Università degli Studi di Padova

Department of Information Engineering

Ph.D. Course in Information Engineering

Heterogeneous multi-agent interaction management: from theory to practice

Beniamino Pozzan

Advisor Prof. Angelo Cenedese

Director and Coordinator Prof. Fabio Vandin

Beniamino Pozzan
Heterogeneous multi-agent interaction management: from theory to practice
Ph.D. Thesis,
Advisor: Prof. Angelo Cenedese

Università degli Studi di Padova
Ph.D. School in Information Engineering
Department of Information engineering
Via Giovanni Gradenigo, 6, Padua, Italy
35131

Abstract

This thesis addresses the topic of controlling, estimating and managing heterogeneous multi-agent systems, namely interacting devices with different actuation and sensing capabilities, and the difficulties involved with migrating from theory alone to practice. For the sensing point of view, great attention is placed in inter-agent bearing sensing capabilities, where the agents can only measure their relative direction vectors expressed in their own local reference frames. This arises from the simplicity of such sensors and their processing pipelines; indeed, a bearing vector can be easily extrapolated from an optical camera and minimum computer vision algorithms. From the control point of view, gradient descent methods and Nonlinear Model Predictive Control play the lead role. The former is well suited for minimizing potential cost functions and for proving stability of the considered systems while the latter perfectly fits in the heterogeneous frameworks, with agents having different actuation constraints. A wide range of scenarios is presented. First of all, the control and stabilization of an heterogeneous bearing-based formation by the use of Bearing Rigidity Theory is investigated and the superiority of an heterogeneous control approach over a mixture of homogeneous ones is showed. Then, the localization of an uncooperative target by a group of seekers from their perturbed bearing measurements is tackled adopting an active-sense approach which maximizes the estimation accuracy. A novel NMPC trajectory controller for tilting quadrotors i proposed to fill the gap between under-actuated coplanar aerial platforms and inefficient tilted multirotors. Finally, the implementation challenges of multi agent controllers are investigated by dealing with the autonomous landing of a drone on a moving platform.

Contents

1 Introduction

In the last decades, *multi-agent* systems have gained increasing attention thanks to their advantages over single agent approaches. Indeed, having a group of small, agile, autonomous devices that act cooperatively to reach a common goal reveals to be more efficient, more robust, more safe than a large, fragile, single agent (Dorri et al., 2018; Cao et al., 2012). In this framework, multi-agent systems have been deeply adopted in search and rescue applications (Drew, 2021), localization (Aragues et al., 2011), cooperative transportation (Ota, 2006) and mapping (Kovacina et al., 2002) to mention a few. However, managing multi-agent systems involves dealing with additional complexity coming from:

- *Communication.* A multi-agent system is truly cooperative only if the single elements of the group exchange information about their own states and measurements. Depending on the nature of the task such communication could involve time delays, packet losses, asynchronous requirements. Additionally, the communication infrastructure or *communication graph* may not be complete. This requires great care to ensure that the control objective are still reached.

- *Coordination.* Once the common objective is defined, there is the need to share it to all the agents. This sometimes involves reevaluating common control strategies for single element systems to better fit the multi-agent scenario. These include leader-follower approaches (Cao et al., 2015), formation control techniques (Oh et al., 2015) and flocking (Olfati-Saber, 2006).

- *Distribution.* Robustness cannot be guaranteed if the control algorithms are centralized, as a failure of the main agent would compromise the whole system. For this reason multi-agent solutions should implement distributed algorithms to increase robustness and flexibility (Xiao et al., 2019).

At the same time, working with multi-agent systems can involve using *heterogeneous* agents, i.e., agents with different actuation, communication and sensing capabilities. Such a choice can span from specific application needs

or constraints and requires the control design to be carried out with great care to maximize the overall performance.

Sensing apparatus - The sensing apparatus of an agent is the first component enabling it to interact with the environment and for this reason it is a crucial part of the design process. When dealing with multi-agent infrastructures, each components normally needs to sense and estimate the relative position and orientation of the other agents. The easiest way to obtain this is to resort to *absolute* position measuring system, such as *GNSS* devices combined with 9dof-IMUs (accelerometers, gyroscopes, magnetometers) to retrieve the agents pose (position and orientation) with respect to a fixed, common and known reference frame. On one hand this approach is by far the simplest from the agents point of view as they just have to exchange their own measurements and it can deliver high accuracy, for example employing Real-time kinematic positioning (RTK) systems. On the other hand, it requires external devices, satellites for GNSS, anchors for anchor-based localization systems such as Ultra-Wide Band (UWB) approaches, for instance. Moreover, depending on an external infrastructure can pose a security risk for many applications, as a failure on the global positioning system could compromise the whole architecture. Finally, full orientation retrieval through IMUs requires a clean, static, unperturbed magnetic field otherwise the heading of the agents cannot be fully resolved. This limits the application of IMUs for robust orientation estimation to outdoor environment.

With this in mind, other sensing approaches should be considered for critical applications in which depending on a centralized positioning system is not an option. In this framework, *inter-agent sensing* perfectly fits. In broad terms, the basic concept of inter-agent sensing is having each agent measuring a quantity $\mathbf{y}_{ij}$ that depend only on the *relative state* of the measuring agent (agent i) and the measured agent (agent j). It can be formulated as

$$\mathbf{y}_{ij} = \mathbf{f}_i(\mathbf{x}_j - \mathbf{x}_i), \tag{1.1}$$

where $\mathbf{f}_i(\cdot)$ is a possibly non linear function that takes as input the relative state $\mathbf{x}_j - \mathbf{x}_i$ of the two agents and outputs the measurement taken by the i-th agent. It is important to notice that each agent can have a different measuring functions, here the concept of heterogeneous sensing. The other crucial aspect is that the measuring function acts on the *difference* between

the two agents states, where this difference could be the standard Euclidean one or even more complex representations if the agents state is a non trivial manifold such as the quaternion manifold $\mathbb{S}^3$. Nevertheless, it should be clear that as long as the agents state difference $\mathbf{x}_j - \mathbf{x}_i$ remains constant, in first approximation, no variation of the the measurement can be detected.[1] This is the first intrinsic property of inter agent measurement. For example, for multi-agent systems acting on Euclidean spaces such as $\mathbb{R}^k$, coordinated translations, namely translation of the measuring and measured agents in which both components keep the same velocity vector are trivial examples of maneuvers that do not affect inter-agent measurements. In $\mathbb{R}^d$, three major inter agent measurements can be listed for their importance in real world applications:

1. *Relative distance measurements* - relative distance measurements are in the form $\mathbf{y}_{ij} = \left\|\mathbf{x}_j - \mathbf{x}_i\right\| \geq 0$, they can be acquired by range sensors. Distance measurements do not provide direction information between the pair ij, but they are very useful for determining communication ranges and avoiding collisions.

2. *Relative bearing measurements* - relative bearing measurements are in the form $\mathbf{y}_{ij} = \frac{\mathbf{x}_j - \mathbf{x}_i}{\left\|\mathbf{x}_j - \mathbf{x}_i\right\|} \in \mathbb{S}^{d-1} = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| = 1\}$, they can be acquired by optical cameras and angle of arrival (AoA) sensors. This kind of measurements are the natural dual for the distance ones as they provide direction data but not distance information.

3. *Relative pose measurements* - relative pose measurements directly acquire $\mathbf{y}_{ij} = \mathbf{x}_j - \mathbf{x}_i \in \mathbb{R}^k$ and they are the only ones that do not cause loss in information. The can be acquired, for example, combining bearing and distance sensors.

Recalling that the initial objective was to have the multi-agent system to be able to estimate the relative position of each components, it should be trivial to see that relative pose measurement sensors are the ones that require the minimum effort: as long as the *communication* graph, namely the graph that determines which agent is able to communicate with, is complete, then

[1] Here *deterministic* measurements are considered, if *stochastic* approaches are instead adopted, the noise distribution could be space dependent, then the above assumption does not hold anymore.

every agent can know its relative position w.r.t any other agent by simply combining the measurements along a path connecting the two. The price to pay for such simplicity is the need to employ more complex sensors. On the other hand, using bearing or distance sensors is less demanding from an hardware point of view, but the relative position retrieval is by far more complex. The communication graph must satisfy stricter requirements and a whole mathematical theory, the *rigidity theory* has been developed for investigating these scenarios.

At the same time, when working with real sensors, it is important to remember that all measurements are affected by noise. While it is easy to deal with noises on pose measurements as they are additive to the actual data of interest, bearing and distance noises are non-additive and definitively non linear, which increases the complexity of localization algorithm.

Agents actuation - The agents actuation capabilities determine how they can actively interact with the environments. In the context of this thesis, the interest is focused on the movement capabilities of the agents, rather than, for instance, grasping capabilities, actuator redundancy, etc. In most of real world applications, agents can be divided in ground agents and aerial ones, noticing that boats and underwater vehicles can be easily approximated by ground or aerial ones, respectively, when considering their configuration space: $\mathbb{R}^2$ for the former and $\mathbb{R}^3$ for the latter. When necessary, an orientation must also be taken into account, this mean a heading angle for ground agents and a full three dimensional frame rotation encapsulation for aerial ones. In the most general sense, a full 3D agent can be modeled as

$$\dot{\mathbf{p}} = \mathbf{v} \tag{1.2a}$$
$$\dot{\mathbf{q}} = 1/2\ \mathbf{q} \circ \boldsymbol{\omega}^{+} \tag{1.2b}$$
$$\dot{\mathbf{v}} = \mathbf{f}_1(\mathbf{p}, \mathbf{q}, \mathbf{v}, \boldsymbol{\omega}, \mathbf{u}) \tag{1.2c}$$
$$\dot{\boldsymbol{\omega}} = \mathbf{f}_2(\mathbf{p}, \mathbf{q}, \mathbf{v}, \boldsymbol{\omega}, \mathbf{u}), \tag{1.2d}$$

where (1.2a) and (1.2b) are the linear and angular kinematics, (1.2c) and (1.2d) are the linear and angular dynamics equations. The state $\mathbf{x}$ is composed by the variables $\mathbf{p} \in \mathbb{R}^3$, the agent position, $\mathbf{q} \in \mathbb{S}^3$, the agent orientation, encoded with a quaternion, w.r.t. a fixed frame, $\mathbf{v} \in \mathbb{R}^3$, the agent linear velocity and finally $\boldsymbol{\omega} \in \mathbb{R}^3$, the agent angular velocity. In eq (1.2b) the

Table 1.1: Characterization of 3D agent actuation capabilities.

condition	actuation	example
$\mathrm{rk}\,\mathbf{F} = 6$	*fully*-actuated	coplanar quadrotor
$\mathrm{rk}\,\mathbf{F} < 6$	*under*-actuated	tilted multicopter
$\mathrm{null}\,\mathbf{F} > 1$	*over*-actuated	tilting quadrotor

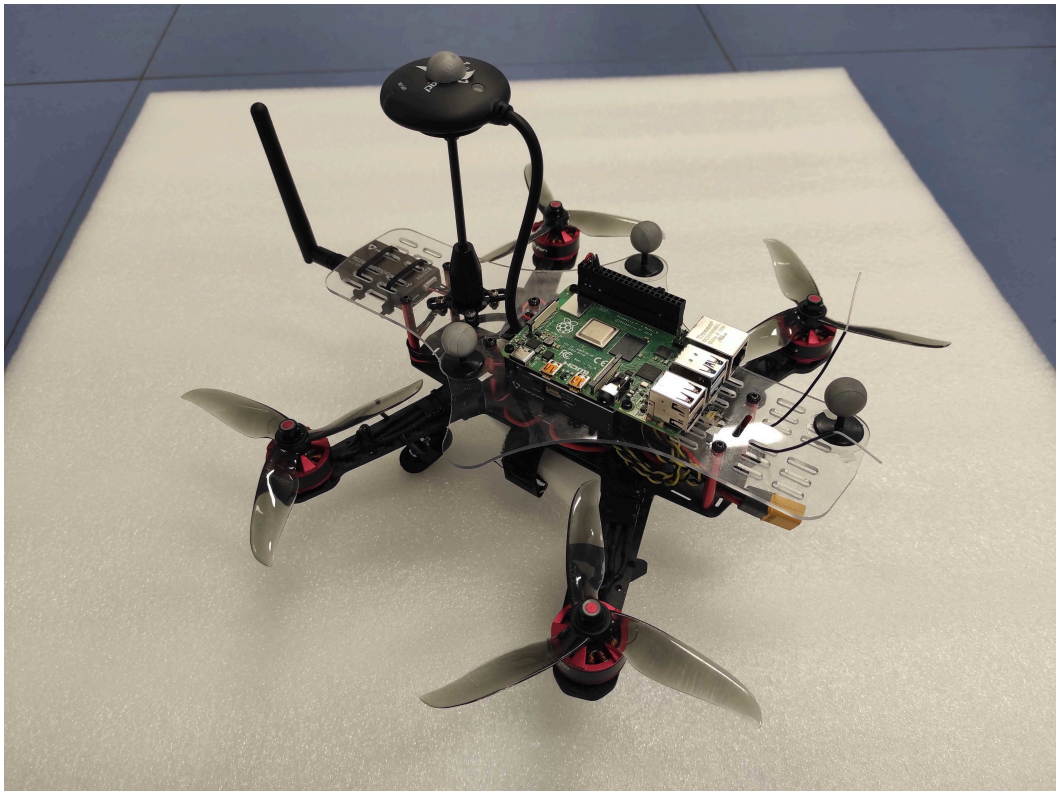


Figure 1.1: Coplanar quadrotor used in the SPARCS lab of the Department of Information Engineering, University of Padova.

standard quaternion algebra is adopted with $\cdot \circ \cdot$ the quaternion product operator and $\boldsymbol{\omega}^+$ the pure quaternion associated to $\boldsymbol{\omega}$. The agent input vector is instead $\mathbf{u} \in \mathbb{R}^m$. The agent *first order* actuation capabilities can be evaluated inspecting the maps $\mathbf{f}_1$ and $\mathbf{f}_2$. Be $\mathbf{F}_1(\mathbf{x}) \in \mathbb{R}^{3\times m}$ and $\mathbf{F}_2(\mathbf{x}) \in \mathbb{R}^{3\times m}$ the so called *allocation* matrices, the linear approximation of $\mathbf{f}_1$ and $\mathbf{f}_2$ w.r.t. the input $\mathbf{u}$,

$$\mathbf{F}_1(\mathbf{x}) = \nabla_{\mathbf{u}}\mathbf{f}_1, \qquad \mathbf{F}_2(\mathbf{x}) = \nabla_{\mathbf{u}}\mathbf{f}_2, \tag{1.3}$$

then the actuation capabilities are determined inspecting the rank and the nullity of $\mathbf{F} = \begin{bmatrix}\mathbf{F}_1 \\ \mathbf{F}_2\end{bmatrix} \in \mathbb{R}^{6\times m}$ and are reported in table 1.1 (Michieletto et al., 2018).

