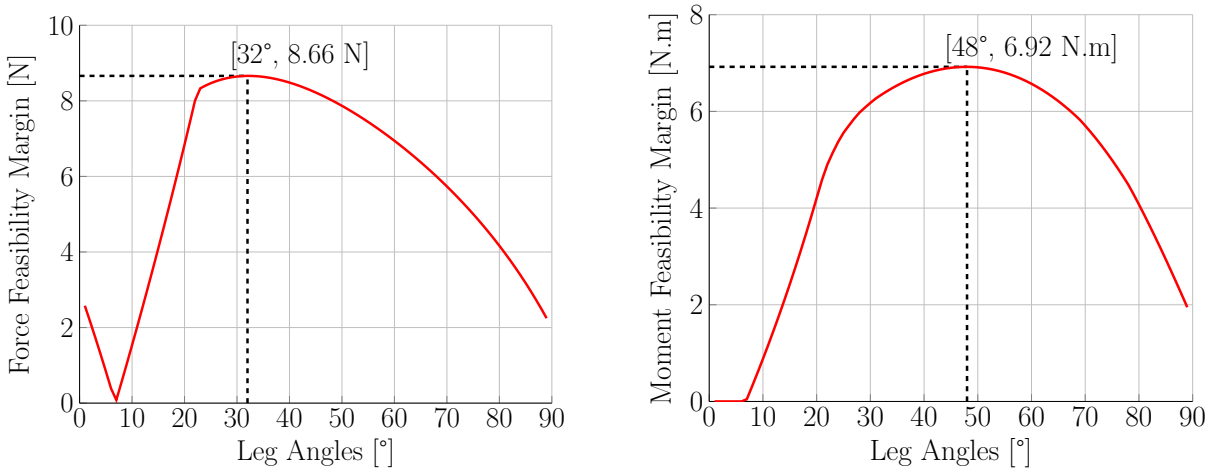
5.5 Determination of Optimal Leg Configuration

In the previous sections, the computation of Available Wrench Set and the analysis of static feasibility with a quantitative metric to evaluate the feasibility in static equilibrium have been extensively studied. While in the previous case studies, the feasibility of different platform orientations with various leg configurations has been evaluated, it is possible to determine the optimal leg angles allowing maximisation of the feasibility margin in a given platform orientation.

As presented in Section 5.4, the computation of static wrench sets is dependent on the robot pose, i.e. the platform orientation and the internal configuration (i.e. leg angles). Therefore, for every single case of the robot pose, the static wrench set will evolve, and so does the static margin. This has complicated the determination of the optimal leg angles for a given orientation of the platform, because it means comparing the static feasibility margins and finding the maximum for different wrench sets corresponding to different leg angles. To simplify the task by reducing the number of cases, it can be reasonably assumed for each given platform orientation that

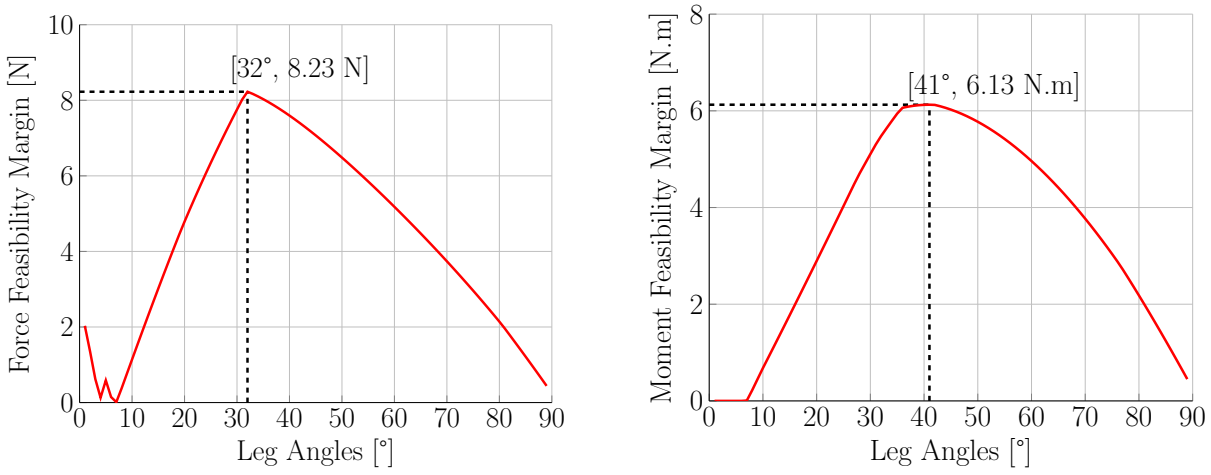
- values of all the leg angles are remaining as the same;
- a minimum step on the leg angles is set to be 1°.

Then, by performing the computational steps presented in Section 5.3 and Section 5.4 for each leg angle selected within the interval of non-singular configurations (i.e. between 0° and 90°) and an angle step of 1°, the optimal one maximising respectively the force and moment feasibility can be found. It is noted that the ATS is represented by half-spaces and only needs to be computed once (as it is constant and pose-independent), which is then mapped to the wrench set using the Jacobian matrix computed according to each case of the leg angles. Fig. 5.10 and Fig. 5.11 show the evolution of feasibility margins for two cases of the platform orientation studied in Section 5.4.3.



(a) Feasibility margin of $\mathcal{F}_p^s$ (b) Feasibility margin of $\mathcal{M}_p^s$

Figure 5.10 – Evolution of feasibility margins of the platform regarding different leg angles in flat platform orientation. The leg angles corresponding to optimal wrench feasibility are found at 32° for the force feasibility (8.66 N) and 48° for the moment feasibility (6.92 N.m).

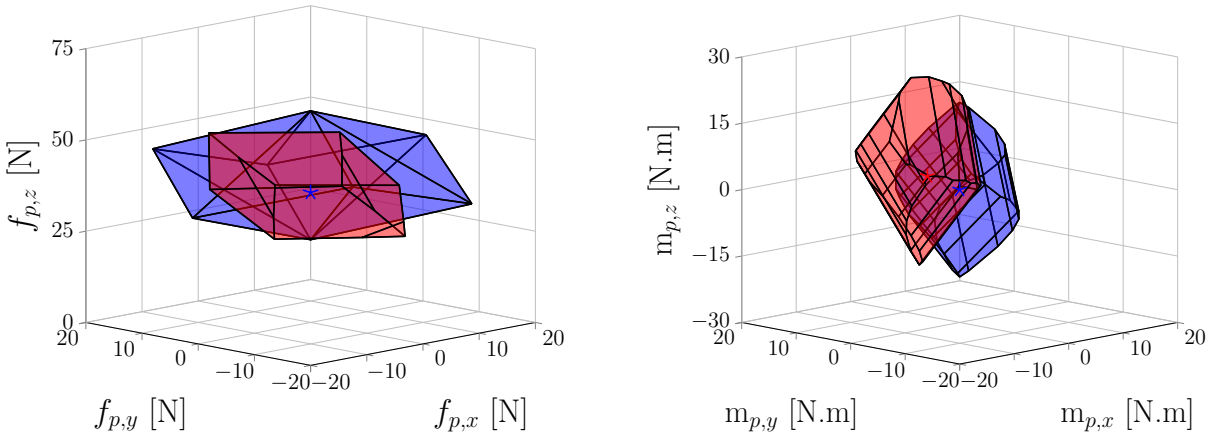


(a) Feasibility margin of $\mathcal{F}_p^s$ (b) Feasibility margin of $\mathcal{M}_p^s$

Figure 5.11 – Evolution of feasibility margins of the platform regarding different leg angles in flat platform orientation. The leg angles corresponding to optimal wrench feasibility are found at 32° for the force feasibility (8.23 N) and 41° for the moment feasibility (6.13 N.m).

It can be found in the results that the optimal leg angles maximising the force feasibility are occasionally 32° for both orientations (which are different with another inclination), a relatively wider configuration compared to the normal one where the leg angles are 45°, while for the moment feasibility, the optimal angles are close to the values of the normal configuration, but with 7° of difference for two different platform orientations. One may additionally remark that the moment feasibility margins with leg angles under 8° are

zero, which means that the balance of the static equilibrium is even infeasible for those leg configurations. This makes the force feasibility margins under such leg configurations become uninterpretable since the computation of available force sets requires to compensate the gravity moment on the platform, which is however infeasible in these leg angles. The static AWS $\mathcal{F}_p^s$, $\mathcal{M}_p^s$ with the optimal leg angles are visualised in Fig. 5.12. These results of the wrench feasibility analysis allow to determine the optimal leg angles which ensure the maximisation of force or moment feasibility according to the specific needs of the manipulation tasks.



(a) Optimal $\mathcal{F}_p^s$ in two orientations

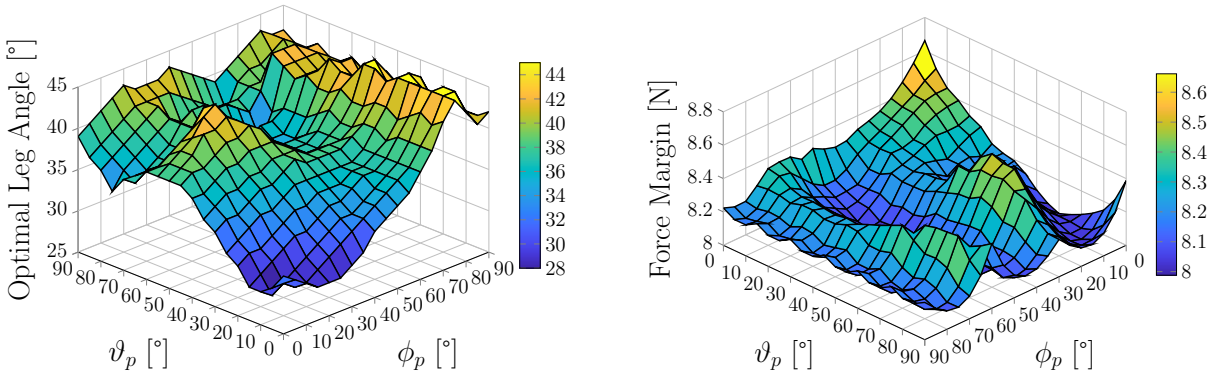
(b) Optimal $\mathcal{M}_p^s$ in two orientations

Figure 5.12 – Optimal Available Wrench Sets of the platform $\mathcal{F}_p^s$ and $\mathcal{M}_p^s$ in flat (blue regions) and inclined (red regions) orientations with the optimal leg angles chosen from Fig. 5.11 to maximise the feasibility margin. The star points correspond to the static equilibrium at each configuration.

Then, the feasibility analysis on the evolution of the platform orientation can be done, in which the optimal leg configuration corresponding to each of the platform orientation cases can be found. The idea of conducting this analysis is also to verify if the platform is wrench feasible at static equilibrium in omnidirectional directions, i.e. with the platform's roll and pitch ranging from -180° to 180°. Remark however that as the leg angles are restricted between 0° and 90° to avoid the singularity, implying the platform is always supported above the UAVs, the orientations with the roll and pitch of the platform remaining within [-90°, 90°] are feasible in such leg configurations. In addition, the configurations of the platform's roll and pitch in the interval of [-90°, 0°] and [0, 90°] are symmetric and should have the same results on the wrench feasibility. Therefore, the cases of the platform angles within [0, 90°] are of interest in this analysis to determine the optimal leg angles and feasibility margins. It is furthermore noticed that the cases of the platform angles within [-180°, -90°] and [90°, 180°] can be also seen as symmetric configurations but with the leg angles being restricted within [-90°, 0°] (i.e. the platform is beneath the

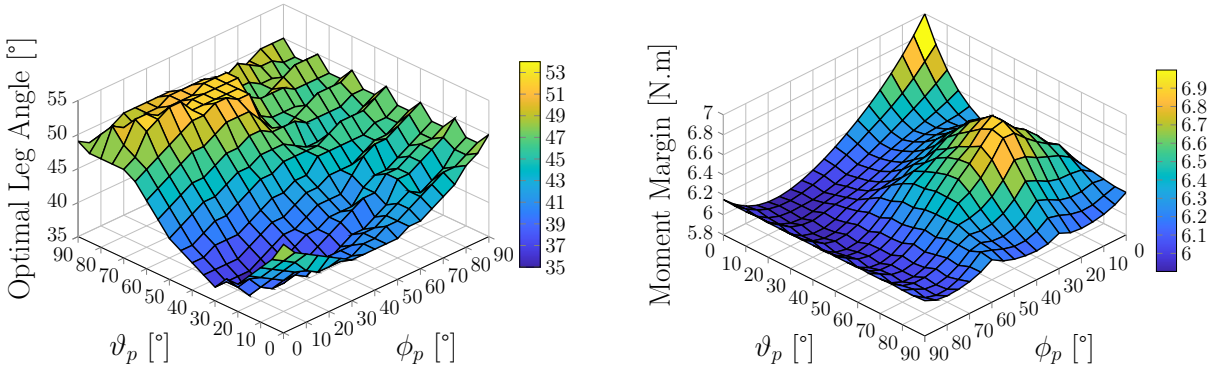
UAVs). Their results are thus easy to derive considering the same computation process. The transition between two singularity-free leg configurations (between the intervals of [-90°, 0°] and [0, 90°]) is however not studied in this thesis. It can be seen as a prerequisite in the setup of the robot assembly according to a specific task.

Apart from the discretisation of leg angles by angle step of 1° finding the maximum feasibility margins at each given orientation, an angle step of 5° on the roll (ϕ_p) and pitch (ϑ_p) of the platform is furthermore taken into account for varying the orientations. The optimal leg angles and their maximum force and moment feasibility margins can be visualised respectively by the 3D mesh plots in Fig. 5.13 and Fig. 5.14. The precision of this analysis is thus 5° in angles of the platform orientation, and 1° for the leg angles.



(a) Optimal leg angles (b) Maximum force feasibility margin

Figure 5.13 – Optimal leg angles for maximising the force feasibility margin.



(a) Optimal leg angles (b) Maximum moment feasibility margin

Figure 5.14 – Optimal leg angles for maximising the moment feasibility margin.

It can be remarked from these results that the values of the optimal leg angles for the force feasibility are located in relatively wider configuration ($\boldsymbol{\theta}_l$ from 28° to 45°), while for the moment feasibility they are more symmetrically distributed around the normal configuration ($\boldsymbol{\theta}_l$ between 35° and 55°). The best wrench-feasible platform orientation is

found at the point where the roll and pitch are zero, but a locally-optimal configuration can also be noticed around the point $\phi_p = 25°, \vartheta_p = 60°$. In addition, the feasibility margins for both force and moment of the platform in optimal leg configurations are always above a value considerably high enough to ensure the robustness of the static equilibrium (about 8 N for the force feasibility and 6 N.m for the moment feasibility), which has validated the capability of the FPR for executing the tasks required to generate omnidirectional wrench in various orientations.

5.6 Conclusion

In this chapter, a detailed analysis of the wrench feasibility of the FPR is conducted. The definition of the vertices depicting the actuation limitation of a single UAV allows to discretise the Available Thrust Set, which can then be mapped by the wrench matrix to obtain the Available Wrench Set of the FPR using respectively the V-representation and H-representation of polytopes. The static available wrench set and the static feasibility by a quantitative metric called static feasibility margin are introduced, which is then applied to determine the optimal leg configurations with different platform orientations to maximise the wrench feasibility of the platform. This analysis is thus a quantitative proof of the capacity of the FPR in performing manipulation tasks, which is applicable to optimal configuration planning, especially for determining the internal leg configurations.

While all the computation in this chapter has been implemented in Matlab, the presented algorithms can potentially be achieved in other programming platforms such as C++ or Python, in order to compute the optimal leg configurations in real-time. However, the computation of wrench sets and the optimal leg angles still needs to solve a relatively large number of QP problems for every single configuration, which may be handled in future works by more efficient algorithms that reduce the complexity of the problem for every case.

CONCLUSION

Summary of the Thesis

The domain of aerial manipulation has experienced rapid development in recent years. Although a variety of aerial manipulator architectures have been proposed and studied in the literature, evolving from single Unmanned Aerial Vehicle (UAV) equipped with onboard actuators to multi-UAV cooperative systems, there still have been a lot of scientific issues awaited to be overcome, such as payload capacity, flight stability, wrench feasibility and manipulability of this kind of aerial robots. For this, a multi-UAV parallel robot named Flying Parallel Robot has been proposed within the thesis of [Six, 2018a], which presents better payload capacity, flight stability and accuracy, and can achieve full manipulability in 3-dimensional space thanks to the design of a parallel passive architecture actuated by multiple UAVs.

On the basis of the previous works for generic modelling and motion control applied to the FPR, the works presented in this thesis have been focusing on decentralized control and estimation techniques dealing with the interaction with the environment, which has extended the capability of the FPR for performing potentially industrial manipulation tasks while facing with challenges that might appear in real-world scenarios. More precisely, the main contents of this manuscript have been as follows:

- **Chapter 1** has been devoted to a literature review of aerial manipulation with different generations of aerial manipulators, and the introduction to the design of Flying Parallel Robot as well as the generic modelling and motion control of the FPR. These previous works have been an inspiration and a solid basis for the works conducted in this thesis.