Figure 1.2: Tilted, fully actuated exarotor. Image courtesy of the *c-square* lab of the Department of Management and Engineering, University of Padova.

Three examples are reported in the following. The *coplanar* quadrotor, figure 1.1 is an *under*-actuated platform as it is no able to generate thrust on its x and y-local axes, even more trivially, it has only four inputs therefore it cannot possibly allocate arbitrary six dimensional thrust and torque vectors. The *tilted* multicopter, figure 1.2 is instead a *fully*-actuated platform (as long as a there are at least 6 rotors and the tilted axes are not oriented in degenerate configurations). While the above mentioned examples have *static* allocation matrices eq (1.3), the *tilting quadrotor*, which will be deeply discussed in chapter 4, has *dynamic* allocation matrix. On one hand, given a fixed state $\mathbf{x}$, in that configuration the tilting quadrotor is under-actuated. On the other, at a different state $\mathbf{x}'$, the platform is still under-actuated but the achievable thrust and torque vectors can change. This allows it to reach an hybrid concept of full actuation as long as it is allowed to reorient its spinning axes. Considering instead ground vehicles, example of fully-actuated devices are robot equipped with omnidirectional wheels, while the unicycle is an example of under-actuated vehicle with state dependent actuation matrix, figure 1.3.

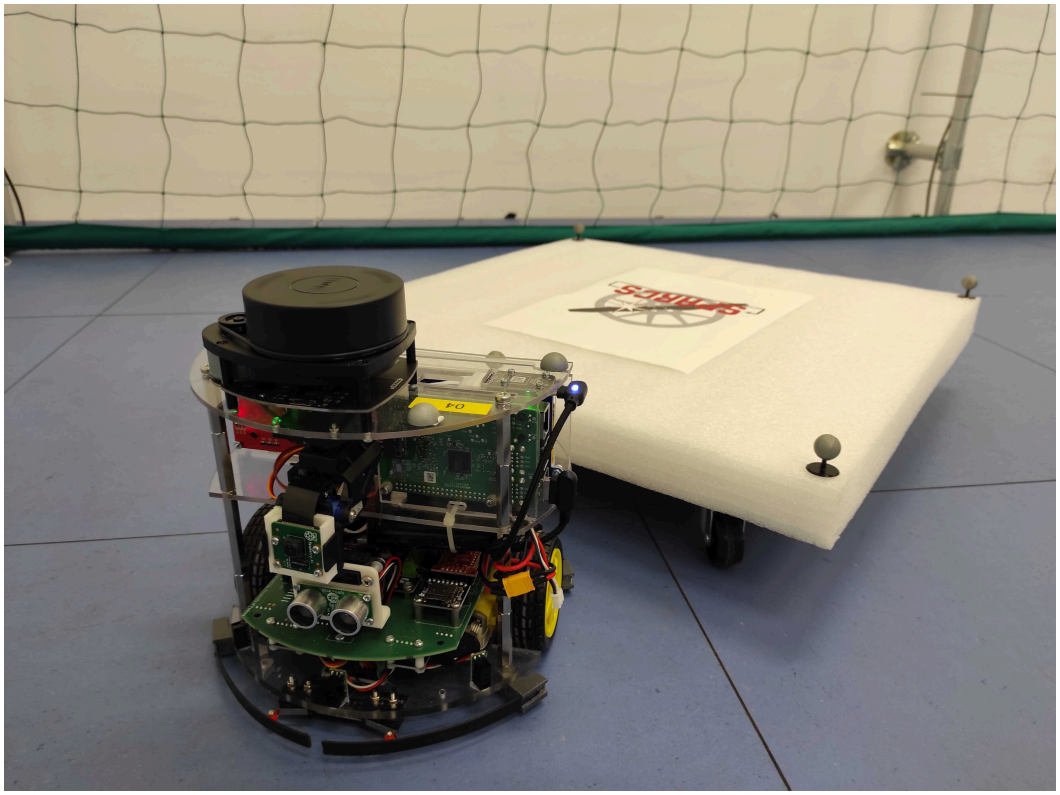


Figure 1.3: An Unmanned Ground Vehicle driven by a *differential drive* system used in the SPARCS lab of the Department of Information Engineering, University of Padova. Only two controllable spinning wheels make it an under-actuated platform that can only rotate and move along its longitudinal axis.

Heterogeneous multi-agent - with a clear picture of what heterogeneous sensing and actuation means, it is then possible to give an example of heterogeneous multi-agent system. Consider the task of inspecting a canyon, one could set up a base station at the top of it and deploy a ground vehicle at its bottom to perform the actual inspection, exploiting is better payload capacity compared to an aerial vehicle. At the same time multiple UAVs could be tasked to provide connectivity between ground station and inspecting robot and localization services.

Thesis structure - This thesis addresses the issue of managing heterogeneous multi-agent system focusing on four connected use cases. First of all, heterogeneous multi-agent *formation*, namely multi-agent system in which the agent are further constrained in keeping a specific, fixed spacial disposition, is considered when the agents can only acquire inter-agent bearing measurements (chapter 2). To this aim, the standard, *homogeneous* rigidity theory is expanded to account for heterogeneous actuation and a specific

gradient descent controller is designed and validated with formal proof and by simulations, showing that is outperforms an hierarchical controller combining standard homogeneous rigidity-based techniques. Moving forward, the problem of bearing-based target localization is discussed and tackled with an active-sense approach (chapter 3). In detail, a complete formation of *seekers* has to localize and uncooperative target using only noisy bearing measurements. As the seeker are able to move, an active-sense based controller is designed such that the seekers move to configurations that reduce the estimation uncertainty. From a rigidity perspective, this turn out to be strictly related to increasing the robustness of the formation. To address the gap between theory and real world applications of bearing-based approaches for multi-agent systems, chapter 4 proposes a Nonlinear Model Predictive Control approach for the tilting quadrotors. Tilting quadrotors can indeed supply the right actuation properties required by the less demanding bearing rigidity theory in $\mathbb{R}^3 \times \mathbb{S}^1$ without the need to use inefficient fully actuation platforms. Finally, an heterogeneous multi-agent application: the autonomous landing of a quadrotor on a moving platform, is discussed in chapter 5. Once again, the aerial vehicle can only sense its bearing vector w.r.t. the target and its altitude to the ground. Model predictive control is exploited to control the relative state of the two agents and manage tracking and landing phases. Simulation and real world experiments show the challenges in applying the theory on the field.

2 Bearing-based heterogenous control

This chapter tackles the problem of heterogeneous formation control under bearing measurement. To achieve that, Bearing Rigidity Theory is exploited and extended to take into account different agent actuation capabilities.

2.1 Introduction

In a broad sense, bearing rigidity theory aims at investigating the stiffness properties of given multi-element systems whose components are mutually constrained in terms of relative orientation (Ahn, 2020). In the last years, the study of such a theory has been deeply encouraged by the emergence of multi-agent systems as an enabling paradigm in several contexts. The bearing rigidity framework, indeed, suitably fits for applications related to the estimation and control of mobile agent formations wherein the involved devices are aware of their orientation w.r.t. some neighbors in the group. In this perspective, bearing constraints are virtual and the rigidity property of the multi-element system relies on the preservation of the agent interactions (Zhao and Zelazo, 2019).

One of the aims of the bearing rigidity theory is the identification of the conditions under which the geometric pattern induced by a set of points in any metric space can be uniquely determined by the bearing vectors between these points (Michieletto et al., 2021). Hence, bearing rigidity notions can be exploited in the design of multi-agent formation control laws, especially by accounting for the system rigidity as an architectural requirement for the convergence of the agents to a desired spatial configuration. Thus, recently, several bearing rigid based formation stabilization approaches have been proposed for multi-agent systems modeled as frameworks embedded in $SE(2)$, $SE(3)$, and more generic smooth manifolds (Michieletto and Cenedese, 2019; Schiano and Giordano, 2017; Chen et al., 2019a; Stacey and Mahony, 2017).

Most of the existing strategies apply to homogeneous formations, intended for groups of agents characterized by the same actuation capabilities. For instance, in (Zelazo et al., 2015), a distributed bearing-only formation control strategy is outlined for a team of unmanned ground vehicles (UGVs), ensuring the global asymptotic stability when the agents sensing interplay is minimal to guarantee the bearing preservation only in case of translations and scaling of the whole multi-robot system. Similarly, in (Schiano et al., 2016), a decentralized formation controller is designed and tested on a group of quadrotors aiming at steering the team of unmanned aerial vehicles (UAVs) towards a formation defined in terms of desired bearings.

Motivated, instead, by the IoT perspective, encouraging also the cooperation among devices with various actuation capabilities, this chapter focuses on *heterogeneous* formations, meant as multi-element systems whose components have different degrees of freedom (dofs) as to controllable variables. Assuming all the involved agents to be characterized by (local) communication and bearing sensing capabilities, the given contribution consists in the design of a distributed control law to stabilize a group of heterogeneous agents by preserving the existing bearing measurements. In doing this, the heterogeneous formations are modeled as *generalized* frameworks, namely frameworks embedded in the differential manifold $\mathbb{R}^3 \times \mathbb{S}^3$ which allows to describe the pose (position and orientation) of a rigid body in the 3D space by adopting the quaternion formalism (Michieletto and Cenedese, 2019), enriched with a mathematical codification of the agents actuation capabilities. The effectiveness of the proposed controller is confirmed by both rigorous proof of convergence and numerical results of an extensive simulations campaign. In particular, accounting for a formation involving (fully actuated) aerial and ground vehicles, the outlined solution is compared with an ad-hoc designed leader-follower combination of the bearing rigidity based controllers in (Michieletto and Cenedese, 2019) and (Zelazo et al., 2015) for the stabilization of planar and (fully-actuated) aerial multi-agents systems, respectively.

The rest of the chapter is organized as follows. section 2.2 recaps the main notion of Bearing Rigidity Theory. section 2.3 is devoted to the modeling of the heterogeneous formations. The proposed distributed control solution

is described insection 2.4 and its effectiveness is discussed in section 2.5. Section 2.6 summarizes the principal strengths of the proposed approach.

2.2 Preliminaries on bearing rigidity theory

Before diving into the specific applications of Bearing Rigidity Theory for heterogeneous formation, in this section the main results for classical homogeneous Rigidity Theory (Michieletto et al., 2021) are summarized. Initially, a formal definition of *formation* is required. We consider formations composed by $n \geq 3$ agents whose configuration belong to the domains $\mathcal{D}_i$, $i \in 1, \dots n$. Such formations can be modeled as a framework pairing the agent *spatial displacement* and *sensing interaction*.[1]

Definition 2.1 (Framework in $\bar{\mathcal{D}}$)**.** A framework in $\bar{\mathcal{D}}$ is an ordered pair $(\mathcal{G}, \mathbf{x})$ consisting of a connected (directed or undirected) graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = n$, and a configuration $\mathbf{x} \in \bar{\mathcal{D}} = \prod_{i=1}^{n} \mathcal{D}_i$.

The two components of the framework characterize a formation in terms of both agents configuration and interaction capabilities. $\mathcal{G}$ describes the available bearing measurements associating each agent to a vertex. It can be directed or undirected reflecting the possibility of the agents interactions to be either unidirectional or bidirectional. Anyway, in rigidity theory it is normally assumed to be time invariant. The formation configuration $\mathbf{x} \in \bar{\mathcal{D}}$ is associated with the set $\{\mathbf{x}_i \in \mathcal{D}_i\}_{i=1}^{n}$ describing the agent configurations so that $\mathbf{x}_i \in \mathcal{D}_i$ coincides with the i-th agent position when it is modeled as a particle point, and with the pair of its position and (partial/full) attitude when the rigid body model is assumed. Our attention is now restricted to *non-degenerate* formations:

Definition 2.2 (Non-Degenerate Formation)**.** A n-agents formation modeled as a framework $(\mathcal{G}, \mathbf{x})$ in $\bar{\mathcal{D}}$ is *non-degenerate* if the agents are univocally

[1] The configuration domain $\mathcal{D}_i$ does not need to be the agent *state*. However, for first order models, the two concepts are equivalent.

placed, i.e., two agents can not have the same position, and not all collinear, namely the matrix of the coordinates describing their positions is of rank greater than 1.

The sensing capability and the configuration of a formation characterizes its bearing rigidity properties. According to the framework model, any edge $e_k = e_{ij} = (v_i, v_j) \in \mathcal{E}$ ($|\mathcal{E}| = m$) represents a bearing measurement $\mathbf{b}_k = \mathbf{b}_{ij} \in \mathcal{M}$ recovered by the i-th agent which is able to sense the j-th agent, $i, j \in \{1 \dots n\}, i \neq j$. The *bearing measurement domain* can thus be defined as $\bar{\mathcal{M}} \coloneqq \mathcal{M}^m$.[2] Depending on the chosen model, the available measurements can be expressed in a common frame or according for local frames attached to each agent; however, in both cases, these are related to the framework configuration as stated in the following definition where an arbitrary edge labelling is introduced.

Definition 2.3 (Bearing Rigidity Function)**.** Given a n-agents formation modeled as a framework $(\mathcal{G}, \mathbf{x})$ in $\bar{\mathcal{D}}$, the *bearing rigidity function* is the map associating the configuration $\mathbf{x} \in \bar{\mathcal{D}}$ to the vector $\mathbf{b}_{\mathcal{G}}(\mathbf{x}) = \left[\mathbf{b}_1^\top \dots \mathbf{b}_m^\top\right]^\top \in \bar{\mathcal{M}}$ stacking all the available bearing measurements.

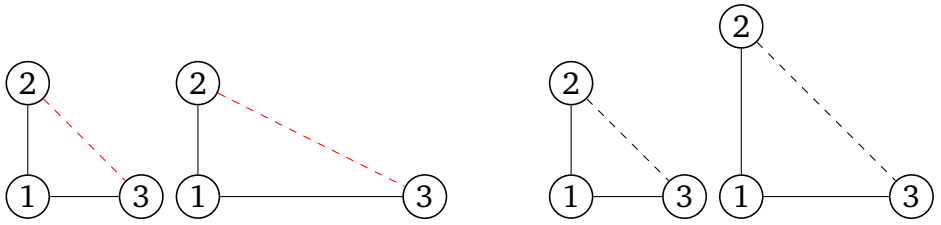
Starting from definition 2.3 it is possible to introduce the first notion related to the bearing rigidity theory, namely the equivalence and congruence of different frameworks.

Definition 2.4 (Bearing Equivalence)**.** Two frameworks $(\mathcal{G}, \mathbf{x})$ and $(\mathcal{G}, \mathbf{x}')$ are *Bearing Equivalent (BE)* if $\mathbf{b}_{\mathcal{G}}(\mathbf{x}) = \mathbf{b}_{\mathcal{G}}(\mathbf{x}')$.

Definition 2.5 (Bearing Congruence)**.** Two frameworks $(\mathcal{G}, \mathbf{x})$ and $(\mathcal{G}, \mathbf{x}')$ are *Bearing Congruent (BC)* if $\mathbf{b}_{\mathcal{K}}(\mathbf{x}) = \mathbf{b}_{\mathcal{K}}(\mathbf{x}')$, where $\mathcal{K}$ is the complete graph associated to $\mathcal{G}$.

Two examples of Bearing Equivalent and Bearing Congruent frameworks are shown in figure 2.1.

[2] It is assumed that all the agents have the same sensing apparatus. Otherwise each agent would have had a different bearing measurement domain $\mathcal{M}_i$ $i \in \{1 \dots n\}$.



(a) Two bearing equivalent frameworks. **(b)** Two bearing congruent frameworks.

Figure 2.1: Example of bearing equivalent (a) and bearing congruent (b) frameworks in $(\mathbb{R}^2)^3$. Solid black lines represent the edges in $\mathcal{E} = \{(1,2),\ (1,3)\}$ while dashed edges are in $\mathcal{K} \setminus \mathcal{E}$. In (a), both frameworks keep unchanged the edges in $\mathcal{E}$ and therefore they are bearing equivalent. However, $(2,3)$ changes as agent (3) is moved: therefore there is no bearing congruence. On the other hand, in (b) the scaling operation preserves also $(2,3)$ and therefore the two frameworks are bearing congruent.

Using the preimage under the bearing rigidity function, the set

$$\mathcal{Q}(\mathbf{x}) = \mathbf{b}_{\mathcal{G}}^{-1}(\mathbf{b}_{\mathcal{G}}(\mathbf{x})) \subseteq \bar{\mathcal{D}}$$

includes all the configurations $\mathbf{x}' \in \bar{\mathcal{D}}$ such that $(\mathcal{G}, \mathbf{x}')$ is BE to $(\mathcal{G}, \mathbf{x})$, while the set

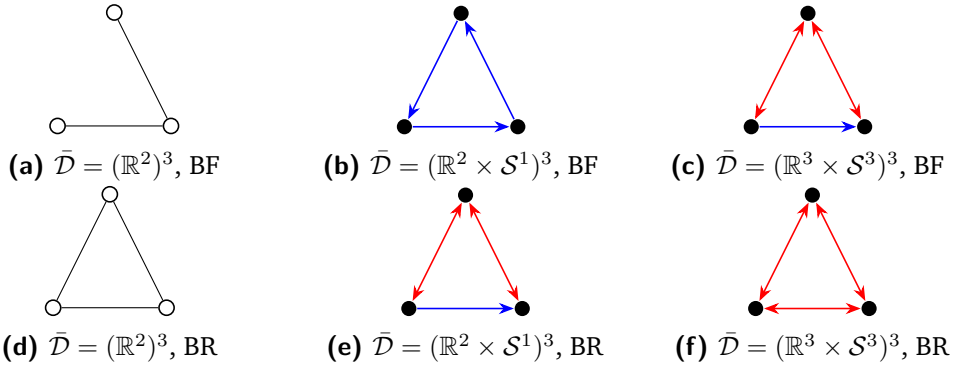
$$\mathcal{C}(\mathbf{x}) = \mathbf{b}_{\mathcal{K}}^{-1}(\mathbf{b}_{\mathcal{K}}(\mathbf{x})) \subseteq \bar{\mathcal{D}}$$

contains all the configurations $\mathbf{x}' \in \bar{\mathcal{D}}$ such that $(\mathcal{G}, \mathbf{x}')$ is BC to $(\mathcal{G}, \mathbf{x})$.

Proposition 2.1. *It holds that* $\mathcal{C}(\mathbf{x}) \subseteq \mathcal{Q}(\mathbf{x})$.

Proof. Assume that the $|\mathcal{E}_{\mathcal{K}}|$ edges of $\mathcal{K}$ are labelled such that the the first m are also the edges of $\mathcal{G}$. Suppose that $\mathbf{x}' \in \mathcal{C}(\mathbf{x})$; then from definitions 2.3 and 2.5, $\mathbf{b}'_k = \mathbf{b}_k$ for $k \in \{1 \dots |\mathcal{E}_{\mathcal{K}}|\}$, where $\mathbf{b}'_k, \mathbf{b}_k \in \mathcal{M}$ are the bearing measurements associated to the k-th edge of $(\mathcal{K}, \mathbf{x}'), (\mathcal{K}, \mathbf{x})$, respectively. This implies that also $\mathbf{b}'_k = \mathbf{b}_k$ for $k \in \{1 \dots m\}$, hence $\mathbf{b}_{\mathcal{G}}(\mathbf{x}) = \mathbf{b}_{\mathcal{G}}(\mathbf{x}')$, which is exactly definition 2.4 and $\mathbf{x}' \in \mathcal{Q}(\mathbf{x})$. □

As the bearing $\mathbf{b}_{\mathcal{K}}$ associated to the complete graph carries the greatest amount of information, it is the reasonable to use it to uniquely characterize the formation shape.



(a) $\bar{\mathcal{D}} = (\mathbb{R}^2)^3$, BF **(b)** $\bar{\mathcal{D}} = (\mathbb{R}^2 \times \mathcal{S}^1)^3$, BF **(c)** $\bar{\mathcal{D}} = (\mathbb{R}^3 \times \mathcal{S}^3)^3$, BF

(d) $\bar{\mathcal{D}} = (\mathbb{R}^2)^3$, BR **(e)** $\bar{\mathcal{D}} = (\mathbb{R}^2 \times \mathcal{S}^1)^3$, BR **(f)** $\bar{\mathcal{D}} = (\mathbb{R}^3 \times \mathcal{S}^3)^3$, BR

Figure 2.2: Examples of BF (a, b, c) and BR (d, e, f) formations embedded in different domains. Unidirected sensing links are represented by black edges. Directed sensing links are represented by blu arrows and red double arrows.

Definition 2.6 (Formation shape)**.** Given a formation modeled as a framework in $\bar{\mathcal{D}}$ $(\mathcal{G}, \mathbf{x})$, its shape is characterized by the collection of *all* the possible bearing measurements, namely, by the vector $\mathbf{b}_{\mathcal{K}}(\mathbf{x}) \in \mathbb{S}^{2^{n(n-1)}}$ where $\mathcal{K}$ is the complete graph associated to $\mathcal{G}$.

The definition of $\mathcal{Q}(\mathbf{x})$ and $\mathcal{C}(\mathbf{x})$ allows to introduce the (local and global) properties of bearing rigidity.

Definition 2.7 (Bearing Rigidity in $\bar{\mathcal{D}}$)**.** A framework $(\mathcal{G}, \mathbf{x})$ is (locally) *Bearing Rigid* in $\bar{\mathcal{D}}$ if there exists a neighborhood $\mathcal{U}(\mathbf{x}) \subseteq \bar{\mathcal{D}}$ of $\mathbf{x}$ such that

$$\mathcal{Q}(\mathbf{x}) \cup \mathcal{U}(\mathbf{x}) = \mathcal{C}(\mathbf{x}) \cup \mathcal{U}(\mathbf{x}). \tag{2.1}$$

Figure 2.2 shows examples of bearing rigid and bearing flexible (non rigid) flexible for different domains.

Definition 2.8 (Global Bearing Rigidity in $\bar{\mathcal{D}}$)**.** A framework $(\mathcal{G}, \mathbf{x})$ is *Globally Bearing Rigid* in $\bar{\mathcal{D}}$ if every framework which is BE to $(\mathcal{G}, \mathbf{x})$ is also BC to $(\mathcal{G}, \mathbf{x})$, or equivalently if $\mathcal{Q}(\mathbf{x}) = \mathcal{C}(\mathbf{x})$.

The meaning of the sets $\mathcal{U}(\mathbf{x}), \mathcal{C}(\mathbf{x}), \mathcal{Q}(\mathbf{x}) \subset \bar{\mathcal{D}}$ is graphically represented in figure 2.3. The requirement of "closeness" in the configurations space is missed in definition 2.8 of global bearing rigidity. As a consequence, this

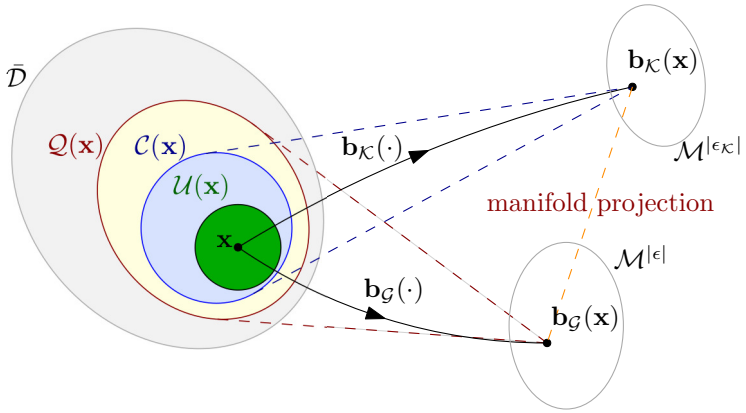


Figure 2.3: Graphical representation of the sets $\mathcal{Q}(\mathbf{x}), \mathcal{C}(\mathbf{x}), \mathcal{U}(\mathbf{x}) \subseteq \bar{\mathcal{D}}$ involved in the definition of bearing rigidity and global bearing rigidity.

property results to be stronger than the previous one as proved in the next theorem.

Theorem 2.2. *A GBR framework $(\mathcal{G}, \mathbf{x})$ is also BR.*

Proof. For a GBR framework $(\mathcal{G}, \mathbf{x})$, it holds that $\mathcal{Q}(\mathbf{x}) = \mathcal{C}(\mathbf{x})$. Consequently, condition (2.1) is valid for $\mathcal{U}(\mathbf{x}) = \bar{\mathcal{D}}$ demonstrating that the framework is BR. □

All the properties previously defined concern rigidity for *static* frameworks. Nevertheless, in real-world scenarios agents belonging to a formation are generally able to move. For this reason, the analysis of bearing rigidity for *dynamic* agents formations, which will serve as base for the development of bearing-based controllers, is performed. Dynamic agents formations can be modeled as frameworks $(\mathcal{G}, \mathbf{x})$ where the configuration can change over time, namely $\mathbf{x} = \mathbf{x}(t) \in \bar{\mathcal{D}}$, while the graph $\mathcal{G}$ is fixed. The goal for the rest of this section is to identify the constraints under which a given dynamic formation can deform while maintaining its rigidity, i.e., preserving the existing bearings among the agents.

For a given formation the *instantaneous variation vector* $\boldsymbol{\delta}(t) \in \bar{\mathcal{I}}$ represents a deformation of $\mathbf{x}(t)$ taking place in an infinitesimal time interval. This vector belongs to the *instantaneous variation domain* $\bar{\mathcal{I}} := \prod_{i=1}^{n} \mathcal{I}_i$ whose identity depends on the space of agent controllable variables through the agent command space $\mathcal{I}_i$. The characterization of $\mathcal{I}_i$ is made by the adopted

dynamic model and the same applies for the function that relate $\frac{d}{dt}\mathbf{x}(t)$ to $\boldsymbol{\delta}(t)$.

Remark. Normally, it holds that $\frac{d}{dt}\mathbf{x}(t) \neq \boldsymbol{\delta}(t)$, namely, the time derivative of the configuration do not coincide with the instantaneous variation vector.

For homogeneous formations, it is assumed that all agents have the same command spaces, that is $\mathcal{I}_i = \mathcal{I}$, and thus $\bar{\mathcal{I}} = \mathcal{I}^n$. The introduction of $\boldsymbol{\delta}(t)$ allows to describe the bearing measurement dynamics in terms of configuration deformations. The relation between $\boldsymbol{\delta}(t)$ and the time derivative of the bearing rigidity function, clarified in the next definition, constitutes the starting point for the study of the rigidity properties of dynamic formations.

Definition 2.9 (Bearing Rigidity Matrix)**.** For a given (dynamic) framework $(\mathcal{G}, \mathbf{x})$, the *bearing rigidity matrix* is the matrix $\mathbf{B}_{\mathcal{G}}(\mathbf{x}(t))$ that satisfies the relation

$$\dot{\mathbf{b}}_{\mathcal{G}}\Big(\mathbf{x}(t)\Big) = \frac{d}{dt}\mathbf{b}_{\mathcal{G}}\Big(\mathbf{x}(t)\Big) = \mathbf{B}_{\mathcal{G}}\Big(\mathbf{x}(t)\Big)\boldsymbol{\delta}(t). \tag{2.2}$$

The dimension of the bearing rigidity matrix typically depends on the spaces $\bar{\mathcal{M}}$ and $\bar{\mathcal{I}}$. Nevertheless, one can observe that the null space of $\mathbf{B}_{\mathcal{G}}(\mathbf{x}(t))$ always identifies the (first-order) deformations of the configuration $\mathbf{x}(t)$ that maintain the bearing measurements unchanged. From a physical perspective, such variations of $(\mathcal{G}, \mathbf{x})$ can be considered as sets of command inputs to provide to the agents to instantaneously drive the formation from the initial configuration $\mathbf{x} = \mathbf{x}(t)$ to a final configuration $\mathbf{x}'$ belonging to $\mathcal{Q}(\mathbf{x})$.