- **Chapter 2** has investigated in depth the modelling of a specific FPR composed of a moving platform and a number of one-DoF legs attached with multirotors. Analytical expressions of geometric and kinematic relations have been derived, relating the robot pose/velocity to the positions/linear velocities of the multirotors. Dynamic modelling has also been done based on numerical algorithms using Khalil's method and Featherstone's Spatial Vector notation. Numerical validation of all the models has been conducted to verify the modelling correctness.

- **Chapter 3** has tackled the problems related to the interaction with the environment by the FPR. Several momentum-based observers for estimating the external wrench exerted on the robot are firstly implemented and compared, which are then used in an impedance-based controller with wrench tracking capability shown to be able to deal with the modelling uncertainties, perturbations and physical interactions in the experimental validation.

- **Chapter 4** has dealt with the decentralized estimation and control strategies for the FPR being more robust and independent on external localisation systems. A vision-based pose estimation technique using ArUco marker system has been adopted, with the estimation results enhanced by a quaternion-based Extended Kalman Filter considering the IMU measurements, for which a methodology of reconstructing a partial set of robot pose and velocity has been shown to be sufficient to construct a teleoperable system. Then several decentralized controllers according to the previously presented motion control and interaction control schemes have been investigated. These decentralized methods have been experimentally validated for precise positioning of the robot by teleoperation, pick-up of a payload and performing contact-based interactions.

- **Chapter 5** has presented a wrench feasibility analysis of the FPR considering the actuation limits of UAVs. A Feasible Thrust Space has been firstly computed taking into account the limited thrust and rotational movements of a UAV. The computation of the Feasible Wrench Space of the FPR has been done using both H-representation and V-representation defining a convex polytope. A quantitative metric corresponding to the feasibility margin has been adopted to analyse quantitatively the feasible spaces, which is then applied to determine the optimal leg configurations to maximise the wrench feasibility in different platform orientations.

This thesis has thus been an extension of the previous works on the FPR to develop its application in the real world and its potential merits in the industrial context. Extensive experimental results have also been a valid demonstration of the FPR that is capable of performing a variety of manipulation tasks such as the transportation of relatively heavy loads, pick and place operations in remote locations, and contact-based inspection or repair of infrastructures.

Perspectives and Future Works

This thesis has presented the control and estimation methodologies for a multi-UAV manipulator performing physical interaction with the environment, with considerations

of real-world constraints and implementations. Nevertheless, there are still places for improvements that may be completed in future works.

First of all, in terms of **control**, an impedance-based method based on the estimation of the external wrench has been implemented, regulating the robot's behaviour by the virtual impedance system when interacting with the environment. The output of this control law is however not ensured to be optimal, resulting in the positioning precision of the platform being centimetric or sometimes decimetric due to the perturbations or the uncertainties from UAVs themselves. A model-based **optimal control** algorithm may therefore be investigated, such as the Nonlinear Model Predictive Control (NMPC) as studied in [Lunni, 2017; Lee, 2020; Tzoumanikas, 2020; Yiğit, 2021a] applied to the FPR to compute the optimal control for minimising both position and wrench tracking errors with more constraints and optimisation conditions considered (like the actuation limits of UAVs and the energy consumptions). Besides, the control law considering a crossing of singularity can also be investigated [Six, 2017a], allowing the FPR to work in different configurations (where the platform is below or above the UAVs) and make possible a transition between two configurations suitable for different scenarios.

Secondly, even though a robot pose estimation technique based on onboard and intrinsic measurements has been adopted to sufficiently control the FPR with teleoperation, the lack of robustness in vision-based techniques still prevents this method to work in non-ideal environments. Therefore, the **pose estimation** based on other sources of sensors might be foreseen. On one hand, the vision-based approaches can be enhanced by redundant detection with other techniques to complete the defective situations for vision; on the other, novel strategies with no vision can be proposed to overcome the drawbacks related to the cameras. Such sources of sensors can be GPS for outdoor applications, additional IMU attached on the platform for measuring its orientation, encoders for measuring the leg angles, and/or Ultra Wide-band (UWB)-based localisation algorithm [Shule, 2020] applied to the FPR.

Thirdly, the **perception of environment** when the robot navigates is totally omitted in this thesis, which has been considered known (by MOCAP) during the experiments in Chapter 3 and handled by a human operator with teleoperation in Chapter 4. An online perception of the environment like detecting the contact surface may be implemented using an additional camera attached to the platform, for example Time-of-Flight camera in [Bodie, 2021a]. The perception may potentially be followed by a complete **vision-based control**, such as eye-in-hand image-based visual servoing (IBVS) of the platform. To further deal with the interaction with the environment, a hybrid vision/force control can be

applied as investigated in [Bellakehal, 2011]. The vision-based control may also be applied to a leg-direction-based visual servoing for maintaining a good internal configuration of the FPR. The application of such arrangements in control has been seen in cable-driven parallel robots (CDPRs) [Dallej, 2019], and classical parallel manipulators [Andreff, 2006; Zhu, 2022]. In the other aspect, the **teleoperation** of the FPR by a human operator may still be a solution for avoiding the onboard perception, but can be enhanced by a bilateral haptic feedback [Lee, 2011; Omari, 2013; Zhang, 2020], which results in a more teleoperable system for accomplishing potential manipulation tasks.

Moreover, along with an initial analysis on the **wrench analysis** of the FPR, the optimal leg configurations are determined to relatively maximise the force and moment feasibility margins of the platform in different orientations. However, this analysis was only done off-line due to the computational time and complexity. A more computationally efficient algorithm may be proposed to extend the wrench feasibility analysis to online optimal planning of the leg angles maximising the feasibility according to the actual orientation of the platform and the specific task the robot is performing.

Last but not least, the improvements on **practical and experimental aspects** might also be necessary to bring the FPR closer to its real-world application. To this point, the quadrotor UAVs used in the FPR for experimental validations are customised using low-cost materials and a minimum set of sensors with an open-source flight controller, which has shown acceptable precisions for testing flights in the laboratory environment. The final applications of the FPR in the real world must be supported by more maturely controlled UAVs with a complete set of embedded sensors including IMU, optical flow, sonar-based or laser-based height sensors and possibly GPS (for outdoor applications).

APPENDIX

Appendix A: Orientation and Rotational Kinematics

In robotics, a variety of mathematical constructs are adopted to represent the orientation (or attitude) of a rigid body in 3-dimensional space. While different ways of parametrising an orientation have their respective advantages and disadvantages, they are equivalent and should have unified conversions between each other. In addition, the rotational kinematics of a rigid body is of great importance to depict its angular motion, which can be equivalently described using different mathematical representations.

This appendix has thus objective to provide a thorough and unified reference on the definition of orientation and rotational kinematics adopted in this manuscript.

A.1 Representations of Orientation

The main mathematical notations to represent the orientation of a rigid body in 3-dimensional space are: **1) the rotation matrix**, **2) a triple of Euler angles**, **3) the unit quaternion**, and **4) the axis-angle representation**. In the following part, a detailed definition of different notations as well as their relations and useful conversions will be given. For the mathematical derivations in this section, one may refer to [Diebel, 2006; Slabaugh, 2020; Flores, 2015; Schwab, 2002].

Rotation Matrix

A rotation matrix is a matrix for rotating a vector while preserving its length by the multiplication with the vector. It is belonging to the *special orthogonal group*, denoted by $SO(3)$. If a matrix $\mathbf{R} \in SO(3)$, then it possesses the following properties:

$$\det \mathbf{R} = \pm 1, \quad \mathbf{R}^{-1} = \mathbf{R}^T \tag{A.36}$$

Rotation matrices for which the determinant $\det \mathbf{R} = 1$ are called *proper* and those for which $\det \mathbf{R} = -1$ are called *improper*. Improper rotations usually consist of a rotation followed by an inversion operation, known as *rotoinversions* and are not representing robot-body transformation [Diebel, 2006]. Therefore, the determinant of the rotation matrix for representing the orientation of a rigid body is constrained to be 1 (proper rotation

matrix), which can be written with elements as follows:

$$\mathbf{R} = \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{A.37}$$

The rotation matrix encodes the attitude of a rigid body to be the matrix that when pre-multiplied by a vector expressed in the body-fixed frame yields the same vector but expressed in the world frame[1]. That is, for a rigid body A, if $\mathbf{z} \in \mathbb{R}^3$ is a vector in the world frame and ${}^A\mathbf{z} \in \mathbb{R}^3$ is the same vector expressed in the body-fixed frame, then the following relations hold:

$$\begin{aligned} \mathbf{z} &= \mathbf{R}_A \, {}^A\mathbf{z} \\ {}^A\mathbf{z} &= \mathbf{R}_A^T \, \mathbf{z} \end{aligned} \tag{A.38}$$

with $\mathbf{R}_A$ the rotation matrix representing the orientation of the rigid body with respect to the world frame. It should be noted that there are two possible conventions for defining the rotation matrix: one mapping from the body-fixed coordinates to the world coordinates as in (A.38); the other one writes the matrix that maps from the world coordinates to the body-fixed coordinates. Although two conventions are equivalent with their conversions being as trivial as performing the transpose of a matrix, it is necessary to be sure which convention is being used so as not to lose the coherence.

The rotation matrix can also depict the relative orientation between two coordinate systems or frames, i.e. for two frames denoted by $\mathfrak{F}_A$ and $\mathfrak{F}_B$, the rotation matrix ${}^A\mathbf{R}_B$ performs a rotation from $\mathfrak{F}_B$ to $\mathfrak{F}_A$, that can be given by

$${}^A\mathbf{R}_B = \mathbf{R}_A^T \, \mathbf{R}_B \tag{A.39}$$

Note that $\mathbf{R}_A$ and $\mathbf{R}_B$ represent respectively the orientation of the rigid body A and B relative to the world frame.

Euler Angles

The most popular way to represent the orientation of a rigid body is a set of Euler angles, because of their simplicity to understand and use. A widely used convention of Euler angles for aerial robotics is ZYX, defined by *yaw*, *pitch*, *roll* angles about successive *z*-*y*-*x* axes of the body-fixed frame. Another convention is ZXZ, defined by successive *z*-*x*-*z* axes, with angles known respectively as *spin*, *nutation* and *precession*.

1. Note that *world frame* or *world coordinate system* is also referred as the global frame or inertial reference frame in the manuscript.

The Euler angles can be parametrized by a vector $\boldsymbol{\eta} \in \mathbb{R}^3$ of three angles defined about the successive body-fixed axes (with a specific order according to the ZYX or ZXZ convention). The orientation represented by Euler angles can thus be obtained by a combination of these three rotations about the single coordinate axis, which is called *coordinate rotation* denoted by $\mathbf{R}_j(\alpha) : \mathbb{R} \to SO(3)$ for $j \in \{x, y, z\}$, defining a rotation of an arbitrary angle α respectively about the x, y and z axis

$$\mathbf{R}_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \tag{A.40}$$

$$\mathbf{R}_y(\alpha) = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \tag{A.41}$$

$$\mathbf{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.42}$$

Based on the unit coordinate rotations, the rotation matrix associated with different conventions of Euler angles can be fully determined. The rotation matrix defined by ZYX Euler angles $\boldsymbol{\eta} = [\phi,\ \vartheta,\ \psi]^T$ (written in roll, pitch and yaw order) is

$$\begin{aligned} \mathbf{R} &= \mathbf{R}_z(\psi)\, \mathbf{R}_y(\vartheta)\, \mathbf{R}_x(\phi) \\ &= \begin{bmatrix} \cos\vartheta\cos\psi & \sin\phi\sin\vartheta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\vartheta\cos\psi + \sin\phi\sin\psi \\ \cos\vartheta\sin\psi & \sin\phi\sin\vartheta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\vartheta\sin\psi - \sin\phi\cos\psi \\ -\sin\vartheta & \sin\phi\cos\vartheta & \cos\phi\cos\vartheta \end{bmatrix} \end{aligned} \tag{A.43}$$

Similarly, the rotation matrix of ZXZ Euler angles $\boldsymbol{\eta} = [\psi_1,\ \phi,\ \psi_2]^T$ (for spin, nutation and precession angles) can be given by

$$\begin{aligned} \mathbf{R} &= \mathbf{R}_z(\psi_1)\, \mathbf{R}_x(\phi)\, \mathbf{R}_z(\psi_2) \\ &= \begin{bmatrix} \cos\psi_1\cos\psi_2 - \sin\psi_1\cos\phi\sin\psi_2 & -\cos\psi_1\sin\psi_2 - \sin\psi_1\cos\phi\cos\psi_2 & \sin\psi_1\sin\phi \\ \sin\psi_1\cos\psi_2 + \cos\psi_1\cos\phi\sin\psi_2 & -\sin\psi_1\sin\psi_2 + \cos\psi_1\cos\phi\cos\psi_2 & -\cos\psi_1\sin\phi \\ \sin\phi\sin\psi_2 & \sin\phi\cos\psi_2 & \cos\phi \end{bmatrix} \end{aligned} \tag{A.44}$$

One disadvantage of using Euler angles is that the singularity might occur at certain angles, which restricts the usage of this representation particularly for control. For instance the singularity occurs at pitch values of $\vartheta = \frac{\pi}{2} + n\pi$ $(n \in \mathbb{Z})$ for ZYX convention, and at nutations values of $\phi = n\pi$ $(n \in \mathbb{Z})$ for ZXZ convention. These singularities found in the various Euler angle representations are said to arise from *gimbal lock*, where the first and

third Euler angles are indistinguishable when the second Euler angle is at some critical value. This means, for the ZYX convention, when the pitch angle is 90°, the vehicle is pointing straight up, making the roll and yaw impossible to be determined. For the ZXZ convention, when the nutation angle is 0° or 180°, the other two angles are ambiguous.

Unit Quaternion

The unit quaternion uses a four-element vector to represent an orientation in 3-dimensional space. With the quaternion space defined by $\mathbb{H}$, a quaternion $\mathfrak{q} \in \mathbb{H}$ can be parametrised by

$$\mathfrak{q} = \begin{bmatrix} \mathfrak{q}_0 & \mathfrak{q}_1 & \mathfrak{q}_2 & \mathfrak{q}_3 \end{bmatrix}^T = \begin{bmatrix} \mathfrak{q}_0 \\ \mathfrak{q}_{1:3} \end{bmatrix} \tag{A.45}$$

which is composed of a scalar value $\mathfrak{q}_0$ and a complex part of the 3-dimensional vector $\mathfrak{q}_{1:3}$. For a unit quaternion, the additional constraint is that

$$\|\mathfrak{q}\| = \sqrt{\mathfrak{q}_0^2 + \mathfrak{q}_1^2 + \mathfrak{q}_2^2 + \mathfrak{q}_3^2} = 1 \tag{A.46}$$

A set of additional definitions and operations may be applied to it, such as the conjugate and inverse of the quaternion given as follows

$$\bar{\mathfrak{q}} = \begin{bmatrix} \mathfrak{q}_0 \\ -\mathfrak{q}_{1:3} \end{bmatrix}, \qquad \mathfrak{q}^{-1} = \frac{\bar{\mathfrak{q}}}{\|\mathfrak{q}\|} \tag{A.47}$$

It can be noted that for a unit quaternion, its inverse is equal to the conjugate of the quaternion, i.e. $\mathfrak{q}^{-1} = \bar{\mathfrak{q}}$ when $\|\mathfrak{q}\| = 1$.

As the unit quaternions are representatives of rigid-body orientations, the combination of two orientations can be done by the multiplication of two quaternions. The multiplication between two arbitrary quaternions $\mathfrak{q}_A$ and $\mathfrak{q}_B$ is defined by

$$\mathfrak{q}_A \circ \mathfrak{q}_B = \begin{bmatrix} \mathfrak{q}_{A,0}\mathfrak{q}_{B,0} - \mathfrak{q}_{A,1:3}^T \mathfrak{q}_{B,1:3} \\ \mathfrak{q}_{A,0}\mathfrak{q}_{B,1:3} + \mathfrak{q}_{B,0}\mathfrak{q}_{A,1:3} + \mathfrak{q}_{A,1:3} \times \mathfrak{q}_{B,1:3} \end{bmatrix} \tag{A.48}$$

with $\circ$ representing the quaternion multiplication operation. (A.48) may be rewritten as a quaternion pre-multiplied by a matrix-valued function of the other quaternion, that is

$$\begin{aligned} \mathfrak{q}_A \circ \mathfrak{q}_B &= \begin{bmatrix} \mathfrak{q}_{A,0} & -\mathfrak{q}_{A,1:3}^T \\ \mathfrak{q}_{A,1:3} & \mathfrak{q}_{A,0}\mathbf{1}_{3\times 3} + [\mathfrak{q}_{A,1:3}]_\times \end{bmatrix} \begin{bmatrix} \mathfrak{q}_{B,0} \\ \mathfrak{q}_{B,1:3} \end{bmatrix} \\ &= \begin{bmatrix} \mathfrak{q}_{B,0} & -\mathfrak{q}_{B,1:3}^T \\ \mathfrak{q}_{B,1:3} & \mathfrak{q}_{B,0}\mathbf{1}_{3\times 3} - [\mathfrak{q}_{B,1:3}]_\times \end{bmatrix} \begin{bmatrix} \mathfrak{q}_{A,0} \\ \mathfrak{q}_{A,1:3} \end{bmatrix} \end{aligned} \tag{A.49}$$

where the skew-symmetric cross product matrix $[.]_\times : \mathbb{R}^3 \to \mathbb{R}^{3\times 3}$ is defined by

$$[\mathbf{v}]_\times = \begin{bmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{bmatrix} \tag{A.50}$$

It is additionally remarked that the cross-product operation and skew-symmetric matrix have the following properties

$$\begin{aligned} \mathbf{v} \times \boldsymbol{\omega} &= -\boldsymbol{\omega} \times \mathbf{v} \\ [\mathbf{v}]_\times \boldsymbol{\omega} &= -[\boldsymbol{\omega}]_\times \mathbf{v} = [\boldsymbol{\omega}]_\times^T \mathbf{v} \end{aligned} \tag{A.51}$$

More compactly, the quaternion multiplication may be written as the second quaternion pre-multiplied by a matrix-valued function of the first quaternion, which is

$$\mathfrak{q}_A \circ \mathfrak{q}_B = \mathbf{Q}(\mathfrak{q}_A)\mathfrak{q}_B = \bar{\mathbf{Q}}(\mathfrak{q}_B)\mathfrak{q}_A \tag{A.52}$$

where the quaternion matrix function, $\mathbf{Q} : \mathbb{H} \to \mathbb{R}^{4\times 4}$ is defined by

$$\mathbf{Q}(\mathfrak{q}) = \begin{bmatrix} \mathfrak{q}_0 & -\mathfrak{q}_{1:3}^T \\ \mathfrak{q}_{1:3} & \mathfrak{q}_0 \mathbf{1}_{3\times 3} + [\mathfrak{q}_{1:3}]_\times \end{bmatrix} = \begin{bmatrix} \mathfrak{q}_0 & -\mathfrak{q}_1 & -\mathfrak{q}_2 & -\mathfrak{q}_3 \\ \mathfrak{q}_1 & \mathfrak{q}_0 & -\mathfrak{q}_3 & \mathfrak{q}_2 \\ \mathfrak{q}_2 & \mathfrak{q}_3 & \mathfrak{q}_0 & -\mathfrak{q}_1 \\ \mathfrak{q}_3 & -\mathfrak{q}_2 & \mathfrak{q}_1 & \mathfrak{q}_0 \end{bmatrix} \tag{A.53}$$

and the closely related conjugate quaternion matrix $\bar{\mathbf{Q}} : \mathbb{H} \to \mathbb{R}^{4\times 4}$ is defined by

$$\bar{\mathbf{Q}}(\mathfrak{q}) = \begin{bmatrix} \mathfrak{q}_0 & -\mathfrak{q}_{1:3}^T \\ \mathfrak{q}_{1:3} & \mathfrak{q}_0 \mathbf{1}_{3\times 3} - [\mathfrak{q}_{1:3}]_\times \end{bmatrix} = \begin{bmatrix} \mathfrak{q}_0 & -\mathfrak{q}_1 & -\mathfrak{q}_2 & -\mathfrak{q}_3 \\ \mathfrak{q}_1 & \mathfrak{q}_0 & \mathfrak{q}_3 & -\mathfrak{q}_2 \\ \mathfrak{q}_2 & -\mathfrak{q}_3 & \mathfrak{q}_0 & \mathfrak{q}_1 \\ \mathfrak{q}_3 & \mathfrak{q}_2 & -\mathfrak{q}_1 & \mathfrak{q}_0 \end{bmatrix} \tag{A.54}$$

Note that the quaternion matrix has the property of $\mathbf{Q}(\bar{\mathfrak{q}})\mathfrak{q} = \mathbf{Q}(\mathfrak{q})^T\mathfrak{q}$.

A unit quaternion can be used to represent the orientation of a rigid body. Consider a vector $\mathbf{z} \in \mathbb{R}^3$ in the global coordinates and the same vector ${}^A\mathbf{z} \in \mathbb{R}^3$ expressed in the body-fixed frame of a rigid body A. The following relation holds:

$$\begin{bmatrix} 0 \\ \mathbf{z} \end{bmatrix} = \mathfrak{q}_A \circ \begin{bmatrix} 0 \\ {}^A\mathbf{z} \end{bmatrix} \circ \bar{\mathfrak{q}}_A \tag{A.55}$$

with $\circ$ being the operation of quaternion multiplication. Moreover, a physical interpreta-

tion of unit quaternion is that it is defined by a rotation angle (the scalar value $\mathfrak{q}_0$) about a spatial axis (given by the complex vector $\mathfrak{q}_{1:3}$), as

$$\mathfrak{q} = \begin{bmatrix} \mathfrak{q}_0 \\ \mathfrak{q}_{1:3} \end{bmatrix} = \begin{bmatrix} \cos\frac{\theta}{2} \\ \sin\frac{\theta}{2}\mathbf{u} \end{bmatrix} \tag{A.56}$$

where $\frac{\theta}{2}$ is the half of the rotation angle and $\mathbf{u}$ is the unit vector defining the rotation axis.

The advantage of using unit quaternions is that there is no singularity in this representation. However, the main disadvantage of unit quaternions is that they are constrained to have unit length, a quadratic constraint that can lead to complications when attempting to optimise over the quaternion parameters. In addition, the interpretation of quaternion values is difficult, and the discontinuity between two quaternions for an identical orientation represented by $\begin{bmatrix} \mathfrak{q}_0 & \mathfrak{q}_{1:3}^T \end{bmatrix}^T$ and $\begin{bmatrix} -\mathfrak{q}_0 & -\mathfrak{q}_{1:3}^T \end{bmatrix}^T$ should be tackled.

Axis-Angle Representation

The axis-angle representation is a notation closely related to the unit quaternion, but with a more intuitive interpretation as it corresponds to a rotation vector codirectional with a rotation axis, denoted by $\mathbf{u}$, and the length being the rotation angle, denoted by θ. The rotation vector representing an orientation is therefore defined by the product of the unit rotation axis and the rotation angle, as

$$\boldsymbol{\theta} = \theta\mathbf{u} \in \mathbb{R}^3 \tag{A.57}$$

Given a vector $\mathbf{z} \in \mathbb{R}^3$, the vector rotated about the axis of rotation defined by the unit vector $\mathbf{u}$ by an angle θ can be obtained using Rodrigues' rotation formula

$$\mathbf{z}' = (\cos\theta)\mathbf{u} + (\sin\theta)(\mathbf{u} \times \mathbf{z}) + (1 - \cos\theta)(\mathbf{u} \cdot \mathbf{z})\mathbf{u} \tag{A.58}$$

However, the same orientation can be represented by an infinite number of rotation vectors, and the relation between the rotation vector and the unit quaternion is quite straightforward as shown in (A.56), which together makes the axis-angle representation less used than its counterparts.

A.2 Conversions between Representations

As the same orientation can be represented by different notations, the unified conversions between them can be determined, among which the conversions between the

commonly used representations that appear in this manuscript are summarised as follows.

Unit Quaternion ⇔ Rotation Matrix

The conversion from the unit quaternion to the rotation matrix can be derived from (A.55), which states

$$\begin{bmatrix} 0 \\ \mathbf{z} \end{bmatrix} = \bar{\mathbf{Q}}(\mathfrak{q}_A)^T \mathbf{Q}(\mathfrak{q}_A) \begin{bmatrix} 0 \\ {}^A\mathbf{z} \end{bmatrix} = \begin{bmatrix} 1 & \mathbf{0}_3^T \\ \mathbf{0}_3 & \mathbf{R}_A(\mathfrak{q}_A) \end{bmatrix} \begin{bmatrix} 0 \\ {}^A\mathbf{z} \end{bmatrix} \tag{A.59}$$

from which the rotation matrix $\mathbf{R}(\mathfrak{q}) : \mathbb{H} \to SO(3)$ corresponding to the unit quaternion $\mathfrak{q}$ can be written by

$$\mathbf{R}(\mathfrak{q}) = \begin{bmatrix} \mathfrak{q}_0^2 + \mathfrak{q}_1^2 - \mathfrak{q}_2^2 - \mathfrak{q}_3^2 & 2(\mathfrak{q}_1\mathfrak{q}_2 - \mathfrak{q}_0\mathfrak{q}_3) & 2(\mathfrak{q}_1\mathfrak{q}_3 + \mathfrak{q}_0\mathfrak{q}_2) \\ 2(\mathfrak{q}_1\mathfrak{q}_2 + \mathfrak{q}_0\mathfrak{q}_3) & \mathfrak{q}_0^2 - \mathfrak{q}_1^2 + \mathfrak{q}_2^2 - \mathfrak{q}_3^2 & 2(\mathfrak{q}_2\mathfrak{q}_3 - \mathfrak{q}_0\mathfrak{q}_1) \\ 2(\mathfrak{q}_1\mathfrak{q}_3 - \mathfrak{q}_0\mathfrak{q}_2) & 2(\mathfrak{q}_2\mathfrak{q}_3 + \mathfrak{q}_0\mathfrak{q}_1) & \mathfrak{q}_0^2 - \mathfrak{q}_1^2 - \mathfrak{q}_2^2 + \mathfrak{q}_3^2 \end{bmatrix} \tag{A.60}$$

The reverse mapping from a rotation matrix to a unit quaternion is slightly more complicated. Inspection from (A.60) yields the following relations

$$\begin{cases} 4\mathfrak{q}_0^2 = 1 + r_{11} + r_{22} + r_{33} \\ 4\mathfrak{q}_1^2 = 1 + r_{11} - r_{22} - r_{33} \\ 4\mathfrak{q}_2^2 = 1 - r_{11} + r_{22} - r_{33} \\ 4\mathfrak{q}_3^2 = 1 - r_{11} - r_{22} + r_{33} \end{cases} \quad \text{and} \quad \begin{cases} 4\mathfrak{q}_0\mathfrak{q}_1 = r_{32} - r_{23} \\ 4\mathfrak{q}_0\mathfrak{q}_2 = r_{13} - r_{31} \\ 4\mathfrak{q}_0\mathfrak{q}_3 = r_{21} - r_{12} \\ 4\mathfrak{q}_1\mathfrak{q}_2 = r_{12} + r_{21} \\ 4\mathfrak{q}_1\mathfrak{q}_3 = r_{13} + r_{31} \\ 4\mathfrak{q}_2\mathfrak{q}_3 = r_{23} + r_{32} \end{cases} \tag{A.61}$$

From these relations, four different inverse conversions can be obtained, written as $\mathfrak{q}^i(\mathbf{R}) : SO(3) \to \mathbb{H}$ for $i \in \{0, 1, 2, 3\}$, and respectively defined by

$$\mathfrak{q}^0(\mathbf{R}) = \frac{1}{2} \begin{bmatrix} (1 + r_{11} + r_{22} + r_{33})^{\frac{1}{2}} \\ (r_{32} - r_{23})/(1 + r_{11} + r_{22} + r_{33})^{\frac{1}{2}} \\ (r_{31} - r_{13})/(1 + r_{11} + r_{22} + r_{33})^{\frac{1}{2}} \\ (r_{21} - r_{12})/(1 + r_{11} + r_{22} + r_{33})^{\frac{1}{2}} \end{bmatrix} \tag{A.62}$$

$$\mathfrak{q}^1(\mathbf{R}) = \frac{1}{2} \begin{bmatrix} (r_{32} - r_{23})/(1 + r_{11} - r_{22} - r_{33})^{\frac{1}{2}} \\ (1 + r_{11} - r_{22} - r_{33})^{\frac{1}{2}} \\ (r_{12} + r_{21})/(1 + r_{11} - r_{22} - r_{33})^{\frac{1}{2}} \\ (r_{13} + r_{31})/(1 + r_{11} - r_{22} - r_{33})^{\frac{1}{2}} \end{bmatrix} \tag{A.63}$$

$$\mathfrak{q}^2(\mathbf{R}) = \frac{1}{2}\begin{bmatrix} (r_{13}-r_{31})/(1-r_{11}+r_{22}-r_{33})^{\frac{1}{2}} \\ (r_{12}+r_{21})/(1-r_{11}+r_{22}-r_{33})^{\frac{1}{2}} \\ (1-r_{11}+r_{22}-r_{33})^{\frac{1}{2}} \\ (r_{23}+r_{32})/(1-r_{11}+r_{22}-r_{33})^{\frac{1}{2}} \end{bmatrix} \tag{A.64}$$

$$\mathfrak{q}^3(\mathbf{R}) = \frac{1}{2}\begin{bmatrix} (r_{21}-r_{12})/(1-r_{11}-r_{22}+r_{33})^{\frac{1}{2}} \\ (r_{13}+r_{31})/(1-r_{11}-r_{22}+r_{33})^{\frac{1}{2}} \\ (r_{23}+r_{32})/(1-r_{11}-r_{22}+r_{33})^{\frac{1}{2}} \\ (1-r_{11}-r_{22}+r_{33})^{\frac{1}{2}} \end{bmatrix} \tag{A.65}$$

Depending on the values of $\mathbf{R}$, some of these functions will produce complex numbers. To avoid this, the following composite function can be defined, in which the best of these four conversions is selected depending on the parameters of $\mathbf{R}$, i.e. $\mathfrak{q}(\mathbf{R}) : SO(3) \to \mathbb{H}$

$$\mathfrak{q}(\mathbf{R}) := \begin{cases} \mathfrak{q}^0(\mathbf{R}) \text{ if } r_{22} > -r_{33},\ r_{11} > -r_{22},\ r_{11} > -r_{33} \\ \mathfrak{q}^1(\mathbf{R}) \text{ if } r_{22} < -r_{33},\ r_{11} > r_{22},\quad r_{11} > r_{33} \\ \mathfrak{q}^2(\mathbf{R}) \text{ if } r_{22} > r_{33},\quad r_{11} < r_{22},\quad r_{11} < -r_{33} \\ \mathfrak{q}^3(\mathbf{R}) \text{ if } r_{22} < r_{33},\quad r_{11} < -r_{22},\ r_{11} < r_{33} \end{cases} \tag{A.66}$$

Euler Angles ⇔ Unit Quaternion

From the definition of unit quaternion by a rotation axis with the rotation angle in (A.56), the unit quaternions $\mathfrak{q}_j(\alpha) : \mathbb{R} \to \mathbb{H}$ for $j \in \{x, y, z\}$ corresponding to the unit coordinate rotation respectively about x, y and z axis can be obtained as follows

$$\mathfrak{q}_x(\alpha) = \begin{bmatrix} \cos\frac{\alpha}{2} & \sin\frac{\alpha}{2} & 0 & 0 \end{bmatrix}^T \tag{A.67}$$

$$\mathfrak{q}_y(\alpha) = \begin{bmatrix} \cos\frac{\alpha}{2} & 0 & \sin\frac{\alpha}{2} & 0 \end{bmatrix}^T \tag{A.68}$$

$$\mathfrak{q}_z(\alpha) = \begin{bmatrix} \cos\frac{\alpha}{2} & 0 & 0 & \sin\frac{\alpha}{2} \end{bmatrix}^T \tag{A.69}$$

Based on these quaternions for unit coordinate rotations, the unit quaternions associated with ZYX and ZXZ Euler angles are respectively

$$\begin{aligned} \mathfrak{q}_{zyx} &= \mathfrak{q}_z(\psi) \circ \mathfrak{q}_y(\vartheta) \circ \mathfrak{q}_x(\phi) \\ &= \begin{bmatrix} \cos\frac{\phi}{2}\cos\frac{\vartheta}{2}\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\sin\frac{\vartheta}{2}\sin\frac{\psi}{2} \\ \sin\frac{\phi}{2}\cos\frac{\vartheta}{2}\cos\frac{\psi}{2} - \cos\frac{\phi}{2}\sin\frac{\vartheta}{2}\sin\frac{\psi}{2} \\ \cos\frac{\phi}{2}\sin\frac{\vartheta}{2}\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\cos\frac{\vartheta}{2}\sin\frac{\psi}{2} \\ \cos\frac{\phi}{2}\cos\frac{\vartheta}{2}\sin\frac{\psi}{2} - \sin\frac{\phi}{2}\sin\frac{\vartheta}{2}\cos\frac{\psi}{2} \end{bmatrix} \end{aligned} \tag{A.70}$$

$$
\begin{aligned}
\mathfrak{q}_{zxz} &= \mathfrak{q}_z(\psi_1) \circ \mathfrak{q}_x(\phi) \circ \mathfrak{q}_z(\psi_2) \\
&= \begin{bmatrix}
\cos\frac{\psi_1}{2}\cos\frac{\phi}{2}\cos\frac{\psi_2}{2} - \sin\frac{\psi_1}{2}\cos\frac{\phi}{2}\sin\frac{\psi_2}{2} \\
\cos\frac{\psi_1}{2}\sin\frac{\phi}{2}\cos\frac{\psi_2}{2} + \sin\frac{\psi_1}{2}\sin\frac{\phi}{2}\sin\frac{\psi_2}{2} \\
\sin\frac{\psi_1}{2}\sin\frac{\phi}{2}\cos\frac{\psi_2}{2} - \cos\frac{\psi_1}{2}\sin\frac{\phi}{2}\sin\frac{\psi_2}{2} \\
\sin\frac{\psi_1}{2}\cos\frac{\phi}{2}\cos\frac{\psi_2}{2} + \cos\frac{\psi_1}{2}\cos\frac{\phi}{2}\sin\frac{\psi_2}{2}
\end{bmatrix}
\end{aligned}
\tag{A.71}
$$