Definition 2.10 (Infinitesimal Variation)**.** For a given (dynamic) framework $(\mathcal{G}, \mathbf{x})$, an *infinitesimal variation* is an instantaneous variation $\boldsymbol{\delta}(t) \in \bar{\mathcal{I}}$ that allows to preserve the relative direction among the interacting agents.

Lemma 2.3. *For a given (dynamic) framework $(\mathcal{G}, \mathbf{x})$, an infinitesimal variation in an instantaneous variation $\boldsymbol{\delta}(t) \in \bar{\mathcal{I}}$ such that $\boldsymbol{\delta}(t) \in \ker\Big(\mathbf{B}_{\mathcal{G}}\Big(\mathbf{x}(t)\Big)\Big)$.*

For a given $(\mathcal{G}, \mathbf{x})$, there are many infinitesimal variations. However, there exists infinitesimal variations that hold for *any* graphs. This follows from the next results.

Theorem 2.4. *Given a (dynamic) framework* $(\mathcal{G}, \mathbf{x})$ *and denoting as* $\mathcal{K}$ *the complete graph associated to* $\mathcal{G}$*, it holds that* $\ker\left(\mathbf{B}_{\mathcal{K}}\left(\mathbf{x}(t)\right)\right) \subseteq \ker\left(\mathbf{B}_{\mathcal{G}}\left(\mathbf{x}(t)\right)\right)$.

Proof. Since each edge of the graph $\mathcal{G}$ belongs to the graph $\mathcal{K}$, the equation set defined by $\mathbf{B}_{\mathcal{G}}\left(\mathbf{x}(t)\right)\boldsymbol{\delta}(t) = \mathbf{0}$ constitutes a subset of the equations set defined by $\mathbf{B}_{\mathcal{K}}\left(\mathbf{x}(t)\right)\boldsymbol{\delta}(t) = \mathbf{0}$. Then $\boldsymbol{\delta}(t) \in \ker\left(\mathbf{B}_{\mathcal{K}}\left(\mathbf{x}(t)\right)\right)$ implies $\boldsymbol{\delta}(t) \in \ker\left(\mathbf{B}_{\mathcal{G}}\left(\mathbf{x}(t)\right)\right)$. □

In the light of theorem 2.4, the notion of *trivial variations* is introduced by considering the infinitesimal variations related to the complete graph $\mathcal{K}$ associated to $\mathcal{G}$. These ensure the measurements preservation for each pair of node in the formation ($\mathbf{x}' \in \mathcal{C}(\mathbf{x})$), i.e., the formation shape preservation.

Definition 2.11 (Trivial Variation)**.** For a given (dynamic) framework $(\mathcal{G}, \mathbf{x})$, a *trivial variation* in an instantaneous variation $\boldsymbol{\delta}(t) \in \bar{\mathcal{I}}$ such that shape uniqueness is preserved.

Lemma 2.5. *For a given (dynamic) framework* $(\mathcal{G}, \mathbf{x})$*, a trivial variation in an instantaneous variation* $\boldsymbol{\delta}(t) \in \bar{\mathcal{I}}$ *such that* $\boldsymbol{\delta}(t) \in \ker\left(\mathbf{B}_{\mathcal{K}}\left(\mathbf{x}(t)\right)\right)$*, where* $\mathbf{B}_{\mathcal{K}}\left(\mathbf{x}(t)\right)$ *is the bearing rigidity matrix computed for the complete graph* $\mathcal{K}$ *associated to* $\mathcal{G}$*.*

Theorem 2.4 is fundamental for the next definition that constitutes the core of the rigidity theory.

Definition 2.12 (Infinitesimal Bearing Rigidity in $\bar{\mathcal{D}}$)**.** A (dynamic) framework $(\mathcal{G}, \mathbf{x})$ is *Infinitesimally Bearing Rigid* in $\bar{\mathcal{D}}$ if

$$\ker\left(\mathbf{B}_{\mathcal{K}}\left(\mathbf{x}(t)\right)\right) = \ker\left(\mathbf{B}_{\mathcal{G}}\left(\mathbf{x}(t)\right)\right)$$

Otherwise, it is *Infinitesimally Bearing Flexible*.

A framework $(\mathcal{G}, \mathbf{x})$ is IBR if all its infinitesimal variation are also trivial. Contrarily a framework is IBF it there exists at least an infinitesimal variation that wraps the configuration $\mathbf{x} = \mathbf{x}(t)$ in $\mathbf{x}' \in \mathcal{Q}(\mathbf{x}) \setminus \mathcal{C}(\mathbf{x})$.

For now on, the time dependency is dropped out to simplify the notation.

2.3 Heterogeneous formation characterization

Given the Bearing Rigidity Theory introduction provided in the previous section, it is possible to move forward to characterize *heterogeneous* formations with $n \geq 3$ agents.

2.3.1 Single agent model

In the most general sense, each i-th agent, $i \in \{1 \dots n\}$, in a heterogeneous formation can be modeled as a rigid body acting in 3D space. Thus, its spatial displacement can be characterized by introducing the local frame $\mathcal{F}_i$ (*body frame*), having origin O_i coincident with the agent center of mass and axes identified by the unit vectors $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ defining the canonical basis of $\mathbb{R}^3$, and the inertial frame $\mathcal{F}_W$ (*world frame*), fixed and common, even if unknown, for all the formation components. Its configuration space is therefore characterized by the position of O_i in the world frame and the orientation of $\mathcal{F}_i$ w.r.t. $\mathcal{F}_W$. The former data can simply be encoded by the vector $\mathbf{p}_i \in \mathbb{R}^3$ of the coordinates of O_i in $\mathcal{F}_W$ while the latter admits a great variety of representations: rotation matrices, Euler Angels, DCM, quaternions. Here, the latter is adopter, as it does not have singularities and it is more computational efficient. Therefore, the unit quaternion $\mathbf{q}_i = [\eta_i \;\; \boldsymbol{\epsilon}_i^\top]^\top \in \mathbb{S}^3$ defines the rotation of $\mathcal{F}_i$ w.r.t. $\mathcal{F}_W$. Finally, the vector $\mathbf{x}_i = [\mathbf{p}_i^\top \;\; \mathbf{q}_i^\top]^\top$ belonging to the differential manifold $\mathbb{R}^3 \times \mathbb{S}^3$ identifies the i-th agent pose in world frame, i.e., its time-varying *configuration*. In view of section 2.2, we have that $\mathcal{D}_i = \mathbb{R}^3 \times \mathbb{S}^3$, $i \in \{1, \dots n\}$.

Introducing the linear velocity $\mathbf{v}_i \in \mathcal{V}_i \subseteq \mathbb{R}^3$ of O_i w.r.t. $\mathcal{F}_W$ and the angular velocity $\boldsymbol{\omega}_i \in \Omega_i \subseteq \mathbb{R}^3$ of $\mathcal{F}_i$ w.r.t. $\mathcal{F}_W$, both in body frame, the i-th agent kinematics is governed by

$$\dot{\mathbf{p}}_i = \mathbf{R}_i \mathbf{v}_i, \qquad \dot{\mathbf{q}}_i = \frac{1}{2}\mathbf{M}(\mathbf{q}_i)\boldsymbol{\omega}_i, \tag{2.3}$$

where $\mathbf{R}_i \in SO(3)$ is the rotation matrix associated to $\mathbf{q}_i$ and the matrix $\mathbf{M}(\mathbf{q}_i) \in \mathbb{R}^{4\times 3}$ maps the agent angular velocity into the time derivative of its quaternion based orientation

$$\mathbf{M}(\mathbf{q}_i) = \begin{bmatrix} \eta_i & -\boldsymbol{\epsilon}_i^\top \\ \boldsymbol{\epsilon}_i & \eta_i \mathbb{1}_3 + [\boldsymbol{\epsilon}_i]_\times \end{bmatrix}, \tag{2.4}$$

with $[\mathbf{x}]_\times \in \mathbb{R}^{3\times 3}$ the skew symmetric matrix associated to $\mathbf{x} \in \mathbb{R}^3$. Note that the quaternion time derivative equation (2.3) is equivalent to eq. (1.2b) in chapter 1.

The linear and angular velocity in eq (2.3) can be interpreted as the controllable variables of the i-th agent. When $\mathcal{V}_i = \Omega_i = \mathbb{R}^3$, the agent is *fully*-actuated: it can translate and rotate in any direction of the 3D space having three translational and three rotational controllable degrees of freedom (cdfos). When $\mathcal{V}_i$ or $\Omega_i \subset \mathbb{R}^3$, instead, the agent is *under*-actuated and its movement is constrained only in some directions, having less than 6 cdofs. In view of section 2.2 we then have that $\dim(\mathcal{I}_i) = c_i \in \{0 \dots 6\}$ are the i-th agent cdofs. The agent actuation capabilities are then characterized by the *selection* map

$$S_i \colon \mathcal{I}_i \to \mathcal{V}_i \times \Omega_i, \quad \boldsymbol{\delta}_i \mapsto \begin{bmatrix} \mathbf{v}_i^\top & \boldsymbol{\omega}_i^\top \end{bmatrix}^\top. \tag{2.5}$$

which translates the infinitesimal variation vectors $\boldsymbol{\delta}_i$ into the admissible linear and angular velocities.

Hereafter the following *decoupling* hypothesis is assumed in regard to the agent translation and rotation movements, nonetheless the control law described in section 2.4 is valid also when this is not in place, with minor suitable changes.

Assumption 2.1. Any i-th agent can provide decoupled translation and rotation commands, meaning that $\boldsymbol{\delta}_i$ in (2.5) is made up of two components that can be independently assigned. Formally, $\boldsymbol{\delta}_i = [\boldsymbol{\delta}_{p,i}^\top, \boldsymbol{\delta}_{o,i}^\top]^\top \in \mathcal{I}_i = \mathcal{I}_{p,i} \times \mathcal{I}_{o,i}$ with $\boldsymbol{\delta}_{p,i}$ and $\boldsymbol{\delta}_{o,i}$, respectively associated to the agent linear and angular velocity, $\mathcal{I}_{p,i}$ and $\mathcal{I}_{o,i}$ representing the *i-th agent instantaneous position* and *orientation variation domains*.

Under assumption 2.1, the i-th agent total number of cdofs results $c_i = c_{t,i} + c_{r,i}$ with $c_{t,i} = \dim(\mathcal{I}_{p,i}) = \dim(\mathcal{V}_i)$ and $c_{r,i} = \dim(\mathcal{I}_{o,i}) = \dim(\Omega_i)$ denoting its translational and rotational cdofs, respectively. Moreover, the *selection* map S_i of eq. (2.5) can be split into the following terms,

$$S_{p,i} \colon \mathcal{I}_{p,i} \to \mathcal{V}_i, \quad \boldsymbol{\delta}_{p,i} \mapsto \mathbf{v}_i \tag{2.6a}$$

$$S_{o,i} \colon \mathcal{I}_{o,i} \to \Omega_i, \quad \boldsymbol{\delta}_{o,i} \mapsto \boldsymbol{\omega}_i, \tag{2.6b}$$

with bijective $S_{p,i}$. In addition, accounting for real-world scenarios, hereafter, the structure of the maps $S_{p,i}, S_{o,i}$ is assumed as follows.

Assumption 2.2. The maps $S_{p,i}, S_{o,i}$ are linear, hence it is

$$\mathbf{v}_i = S_{p,i}(\boldsymbol{\delta}_{p,i}) = \mathbf{S}_{p,i}\boldsymbol{\delta}_{p,i} \qquad \mathbf{S}_{p,i} \in \mathbb{R}^{3\times c_{p,i}}, \tag{2.7a}$$

$$\boldsymbol{\omega}_i = S_{o,i}(\boldsymbol{\delta}_{o,i}) = \mathbf{S}_{o,i}\boldsymbol{\delta}_{o,i} \qquad \mathbf{S}_{o,i} \in \mathbb{R}^{3\times c_{o,i}}. \tag{2.7b}$$

The next example aims at clarifying the introduced model.

Example 2.1. For a UGV that can can rotate only around the z-axis of its body frame and translate on the (xy)-plane, we have $\mathcal{V}_i = \operatorname{span}\{\mathbf{e}_1, \mathbf{e}_2\}$ and $\Omega_i = \operatorname{span}\{\mathbf{e}_3\}$, and thus $\mathcal{I}_{p,i} = \mathbb{R}^2$, $\mathcal{I}_{o,i} = \mathbb{R}$, $\mathbf{S}_{p,i} = [\mathbf{e}_1 \;\; \mathbf{e}_2]$, and $\mathbf{S}_{o,i} = \mathbf{e}_3$.

Example 2.2. An under actuated unmanned aerial vehicle (4dofsUAVs) can freely translate in the 3D space but only rotate around its vertical axis. Namely, it has $\mathcal{V}_i = \mathbb{R}^3$ and $\Omega_i = \operatorname{span}(\mathbf{e}_3)$. Being fully actuated w.r.t. the translations $\mathcal{I}_{p,i} = \mathbb{R}^3$ and $\mathbf{S}_{p,i} = \mathbf{I}_3$. Instead, it has only one rotational dof; therefore, $\mathcal{I}_{o,i} = \mathbb{R}$ and $\mathbf{S}_{o,i} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^\top$.