The reverse mapping from the unit quaternion to Euler angles in different conventions can be identified from (A.43), (A.44) and (A.60), which can be written by

$$
\boldsymbol{\eta}_{zyx} = \begin{bmatrix} \phi \\ \vartheta \\ \psi \end{bmatrix} = \begin{bmatrix} \texttt{atan2}(r_{32}, r_{33}) \\ \texttt{asin}(-r_{31}) \\ \texttt{atan2}(r_{21}, r_{11}) \end{bmatrix} = \begin{bmatrix} \texttt{atan2}\Big(2(\mathfrak{q}_0\mathfrak{q}_1 + \mathfrak{q}_2\mathfrak{q}_3), \mathfrak{q}_0^2 - \mathfrak{q}_1^2 - \mathfrak{q}_2^2 + \mathfrak{q}_3^2\Big) \\ \texttt{asin}\Big(2(\mathfrak{q}_0\mathfrak{q}_2 - \mathfrak{q}_1\mathfrak{q}_3)\Big) \\ \texttt{atan2}\Big(2(\mathfrak{q}_0\mathfrak{q}_3 + \mathfrak{q}_1\mathfrak{q}_3), \mathfrak{q}_0^2 + \mathfrak{q}_1^2 - \mathfrak{q}_2^2 - \mathfrak{q}_3^2\Big) \end{bmatrix} \tag{A.72}
$$

$$
\boldsymbol{\eta}_{zxz} = \begin{bmatrix} \psi_1 \\ \phi \\ \psi_2 \end{bmatrix} = \begin{bmatrix} \texttt{atan2}(r_{13}, -r_{23}) \\ \texttt{acos}(r_{33}) \\ \texttt{atan2}(r_{31}, r_{32}) \end{bmatrix} = \begin{bmatrix} \texttt{atan2}\Big(2(\mathfrak{q}_0\mathfrak{q}_2 + \mathfrak{q}_1\mathfrak{q}_3), 2(\mathfrak{q}_0^2\mathfrak{q}_1^2 - \mathfrak{q}_2^2\mathfrak{q}_3^2)\Big) \\ \texttt{acos}(\mathfrak{q}_0^2 - \mathfrak{q}_1^2 - \mathfrak{q}_2^2 + \mathfrak{q}_3^2) \\ \texttt{atan2}\Big(2(\mathfrak{q}_1\mathfrak{q}_3 - \mathfrak{q}_0\mathfrak{q}_2), 2(\mathfrak{q}_2^2\mathfrak{q}_3^2 + \mathfrak{q}_0^2\mathfrak{q}_1^2)\Big) \end{bmatrix} \tag{A.73}
$$

where `atan2`, `asin` and `acos` are the arctan, arcsin and arccos functions implemented in computer languages that produce correct results for all possible orientations.

A.3 Rotational Kinematics

Rotational kinematics is the study of rigid body motion irrespective of the forces and moments involved. In this section, the kinematic relations depicting the rotational movement of a rigid body using different representations are presented, including the relationship of derivatives of rotation matrix, Euler angles and unit quaternion with respect to the angular velocity of the rigid body as in [Zhao, 2016; Diebel, 2006; Schwab, 2002].

Derivative of Rotation Matrix

Let $\mathbf{R}_A \in SO(3)$ be the rotation matrix of a rigid body A, satisfying $\mathbf{R}_A^T\,\mathbf{R}_A = \mathbf{1}_{3\times3}$ (i.e. $\texttt{det}\ \mathbf{R}_A = 1$). The time derivative of the rotation matrix is defined by

$$
\begin{aligned}
\dot{\mathbf{R}}_A &= [\boldsymbol{\omega}_A]_\times \mathbf{R}_A \\
\dot{\mathbf{R}}_A &= \mathbf{R}_A [{}^A\boldsymbol{\omega}_A]_\times
\end{aligned}
\tag{A.74}
$$

where $[.]_\times$ represents the skew-symmetric matrix defined in (A.50), $\boldsymbol{\omega}_A$ and ${}^A\boldsymbol{\omega}_A$ are the angular velocity of the body A expressed respectively in the world frame and the body-fixed frame.

Similarly, the derivative of a rotation matrix depicting a relative rotation between two frames, such as ${}^{A}\mathbf{R}_{B}$ from $\mathfrak{F}_B$ to $\mathfrak{F}_A$, can be given by

$$\begin{aligned} {}^{A}\dot{\mathbf{R}}_{B} &= [{}^{A}\boldsymbol{\omega}_{B/A}]_{\times}\, {}^{A}\mathbf{R}_{B} \\ {}^{A}\dot{\mathbf{R}}_{B} &= {}^{A}\mathbf{R}_{B}[{}^{B}\boldsymbol{\omega}_{B/A}]_{\times} \end{aligned} \tag{A.75}$$

where ${}^{A}\boldsymbol{\omega}_{B/A}$ is the angular velocity of $\mathfrak{F}_B$ (relative to $\mathfrak{F}_A$) expressed in $\mathfrak{F}_A$, and ${}^{B}\boldsymbol{\omega}_{B/A}$ is the same quantity expressed in $\mathfrak{F}_B$.

Rates of Euler Angles

The relationship between the angular velocity of a rigid body ${}^{A}\boldsymbol{\omega}_{A}$ expressed in its body-fixed frame and the rates of changes of Euler angles $\dot{\boldsymbol{\eta}}_{A}$ can be defined by

$${}^{A}\boldsymbol{\omega}_{A} = \mathbf{D}\dot{\boldsymbol{\eta}}_{A} \tag{A.76}$$

with the mapping matrix $\mathbf{D} \in \mathbb{R}^{3\times3}$. Depending on the convention of Euler angles, this matrix takes different forms which can be derived from the definition of the order of Euler angles, i.e. for ZYX Euler angles

$$\begin{aligned} {}^{A}\boldsymbol{\omega}_{A} &= \dot{\phi}\, \mathbf{R}_{x}^{T}(\phi) \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \dot{\vartheta}\, \mathbf{R}_{x}^{T}(\phi)\mathbf{R}_{y}^{T}(\vartheta) \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \dot{\psi}\, \mathbf{R}_{x}^{T}(\phi)\mathbf{R}_{y}^{T}(\vartheta)\mathbf{R}_{z}^{T}(\psi) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\ &= \mathbf{D}_{zyx} \begin{bmatrix} \dot{\phi} \\ \dot{\vartheta} \\ \dot{\psi} \end{bmatrix} \end{aligned} \tag{A.77}$$

with $\mathbf{R}_x(\phi)$, $\mathbf{R}_y(\vartheta)$ and $\mathbf{R}_z(\psi)$ being the unit coordinate rotations. Then the matrix $\mathbf{D}$ can be calculated as

$$\mathbf{D}_{zyx} = \begin{bmatrix} 1 & 0 & -\sin\vartheta \\ 0 & \cos\phi & \cos\vartheta\sin\phi \\ 0 & -\sin\phi & \cos\vartheta\cos\phi \end{bmatrix} \tag{A.78}$$

The matrix mapping the rates of ZXZ Euler angles to body-fixed angular velocity can be derived using the same procedure, which is however omitted here as this convention is not used to represent the rotational movement of a rigid body in this manuscript.

The reverse mapping from the body-fixed angular velocity to the Euler angle rates can be known by inverting the matrix $\mathbf{D}$ as

$$\dot{\boldsymbol{\eta}}_{A} = (\mathbf{D}^{-1}){}^{A}\boldsymbol{\omega}_{A} \tag{A.79}$$

where the inverse of the mapping matrix is written by

$$\mathbf{D}_{zyx}^{-1} = \begin{bmatrix} 1 & \sin\phi\tan\vartheta & \cos\phi\tan\vartheta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\vartheta & \cos\phi/\cos\vartheta \end{bmatrix} \tag{A.80}$$

The second-order kinematics related to the Euler angles in ZYX convention can be additionally derived from (A.76), which holds

$${}^{A}\dot{\boldsymbol{\omega}}_{A} = \mathbf{D}\ddot{\boldsymbol{\eta}}_{A} + \dot{\mathbf{D}}\dot{\boldsymbol{\eta}}_{A} \tag{A.81}$$

where ${}^{A}\dot{\boldsymbol{\omega}}_{A}$ is the angular acceleration of the rigid body expressed in its body-fixed frame $\mathfrak{F}_{A}$, $\dot{\boldsymbol{\eta}}_{A}$ and $\ddot{\boldsymbol{\eta}}_{A}$ represent the rates and second-order rates of the Euler angles, and $\dot{\mathbf{D}}$ is the time derivative of $\mathbf{D}$ written as

$$\dot{\mathbf{D}}_{zyx} = \begin{bmatrix} 0 & 0 & -\cos\vartheta\cdot\dot{\vartheta} \\ 0 & -\dot{\phi}\sin\phi & -\dot{\vartheta}\sin\vartheta\sin\phi + \dot{\phi}\cos\vartheta\cos\phi \\ 0 & -\dot{\phi}\cos\phi & -\dot{\vartheta}\sin\vartheta\cos\phi - \dot{\phi}\cos\vartheta\sin\phi \end{bmatrix} \tag{A.82}$$

Derivative of Unit Quaternion

Consider a unit quaternion of a rigid body written by $\mathfrak{q}_{A}$, its derivative $\dot{\mathfrak{q}}_{A}$ is related to the angular velocity ${}^{A}\boldsymbol{\omega}_{A}$ expressed in the body-fixed frame, which can be written by the following relationship

$$\begin{bmatrix} 0 \\ {}^{A}\boldsymbol{\omega}_{A} \end{bmatrix} = 2\bar{\mathfrak{q}}_{A} \circ \dot{\mathfrak{q}}_{A} = 2\mathbf{Q}(\mathfrak{q}_{A})^{T}\dot{\mathfrak{q}}_{A} \tag{A.83}$$

and with the inverse mapping

$$\dot{\mathfrak{q}}_{A} = \frac{1}{2}\mathfrak{q}_{A} \circ \begin{bmatrix} 0 \\ {}^{A}\boldsymbol{\omega}_{A} \end{bmatrix} = \frac{1}{2}\mathbf{Q}(\mathfrak{q}_{A}) \begin{bmatrix} 0 \\ {}^{A}\boldsymbol{\omega}_{A} \end{bmatrix} \tag{A.84}$$

Note that the relationship with respect to the angular velocity expressed in the world frame can also be derived, for which the reader may refer to [Schwab, 2002].

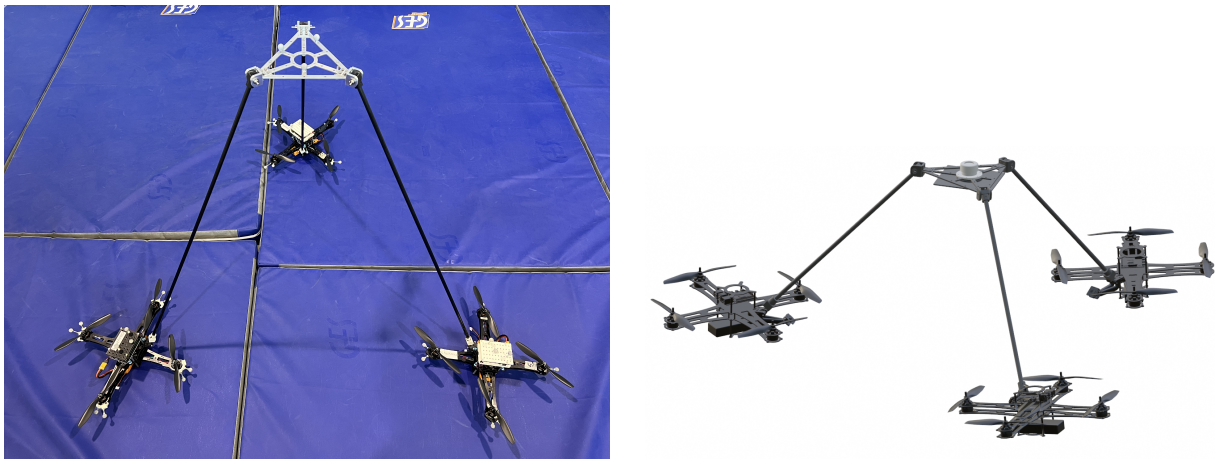
The angular acceleration of a rigid body is found by differentiation of the expression in (A.83), resulting in

$$\begin{bmatrix} 0 \\ {}^{A}\dot{\boldsymbol{\omega}}_{A} \end{bmatrix} = 2\mathbf{Q}(\mathfrak{q}_{A})^{T}\ddot{\mathfrak{q}}_{A} + 2 \begin{bmatrix} \|\dot{\mathfrak{q}}_{0}\|^{2} \\ \mathbf{0}_{3} \end{bmatrix} \tag{A.85}$$

Appendix B: Prototype and Experimental Implementation

The prototype of the Flying Parallel Robot (FPR) studied within this thesis is composed of a triangle-form platform and three legs attached with quadrotors, as shown in Fig. B.1(a). The structure of the passive architecture is designed, manufactured and assembled in the laboratory, with the original CAD model illustrated in Fig. B.1(b).

The aluminium platform weighs 250 g and the rigid legs are each 1.043 m long with a mass of 66 g. Each of them is assembled by a carbon-fibre tube and 3D-printed connection parts. The spherical joints and the bearings (for revolute joints) are based on standard parts, which are sufficiently lubricated to reduce friction in the joints. The custom quadrotors used in the FPR are based on a 34-cm Lynxmotion Crazy2Fly frame, with SunnySky 1250kV brushless DC motors, 8045 dual-blade propellers and a 3-cell LiPo battery, weighing about 1 kg. The detailed information on geometric and dynamic parameters are summarized in Table B.1, which are obtained from the CAD model in Catia V5.



(a) Prototype of the FPR with three quadrotors (b) CAD model of the FPR prototype

Figure B.1 – FPR prototype and its original CAD design for the experimental validation.

In terms of low-level quadrotor control, an open-source autopilot is chosen, which is based on PX4 (software) and Pixhawk 4 Mini/Pixhawk 5 (hardware) [Dronecode, 2021]. The flight control board can achieve different levels of quadrotor control, such as position, velocity, attitude and angular rate, with onboard IMU sensor and EKF filtering technique sufficiently estimating the state of the quadrotor, such as its attitude and linear velocity. A custom-built PX4 firmware is installed on the Pixhawk performing the attitude control such as the one presented in Section 3.3.3, which has been experimentally tuned with aggressive attitude and angular rate gains to ensure rapid convergence to the attitude

Symbol	Parameter	Value	Unit
r	Platform radius	0.127	[m]
l	Leg length	1.043	[m]
d	Spherical-joint offset	0.069	[m]
β	Yaw constant	45	[°]
m_p	Platform mass	0.250	[kg]
m_l	Leg mass	0.066	[kg]
$\mathbf{s}_p$	Platform CoM	$[0,\ 0,\ 0.010]^T$	[m]
$\mathbf{s}_l$	Leg CoM	$[0.599,\ 0,\ 0]^T$	[m]
$\mathbf{I}_p$	Platform MoI*	$\texttt{diag}(\{5.4e^{-4}, 5.4e^{-4}, 0.001\})$	[kg.m^2]
$\mathbf{I}_l$	Leg MoI*	$\texttt{diag}(\{1.6e^{-6}, 0.035, 0.035\})$	[kg.m^2]
$m_{b,1}, m_{b,2}, m_{b,3}$	Quadrotor masses	$\{1.007,\ 1.028,\ 1.018\}$	[kg]

Table B.1 – Geometric and dynamic parameters of the FPR. *MoI refers to the Moment of Inertia. $\texttt{diag}(.) : \mathbb{R}^n \rightarrow \mathbb{R}^{n\times n}$ is the diagonal matrix of a given array of values.

setpoints and robustness to the disturbances. The location of the spherical joint is given by the offset d and the yaw constant β in Table B.1, which is away from the CoM of the quadrotor and considered as a disturbance on the onboard attitude controller. On each quadrotor, a Raspberry Pi 4B is mounted working as a companion computer to handle the communication with the Pixhawk either via serial ports by microRTPS protocol [eProsima, 2022] or through Ethernet by UDP protocol. It is remarked that the Ethernet connection seemed to be more robust in terms of bandwidth and stability compared to serial communication channels. The communication between the ground computer and the companion computers is handled by ROS2 Galactic, performed through a 5-GHz Wifi network.

Appendix C: Quadrotor Thrust Regulation

The onboard flight control of the quadrotor UAV used in the FPR prototype (detailed in Appendix B) is achieved by open-source hardware and software (Pixhawk and PX4). When received the thrust magnitude setpoint (i.e. f_t^d) from a high-level controller, the onboard companion computer (i.e. Raspberry Pi) computes a scalar value as a percentage of maximum thrust and sends to the Pixhawk, i.e.

$$u_f = \frac{f_t^d}{f_{max}} \tag{C.1}$$

with u_f a value between $[0, 1]$ and f_{max} a parameter characterising the maximum thrust magnitude in the Pixhawk.

The desired thrust is then mapped by Electronic Speed Control (ESC) units to the motor speeds propelling the rotors. The regulation of motor speeds by the ESCs is however in an open loop without any feedback, which makes the actual thrust produced by a quadrotor being deviated from its desired values. When the battery is fully charged, the actual thrust produced by the quadrotor is sufficiently close to the desired value, while the mismatch between the desired and actual thrusts becomes non-negligible with low battery levels. This problem has been particularly recognised when the flight lasts a relatively long duration, during which the tracking of z position of a UAV is gradually degenerated due to drops in the battery level.

However, it has been noticed that the maximum thrust that the quadrotor can produce decreases linearly when the battery level drops. Therefore, an online regulation of the thrust commands u_f adapting the maximum thrust parameter f_{max} according to the actual battery level can be done, which is given by a linear relationship as

$$f_{max} = a_f V + b_f \tag{C.2}$$

with a and b two constants and V being the battery level in voltage.

To determine the linear relationship between the battery level and maximum thrust parameter in (C.2), a hovering flight of a single UAV has been done using a simple PD-based position controller as in [Kamel, 2017]. Note that the integral gain of a classical PID controller is disabled to not compensate for the thrust changes due to the evolution of the battery level during the hovering flight. An acceleration-based observer such as the one investigated [Tomić, 2017] is additionally implemented to estimate the evolution of the maximum thrust f_{max} during the flight, which can be summarised as follows:

Consider the 3-dimensional thrust force produced by a quadrotor i expressed in its own frame $\mathfrak{F}_{bi}$ written by ${}^{bi}\mathbf{f}_i = [0, 0, f_{t,i}]$. It can be known from the linear acceleration $\mathbf{a}_i$ of the quadrotor i expressed in $\mathfrak{F}_0$ by

$$ {}^{bi}\mathbf{f}_i = m_i \mathbf{R}_i^T (\mathbf{a}_i - \mathfrak{g}) \tag{C.3} $$

with m_i the mass of the quadrotor, $\mathbf{R}_i$ the rotation matrix representing its orientation and $\mathfrak{g} = \begin{bmatrix} 0 & 0 & -g \end{bmatrix}$ (for $g = 9.81 m/s^2$) being the gravity vector in $\mathfrak{F}_0$. One may remark that the term $\mathbf{a}_{acc} := \mathbf{R}_i^T(\mathbf{a}_i - \mathfrak{g})$ is directly measured by the accelerometer of the onboard IMU sensor, of which the z-axis element is related to $f_{t,i}$ and can be denoted by $a_{acc,z}$.

Knowing from (C.1) that $f_{max} = f_t^d / u_f$, an estimation on the maximum thrust parameter can be formulated based on the relation of (C.3) such that

$$ \hat{f}_{max}(t) = K_I \int_0^t \Big(\frac{m_i a_{acc,z}}{u_f} - \hat{f}_{max}(t - \Delta t) \Big) dt \tag{C.4} $$

where $\hat{f}_{max}(t)$, $\hat{f}_{max}(t - \Delta t)$ are the estimated maximum thrust parameter respectively at current and previous timestamps, u_f is the actual control value sent to the flight controller, and K_I is the estimation gain.

This acceleration-based estimation is equivalent to a first-order filter. That is, by deriving the relation of (C.4), one can obtain the estimation error dynamics as

$$ \dot{\hat{f}}_{max} + K_I (\hat{f}_{max} - f_{max}) = 0 \tag{C.5} $$

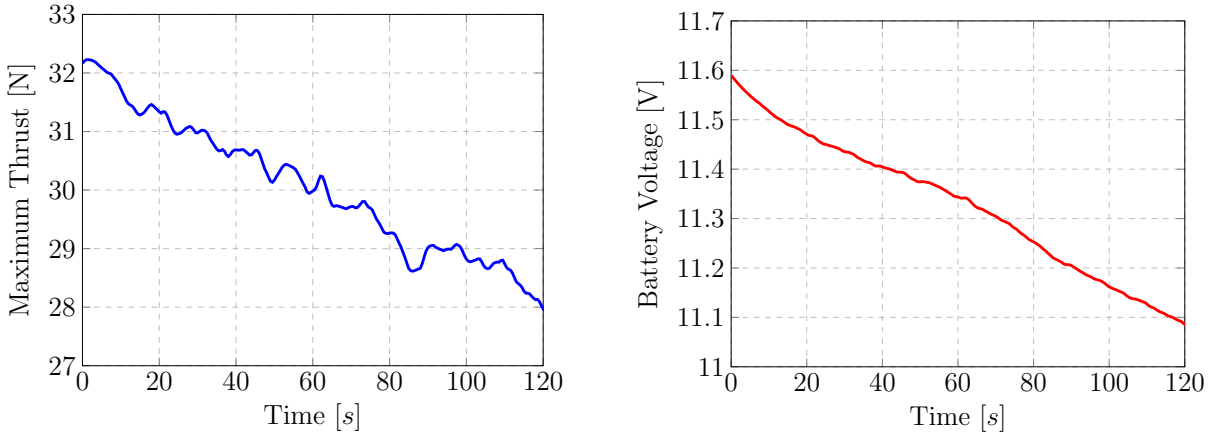
where the estimation error is given by $(\hat{f}_{max} - f_{max})$ with $f_{max} = \frac{m_i a_{acc,z}}{u_f}$ as the actual maximum thrust computed from the direct measurements. $\hat{f}_{max}$ converges asymptotically to its actual value f_{max}.

It is additionally remarked that the estimated values of $\hat{f}_{max}$ might not be smooth enough due to the noisy acceleration measurements. To avoid this, (C.4) can be further adapted to (C.6) using measurements of the linear velocity which are much better filtered by the onboard EKF technique.

$$ \hat{f}_{max}(t) = K_I \Big[\int_0^t \Big(- \frac{m_i {}^{bi}\mathfrak{g}_z}{u_f} - \hat{f}_{max}(t - \Delta t) \Big) dt + \frac{{}^{bi}\mathbf{v}_{i,z}}{u_f} \Big] \tag{C.6} $$

where ${}^{bi}\mathfrak{g}_z$ represents the z-axis element of the gravity vector expressed in $\mathfrak{F}_{bi}$ by ${}^{bi}\mathfrak{g} = \mathbf{R}_i^T \mathfrak{g}$, and ${}^{bi}\mathbf{v}_{i,z}$ is the z-axis linear velocity of the quadrotor i expressed in $\mathfrak{F}_{bi}$.

The results on the evolution of the estimated maximum thrust parameter and the battery level recorded on one quadrotor are plotted in Fig. C.1. The linear regression for obtaining the relationship of (C.2) has been done (shown in Fig. C.2), which has resulted in constant parameters for regulating the maximum thrust parameter according to the battery level on all the quadrotors given in Table C.1.



(a) Maximum thrust estimates

(b) Battery level measurements

Figure C.1 – Evolution of the estimated maximum thrust and the battery voltage level of a quadrotor during a hovering flight over 120 s.

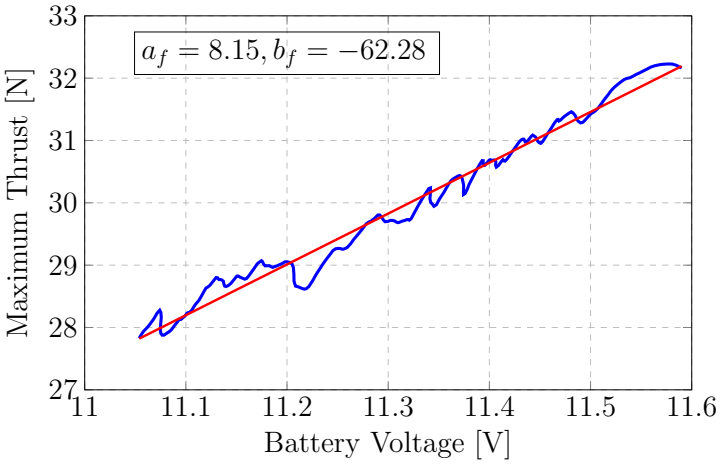


Figure C.2 – Linear regression between battery levels and maximum thrust estimates.

Quadrotor	a_f [N/V]	b_f [N]
1	8.02	-61.66
2	8.22	-64.01
3	8.15	-62.28

Table C.1 – Constant parameters for thrust regulation as a function of battery levels according to (C.2) for three quadrotors.

Appendix D: Calibration of Fixed Transformations

As presented in Section 4.2.1, the estimation of relative pose between each UAV and the platform requires two fixed transformations to be known, which are ${}^{bi}\mathbf{T}_{Ci}$ from the camera frame to multirotor's body frame, and ${}^{p}\mathbf{T}_{M}$ from the ArUco marker frame to the platform frame. A calibration process is done to obtain the values of these two transformation matrices using both marker pose estimation algorithm and the MOCAP data. It is noted that to reduce the calibration errors, instead of detecting a single marker using ArUco algorithm, a ChArUco marker detection algorithm available in OpenCV [OpenCV, 2022] is chosen. The ChArUco maker is a chessboard composed of 3×3 squares and four ArUco markers inside the white squares. The pose estimation is therefore more accurate as the algorithm detects the corner points of four markers and the black squares, and estimates the pose based on these detected features. However, the size of four ArUco markers in a ChArUco marker is too small, limiting this method only for calibration (detecting in a short distance), but not for the pose estimation discussed in Section 4.2.1.

For calibrating ${}^{bi}\mathbf{T}_{Ci}$, a ChArUco marker is placed in the flight arena, whose pose relative to the camera frame of the quadrotor is being estimated by the ChArUco algorithm (as shown in Fig. D.1). To distinguish with the ArUco marker attached below the platform, the ChArUco marker for calibration has a maker frame $\mathfrak{F}_{M_c}$. The ground-truth poses of the UAV and the marker are recorded by the MOCAP system, which allows to know their global poses written in transformation matrices as

$$\mathbf{T}_{bi} = \begin{bmatrix} \mathbf{R}_i & \mathbf{p}_i \\ \mathbf{0}_3^T & 1 \end{bmatrix}, \quad \mathbf{T}_{M_c} = \begin{bmatrix} \mathbf{R}_{M_c} & \mathbf{p}_{M_c} \\ \mathbf{0}_3^T & 1 \end{bmatrix} \tag{D.1}$$

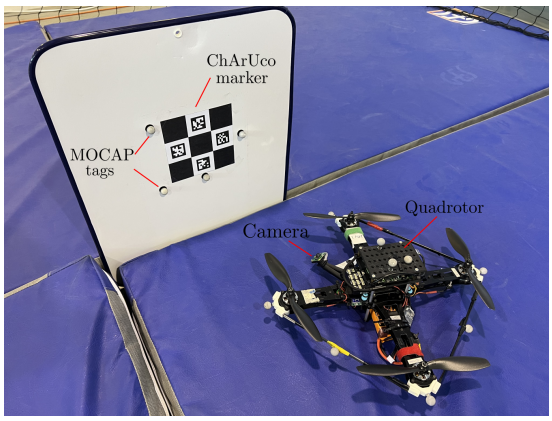
where $\mathbf{R}_i$, $\mathbf{p}_i$ are the rotation matrix and translation vector for the pose of the UAV i, and $\mathbf{R}_{M_c}$, $\mathbf{p}_{M_c}$ represent the pose of the marker, with all the quantities expressed in the global frame $\mathfrak{F}_0$. From these global poses, the relative pose from the marker frame to the body frame of UAV i can be known by

$${}^{bi}\mathbf{T}_{M_c} = \mathbf{T}_{bi}^{-1}\mathbf{T}_{M_c} \tag{D.2}$$

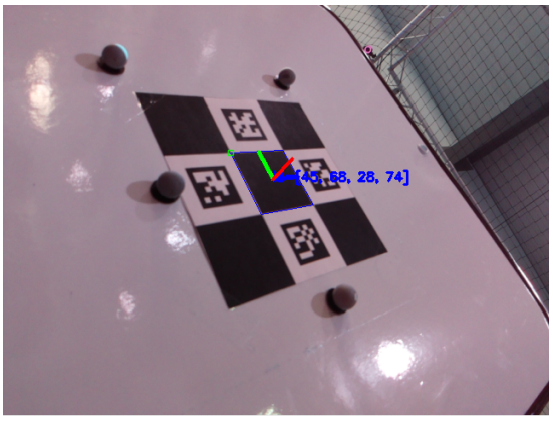
with $\mathbf{T}_{bi}^{-1}$ being the inverse transformation that has been defined in (4.3).

Knowing the relative pose of the marker frame expressed in the camera frame estimated by the algorithm, written as ${}^{Ci}\mathbf{T}_{M_c}$, the fixed transformation of ${}^{bi}\mathbf{T}_{Ci}$ can finally be known by

$${}^{bi}\mathbf{T}_{Ci} = {}^{bi}\mathbf{T}_{M_c}\,{}^{Ci}\mathbf{T}_{M_c}^{-1} \tag{D.3}$$



(a) Calibration configuration



(b) Pose estimation by ChArUco algorithm

Figure D.1 – Calibration of the camera pose relative to the multirotor's frame, i.e. ${}^{bi}\mathbf{T}_{Ci}$, using ChArUco pose estimation algorithm and MOCAP data.

The calibration of the fixed transformation ${}^{p}\mathbf{T}_{M}$ can be done using a similar process. However, it is remarked that the global poses of the platform and the marker can be directly detected by MOCAP, written respectively by $\mathbf{T}_p$ and $\mathbf{T}_M$ expressed in $\mathfrak{F}_0$. The pose of the marker relative to the platform frame is therefore known by

$$ {}^{p}\mathbf{T}_{M} = \mathbf{T}_{p}^{-1}\mathbf{T}_{M} \tag{D.4} $$

with the inverse transformation $\mathbf{T}_{p}^{-1}$.

The fixed transformations have thus been calibrated, which are expressed respectively by a rotation matrix and a translation vector. For the orientation, a simpler representation can be chosen to reduce the number of parameters for saving these constant values. The Euler angles in ZXZ convention are selected because they are simple and intuitive to use. The conversion from the rotation matrix to the ZXZ Euler angles is given in (A.73). Finally, the constant parameters for all the fixed transformations are summarised in Table D.1. Noted that the fixed transformation ${}^{bi}\mathbf{T}_{Ci}$ for all the UAVs are supposed to be identical.

Transformation	$\mathbf{p}$ [cm]			$\boldsymbol{\eta}_{zxz}$ [rad]		
	x	y	z	ψ_1	ϕ	ψ_2
Camera Pose	8.45	−9.35	0.79	1.561	0.708	0.829
Marker Pose	−0.83	0.26	−2.47	−0.716	3.136	0.837

Table D.1 – Constant parameters (translation vector and ZXZ Euler angles) of the fixed transformations for the camera pose and the ArUco marker pose.

BIBLIOGRAPHY

Abbaraju, P., X. Ma, G. Jiang, M. Rastgaar, and R. M. Voyles (2021). Aerodynamic Modeling of Fully-Actuated Multirotor UAVs with Nonparallel Actuators. In: *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Aceituno-Cabezas, B., C. Mastalli, H. Dai, M. Focchi, A. Radulescu, D. G. Caldwell, J. Cappelletto, J. C. Grieco, G. Fernández-López, and C. Semini (2018). Simultaneous Contact, Gait, and Motion Planning for Robust Multilegged Locomotion via Mixed-Integer Convex Optimization. *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pages: 2531–2538.

Alexis, K., G. Darivianakis, M. Burri, and R. Siegwart (2016). Aerial robotic contact-based inspection: planning and control. *Autonomous Robots*, vol. 40, no. 4, pages: 631–655.

Andreff, N. and P. Martinet (2006). Unifying Kinematic Modeling, Identification, and Control of a Gough–Stewart Parallel Robot Into a Vision-Based Framework. *IEEE Transactions on Robotics*, vol. 22, no. 6, pages: 1077–1086.

Anzai, T., M. Zhao, S. Nozawa, F. Shi, K. Okada, and M. Inaba (2018). Aerial Grasping Based on Shape Adaptive Transformation by HALO: Horizontal Plane Transformable Aerial Robot with Closed-Loop Multilinks Structure. In: *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*.

Augugliaro, F., A. Mirjan, F. Gramazio, M. Kohler, and R. D'Andrea (2013). Building Tensile Structures with Flying Machines. In: *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE.

Azar, A. T. and F. E. Serrano (2015). Design and Modeling of Anti Wind Up PID Controllers. In: *Complex System Modelling and Control Through Intelligent Soft Computations*. Springer International Publishing.

Barawkar, S., M. Radmanesh, M. Kumar, and K. Cohen (2017). Admittance based force control for collaborative transportation of a common payload using two uavs. In: *Proceedings of the 2017 Dynamic Systems and Control Conference*, vol. 58295, American Society of Mechanical Engineers.

Barrau, A. and S. Bonnabel (2017). The Invariant Extended Kalman Filter as a Stable Observer. *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pages: 1797–1812.

Bartelds, T., A. Capra, S. Hamaza, S. Stramigioli, and M. Fumagalli (2016). Compliant Aerial Manipulators: Toward a New Generation of Aerial Robotic Workers. *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pages: 477–483.