As the considered agents belongs to $\mathbb{R}^3 \times \mathbb{S}^3$ and each agent is characterized by the local frames $\mathcal{F}_i$, their bearing measurements are also acquired w.r.t. those frames:

$$\mathbf{b}_{ij} = \mathbf{R}_i^\top \bar{\mathbf{p}}_{ij} \in \mathbb{S}^2, \quad \bar{\mathbf{p}}_{ij} = \frac{\mathbf{p}_j - \mathbf{p}_i}{\left\|\mathbf{p}_j - \mathbf{p}_i\right\|} \in \mathbb{S}^2. \tag{2.8}$$

In eq. (2.8), $\bar{\mathbf{p}}_{ij}$ is the bearing vector of the j-th agent acquired by the i-th one expressed in the global reference frame. $\mathbf{R}_i^\top \in \mathbb{R}^{3\times 3}$ rotates the measurement

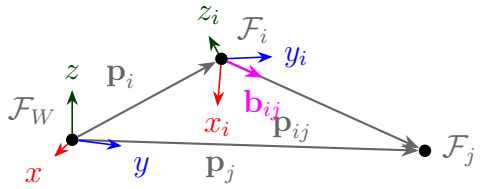


Figure 2.4: agents sensing interaction - magenta arrow indicates the bearing measurement of i-th agent w.r.t. its j-th neighbor.

back to the i-th local frame. A graphical representation of the vector $\mathbf{b}_{ij}$ is given in figure 2.4.

Finally, each agent is also supposed to communicate with its neighbors. In particular, it can share the retrieved measurements, allowing the sensed formation components to estimate their relative orientation.

2.3.2 Networked system model

Stacking the position and orientation of all the formation components into the vectors $\mathbf{p} = [\mathbf{p}_1^\top \dots \mathbf{p}_n^\top]^\top \in \mathbb{R}^{3n}$ and $\mathbf{q} = [\mathbf{q}_1^\top \dots \mathbf{q}_n^\top]^\top \in \mathbb{S}^{3^n}$, respectively, the time-varying *configuration of the whole system* is, thus, represented by $\mathbf{x} = [\mathbf{p}^\top \quad \mathbf{q}^\top]^\top \in \mathbb{R}^{3n} \times \mathbb{S}^{3^n}$. Furthermore, introducing the vector $\mathbf{u} = [\mathbf{v}_1^\top \dots \mathbf{v}_n^\top, \boldsymbol{\omega}_1^\top \dots \boldsymbol{\omega}_n^\top]^\top \in \prod_{i=1}^n \mathcal{V}_i \times \prod_{i=1}^n \Omega_i$ and accounting for (2.3), the evolution of the formation is governed by

$$\dot{\mathbf{x}} = \begin{bmatrix} \mathbf{D}_1(\mathbf{q}) & \mathbf{0}_{3n\times 3n} \\ \mathbf{0}_{4n\times 3n} & \mathbf{D}_2(\mathbf{q}) \end{bmatrix} \mathbf{u} = \mathbf{D}(\mathbf{q})\mathbf{u}, \tag{2.9}$$

where $\mathbf{D}_1(\mathbf{q}) = \mathrm{diag}(\mathbf{R}_i) \in \mathbb{R}^{3n\times 3n}$ and $\mathbf{D}_2(\mathbf{q}) = \mathrm{diag}(\frac{1}{2}\mathbf{M}(\mathbf{q}_i)) \in \mathbb{R}^{4n\times 3n}$ are diagonal block matrices. In detail, under assumption 2.2, the relation (2.9) can be rewritten introducing the *commands vector* $\boldsymbol{\delta} = [\boldsymbol{\delta}_{p,1}^\top \dots \boldsymbol{\delta}_{p,n}^\top \, \boldsymbol{\delta}_{o,1}^\top \dots \boldsymbol{\delta}_{o,n}^\top]^\top$ belonging to the *instantaneous variation domain* $\bar{\mathcal{I}} = \prod_{i=1}^n \mathcal{I}_{p,i} \times \prod_{i=1}^n \mathcal{I}_{o,i}$ having dimension $c = c_t + c_r$ with $c_t = \sum_{i=1}^n c_{t,i}$ and $c_r = \sum_{i=1}^n c_{r,i}$. It holds that

$$\dot{\mathbf{x}} = \mathbf{D}(\mathbf{q}) \begin{bmatrix} \mathbf{S}_p & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_o \end{bmatrix} \boldsymbol{\delta} = \mathbf{D}(\mathbf{q})\mathbf{S}\boldsymbol{\delta}, \tag{2.10}$$

with $\mathbf{S}_p = \mathrm{diag}\left(\mathbf{S}_{p,i}\right) \in \mathbb{R}^{3n\times c_t}, \mathbf{S}_o = \mathrm{diag}\left(\mathbf{S}_{o,i}\right) \in \mathbb{R}^{3n\times c_o}$. Given these premises, any heterogeneous formation can be modeled as a *generalized framework*

which extends the one of definition 2.1, whose dynamic behavior is described by eq (2.10).

Definition 2.13 (Generalized Framework)**.** A *generalized framework* is an ordered triple $(\mathcal{G}, \mathbf{x}, \mathcal{H})$ consisting of a connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = n \geq 3$ and $|\mathcal{E}| = m$, a configuration $\mathbf{x} \in \mathbb{R}^{3n} \times \mathbb{S}^{3^n}$, and a collection of instantaneous variation domains $\mathcal{H} = \{\mathcal{I}_1 \dots \mathcal{I}_n\}$.

Note that the agent sensing is not assumed to be be bidirectional, as $\mathbf{b}_{ij}$ and $\mathbf{b}_{ji}$ carry different information as an effect of being acquired in the local agent frames. Therefore, the sensing graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is *directed*.

To summarize, heterogeneous formations differs from homogeneous ones for the following aspects:

- All agents use the same embedding domain $\mathcal{D} = \mathbb{R}^3 \times \mathcal{S}^3$ even if the belong to subspaces of it.
- The difference in true embedding domains is modeled in terms of actuation capabilities and then it is captured by agent-specific instantaneous variation domains $\mathcal{I}_i$.

2.4 Bearing rigidity-based formation control

With a 3D heterogeneous formation properly defined, it is now possible to address the main task of this chapter, the design of a distributed controller aiming at stabilizing those formations through the solution of the following problem.

Problem 2.1 ((Bearing Based) Formation Stabilization)**.** For a given heterogeneous formation modeled as a generalized framework $(\mathcal{G}, \mathbf{x}, \mathcal{H})$ subject to (2.10), consider a desired formation shape described by $\mathbf{b}^*_{\mathcal{K}} \in \mathbb{S}^{2^{n(n-1)}}$ which is *feasible* meaning that it exists a configuration $\mathbf{x}^*$ such that $\mathbf{b}^*_{\mathcal{K}} =$

$\mathbf{b}_{\mathcal{K}}(\mathbf{x}^*)$. The *(bearing based) formation stabilization problem* consists in asymptotically zeroing the *shape error* $\mathbf{e}_{\mathcal{K}}(\mathbf{x}) \in \mathbb{R}^{3n(n-1)}$ defined as

$$\mathbf{e}_{\mathcal{K}}(\mathbf{x}) = \mathbf{b}_{\mathcal{K}}(\mathbf{x}) - \mathbf{b}^*_{\mathcal{K}}. \tag{2.11}$$

Problem 2.1 is here faced resting on the main notions of the bearing rigidity theory, applied to heterogeneous formations.

2.4.1 Stabilization control Law

Inspired by (Zhao and Zelazo, 2019; Schiano et al., 2016; Zelazo et al., 2015; Michieletto and Cenedese, 2019), the bearing rigidity matrix is here exploited in the solution of problem 2.1. Indeed, accounting for the *bearing error* $\mathbf{e}_{\mathcal{G}}(\mathbf{x}) \in \mathbb{R}^{3m}$ as

$$\mathbf{e}_{\mathcal{G}}(\mathbf{x}) = \mathbf{b}_{\mathcal{G}}(\mathbf{x}) - \mathbf{b}_{\mathcal{G}}(\mathbf{x}^*) = \mathbf{b}_{\mathcal{G}}(\mathbf{x}) - \mathbf{b}^*_{\mathcal{G}}, \tag{2.12}$$

it is possible to prove that its dynamics $\dot{\mathbf{e}}_{\mathcal{G}}(\mathbf{x}) = \dot{\mathbf{b}}_{\mathcal{G}}(\mathbf{x}) = \mathbf{B}_{\mathcal{G}}(\mathbf{x})\boldsymbol{\delta}$ asymptotically converges to zero by selecting the command vector as follows with $k_c > 0$ be a tunable gain

$$\boldsymbol{\delta} = k_c \mathbf{B}^\top_{\mathcal{G}}(\mathbf{x})\mathbf{b}^*_{\mathcal{G}}. \tag{2.13}$$

Proposition 2.6. *Given a generalized framework $(\mathcal{G}, \mathbf{x}, \mathcal{H})$ subject to (2.10), and a feasible bearing measurements vector $\mathbf{b}^*_{\mathcal{G}} \in \mathbb{S}^{2^m}$, it holds that $\mathbf{e}_{\mathcal{G}}(\mathbf{x}) = \mathbf{0}_{3m}$ is an asymptotically stable equilibrium point for the dynamics of the bearing error (2.12) driven by the control law (2.13), namely for the system*

$$\dot{\mathbf{e}}_{\mathcal{G}}(\mathbf{x}) = -k_c \mathbf{B}_{\mathcal{G}}(\mathbf{x})\mathbf{B}^\top_{\mathcal{G}}(\mathbf{x})\mathbf{e}_{\mathcal{G}}(\mathbf{x}). \tag{2.14}$$

Proof. Let consider the positive definite Lyapunov function

$$\mathrm{V}(\mathbf{e}_{\mathcal{G}}(\mathbf{x})) = \frac{1}{2k_c}\mathbf{e}_{\mathcal{G}}(\mathbf{x})^\top \mathbf{e}_{\mathcal{G}}(\mathbf{x}), \tag{2.15}$$

whose derivative $\dot{\mathrm{V}}(\mathbf{e}_{\mathcal{G}}(\mathbf{x})) = -\mathbf{e}_{\mathcal{G}}(\mathbf{x})^\top \mathbf{B}_{\mathcal{G}}(\mathbf{x})\mathbf{B}^\top_{\mathcal{G}}(\mathbf{x})\mathbf{e}_{\mathcal{G}}(\mathbf{x})$ is negative semi-definite since the product $\mathbf{B}_{\mathcal{G}}(\mathbf{x})\mathbf{B}^\top_{\mathcal{G}}(\mathbf{x})$ is a positive semi-definite matrix for any $\mathbf{x} \in \mathbb{R}^{3n} \times \mathbb{S}^{3^n}$. It follows that $\mathbf{e}_{\mathcal{G}}(\mathbf{x}) = \mathbf{0}_{3m}$ is a simple stable

equilibrium point for the bearing error dynamics, which converges to the set $\mathcal{Z} = \{\mathbf{e}_\mathcal{G}(\mathbf{x}) \mid \mathbf{x} \in \mathcal{U}(\mathbf{x}^*), \dot{\mathrm{V}}(\mathbf{e}_\mathcal{G}(\mathbf{x})) = 0\}$, where $\mathcal{U}(\mathbf{x}^*)$ is a neighborhood of $\mathbf{x}^*$. Now, exploiting the properties of the adjoint operator, the definition (2.12) and the fact that $\mathbf{B}_\mathcal{G}^\top(\mathbf{x})\mathbf{b}_\mathcal{G}(\mathbf{x}) = \mathbf{0}_c$ due to (2.24), one can realize that the equilibrium condition $\dot{\mathrm{V}}(\mathbf{e}_\mathcal{G}(\mathbf{x})) = 0$ implies that $\mathbf{B}_\mathcal{G}^\top(\mathbf{x})\mathbf{b}_\mathcal{G}(\mathbf{x}^*) = \mathbf{0}_c$. Then, accounting for the Taylor's expansion given $\mathbf{x}^* = \mathbf{x} + d\mathbf{x}$, it follows that

$$\begin{aligned}\mathbf{B}_\mathcal{G}^\top(\mathbf{x})\mathbf{b}_\mathcal{G}(\mathbf{x} + d\mathbf{x}) &\simeq \mathbf{B}_\mathcal{G}^\top(\mathbf{x})\Big(\mathbf{b}_\mathcal{G}(\mathbf{x}) + \nabla_\mathbf{x}\mathbf{b}_\mathcal{G}(\mathbf{x})d\mathbf{x}\Big), \\ &= \mathbf{B}_\mathcal{G}^\top(\mathbf{x})\nabla_\mathbf{x}\mathbf{b}_\mathcal{G}(\mathbf{x})d\mathbf{x}. \end{aligned} \tag{2.16}$$

Hence, according to (2.10), there exists $\boldsymbol{\delta} \in \bar{\mathcal{I}}$ such that

$$\mathbf{B}_\mathcal{G}^\top(\mathbf{x})\mathbf{e}_\mathcal{G}(\mathbf{x}) \simeq -\mathbf{B}_\mathcal{G}^\top(\mathbf{x})^\top\nabla_\mathbf{x}\mathbf{b}_\mathcal{G}(\mathbf{x})\mathbf{D}(\mathbf{q})\mathbf{S}\boldsymbol{\delta}. \tag{2.17}$$

On the other hand, exploiting the chain rule, the relation (2.10) and the bearing matrix definition, one can observe that

$$\nabla_\mathbf{x}\mathbf{b}_\mathcal{G}(\mathbf{x})\mathbf{D}(\mathbf{q})\mathbf{S} = \mathbf{B}_\mathcal{G}(\mathbf{x}), \tag{2.18}$$

leading to the conclusion that the condition $\dot{\mathrm{V}}(\mathbf{e}_\mathcal{G}(\mathbf{x})) = 0$ implies $\mathbf{B}_\mathcal{G}^\top(\mathbf{x})\mathbf{B}_\mathcal{G}(\mathbf{x})\boldsymbol{\delta} = \mathbf{0}_c$. In the light of this fact, given that $\ker(\mathbf{B}_\mathcal{G}^\top(\mathbf{x})\mathbf{B}_\mathcal{G}(\mathbf{x})) = \ker(\mathbf{B}_\mathcal{G}(\mathbf{x}))$, the elements in the set $\mathcal{Z}$ are associated to $\boldsymbol{\delta} \in \ker(\mathbf{B}_\mathcal{G}(\mathbf{x}))$. Hence, $\mathbf{e}_\mathcal{G}(\mathbf{x}) = \mathbf{b}_\mathcal{G}(\mathbf{x}) - \Big(\mathbf{b}_\mathcal{G}(\mathbf{x}) + \mathbf{B}_\mathcal{G}(\mathbf{x})\boldsymbol{\delta}\Big) = \mathbf{0}_{3m}$ belongs to $\mathcal{Z}$. □

Given these premises, it is then possible to prove that the infinitesimal rigidity property introduced in definition 2.12 is a sufficient condition for the solution of problem 2.1.