Bellakehal, S., N. Andreff, Y. Mezouar, and M. Tadjine (2011). Vision/force control of parallel robots. *Mechanism and Machine Theory*, vol. 46, no. 10, pages: 1376–1395.

Bellens, S., J. De Schutter, and H. Bruyninckx (2012). A hybrid pose/wrench control framework for quadrotor helicopters. In: *Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA)*.

Bellicoso, C. D., L. R. Buonocore, V. Lippiello, and B. Siciliano (2015). Design, Modeling and Control of a 5-DoF Light-Weight Robot Arm for Aerial Manipulation. In: *Proceedings of the 2015 23rd Mediterranean Conference on Control and Automation (MED)*, IEEE.

Bernard, M., K. Kondak, and G. Hommel (2010). Load transportation system based on autonomous small size helicopters. *The Aeronautical Journal*, vol. 114, no. 1153, pages: 191–198.

Bernard, M., K. Kondak, I. Maza, and A. Ollero (2011). Autonomous Transportation and Deployment with Aerial Robots for Search and Rescue Missions. vol. 28, no. 6, pages: 914–931.

Bodie, K., M. Brunner, M. Pantic, S. Walser, P. Pfndler, U. Angst, R. Siegwart, and J. Nieto (2019). An Omnidirectional Aerial Manipulation Platform for Contact-Based Inspection. In: *Proceedings of the 2019 Robotics: Science and Systems (RSS)*.

Bodie, K., M. Brunner, M. Pantic, S. Walser, P. Pfndler, U. Angst, R. Siegwart, and J. Nieto (2021). Active Interaction Force Control for Contact-Based Inspection With a Fully Actuated Aerial Vehicle. *IEEE Transactions on Robotics*, vol. 37, no. 3, pages: 709–722.

Bodie, K., M. Tognon, and R. Siegwart (2021). Dynamic End Effector Tracking With an Omnidirectional Parallel Aerial Manipulator. *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pages: 8165–8172.

Bonnabel, S., P. Martin, and E. Salaün (Dec. 2009). Invariant Extended Kalman Filter: theory and application to a velocity-aided attitude estimation problem. In: *Proceedings of the 48th IEEE Conference on Decision and Control (CDC)*, URL: https://hal-mines-paristech.archives-ouvertes.fr/hal-00494342.

Borst, C., M. Fischer, and G. Hirzinger (2004). Grasp Planning: How to Choose a Suitable Task Wrench Space. In: *Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA)*.

Bosscher, P., A. T. Riechel, and I. Ebert-Uphoff (2006). Wrench-feasible workspace generation for cable-driven robots. *IEEE Transactions on Robotics*, vol. 22, no. 5, pages: 890–902.

Bouchard, S., C. Gosselin, and B. Moore (2010). On the ability of a cable-driven robot to generate a prescribed set of wrenches. *Journal of Mechanisms and Robotics*, vol. 2, no. 1, pages: 1–10.

Bourquardez, O., R. Mahony, N. Guenard, F. Chaumette, T. Hamel, and L. Eck (2009). Image-Based Visual Servo Control of the Translation Kinematics of a Quadrotor Aerial Vehicle. *IEEE Transactions on Robotics*, vol. 25, no. 3, pages: 743–749.

Brescianini, D., M. Hehn, and R. D'Andrea (2013). *Nonlinear Quadrocopter Attitude Control*. Technical report. ETH Zurich.

Brescianini, D. and R. D'Andrea (2018). Computationally Efficient Trajectory Generation for Fully Actuated Multirotor Vehicles. *IEEE Transactions on Robotics*, vol. 34, no. 3, pages: 555–571.

Brilliant (2022). *Finite Elements*. URL: https://brilliant.org/wiki/finite-elements/.

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

Bruno, S. and V. Luigi (2000). *Robot Force Control*. Springer Science & Business Media.

Büeler, B. and K. Enge Andreas and Fukuda (2000). Exact Volume Computation for Polytopes: A Practical Study. In: *Polytopes — Combinatorics and Computation*. Basel: Birkhäuser Basel.

Caballero, A., A. Suarez, F. Real, V. M. Vega, M. Bejar, A. Rodriguez-Castano, and A. Ollero (2018). First experimental results on motion planning for transportation in aerial long-reach manipulators with two arms. In: *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Caccavale, F., G. Giglio, G. Muscio, and F. Pierri (2015). Cooperative impedance control for multiple UAVs with a robotic arm. In: *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Cano, R., C. Pérez, F. Pruaño, A. Ollero, and G. Heredia (2013). Mechanical Design of a 6-DOF Aerial Manipulator for assembling bar structures using UAVs. In: *Proceedings of the 2nd IFAC Workshop on Research, Education and Development of Unmanned Aerial Systems*.

Car, M., A. Ivanovic, M. Orsag, and S. Bogdan (2018). Impedance Based Force Control for Aerial Robot Peg-in-Hole Insertion Tasks. In: *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Cardona, G. A., D. S. D'Antonio, C.-I. Vasile, and D. Saldaña (2021). Non-Prehensile Manipulation of Cuboid Objects Using a Catenary Robot. In: *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Carpentier, J., R. Budhiraja, and N. Mansard (2017). Learning feasibility constraints for multi-contact locomotion of legged robots. In: *Proceedings of the 2017 Robotics: Science and Systems (RSS)*, vol. 13.

Carpentier, J. and N. Mansard (2018). Multi-contact Locomotion of Legged Robots. *IEEE Transactions on Robotics*, vol. 34, no. 6, pages: 1441–1460.

Cataldi, E., G. Muscio, M. A. Trujillo, Y. Rodriguez, F. Pierri, G. Antonelli, F. Caccavale, A. Viguria, S. Chiaverini, and A. Ollero (2016). Impedance Control of an aerial-manipulator: Preliminary results. In: *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Chaumette, F. (2004). Image moments: a general and useful set of features for visual servoing. *IEEE Transactions on Robotics*, vol. 20, no. 4, pages: 713–723.

Chen, Y., W. Zhan, B. He, L. Lin, Z. Miao, X. Yuan, and Y. Wang (2020). Robust Control for Unmanned Aerial Manipulator Under Disturbances. *IEEE Access*, vol. 8, pages: 129869–129877.

Chung, S.-J., A. A. Paranjape, P. Dames, S. Shen, and V. Kumar (2018). A Survey on Aerial Swarm Robotics. *IEEE Transactions on Robotics*, vol. 34, no. 4, pages: 837–855.

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

Dallej, T., M. Gouttefarde, N. Andreff, P.-E. Hervé, and P. Martinet (2019). Modeling and vision-based control of large-dimension cable-driven parallel robots using a multiple-camera setup. *Mechatronics*, vol. 61, pages: 20–36.

Danko, T. W., K. P. Chaney, and P. Y. Oh (2015). A Parallel Manipulator for Mobile Manipulating UAVs. In: *Proceedings of the 2015 IEEE Conference on Technologies for Practical Robot Applications (TePRA)*, vol. August, IEEE.

De Luca, A., A. Albu-Schäffer, S. Haddadin, and G. Hirzinger (2006). Collision detection and safe reaction with the DLR-III lightweight manipulator arm. In: *Proceedings of the 2006 IEEE International Conference on Intelligent Robots and Systems (IROS)*.

Delos, V. and D. Teissandier (2015). Minkowski Sum of Polytopes Defined by Their Vertices. *Journal of Applied Mathematics and Physics*, no. 3, pages: 62–67.

Deserno, M. (2004). *How to generate equidistributed points on the surface of a sphere*. Technical report. Carnegie Mellon University.

Diebel, J. (2006). *Representing attitude: Euler angles, unit quaternions, and rotation vectors*. Technical report. Stanford University, Stanford, CA, USA.

Dronecode (2021). *Pixhawk 4 Mini, PX4 User Guide*. URL: https://docs.px4.io/v1.12/en/flight_controller/pixhawk4_mini.html (updated on 02/19/2021).

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

eProsima (2022). *Fast DDS Documentation*. URL: https://fast-dds.docs.eprosima.com/en/latest/ (updated on 06/09/2022).

Erskine, J., A. Chriette, and S. Caro (2018). Wrench Capability Analysis of Aerial Cable Towed Systems. In: *Proceedings of the 2018 ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE)*, no. August.

Erskine, J., A. Chriette, and S. Caro (2019a). Control and Configuration Planning of an Aerial Cable Towed System. In: *Proceedings of the 2019 International Conference on Robotics and Automation (ICRA)*.

Erskine, J., A. Chriette, and S. Caro (2019b). Wrench Analysis of Cable-Suspended Parallel Robots Actuated by Quadrotors UAVs. *ASME Journal of Mechanisms and Robotics*, vol. 11, no. 2.

Falanga, D., K. Kleber, S. Mintchev, D. Floreano, and D. Scaramuzza (2019). The Foldable Drone: A Morphing Quadrotor that can Squeeze and Fly. *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pages: 209–216.

Featherstone, R. (2008). *Rigid Body Dynamics Algorithms*. Springer.

Feng, K., J. Li, X. Zhang, C. Shen, Y. Bi, T. Zheng, and J. Liu (2017). A New Quaternion-Based Kalman Filter for Real-Time Attitude Estimation Using the Two-Step Geometrically-Intuitive Correction Algorithm. *Sensors (Basel, Switzerland)*, vol. 17.

Fernbach, P., S. Tonneau, and M. Taix (2018). CROC: Convex Resolution of Centroidal Dynamics Trajectories to Provide a Feasibility Criterion for the Multi Contact Planning Problem. In: *Proceedings of the 2018 IEEE International Conference on Intelligent Robots and Systems (IROS)*.

Fink, J., N. Michael, S. Kim, and V. Kumar (2011). Planning and control for cooperative manipulation and transportation with aerial robots. *The International Journal of Robotics Research*, vol. 30, no. 3, pages: 324–334.

Flores, P. (2015). Euler Angles, Bryant Angles and Euler Parameters. In: *Concepts and Formulations for Spatial Multibody Dynamics*. Cham: Springer International Publishing.

Foehn, P., D. Falanga, N. Kuppuswamy, R. Tedrake, and D. Scaramuzza (2017). Fast Trajectory Optimization for Agile Quadrotor Maneuvers with a Cable-Suspended Payload. In: *Proceedings of the 2017 Robotics: Science and System (RSS)*, vol. 13.

Forte, F., R. Naldi, A. MacChelli, and L. Marconi (2012). Impedance Control of an Aerial Manipulator. In: *Proceedings of the 2012 American Control Conference (ACC)*, IEEE.

Forte, F., R. Naldi, A. Macchelli, and L. Marconi (2014). On the Control of an Aerial Manipulator Interacting with the Environment. In: *Proceedings of the 2014 IEEE International Conference on Robotics & Automation (ICRA)*, IEEE.

Franchi, A., R. Carli, D. Bicego, and M. Ryll (2018). Full-Pose Tracking Control for Aerial Robotic Systems With Laterally Bounded Input Force. *IEEE Transactions on Robotics*, vol. 34, no. 2, pages: 534–541.

Fraundorfer, F., L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, and M. Pollefeys (2012). Vision-based autonomous mapping and exploration using a quadrotor MAV. In: *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Fresk, E. and G. Nikolakopoulos (2013). Full quaternion based attitude control for a quadrotor. In: *Proceedings of the 2013 European Control Conference (ECC)*.

Fumagalli, M., R. Naldi, A. MacChelli, R. Carloni, S. Stramigioli, and L. Marconi (2012). Modeling and Control of a Flying Robot for Contact Inspection. In: *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE.

Fumagalli, M., S. Stramigioli, and R. Carloni (2016). Mechatronic Design of a Robotic Manipulator for Unmanned Aerial Vehicles. In: *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE.

Gabrich, B., D. Saldaña, V. Kumar, and M. Yim (2018). A Flying Gripper Based on Cuboid Modular Robots. In: *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*.

Gagliardini, L., S. Caro, M. Gouttefarde, and A. Girin (2016). Discrete reconfiguration planning for Cable-Driven Parallel Robots. *Mechanism and Machine Theory*, vol. 100, pages: 313–337.

Gagliardini, L., M. Gouttefarde, and S. Caro (2018). Determination of a Dynamic Feasible Workspace for Cable-Driven Parallel Robots. In: *Advances in Robot Kinematics 2016*. Springer International Publishing.

Garofalo, G., N. Mansfeld, J. Jankowski, and C. Ott (2019). Sliding mode momentum observers for estimation of external torques and joint acceleration. In: *Proceedings of the 2019 IEEE International Conference on Robotics and Automation (ICRA)*, vol. May.

Garrido-Jurado, S., R. Muñoz-Salinas, F. Madrid-Cuevas, and R. Medina-Carnicer (2016). Generation of fiducial marker dictionaries using Mixed Integer Linear Programming. *Pattern Recognition*, vol. 51, pages: 481–491.

Gassner, M., T. Cieslewski, and D. Scaramuzza (2017). Dynamic Collaboration without Communication: Vision-Based Cable-Suspended Load Transport with Two Quadrotors. In: *Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA)*.

Gentili, L., R. Naldi, and L. Marconi (2008). Modeling and control of VTOL UAVs interacting with the environment. In: *Proceedings of the 47th IEEE Conference on Decision and Control (CDC)*, IEEE.

Ghadiok, V., J. Goldin, and W. Ren (2011). Autonomous Indoor Aerial Gripping Using a Quadrotor. In: *Proceedings of the 2011 IEEE International Conference on Intelligent Robots and Systems (IROS)*, IEEE.

Gioioso, G., A. Franchi, G. Salvietti, S. Scheggi, and D. Prattichizzo (2014). The flying hand: A formation of UAVs for cooperative aerial tele-manipulation. In: *Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA)*.

Goor, P. van, T. Hamel, and R. Mahony (2022). Equivariant Filter (EqF). *IEEE Transactions on Automatic Control*, pages: 1–13.

Gouttefarde, M., J.-P. Merlet, and D. Daney (2007). Wrench-Feasible Workspace of Parallel Cable-Driven Mechanisms. In: *Proceedings of the 2007 IEEE International Conference on Robotics and Automation (ICRA)*.

Gouttefarde, M., D. Daney, and J. P. Merlet (2011). Interval-analysis-based determination of the wrench-feasible workspace of parallel cable-driven robots. *IEEE Transactions on Robotics*, vol. 27, no. 1, pages: 1–13.

Grünbaum, B. (2003). *Convex Polytopes*. Springer New York, NY.

Guerrero-Castellanos, J. F., A. Vega-Alonzo, S. Durand, N. Marchand, V. R. Gonzalez-Diaz, J. Castañeda-Camacho, and W. F. Guerrero-Sánchez (2019). Leader-Following Consensus and Formation Control of VTOL-UAVs with Event-Triggered Communications. *Sensors*, vol. 19, no. 24.

Hamandi, M., F. Usai, Q. Sablé, N. Staub, M. Tognon, and A. Franchi (2021). Design of Multirotor Aerial Vehicles: A Taxonomy Based on Input Allocation. *The International Journal of Robotics Research*, vol. 40, no. 8-9, pages: 1015–1044.

Hamaza, S., I. Georgilas, and T. Richardson (2018). An Adaptive-Compliance Manipulator for Contact-Based Aerial Applications. In: *Proceedings of the 2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, vol. July.

Herceg, M., M. Kvasnica, C. N. Jones, and M. Morari (2013). Multi-Parametric Toolbox 3.0. In: *Proceedings of the 2013 European Control Conference (ECC)*.

Hou, Z. and I. Fantoni (2018). Interactive Leader–Follower Consensus of Multiple Quadrotors Based on Composite Nonlinear Feedback Control. *IEEE Transactions on Control Systems Technology*, vol. 26, no. 5, pages: 1732–1743.

Huber, F., K. Kondak, K. Krieger, D. Sommer, M. Schwarzbach, M. Laiacker, I. Kossyk, S. Parusel, S. Haddadin, and A. Albu-Schaffer (2013). First Analysis and Experiments in Aerial Manipulation Using Fully Actuated Redundant Robot Arm. In: *Proceedings of the 2013 IEEE International Conference on Intelligent Robots and Systems (IROS)*.

Islam, S., P. X. Liu, and A. El Saddik (2015). Robust Control of Four-Rotor Unmanned Aerial Vehicle With Disturbance Uncertainty. *IEEE Transactions on Industrial Electronics*, vol. 62, no. 3, pages: 1563–1571.

Jamshidifar, H. and A. Khajepour (2020). Static Workspace Optimization of Aerial Cable Towed Robots With Land-Fixed Winches. *IEEE Transactions on Robotics*, vol. 36, no. 5, pages: 1603–1610.

Jiang, Q. and V. Kumar (2013). The Inverse Kinematics of Cooperative Transport With Multiple Aerial Robots. *IEEE Transactions on Robotics*, vol. 29, no. 1, pages: 136–145.

Jimenez-Cano, A. E., J. Martin, G. Heredia, A. Ollero, and R. Cano (2013). Control of an aerial robot with multi-link arm for assembly tasks. In: *Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE.

Kamel, M., K. Alexis, and R. Siegwart (2016). Design and modeling of dexterous aerial manipulator. In: *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, IEEE.