Proposition 2.7. *Consider a desired formation shape defined by* $\mathbf{b}_\mathcal{K}^* \in \mathbb{S}^{2^{n(n-1)}}$. *For any IBR generalized framework* $(\mathcal{G}, \mathbf{x}, \mathcal{H})$ *whose corresponding configuration* $\mathbf{x}$ *is in the neighborhood* $\mathcal{U}(\mathbf{x}^*)$ *of* $\mathbf{x}^*$ *such that* $\mathbf{b}_\mathcal{K}^* = \mathbf{b}_\mathcal{K}(\mathbf{x}^*)$, *the control law* (2.13) *solves the formation stabilization problem.*

Proof. For any IBR generalized framework $(\mathcal{G}, \mathbf{x}, \mathcal{H})$ with $\mathbf{x} \in \mathcal{U}(\mathbf{x}^*)$, $\mathbf{b}_\mathcal{G}(\mathbf{x}) = \mathbf{b}_\mathcal{G}(\mathbf{x}^*)$ implies $\mathbf{b}_\mathcal{K}(\mathbf{x}) = \mathbf{b}_\mathcal{K}(\mathbf{x}^*)$, and viceversa (Michieletto et al., 2021). Thus the shape error (2.11) asymptotically converges to zero as long as the

bearing error (2.12) asymptotically converges to zero. This concludes the proof in the light of proposition 2.6. □

Starting from the expression of a single bearing measurement, it is possible to show that the proposed control law is also distributed. First of all, the time derivative of eq. (2.8) is

$$\begin{aligned}
\dot{\mathbf{b}}_{ij} &= \frac{d}{dt}\left(\mathbf{R}_i^\top \frac{\mathbf{p}_j - \mathbf{p}_i}{\left\|\mathbf{p}_j - \mathbf{p}_i\right\|}\right) \\
&= \mathbf{R}_i^\top \frac{d}{dt}\left(\frac{\mathbf{p}_j - \mathbf{p}_i}{\left\|\mathbf{p}_j - \mathbf{p}_i\right\|}\right) + \frac{d}{dt}\left(\mathbf{R}_i^\top\right)\frac{\mathbf{p}_j - \mathbf{p}_i}{\left\|\mathbf{p}_j - \mathbf{p}_i\right\|} && \text{(2.19a)} \\
&= \frac{1}{d_{ij}}\mathbf{P}(\mathbf{b}_{ij})(\mathbf{R}_i^\top \mathbf{R}_j \mathbf{v}_j - \mathbf{v}_i) + \left[\mathbf{b}_{ij}\right]_\times \boldsymbol{\omega}_i && \text{(2.19b)} \\
&= \frac{1}{d_{ij}}\mathbf{P}(\mathbf{b}_{ij})(\mathbf{R}_i^\top \mathbf{R}_j \mathbf{S}_{p,j}\boldsymbol{\delta}_{p,j} - \mathbf{S}_{p,i}\boldsymbol{\delta}_{p,i}) + \left[\mathbf{b}_{ij}\right]_\times \mathbf{S}_{o,i}\boldsymbol{\delta}_{o,i}
\end{aligned}$$

where $d_{ij} = \left\|\mathbf{p}_j - \mathbf{p}_i\right\|$ is the distance between the i-th and the j-th agent, $\mathbf{P}(\cdot)$ is the orthogonal projection operator:

$$\mathbf{P} : \mathbb{R}^3 \to \mathbb{R}^{3\times 3}, \quad \mathbf{x} \mapsto \mathbf{I}_3 - \frac{\mathbf{x}^\top \mathbf{x}}{\mathbf{x}\mathbf{x}^\top}, \tag{2.20}$$

and

$$[\cdot]_\times : \mathbb{R}^3 \to \mathbb{R}^{3\times 3}, \quad \mathbf{x} = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \mapsto \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix} \tag{2.21}$$

is the skew symmetric operator.

Going from eq (2.19a) to eq (2.19b) requires some additional steps. The rotational component results from:

$$\begin{aligned}
\frac{d}{dt}\left(\mathbf{R}_i^\top\right)\frac{\mathbf{p}_j - \mathbf{p}_i}{\left\|\mathbf{p}_j - \mathbf{p}_i\right\|} &= \left(\mathbf{R}_i \left[\boldsymbol{\omega}_i\right]_\times\right)^\top \bar{\mathbf{p}}_{ij} \\
&= \left[\boldsymbol{\omega}_i\right]_\times^\top \mathbf{R}_i^\top \bar{\mathbf{p}}_{ij} = -\left[\boldsymbol{\omega}_i\right]_\times \mathbf{b}_{ij} = \left[\mathbf{b}_{ij}\right]_\times \boldsymbol{\omega}_i.
\end{aligned} \tag{2.22}$$

While the translational one is

$$\begin{aligned}
\mathbf{R}_i^\top \frac{d}{dt}\left(\frac{\mathbf{p}_j-\mathbf{p}_i}{\left\|\mathbf{p}_j-\mathbf{p}_i\right\|}\right) &= \mathbf{R}_i^\top \frac{d}{dt}\frac{\mathbf{p}_{ij}}{\left\|\mathbf{p}_{ij}\right\|} = \mathbf{R}_i^\top \frac{\dot{\mathbf{p}}_{ij}\left\|\mathbf{p}_{ij}\right\| - \mathbf{p}_{ij}\frac{d}{dt}\left\|\mathbf{p}_{ij}\right\|}{\left\|\mathbf{p}_{ij}\right\|^2} \\
&= \mathbf{R}_i^\top \frac{1}{\left\|\mathbf{p}_{ij}\right\|}\left(\dot{\mathbf{p}}_{ij} - \frac{\mathbf{p}_{ij}\mathbf{p}_{ij}^\top}{\left\|\mathbf{p}_{ij}\right\|^2}\dot{\mathbf{p}}_{ij}\right) \\
&= \frac{1}{d_{ij}}\mathbf{R}_i^\top\left(\mathbf{I}_3 - \frac{\mathbf{p}_{ij}\mathbf{p}_{ij}^\top}{\left\|\mathbf{p}_{ij}\right\|^2}\right)\dot{\mathbf{p}}_{ij} \\
&= \frac{1}{d_{ij}}\mathbf{R}_i^\top\mathbf{P}(\mathbf{p}_{ij})(\mathbf{R}_j\mathbf{v}_j - \mathbf{R}_i\mathbf{v}_i) \\
&= \frac{1}{d_{ij}}\mathbf{R}_i^\top\mathbf{P}(\mathbf{p}_{ij})\mathbf{R}_i(\mathbf{R}_i^\top\mathbf{R}_j\mathbf{v}_j - \mathbf{v}_i) \\
&= \frac{1}{d_{ij}}\mathbf{P}(\mathbf{b}_{ij})(\mathbf{R}_i^\top\mathbf{R}_j\mathbf{v}_j - \mathbf{v}_i).
\end{aligned} \tag{2.23}$$

Then, the k-th row of the bearing rigidity matrix, the one associated to the measurement $\mathbf{b}_{ij}$ takes expression

$$\begin{cases}
\begin{bmatrix} \mathbf{0}_{3\times 3(i-1)} & -d_{ij}\mathbf{P}(\mathbf{b}_{ij})\mathbf{S}_{p,i} & \mathbf{0}_{3\times 3(j-i-1)} & d_{ij}\mathbf{P}(\mathbf{b}_{ij})\mathbf{R}_i^\top\mathbf{R}_j\mathbf{S}_{p,j} \\ & \mathbf{0}_{3\times 3(n-j+i-1)} & [\mathbf{b}_{ij}]_\times\mathbf{S}_{o,i} & \mathbf{0}_{3\times 3(n-i)} \end{bmatrix} & \text{if } i<j, \\
\begin{bmatrix} \mathbf{0}_{3\times 3(j-1)} & d_{ij}\mathbf{P}(\mathbf{b}_{ij})\mathbf{R}_i^\top\mathbf{R}_j\mathbf{S}_{p,j} & \mathbf{0}_{3\times 3(i-j-1)} & -d_{ij}\mathbf{P}(\mathbf{b}_{ij})\mathbf{S}_{p,i} \\ & \mathbf{0}_{3\times 3(n-1)} & [\mathbf{b}_{ij}]_\times\mathbf{S}_{o,i} & \mathbf{0}_{3\times 3(n-i)} \end{bmatrix} & \text{if } i<j.
\end{cases} \tag{2.24}$$

Denoting with $\mathcal{N}_i$ the set of neighbors of any i-th agent, $i \in \{1\dots n\}$, the control law (2.13) can be rewritten in terms of agent commands as follows, revealing its distributed nature

$$\begin{cases}
\boldsymbol{\delta}_{p,i} = -k_c\sum_{j\in\mathcal{N}_i} d_{ij}\mathbf{S}_{p,i}^\top\mathbf{P}^\top(\mathbf{b}_{ij})\mathbf{b}_{ij}^* \\
\qquad\qquad + k_c\sum_{j:i\in\mathcal{N}_j} d_{ij}\mathbf{S}_{p,i}^\top\mathbf{R}_i^\top\mathbf{R}_j\mathbf{P}^\top(\mathbf{b}_{ji})\mathbf{b}_{ji}^*, \\
\boldsymbol{\delta}_{o,i} = k_c\sum_{j\in\mathcal{N}_i}\mathbf{S}_{o,i}^\top[\mathbf{b}_{ij}]_\times^\top\mathbf{b}_{ij}^*.
\end{cases} \tag{2.25}$$

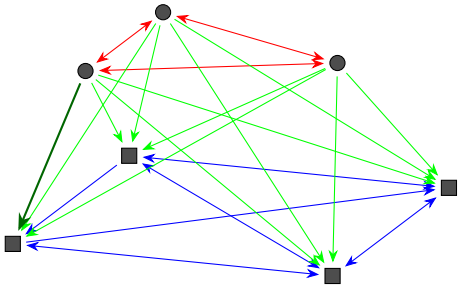


Figure 2.5: Desired formation shape - UAVs are represented by circles, UGVs by squares; red arrows and blue arrows refer to the edges of $\mathcal{G}_A$ and $\mathcal{G}_G$, respectively, the dark green edge is the one exploited by the leader-follower controller.

According to (2.25), each agent computes its commands exploiting the recorded bearing measurements ($\mathcal{N}_i$) and those gathered from the agents it is sensed by ($j : i \in \mathcal{N}_j$).[3]

2.5 Numerical results

To assess the performance of the stabilization control law (2.25), in this section this is compared with a hierarchical combination of existing rigidity based controllers designed for homogeneous formations. In doing this, the intent is both to show the effectiveness of the proposed solution and to highlight its structural simplicity mainly deriving from the general frameworks model.

2.5.1 Preliminary comparative assessment

The attention is focused on a heterogeneous formation composed of three fully-actuated UAVs and four fully-actuated UGVs. As per problem 2.1, the control goal consists in the stabilization of the given formation toward a desired shape: at the beginning all the agents are randomly placed on the (x, y)-plane of $\mathcal{F}_W$; whereas, in the final desired shape the UGVs are required to be located on the corners of a square, while the UAVs fly over them in a triangular configuration with a specific alignment between the two planar shapes, as shown in figure 2.5.

[3] Since the interaction graph is assumed to be directed, the commands computation has to be preceded with the measurement communication.

The performance of the controller (2.25) is evaluated w.r.t. an ad-hoc strategy that hierarchically solves problem 2.1 by focusing on the two homogeneous sub-formations composed of only UAVs and only UGVs.

Such a strategy envisages to control them in a separate and parallel way and to simultaneously act adjusting the relative displacement $\bar{\mathbf{p}} \in \mathbb{R}^3$ between the centers of mass, the relative orientation $\bar{\mathbf{q}} \in \mathbb{S}^3$ between the local frames of a generic couple made of a UGV and a UAV, and the whole formation scale factor $\rho \in \mathbb{R}_+$. More specifically, the rigidity based distributed controllers proposed in (Zelazo et al., 2015; Michieletto and Cenedese, 2019) are employed to steer the two sub-formations so that their components achieve the desired poses. In doing this, the aerial and ground sub-formations are modeled as (homogeneous) frameworks $(\mathcal{G}_\mathrm{A}, \mathbf{x}_\mathrm{A})$ embedded in $SE(3)^3$ and $(\mathcal{G}_\mathrm{G}, \mathbf{x}_\mathrm{G})$ embedded in $SE(2)^4$, respectively. It is possible to verify that both $(\mathcal{G}_\mathrm{A}, \mathbf{x}_\mathrm{A})$ and $(\mathcal{G}_\mathrm{G}, \mathbf{x}_\mathrm{G})$ are IBR (definition 2.12). Concurrently to the sub-formations stabilization, a leader-follower inspired strategy is employed to adjust the parameters of the whole formation.

The multi-UGV subsystem acts as reference generator for the multi-UAV one. Knowing the complete formation desired shape $\mathbf{b}^*_\mathcal{K}$ and its own state $\mathbf{x}_\mathrm{G}$, it computes the reference values for the matching parameters $\bar{\mathbf{p}}^*$, $\bar{\mathbf{q}}^*$ and ρ^*.