Kamel, B., B. Yasmina, B. Laredj, I. Benaoumeur, and A.-F. Zoubir (2017). Dynamic modeling, simulation and PID controller of unmanned aerial vehicle UAV. In: *Proceedings of the 2017 Seventh International Conference on Innovative Computing Technology (INTECH)*.

Kamel, M., S. Verling, O. Elkhatib, C. Sprecher, P. Wulkop, Z. Taylor, R. Siegwart, and I. Gilitschenski (2018). The Voliro Omniorientational Hexacopter: An Agile and Maneuverable Tiltable-Rotor Aerial Vehicle. *IEEE Robotics and Automation Magazine*, vol. 25, no. 4, pages: 34–44.

Keemink, A. Q., M. Fumagalli, S. Stramigioli, and R. Carloni (2012). Mechanical Design of a Manipulation System for Unmanned Aerial Vehicles. In: *Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE.

Kermorgant, O. (2022). *Modeling and Control of Robot Manipulators*.

Khalil, H. (2002). *Nonlinear Systems*. Pearson Education. Prentice Hall.

Khalil, W. and E. Dombre (2002). *Modeling, Identification and Control of Robots*. Butterworth-Heinemann.

Khalil, W. (2010). Dynamic modeling of robots using recursive Newton-Euler techniques. In: *Proceedings of the 7th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, vol. 1.

Kim, S., S. Choi, and H. J. Kim (2013). Aerial Manipulation Using a Quadrotor with a Two DOF Robotic Arm. In: *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE.

Kim, S., H. Seo, S. Choi, and H. J. Kim (2016). Vision-Guided Aerial Manipulation Using a Multirotor with a Robotic Arm. vol. 21, no. 4, pages: 1912–1923.

Kim, H., H. Lee, S. Choi, Y. K. Noh, and H. J. Kim (2017). Motion planning with movement primitives for cooperative aerial transportation in obstacle environment. In: *Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA)*.

Kim, H., H. Seo, C. Y. Son, H. Lee, S. Kim, and H. J. Kim (2018a). Cooperation in the Air: A Learning-Based Approach for the Efficient Motion Planning of Aerial Manipulators. *IEEE Robotics and Automation Magazine*, vol. 25, no. 4, pages: 76–85.

Kim, S., H. Seo, J. Shin, and H. J. Kim (2018b). Cooperative Aerial Manipulation Using Multirotors With Multi-DOF Robotic Arms. *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 2, pages: 702–713.

Kim, C., H. Lee, M. Jeong, and H. Myung (2021). A Morphing Quadrotor that Can Optimize Morphology for Transportation. In: *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

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

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

Kotarski, D. and J. Kasać (2018). Generalized Control Allocation Scheme for Multirotor Type of UAVs. In: *Drones*. Rijeka: IntechOpen.

Krug, R., A. J. Lilienthal, D. Kragic, and Y. Bekiroglu (2016). Analytic grasp success prediction with tactile feedback. In: *Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE.

Laiacker, M., F. Huber, and K. Kondak (2016). High Accuracy Visual Servoing for Aerial Manipulation using a 7 Degrees of Freedom Industrial Manipulator. In: *Proceedings of the 2016 IEEE International Conference on Intelligent Robots and Systems (IROS)*, vol. October, IEEE.

Lee, D., A. Franchi, P. R. Giordano, H. I. Son, and H. H. Bülthoff (2011). Haptic teleoperation of multiple unmanned aerial vehicles over the internet. In: *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*.

Lee, H., H. Kim, and H. J. Kim (2015). Path planning and control of multiple aerial manipulators for a cooperative transportation. In: *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Lee, D., H. Seo, D. Kim, and H. J. Kim (2020). Aerial Manipulation using Model Predictive Control for Opening a Hinged Door. In: *Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA)*, no. March.

Li, F. and L. Chang (2020). MEKF With Navigation Frame Attitude Error Parameterization for INS/GPS. *IEEE Sensors Journal*, vol. 20, no. 3, pages: 1536–1549.

Li, Z., J. Erskine, S. Caro, and A. Chriette (2020). Design and Control of a Variable Aerial Cable Towed System. *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pages: 636–643.

Li, G., R. Ge, and G. Loianno (2021). Cooperative Transportation of Cable Suspended Payloads With MAVs Using Monocular Vision and Inertial Sensing. *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pages: 5316–5323.

Li, Z. (2021). *Theoretical developments and experimental evaluation of a novel collaborative multi-drones grasping and manipulation system of large objects*. PhD thesis. Ecole Centrale de Nantes, France.

Li, Z., X. Song, V. Bégoc, A. Chriette, and I. Fantoni (2021). Dynamic Modeling and Controller Design of a Novel Aerial Grasping Robot. In: *23rd CISM IFToMM Symposium on Robot Design, Dynamics and Control (RoManSy 2020)*. vol. 601. Springer.

Li, Z., V. Bégoc, A. Chriette, and I. Fantoni (June 2022). Wrench Capability Analysis and Control Allocation of a Collaborative Multi-Drone Grasping Robot. *Journal of Mechanisms and Robotics*, vol. 15, no. 2.

Lippiello, V., J. Cacace, A. Santamaria-Navarro, J. Andrade-Cetto, M. Á. Trujillo, Y. R. Esteves, and A. Viguria (2016). Hybrid Visual Servoing With Hierarchical Task Composition for Aerial Manipulation. *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pages: 259–266.

Liu, H., D. Derawi, J. Kim, and Y. Zhong (2013). Robust optimal attitude control of hexarotor robotic vehicles. *Nonlinear Dynamics*, vol. 74, no. 4, pages: 1155–1168.

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

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

Lofberg, J. (2004). YALMIP : a toolbox for modeling and optimization in MATLAB. In: *Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA)*.

Loianno, G. and V. Kumar (2018). Cooperative Transportation Using Small Quadrotors Using Monocular Vision and Inertial Sensing. *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pages: 680–687.

Loianno, G., V. Spurny, J. Thomas, T. Baca, D. Thakur, D. Hert, R. Penicka, T. Krajnik, A. Zhou, A. Cho, M. Saska, and V. Kumar (2018). Localization, Grasping, and

Transportation of Magnetic Objects by a Team of MAVs in Challenging Desert-Like Environments. *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pages: 1576–1583.

Loloei, A. Z., M. M. Aref, and H. D. Taghirad (2009). Wrench feasible workspace analysis of cable-driven parallel manipulators using LMI approach. In: *Proceedings of the 2009 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, no. May.

Lunni, D., A. Santamaria-Navarro, R. Rossi, P. Rocco, L. Bascetta, and J. Andrade-Cetto (2017). Nonlinear Model Predictive Control for Aerial Manipulation. In: *Proceedings of the 2017 International Conference on Unmanned Aircraft Systems (ICUAS)*.

Mahony, R., V. Kumar, and P. Corke (2012). Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robotics and Automation Magazine*, vol. 19, no. 3, pages: 20–32.

Manubens, M., D. Devaurs, L. Ros, and J. Cortés (2013). Motion planning for 6-D manipulation with aerial towed-cable systems. In: *Proceedings of the 2013 Robotics: Science and Systems (RSS)*.

Marconi, L., R. Naldi, and L. Gentili (2011). Modelling and control of a flying robot interacting with the environment. *Automatica*, vol. 47, no. 12, pages: 2571–2583.

Marconi, L. and R. Naldi (2012). Control of Aerial Robots: Hybrid Force and Position Feedback for a Ducted Fan. *IEEE Control Systems Magazine*, vol. 32, no. 4, pages: 43–65.

Markley, F. L. (2003). Attitude Error Representations for Kalman Filtering. *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 2, pages: 311–317.

Matoušek, J. and G. Bernd (2007). *Understanding and Using Linear Programming*. Springer Berlin, Heidelberg.

Mayhew, C. G., R. G. Sanfelice, and A. R. Teel (2011). On quaternion-based attitude control and the unwinding phenomenon. In: *Proceedings of the 2011 American Control Conference (ACC)*, no. 3.

Mellinger, D. and V. Kumar (2011). Minimum Snap Trajectory Generation and Control for Quadrotors. In: *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA)*.

Mellinger, D., Q. Lindsey, M. Shomin, and V. Kumar (2011). Design, Modeling, Estimation and Control for Aerial Grasping and Manipulation. In: *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE.

Mellinger, D., M. Shomin, N. Michael, and V. Kumar (2013). Cooperative Grasping and Transport using Multiple Quadrotors. In: *Distributed Autonomous Robotic Systems*. Springer, Berlin, Heidelberg.

Michael, N., J. Fink, and V. Kumar (Sept. 2011). Cooperative manipulation and transportation with aerial robots. *Autonomous Robots*, vol. 30, no. 1, pages: 73–86.

Miller, A. T. and P. K. Allen (2004). Grasp it! A Versatile Simulator for Robotic Grasping. *IEEE Robotics and Automation Magazine*, no. December.

Mohammadi, M., A. Franchi, D. Barcelli, and D. Prattichizzo (2016). Cooperative Aerial Tele-Manipulation with Haptic Feedback. In: *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Mohiuddin, A., Y. Zweiri, R. Almadhoun, T. Taha, and D. Gan (2020). Energy distribution in Dual-UAV collaborative transportation through load sharing. *Journal of Mechanisms and Robotics*, no. April, pages: 1–14.

Moreno, J. A. and M. Osorio (2008). A Lyapunov approach to second-order sliding mode controllers and observers. In: *Proceedings of the 2008 IEEE Conference on Decision and Control*, no. Dec.

Morton, K. and L. F. G. Toro (2016). Development of a Robust Framework for an Outdoor Mobile Manipulation UAV. In: *Proceedings of the 2016 IEEE Aerospace Conference*, IEEE.

Naldi, R., A. Macchelli, N. Mimmo, and L. Marconi (2018). Robust Control of an Aerial Manipulator Interacting with the Environment. *IFAC-PapersOnLine*, vol. 51, no. 13, pages: 537–542.

Nava, G., Q. Sablé, M. Tognon, D. Pucci, and A. Franchi (2020). Direct Force Feedback Control and Online Multi-Task Optimization for Aerial Manipulators. *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pages: 331–338.

Nayak, V., C. Papachristos, and K. Alexis (2018). Design and Control of an Aerial Manipulator for Contact-based Inspection. *ArXiv*, pages: 1–5.

Nguyen, H. N. and D. Lee (2013). Hybrid force/motion control and internal dynamics of quadrotors for tool operation. In: *Proceedings of the 2013 IEEE International Conference on Intelligent Robots and Systems (IROS)*, no. November.

Nguyen, H.-N., S. Park, and D. Lee (2015). Aerial tool operation system using quadrotors as Rotating Thrust Generators. In: *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Nguyen, H. N., S. Park, J. Park, and D. Lee (2018). A Novel Robotic Platform for Aerial Manipulation Using Quadrotors as Rotating Thrust Generators. *IEEE Transactions on Robotics*, vol. 34, no. 2, pages: 353–369.

Ollero, A., 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 (2018). The AEROARMS Project: Aerial Robots with Advanced Manipulation Capabilities for Inspection and Maintenance. *IEEE Robotics Automation Magazine*, vol. 25, no. 4, pages: 12–23.

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

Omari, S., M.-D. Hua, G. Ducard, and T. Hamel (2013). Bilateral Haptic Teleoperation of VTOL UAVs. In: *Proceedings of the 2013 IEEE International Conference on Robotics and Automation*.

Open-Robotics (2022). *ROS2 Documentation - ament_cmake user documentation*. URL: https://docs.ros.org/en/galactic/.

OpenCV (2022). *Detection of Diamond Markers*. URL: https://docs.opencv.org/4.6.0/d5/d07/tutorial_charuco_diamond_detection.html (updated on 01/05/2022).

Orsag, M., C. Korpela, S. Bogdan, and P. Oh (2014). Valve Turning using a Dual-Arm Aerial Manipulator. In: *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, IEEE.

Orsolino, R., M. Focchi, C. Mastalli, H. Dai, D. G. Caldwell, and C. Semini (2018). Application of wrench-based feasibility analysis to the online trajectory optimization of legged robots. *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pages: 3363–3370.

Orsolino, R., M. Focchi, S. Caron, G. Raiola, V. Barasuol, D. G. Caldwell, and C. Semini (2020). Feasible Region: An Actuation-Aware Extension of the Support Region. *IEEE Transactions on Robotics*, vol. 36, no. 4, pages: 1239–1255.

Palunko, I., P. Cruz, and R. Fierro (2012). Agile Load Transportation : Safe and Efficient Load Manipulation with Aerial Robots. *IEEE Robotics and Automation Magazine*, vol. 19, no. 3, pages: 69–79.

Palunko, I., A. Faust, P. Cruz, L. Tapia, and R. Fierro (2013). A Reinforcement Learning Approach Towards Autonomous Suspended Load Manipulation Using Aerial Robots. In: *Proceedings of the 2013 EEE International Conference on Robotics and Automation (ICRA)*, IEEE.

Park, S., J. Lee, J. Ahn, M. Kim, J. Her, G. Yang, and D. Lee (2018). ODAR: Aerial Manipulation Platform Enabling Omnidirectional Wrench Generation. *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 4, pages: 1907–1918.

Park, S., Y. Lee, J. Heo, and D. Lee (2019). Pose and Posture Estimation of Aerial Skeleton Systems for Outdoor Flying. In: *Proceedings of the 2019 International Conference on Robotics and Automation (ICRA)*.

Paul, H., R. Miyazaki, R. Ladig, and K. Shimonomura (2019). Landing of a Multirotor Aerial Vehicle on an Uneven Surface Using Multiple On-board Manipulators. In: *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*.

Perozo, M. A., J. Dussine, A. Yiğit, L. Cuvillon, S. Durand, and J. Gangloff (2022). Optimal Design and Control of an Aerial Manipulator with Elastic Suspension Using Unidirectional Thrusters. In: *Proceedings of the 2022 International Conference on Robotics and Automation (ICRA)*.

Pounds, P. E., D. R. Bersak, and A. M. Dollar (2011). Grasping from the air: Hovering capture and load stability. In: *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE.

Pounds, P. E., D. R. Bersak, and A. M. Dollar (2012). Stability of small-scale UAV helicopters and quadrotors with added payload mass under PID control. *Autonomous Robots*, vol. 33, no. 1-2, pages: 129–142.

Pounds, P. E. and A. M. Dollar (2014). Aerial Grasping from a Helicopter UAV Platform. In: *Experimental Robotics: The 12th International Symposium on Experimental Robotics*. Springer Berlin Heidelberg.

Qu, H., Y. Fang, and S. Guo (2015). Theory of degrees of freedom for parallel mechanisms with three spherical joints and its applications. *Chinese Journal of Mechanical Engineering*, vol. 28, pages: 737–746.

Qualisys (2022). *Qualisys Motion Capture Systems*. URL: https://www.qualisys.com/.

Rad, M. and V. Lepetit (2017). BB8: A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects without Using Depth. In: *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy.

Rajappa, S., M. Ryll, H. H. Bulthoff, and A. Franchi (2015). Modeling, Control and Design Optimization for a Fully-Actuated Hexarotor Aerial Vehicle with Tilted Propellers. In: *Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA)*.

Rashad, R., J. B. C. Engelen, and S. Stramigioli (2019). Energy Tank-Based Wrench/Impedance Control of a Fully-Actuated Hexarotor: A Geometric Port-Hamiltonian Approach. In: *Proceedings of the 2019 International Conference on Robotics and Automation (ICRA)*.

Riechel, A. T. and I. Ebert-Uphoff (2004). Force-Feasible Workspace analysis for underconstrained, point-mass cable robots. In: *Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2004, no. 5.

Romano, M., Y. Chen, O. Marshall, and E. Atkins (2021). Nailed It: Autonomous Roofing with a Nailgun-Equipped Octocopter. In: *Proceedings of the 2021 AIAA Aviation Forum*.

Romero-Ramirez, F. J., R. Muñoz-Salinas, and R. Medina-Carnicer (2018). Speeded up detection of squared fiducial markers. *Image and Vision Computing*, vol. 76, pages: 38–47.

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

Rossi, R., A. Santamaria-Navarro, J. Andrade-Cetto, and P. Rocco (2017). Trajectory Generation for Unmanned Aerial Manipulators Through Quadratic Programming. vol. 2, no. 2, pages: 389–396.

Ruggiero, F., J. Cacace, H. Sadeghian, and V. Lippiello (2014). Impedance control of VToL UAVs with a momentum-based external generalized forces estimator. In: *Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA)*.

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

Ruggiero, F., J. Cacace, H. Sadeghian, and V. Lippiello (2015b). Passivity-based control of VToL UAVs with a momentum-based estimator of external wrench and unmodeled dynamics. *Robotics and Autonomous Systems*, vol. 72, no. May, pages: 139–151.

Ruggiero, F., V. Lippiello, and A. Ollero (2018). Aerial manipulation: A literature review. *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pages: 1957–1964.

Ryll, M., D. Bicego, and A. Franchi (2016). Modeling and Control of FAST-Hex: a Fully–Actuated by Synchronized–Tilting Hexarotor. In: *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Ryll, M., G. Muscio, F. Pierri, E. Cataldi, G. Antonelli, F. Caccavale, and A. Franchi (2017). 6D Physical Interaction with a Fully Actuated Aerial Robot. In: *Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA)*.