Those references are passed to the multi-UAV subsystem where are used to drive three independent PID controller that match the two sub-formation relative displacement, orientation and scaling:

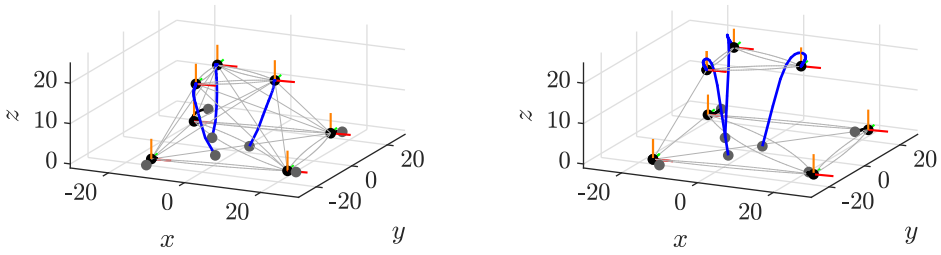
$$\bar{\mathbf{v}}_{\mathrm{A},w} = PID_v(\bar{\mathbf{p}}^* - \bar{\mathbf{p}}), \tag{2.26}$$

$$\bar{\boldsymbol{\omega}}_{\mathrm{A},w} = PID_q(\bar{\mathbf{q}}^* - \bar{\mathbf{q}}), \tag{2.27}$$

$$s_\mathrm{A} = PID_\rho(\rho^* - \rho). \tag{2.28}$$

The first PID, eq (2.26) computes the UAV-subsystem required common linear velocity in the inertial reference frame $\bar{\mathbf{v}}_{\mathrm{A},w} \in \mathbb{R}^3$ to zero the relative displacement error $\bar{\mathbf{p}}^* - \bar{\mathbf{p}}$. The second PID, eq (2.27) computes the UAV-subsystem required common angular velocity in the inertial reference frame $\bar{\boldsymbol{\omega}}_{\mathrm{A},w} \in \mathbb{R}^3$ to zero the relative orientation error $\bar{\mathbf{q}}^* - \bar{\mathbf{q}}$.[4] The third PID, eq (2.28) computes the UAV-subsystem required common scale $s_\mathrm{A} > 0$ to

[4]With abuse of notation, here we use the difference operator to indicate $(\bar{\mathbf{q}}^*)^{-1}\bar{\mathbf{q}}$.



(a) SA controller **(b)** MA controller

Figure 2.6: (a) Performance of the SA controller, (b) performance of the MA controller - the trajectories of the UAVs are depicted in blue, the ones of the UGVs in black.

zero the relative scale error $\rho^* - \rho$. Once the three corrective terms are compute, the single agent linear and angular velocities are derived:

$$\begin{aligned} \mathbf{v}_i &= \mathbf{R}_i^\top (\bar{\mathbf{v}}_{\mathrm{A},w} + s_\mathrm{A}(\mathbf{p}_i - \bar{\mathbf{p}}_\mathrm{A}) + \left[\bar{\boldsymbol{\omega}}_{\mathrm{A},w}\right]_\times (\mathbf{p}_i - \bar{\mathbf{p}}_\mathrm{A}), \\ \boldsymbol{\omega}_i &= \mathbf{R}_i^\top \bar{\boldsymbol{\omega}}_{\mathrm{A},w}, \end{aligned} \tag{2.29}$$

where $\bar{\mathbf{p}}_\mathrm{A} \in \mathbb{R}^3$ is the barycenter of the UAVs subformation. The specific structure of eqs (2.29) ensures that the leader-follower approach does not deteriorate the performances of the sub-formation stabilization controller. Indeed, $\bar{\mathbf{v}}_{\mathrm{A},w}$, $\bar{\boldsymbol{\omega}}_{\mathrm{A},w}$ and s_A are combined such that the resulting actions are trivial motions (definition 2.11).

Hereafter, the described control strategy is referred as *multi-action* (MA) controller, while the stabilization law (2.25) is indicated as *single-action* (SA) controller.

Figure 2.6a reports the trajectories followed by the agents from their initial to final positions (grey and black dots, respectively) thanks to the implementation of the SA controller (2.25). From figure 2.6b, instead, one can observe that the agents trajectories are more complicated when the MA controller is employed. This is due to the fact that the UAVs are required to simultaneously reach their desired poses and to rearrange w.r.t. the whole desired formation shape. Conversely, the SA controller aims at equally distribute the effort among all the agents (based on the sensing graph $\mathcal{G}$).

2.5.2 Monte-Carlo campaign validation

A Monte-Carlo (MC) simulative campaign has also been conducted accounting for $N = 100$ different realizations of initial conditions of the given formation stabilization problem. To approximate real-world behavior, and in accordance with the existing literature, all bearing measurements have been corrupted with additive Gaussian noise having zero mean and standard deviation $\sigma = 0.5°$. For a thorough discussion about noise generation on the unit sphere, please refer to section 3.2. In particular, two different MC tests (with N runs each) have been considered, highlighting the intrinsic trade-off between stabilization speed and control effort.

The first evaluated performance index consists in the *formation settling time* $t_s > 0$, corresponding to the average time required to sufficiently align the bearing measurements to the given, desired ones. Formally, t_s is computed as the average time required by

$$\alpha(t) = \frac{1}{m}\sum_{i=1}^{m} \arccos\left(\mathbf{b}_i^\top(\mathbf{x}(t))\mathbf{b}_i^*\right), \tag{2.30}$$

which represents the average of the unsigned angular error between the desired bearings $\mathbf{b}_i^*$ and the measured ones $\mathbf{b}_i$, to go below the threshold $\bar{\alpha} = 0.75°$. Motivated by the MC approach, the Empirical Cumulative Distribution Function (ECDF) is considered. This is defined as

$$\hat{F}_{t_s}(t) = \frac{1}{N}\sum_{k=1}^{N} \mathbb{1}_{t_s,k}(t), \tag{2.31}$$

where, for each k-th trial, the indicator function $\mathbb{1}_{t_s,k}(t)$ is equal to one when $t \geq t_s$ and zero otherwise. Conversely, the control effort is investigated through the computation of the ECDF of the input energy required to reach the settling condition. Formally, this is computed as

$$\hat{F}_{E_s}(E) = \frac{1}{N}\sum_{k=1}^{N} \mathbb{1}_{E_s,k}(E), \tag{2.32}$$

where the indicator function $\mathbb{1}_{E_s,k}(E)$ accounts for the number of trials where $E \geq E_s$ with $E_s = \int_0^{t_s} \|\boldsymbol{\delta}(s)\|\, ds$.

In the first MC test, the tunable parameters of the SA and MA controllers have been set so that the two solutions turn out to be equivalent in terms of *control effort*. The results are reported on figure 2.7a. Observe that the proposed SA controller (blue line) outperforms the MA controller (orange line) as concerns the selling time (left panel). For the sake of completeness, the settling time and control effort ECDFs of the UAVs and UGVs subformations used in the multi-action controller are highlighted. Two observations are in place.

- First, the UAVs achieve their desired poses ten times faster than the UGVs: this highlights the key role played by the underlying topology.
- Second, the gap between the settling time ECDF restricted to the ground multi-agent system stabilization and of the whole MA controller employment points out that the settling time strongly depends on the alignment between the two sub-formations.

In this direction, the slower sub-formation stabilization performance can be interpreted as a lower bound for the settling time in case of MA controller adoption.

The results of the second MC test are reported in figure 2.7b. In this case, the parameters of the SA and MA controllers are tuned so that the two approaches exhibit the same settling times. In these conditions, the SA approach requires a lower control effort (right panel) implying that the outlined formation controller is more energy-efficient as compared to the MA one but not more effective in terms of settling time (left panel).

The table 2.1 summarizes the main results, specifying the average settling times and the average settling energies for both the MC tests[5], and confirms the overall better performance for the heterogeneous SA controller.

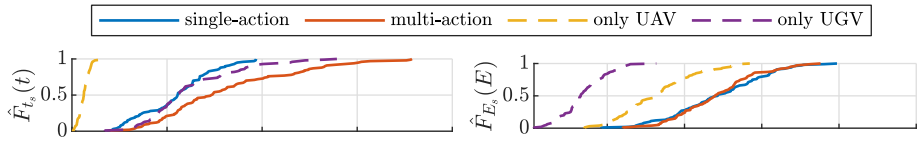
2.6 Discussion

Here, the principal aspects of the designed heterogeneous formation control are highlighted. First of all, the proposed distributed approach (2.25)

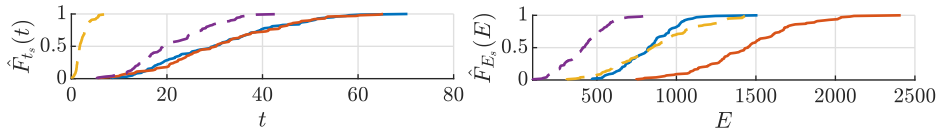
[5]These results are given only in terms of average because the wide range of MC realizations yields high values of variance over the whole spectrum of simulations.

		t_s	E_s
same control effort	**SA**	22.1s	1220
	MA	32.6s	1212
same settling time	**SA**	30.9s	829
	MA	31.5s	1447

Table 2.1: Results of MC tests - average value of settling time and control effort.



(a) Matched control effort E_s



(b) Matched settling time tr_s

Figure 2.7: Results of MC tests - the figures shown the trend of the control input ECDFs (2.31) (left column) and settling time ECDFs (2.32) (right columns) in correspondence to the two MC test. (a) matched control effort scenario. (b) matched settling time scenario. Solid blue lines represent the single-action controller, solid red lines represent the multi-action controller, dashed yellow and purple lines represent the individual UAVs and UGVs subformations considered in the multi-action controller, respectively.

depends on relative bearing information but also also on the inter-agent distances and relative orientations. Nonetheless, these last can be estimated through distributed consensus algorithms without employing additional sensors (Kia et al., 2019). Then, contrarily to most of the existing formation stabilization schemes, the designed control law (2.13) is not a classical gradient descent procedure, distinguishing w.r.t. the standard bearing based rigid formation controllers (Zelazo et al., 2015; Schiano et al., 2016; Michieletto and Cenedese, 2019). However, taking into account (2.18), one can observe that the proposed controller can be interpreted as a gradient descent solution followed by a re-projection operation necessary to guarantee that the configuration derived from the application of (2.13) is in $\mathbb{R}^{3n} \times \mathbb{S}^{3^n}$. Such a re-projection is mainly due to the adopted rotation representation. Finally, the proposed controller presents a single parameter to tune, having a simple structure if compared with the hierarchical MA approach introduced in section 2.5. This, indeed, results to be structurally more complex involving multiple controllers operating at different levels (each of them, then, involves one or more parameters to tune) and more demanding in terms of agents interactions (sub-formations are required to communicate in a bilateral manner).

In this chapter the problem of stabilizing an heterogeneous formation resorting on only inter-agent bearing measurements has been addressed in the mathematical framework of the bearing rigidity theory. Although the stability of the proposed controller has been proven for noiseless scenario and the robustness to disturbances on the measurements has been investigated by the MC simulation campaign, the topic of noisy rigidity theory and noisy bearing based approaches remains open and of interest, as every real life applications must deal with non-perfect sensing apparatus. In the next chapter addresses such issue from the point of view of the noisy bearing based localization task, in which a team of agents must localize an uncooperative target by measuring it perturbed bearing vectors.

3 Bearing-based target localization

In this chapter bearing only measurements are adopted by a group of seeker to estimate the position of an uncooperative target. The seeker actuation capabilities are exploited by the active-sense paradigm to steer the agent and reduce the estimation uncertainty.

3.1 Introduction

The target localization task involves the estimation of the position of an object, a person, or an event based on the data collected by some sensing agents while surveying a specific area of interest. This issue spans multiple application scenarios, including environmental monitoring (e.g., radiation sources identification and forest fires detection), territorial surveillance (e.g., intruders and/or potential threats recognition), and search-and-rescue missions (e.g., localization of victims of catastrophic events)(Robin and Lacroix, 2016).

Due to its widespread occurrence, the target localization problem remains an intriguing research subject within the field of control. The current solving methods vary depending on several application factors such as the number of target and seeker agents, their ability to act (distinguishing between static and dynamic agents), and the type of involved sensing data, including bearing and/or distance measurements. In this chapter, the attention is focused on the localization of a single static target through a group of dynamic seeker agents having bearing sensing capabilities. In particular, the task is addressed by accounting for the twofold aspect of the target position estimate and the seeker agents control.

Related works - The problem of determining the unknown position of a certain object by exploiting a given set of bearing measurements dates back to (Stansfield, 1947) where the expected root-mean-square error on the target position

estimate is adopted as the index for measuring the estimation reliability. Later, multiple works cope with the bearing-based target localization task. Without claiming to be exhaustive, (Kaplan et al., 2001) and (Bishop et al., 2009) are mentioned, which specifically focus on the measurements features. Indeed, in the former, a Maximum Likelihood approach is exploited to face the non-linearities affecting the measurement noise, while in the latter the task is formalized and addressed in a constrained geometric optimization framework resting on the relative direction of the bearings. More recently, similar measurement-aware methods have been proposed for the localization of multiple static targets by means of a multi-agent formation in the context of obstacle avoidance task (Chun and Tian, 2020), and for the localization of a single dynamic target moving on a plane and tracked by a group of seeker agents (Dou et al., 2020; Chen et al., 2023). In these works, the seeker agents are steered along continuous predetermined (elliptical and circular, respectively) trajectories, independently of the target localization uncertainty. Conversely, trajectory optimization is studied in (He et al., 2019) where a single seeker agent is required to track a target having unknown dynamics. Along the same line, in (Xu, 2020) the trajectories of multiple sensing agents are designed in order to minimize the mean-squared error on the estimation of a single target position. In that work, the exploited measurements are both angle and time of arrival.

Contributions - As in (Xu, 2020), an estimation and control framework to displace a group of seeker agents in order to optimize the accuracy of the estimated position of a target is proposed. On the other hand, the focus is placed on bearing measurements. More in detail, the main contributions are twofold:

- on the estimation side, a procedure exploiting the popular weighted least square approach and entailing the iterative computation of the covariance matrix on the estimated target position is proposed;
- on the control side, a regulation law for the seeker agents based on the active-sensing paradigm (Varotto et al., 2021) is advised.

The outlined framework is validated by the numerical results of an extensive Monte Carlo simulation campaign. The claim here is that the goodness of the estimation outcomes is also guaranteed by an ad-hoc initialization procedure for the proposed localization algorithm.