Ryll, M., G. Muscio, F. Pierri, E. Cataldi, G. Antonelli, F. Caccavale, D. Bicego, and A. Franchi (2019). 6D Interaction Control with Aerial Robots: The Flying End-Effector Paradigm. *International Journal of Robotics Research*, vol. 38, no. 9, pages: 1045–1062.

Sabatini, A. (2006). Quaternion-Based Extended Kalman Filter for Determining Orientation by Inertial and Magnetic Sensing. *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 7, pages: 1346–1356.

Saint-Sevin, M., V. Bégoc, S. Briot, A. Chriette, and I. Fantoni (2018). Design and Optimization of a Multi-drone Robot for Grasping and Manipulation of Large Size Objects. In: *22nd CISM IFToMM Symposium on Robot Design, Dynamics and Control (RoManSy 2018)*. vol. 584. Springer.

Sanalitro, D., H. J. Savino, M. Tognon, J. Cortés, and A. Franchi (2020). Full-Pose Manipulation Control of a Cable-Suspended Load with Multiple UAVs under Uncertainties. *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pages: 2185–2191.

Schiano, F., A. Franchi, D. Zelazo, and P. R. Giordano (2016). A rigidity-based decentralized bearing formation controller for groups of quadrotor UAVs. In: *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, vol. 2016-November.

Scholten, J. L., M. Fumagalli, S. Stramigioli, and R. Carloni (2013). Interaction Control of an UAV Endowed with a Manipulator. In: *Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE.

Schwab, A. L. (2002). *Quaternions, Finite Rotation and Euler Parameters*. Technical report. Delft University of Technology, Delft, Netherlands.

Seraji, H. and R. Colbaugh (1997). Force tracking in impedance control. *International Journal of Robotics Research*, vol. 16, no. 1, pages: 97–117.

Serra, P., R. Cunha, T. Hamel, D. Cabecinhas, and C. Silvestre (2016). Landing of a Quadrotor on a Moving Target Using Dynamic Image-Based Visual Servo Control. *IEEE Transactions on Robotics*, vol. 32, no. 6, pages: 1524–1535.

Shinners, S. M. (1998). *Modern control system theory and design*. John Wiley & Sons.

Shule, W., C. M. Almansa, J. P. Queralta, Z. Zou, and T. Westerlund (2020). UWB-Based Localization for Multi-UAV Systems and Collaborative Heterogeneous Multi-Robot Systems. *Procedia Computer Science*, vol. 175, pages: 357–364.

Siciliano, B., L. Sciavicco, L. Villani, and G. Oriolo (2010). *Robotics: Modelling, Planning, and Control*. Springer Science & Business Media.

Six, D., S. Briot, A. Chriette, and P. Martinet (Aug. 2017a). A Controller Avoiding Dynamic Model Degeneracy of Parallel Robots During Singularity Crossing. *Journal of Mechanisms and Robotics*, vol. 9, no. 5.

Six, D., A. Chriette, S. Briot, and P. Martinet (2017b). Dynamic Modeling and Trajectory Tracking Controller of a Novel Flying Parallel Robot. *IFAC-PapersOnLine*, vol. 50, no. 1, pages: 2241–2246. 20th IFAC World Congress.

Six, D. (2018). *Conception et commande de robots parallèles volants*. PhD thesis. Ecole Centrale de Nantes, France.

Six, D., S. Briot, A. Chriette, and P. Martinet (2018). The Kinematics, Dynamics and Control of a Flying Parallel Robot with Three Quadrotors. *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pages: 559–566.

Slabaugh, G. G. (2020). *Computing Euler angles from a rotation matrix*.

Smrcka, D., T. Baca, T. Nascimento, and M. Saska (2021). Admittance Force-Based UAV-Wall Stabilization and Press Exertion for Documentation and Inspection of Historical Buildings. In: *Proceedings of the 2021 International Conference on Unmanned Aircraft Systems (ICUAS)*.

Sreenath, K. and V. Kumar (2013). Dynamics, Control and Planning for Cooperative Manipulation of Payloads Suspended by Cables from Multiple Quadrotor Robots. In: *Proceedings of the 2013 Robotics: Science and Systems (RSS)*.

Sreenath, K., N. Michael, and V. Kumar (2013). Trajectory Generation and Control of a Quadrotor with a Cable-Suspended Load - A Differentially-Flat Hybrid System. In:

Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA), IEEE.

Staub, N., D. Bicego, Q. Sable, V. Arellano, S. Mishra, and A. Franchi (2018). Towards a Flying Assistant Paradigm: The OTHex. In: *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*.

Su, Y., P. Yu, M. J. Gerber, L. Ruan, and T.-C. Tsao (2021). Nullspace-Based Control Allocation of Overactuated UAV Platforms. *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pages: 8094–8101.

Suarez, A., A. E. Jimenez-Cano, V. M. Vega, G. Heredia, A. Rodriguez-Castano, and A. Ollero (2017a). Lightweight and human-size dual arm aerial manipulator. In: *Proceedings of the 2017 International Conference on Unmanned Aircraft Systems (ICUAS)*.

Suarez, A., P. R. Soria, G. Heredia, B. Arrue, and A. Ollero (2017b). Anthropomorphic, Compliant, and Lightweight Dual Arm for Aerial Manipulation. In: *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Suarez, A., A. E. Jimenez-Cano, V. M. Vega, G. Heredia, A. Rodriguez-Castaño, and A. Ollero (2018). Design of a lightweight dual arm system for aerial manipulation. vol. 50, no. April 2017, pages: 30–44.

Sugito, N., M. Zhao, T. Anzai, T. Nishio, K. Okada, and M. Inaba (2022). Aerial Manipulation Using Contact with the Environment by Thrust Vectorable Multilinked Aerial Robot. In: *Proceedings of the 2022 International Conference on Robotics and Automation (ICRA)*.

Szasz, R., M. Allenspach, M. Han, M. Tognon, and R. K. Katzschmann (2022). Modeling and Control of an Omnidirectional Micro Aerial Vehicle Equipped with a Soft Robotic Arm. In: *Proceedings of the 2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)*.

Tagliabue, A., M. Kamel, S. Verling, R. Siegwart, and J. Nieto (2017). Collaborative transportation using MAVs via passive force control. In: *Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA)*.

Tagliabue, A., M. Kamel, R. Siegwart, and J. Nieto (2019). Robust collaborative object transportation using multiple MAVs. *The International Journal of Robotics Research*, vol. 38, no. 9, pages: 1020–1044.

Tekin, B., S. N. Sinha, and P. Fua (2017). Real-Time Seamless Single Shot 6D Object Pose Prediction. *CoRR*.

Thapa, S., H. Bai, and J. Acosta (2020). Cooperative Aerial Manipulation with Decentralized Adaptive Force-Consensus Control. *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 97, no. 1, pages: 171–183.

Thomas, J., J. Polin, K. Sreenath, and V. Kumar (2013). Avian-inspired grasping for quadrotor Micro UAVs. In: *Proceedings of the ASME 2013 International Design Engi-*

neering Technical Conference & Computers and Information in Engineering Conference (IDETC/CIE), vol. 6A, no. August.

Tognon, M., S. S. Dash, and A. Franchi (2016). Observer-Based Control of Position and Tension for an Aerial Robot Tethered to a Moving Platform. *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pages: 732–737.

Tognon, M., B. Yuksel, G. Buondonno, and A. Franchi (2017). Dynamic Decentralized Control for Protocentric Aerial Manipulators. In: *Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA)*.

Tognon, M. and A. Franchi (2018). Omnidirectional Aerial Vehicles With Unidirectional Thrusters: Theory, Optimal Design, and Control. *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pages: 2277–2282.

Tognon, M., C. Gabellieri, L. Pallottino, and A. Franchi (2018). Aerial Co-Manipulation With Cables: The Role of Internal Force for Equilibria, Stability, and Passivity. *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pages: 2577–2583.

Tognon, M., H. T. Chávez, E. Gasparin, Q. Sablé, D. Bicego, A. Mallet, M. Lany, G. Santi, B. Revaz, J. Cort, and A. Franchi (2019). A Truly Redundant Aerial Manipulator System with Application to Push-and-Slide Inspection in Industrial Plants. *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pages: 1846–1851.

Tomić, T., C. Ott, and S. Haddadin (2017). External Wrench Estimation, Collision Detection, and Reflex Reaction for Flying Robots. *IEEE Transactions on Robotics*, vol. 33, no. 6, pages: 1467–1482.

Torre, A., D. Mengoli, R. Naldi, F. Forte, A. MacChelli, and L. Marconi (2012). A Prototype of Aerial Manipulator. In: *Proceedings of the 2012 IEEE International Conference on Intelligent Robots and Systems (IROS)*, IEEE.

Trujillo, M. Á., J. R. Martínez-De Dios, C. Martín, A. Viguria, and A. Ollero (2019). Novel Aerial Manipulator for Accurate and Robust Industrial NDT Contact Inspection: A New Tool for the Oil and Gas Inspection Industry. *Sensors*, vol. 19, no. 6, pages: 1–24.

Tsukagoshi, H., M. Watanabe, T. Hamada, D. Ashlih, and R. Iizuka (2015). Aerial Manipulator with perching and Door-opening Capability. In: *Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA)*, vol. May, IEEE.

Turpin, M., N. Michael, and V. Kumar (2012). Decentralized Formation Control with Variable Shapes for Aerial Robots. In: *Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA)*.

Tzoumanikas, D., F. Graule, Q. Yan, D. Shah, M. Popovic, and S. Leutenegger (2020). Aerial Manipulation Using Hybrid Force and Position NMPC Applied to Aerial Writing. In: *Proceedings of the 2020 Robotics: Science and Systems (RSS)*.

Vásárhelyi, G., C. Virágh, G. Somorjai, N. Tarcai, T. Szörenyi, T. Nepusz, and T. Vicsek (2014). Outdoor flocking and formation flight with autonomous aerial robots. In: *Pro-*

ceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems.

Vempati, A. S., M. Kamel, N. Stilinovic, Q. Zhang, D. Reusser, I. Sa, J. Nieto, R. Siegwart, and P. Beardsley (2018). PaintCopter: An Autonomous UAV for Spray Painting on Three-Dimensional Surfaces. *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pages: 2862–2869.

Welch, G. and G. Bishop (1995). *An Introduction to the Kalman Filter*. Technical report. University of North Carolina, Chapel Hill, NC, USA.

Werink, R. (2019). *On the Control Allocation of Fully-Actuated and Over-Actuated Multirotor UAVs*. Technical report. University of Twente, Enschede, Netherlands.

Wopereis, H. W., J. J. Hoekstra, T. H. Post, G. A. Folkertsma, S. Stramigioli, and M. Fumagalli (2017). Application of Substantial and Sustained Force to Vertical Surfaces using a Quadrotor. In: *Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA)*.

Yang, H. and D. Lee (2015). Hierarchical cooperative control framework of multiple quadrotor-manipulator systems. In: *Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA)*.

Yang, H., S. Park, J. Lee, J. Ahn, D. Son, and D. Lee (2018). LASDRA: Large-Size Aerial Skeleton System with Distributed Rotor Actuation. In: *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*.

Yiğit, A., M. A. Perozo, L. Cuvillon, S. Durand, and J. Gangloff (2021a). Improving Dynamics of an Aerial Manipulator with Elastic Suspension Using Nonlinear Model Predictive Control. In: *Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA)*.

Yiğit, A., M. A. Perozo, L. Cuvillon, S. Durand, and J. Gangloff (2021b). Novel Omnidirectional Aerial Manipulator With Elastic Suspension: Dynamic Control and Experimental Performance Assessment. *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pages: 612–619.

Yiğit, A., M. A. Perozo, M. Ouafo, L. Cuvillon, S. Durand, and J. Gangloff (2021c). Aerial Manipulator Suspended from a Cable-Driven Parallel Robot: Preliminary Experimental Results. In: *Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Yu, P., Y. Su, M. J. Gerber, L. Ruan, and T.-C. Tsao (2021). An Over-Actuated Multi-Rotor Aerial Vehicle With Unconstrained Attitude Angles and High Thrust Efficiencies. *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pages: 6828–6835.

Yüksel, B., N. Staub, and A. Franchi (2016). Aerial Robots with Rigid/Elastic-joint Arms: Single-joint Controllability Study and Preliminary Experiments. In: *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Yüksel, B., C. Secchi, H. H. Bülthoff, and A. Franchi (2019). Aerial physical interaction via IDA-PBC. *International Journal of Robotics Research*, vol. 38, no. 4, pages: 403–421.

Zhang, G., Y. He, B. Dai, F. Gu, L. Yang, J. Han, and G. Liu (2019). Aerial grasping of an object in the strong wind: Robust control of an aerial manipulator. *Applied Sciences*, vol. 9, no. 11.

Zhang, D., G. Yang, and R. P. Khurshid (2020). Haptic Teleoperation of UAVs Through Control Barrier Functions. *IEEE Transactions on Haptics*, vol. 13, no. 1, pages: 109–115.

Zhao, S. (2016). Time Derivative of Rotation Matrices: A Tutorial. *ArXiv*.

Zhao, M., K. Kawasaki, X. Chen, S. Noda, K. Okada, and M. Inaba (2017). Whole-body Aerial Manipulation by Transformable Multirotor with Two-dimensional Multilinks. In: *Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA)*.

Zhao, M., F. Shi, T. Anzai, K. Chaudhary, X. Chen, K. Okada, and M. Inaba (2018). Flight Motion of Passing Through Small Opening by DRAGON: Transformable Multilinked Aerial Robot. In: *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Zhao, M., F. Shi, T. Anzai, K. Okada, and M. Inaba (2020). Online Motion Planning for Deforming Maneuvering and Manipulation by Multilinked Aerial Robot Based on Differential Kinematics. *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pages: 1602–1609.

Zhao, M., K. Nagato, K. Okada, M. Inaba, and M. Nakao (2022). Forceful Valve Manipulation With Arbitrary Direction by Articulated Aerial Robot Equipped With Thrust Vectoring Apparatus. *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pages: 4893–4900.

Zhou, S., G. Flores, E. Bazan, R. Lozano, and A. Rodriguez (2015). Real-time object detection and pose estimation using stereo vision. An application for a Quadrotor MAV. In: *Proceedings of the 2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*.

Zhu, M., S. Briot, and A. Chriette (2022). Sensor-based design of a Delta parallel robot. *Mechatronics*, vol. 87, pages: 102893.

Titre : Commande décentralisée et estimation d'un robot parallèle volant en interaction avec l'environnement

Mots clés : Systèmes Aériens: Mécanique et Contrôle; Robot Parallèle; Système Multi-Robots; Estimation basée sur la Vision; Commande Décentralisée; Manipulation Aérienne.

Résumé : La manipulation aérienne est un nouveau domaine où les drones multirotors sont équipés d'effecteurs pour la saisie, le transport et la manipulation des objets. Afin d'améliorer la capacité de charge et la manipulabilité dans l'espace 3D, un robot parallèle volant (FPR) a été proposé, dans lequel un certain nombre de drones sont utilisés pour soutenir collaborativement une architecture parallèle et passive.

Dans cette thèse, les méthodes d'estimation et de contrôle considérant l'interaction avec l'environnement sont adressées et appliquées à un FPR spécifique composé d'une plateforme mobile et des jambes rigides attachées à des quadrirotors qui actionnent le système.

La deuxième contribution de cette thèse est la proposition de stratégies décentralisées basées sur des mesures embarquées et intrinsèques des drones. Les expériences montrent l'efficacité des méthodes proposées pour la régulation de la configuration du robot, la réalisation des tâches de positionnement précis par téléopération et des interactions basées sur le contact. En outre, une analyse détaillée sur les torseurs admissibles du FPR est présentée, qui permet de calculer l'espace de torseur admissible (AWS) de la plateforme et de déterminer les configurations optimales des jambes maximisant la faisabilité de torseurs du robot en fonction des exigences spécifiques de la tâche.

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

Keywords : Aerial Systems: Mechanics and Control; Parallel Robot; Multi-Robot Systems; Vision-based Estimation; Decentralized Control; Aerial Manipulation.

Abstract : Aerial manipulation is an emerging domain where multirotor Unmanned Aerial Vehicles (UAVs) are equipped with onboard end-effectors for grasping, transporting and manipulating objects. To enhance the payload capacity and achieve full manipulability in 3-dimensional space, a flying parallel robot (FPR) was previously proposed in which a number of UAVs are used to collectively support a passive parallel architecture.

In this thesis, the estimation and control methods dealing with the interaction with the environment are addressed, which are applied to a specific FPR composed of a moving platform and a number of rigid legs attached with quadrotors actuating the system.

The second main contribution of this thesis is the proposal of decentralized strategies based on onboard and intrinsic measurements of the UAVs. Experiments show the effectiveness of the proposed methods in regulating the robot configuration, achieving precise positioning tasks through teleoperation and performing contact-based interactions with an object in the environment. In addition, a detailed analysis of the wrench feasibility of the FPR is presented, allowing to compute the Available Wrench Set (AWS) of the platform and determine the optimal leg configurations maximising the wrench feasibility of the robot accordingly to specific task requirements.