Chapter Structure - The rest of the chapter is organized as follows. The bearing-based target localization task is detailed in section 3.2 with special regard to the measurements modeling. In section 3.3 the algorithm to estimate the target position in 3D space is illustrated. Then, in section 3.4, the control strategy to reduce the uncertainty on the estimated target position is investigated. The performance of the designed estimation and control framework is discussed in section 3.5.

3.2 Target localization task

In this chapter, the *bearing-based target localization task* consisting in the estimation of the position of an unknown target through the exploitation of a set of noisy bearing measurements collected by a group of seeker agents is investigated.

The target position $\mathbf{p}_t \in \mathbb{R}^3$ is assumed to be fixed over time, while any i-th seeker agent, $i \in \mathcal{S} = \{1 \dots n\}, n \geq 2$, is modeled as a particle point having single-integrator dynamics. Denoting with $\mathbf{p}_i(t) \in \mathbb{R}^3$ the time-varying position of the i-th seeker agent, it thus holds that

$$\dot{\mathbf{p}}_i(t) = \mathbf{u}_i(t), \tag{3.1}$$

with $\mathbf{u}_i(t) \in \mathbb{R}^3$ being the control input chosen in order to guarantee that $\mathbf{p}_i(t) \neq \mathbf{p}_t, \forall t$. This assumption makes the considered seekers *fully*-actuated platforms in $\mathbb{R}^3$. Any i-th seeker agent is supposed to know its position with a certain level of accuracy, for example by adopting RTK positioning sensors. To ease the notation, hereafter, the time dependency is dropped when not strictly necessary.

Any i-th seeker agent is also supposed to be capable of acquiring a noisy measurement of the bearing with respect to the target with sampling period $T > 0$. Specifically, the bearing $\mathbf{b}_{it} \in \mathbb{S}^2$ is defined as

$$\mathbf{b}_{it} = \mathbf{f}_i(\mathbf{p}_t) = \frac{\mathbf{p}_t - \mathbf{p}_i}{\|\mathbf{p}_t - \mathbf{p}_i\|}, \tag{3.2}$$

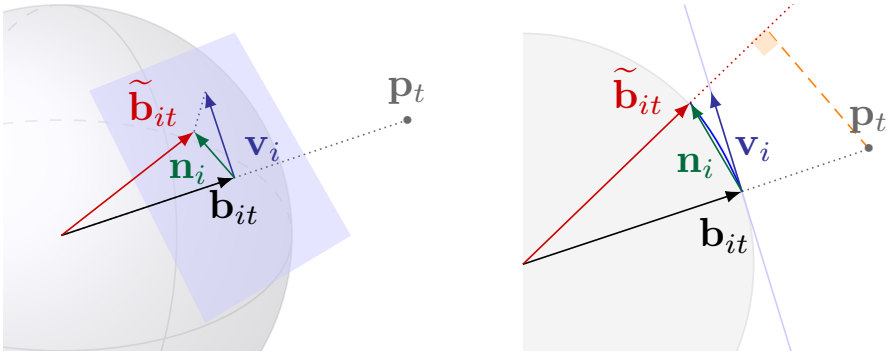


Figure 3.1: 3D and 2D graphical description of the relation between bearing vector $\mathbf{b}_{it}$ and its noisy measurement $\widetilde{\mathbf{b}}_{it}$.

where $\mathbf{f}_i(\cdot) : \mathbb{R}^3 \to \mathbb{S}^2$ is the bearing rigidity function associated to the i-th seeker agent embedded in $\mathcal{D}_i = \mathbb{R}^3$.[1] The measurement recorded at any $t = kT$, $k \in \mathbb{N}$, is then modeled as

$$\widetilde{\mathbf{b}}_{it} = \exp_{\mathbf{b}_{it}} \mathbf{v}_i = \cos(\|\mathbf{v}_i\|)\mathbf{b}_{it} + \frac{\sin(\|\mathbf{v}_i\|)}{\|\mathbf{v}_i\|}\mathbf{v}_i \tag{3.3}$$

where, for any vector $\mathbf{b} \in \mathbb{S}^2$, the exponential map $\exp_{\mathbf{b}}(\cdot) : T_{\mathbf{b}}\mathbb{S}^2 \to \mathbb{S}^2$ projects onto $\mathbb{S}^2$ the vectors belonging to the tangent space $T_{\mathbf{b}}\mathbb{S}^2$ of $\mathbf{b}$. In (3.3), the vector $\mathbf{v}_i \in T_{\mathbf{b}_{it}}\mathbb{S}^2$ represents a perturbation that acts orthogonally to $\mathbf{b}_{it}$ and takes into account the sensor noise and the measurement uncertainties. In the following, this is modeled as a normally distributed Gaussian random vector having zero mean and positive semi-definite covariance matrix $\mathbf{\Sigma}_{\mathbf{v}_i} \in \mathbb{R}^{3\times3}$. In particular, since $\mathbf{v}_i \in T_{\mathbf{b}_{it}}\mathbb{S}^2 \subset \mathbb{R}^3$, the covariance matrix is required to have a zero eigenvalue corresponding to the eigenvector $\mathbf{b}_{it}$. To ensure this property, $\mathbf{\Sigma}_{\mathbf{v}_i}$ is defined as

$$\mathbf{\Sigma}_{\mathbf{v}_i} = \mathbf{P}(\mathbf{b}_{it})\mathbf{\Sigma}_{\bar{\mathbf{v}}_i}\mathbf{P}(\mathbf{b}_{it})^\top, \tag{3.4}$$

where $\mathbf{\Sigma}_{\bar{\mathbf{v}}_i} \in \mathbb{R}^{3\times3}$ is a suitably selected positive definite matrix . A graphical representation of the bearing measurement (3.3) is given in figure 3.1, highlighting the relation between the bearing $\mathbf{b}_{it}$ and the perturbation vector $\mathbf{v}_i$.

Finally, the seeker agents are assumed to constitute an ideal formation in terms of communication, meaning that each group component can exchange

[1] Here the seekers are modeled as points in the 3D space and therefore they do not have any frame attached to them. The bearing measurements are acquired in the common frame $\mathcal{F}_W$.

data with all the others without delays, interference, and/or information corruption and degradation. Accounting for the graph-based representation of the formations, the seeker agents group is thus modeled as an homogeneous framework embedded in $\bar{\mathcal{D}} = \mathbb{R}^{3n}$ with a complete indirect communication graph.

The latter assumption will play a key role in the formulation of the target position estimator of section 3.3 and the seeker controllers of section 3.4. Nevertheless, keep in mind that, contrarily to the common practice used in standard bearing rigidity theory, here the sensing graph and the communication graph of the seeker formation are completely different. The former has no edges, as the seekers do not measure each other. The latter is complete.

A note about perturbation of bearing vectors - Bearing vectors on the three dimensional sphere $\mathbb{S}^2$ are implicitly embedded in $\mathbb{R}^3$ as $\mathbb{S}^2 \subset \mathbb{R}^3$. Therefore, the trivial solution to perturb a given bearing vector $\mathbf{b}_0 \in \mathbb{S}^2$ is to add a random vector $\mathbf{n} \in \mathbb{R}^3$ and the normalize the result,

$$\begin{aligned} \mathbf{b}'_1 &= \mathbf{b}_0 + \mathbf{n} \in \mathbb{R}^3, \\ \mathbf{b}_1 &= \frac{\mathbf{b}'}{\left\|\mathbf{b}'\right\|} \in \mathbb{S}^2. \end{aligned} \tag{3.5}$$

While $\mathbf{b}'_1$ maintains all the statistical properties of $\mathbf{n}$, the following normalization phase introduces non-linearities in $\mathbf{b}_1$ and very little can be said about its statical distribution. A slightly better approach shapes the initial random vector such that it belongs to the tangent plane of $\mathbf{b}_0$, then it adds it to the original bearing and finally it normalizes the results,

$$\begin{aligned} \mathbf{b}'_2 &= \mathbf{b}_0 + \mathbf{P}(\mathbf{b}_0)\mathbf{n} \in \mathbb{R}^3, \\ \mathbf{b}_2 &= \frac{\mathbf{b}'}{\left\|\mathbf{b}'\right\|} \in \mathbb{S}^2. \end{aligned} \tag{3.6}$$

This approach lets $\mathbf{b}'_2$ inherit the statistical properties of $\mathbf{n}$ and at the same time constrains the perturbation direction to belong to the tangent space $\mathbf{P}(\mathbf{b}_0)$. The normalization still remains. The last approach is the one adopted in eq (3.3). The random vector $\mathbf{n}$ is drawn from the tangent space of $\mathbf{b}_0$ and

then the manifold exponential operator is used to project it back to the $\mathbb{S}^2$ sphere,

$$\begin{aligned} \mathbf{n}' &= \mathbf{P}(\mathbf{b}_0)\mathbf{n} \in \mathbb{R}^3, \\ \mathbf{b}_3 &= \exp_{\mathbf{b}_0} \mathbf{n}' \in \mathbb{S}^2 \end{aligned} \tag{3.7}$$

the exponential map introduces a major useful property: the length l of the arc identified by the two bearings $\mathbf{b}_0$ and $\mathbf{b}_3$ is equal to the norm of the perturbation $\mathbf{n}'$. Consequently, if $\mathbf{n}$ is drawn from a normal distribution, the distribution of the angle between the $\mathbf{b}_0$ and $\mathbf{b}_3$, namely $\theta = \arccos(\mathbf{b}_0^\top \mathbf{b}_3) = l$, is distributed according to a Generalized Chi Square distribution.

3.3 WLS-based target position estimation

The noisy bearing measurements introduced in section 3.2 can be interpreted according to the popular additive noise model. Indeed, the expression (3.3) of $\widetilde{\mathbf{b}}_{it}$ can be rewritten as

$$\widetilde{\mathbf{b}}_{it} = \mathbf{b}_{it} + \mathbf{n}_i = \mathbf{f}_i(\mathbf{p}_t) + \mathbf{n}_i, \tag{3.8}$$

where $\mathbf{n}_i \in \mathbb{R}^3, i \in \mathcal{S}$, is a function of both the bearing $\mathbf{b}_{it} \in \mathbb{S}^2$ and the perturbation vector $\mathbf{v}_i \in \mathbb{S}^2$, namely

$$\mathbf{n}_i = (\cos(\|\mathbf{v}_i\|) - 1)\, \mathbf{b}_{it} + \frac{\sin(\|\mathbf{v}_i\|)}{\|\mathbf{v}_i\|}\mathbf{v}_i. \tag{3.9}$$

Note that figure 3.1 clarifies also the relation between the vectors $\mathbf{v}_i \in T_{\mathbf{b}_{it}}\mathbb{S}^2 \subset \mathbb{R}^3$ and $\mathbf{n}_i \in \mathbb{R}^3$. All the seeker agents measurements can then be collected in the so-called *bearing measurements vector* $\widetilde{\mathbf{b}}_S \in \mathbb{S}^{2^n}$, defined as

$$\widetilde{\mathbf{b}}_S = \begin{bmatrix} \widetilde{\mathbf{b}}_1 \\ \vdots \\ \widetilde{\mathbf{b}}_n \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1(\mathbf{p}_t) \\ \vdots \\ \mathbf{f}_n(\mathbf{p}_t) \end{bmatrix} + \begin{bmatrix} \mathbf{n}_1 \\ \vdots \\ \mathbf{n}_n \end{bmatrix} = \mathbf{f}_S(\mathbf{p}_t) + \mathbf{n}_S, \tag{3.10}$$

where $\mathbf{f}_S(\cdot) : \mathbb{R}^{3n} \to \mathbb{S}^{2^n}$ is the standard bearing function when the $\mathbb{R}^3$ domain is taken into account as in section 2.2.

Since any $\mathbf{v}_i \in \mathbb{R}^3$ in (3.3) is modeled as a normally distributed Gaussian random vector, $\mathbf{n}_S \in \mathbb{R}^3$ in (3.10) is supposed to approximates a Gaussian random vector with i.i.d. components. The soundness of this assumption is

confirmed by the numerical results in section 3.5.1. In light of this fact, the bearing-based target localization task is tackled by adopting the well-stated weighted least square (WLS) approach. In doing this, the target position estimate $\hat{\mathbf{p}}_t \in \mathbb{R}^3$ is determined as

$$\hat{\mathbf{p}}_t = \underset{\mathbf{p}\in\mathbb{R}^3}{\operatorname{argmin}} \left\| \widetilde{\mathbf{b}}_S - \mathbf{f}_S(\mathbf{p}) \right\|_{\mathbf{W}}^2, \tag{3.11}$$

where $\mathbf{W} \in \mathbb{R}^{3n\times 3n}$ is a positive definite weight matrix whose selection is discussed in the following.

3.3.1 Iterative WLS solution

The WLS problem (3.11) can be solved via the iterative procedure described in algorithm 1 wherein, at each iteration, it is computed the solution of the linearized WLS problem based on the linearization of the bearing rigidity function around the current target position estimate (lines 3-5).

Remarkably, the Jacobian matrix $\mathbf{F}_S(\cdot) \in \mathbb{R}^{3n\times 3}$ of the function $\mathbf{f}_S(\cdot)$ can be computed in closed form and it results from the stacking of scaled projection operators, namely

$$\mathbf{F}_S(\hat{\mathbf{p}}_t) = \left.\frac{\partial \mathbf{f}_S(\mathbf{p})}{\partial \mathbf{p}}\right|_{\mathbf{p}=\hat{\mathbf{p}}_t} = \begin{bmatrix} \frac{1}{\|\hat{\mathbf{p}}_t - \mathbf{p}_1\|}\mathbf{P}(\mathbf{f}_1(\hat{\mathbf{p}}_t)) \\ \vdots \\ \frac{1}{\|\hat{\mathbf{p}}_t - \mathbf{p}_n\|}\mathbf{P}(\mathbf{f}_n(\hat{\mathbf{p}}_t)) \end{bmatrix}. \tag{3.12}$$

It can be observed that (3.12) corresponds to the bearing rigidity matrix associated with the multi-agent formation embedded in $\mathbb{R}^3$ and composed by n seeker agents and the target whose sensing capabilities are modeled by a directed star graph having the target as the center of the ingoing edges representing the bearing measurements acquired by the seekers.

The algorithm stops when $\|\Delta\hat{\mathbf{p}}_t\| \in \mathbb{R}^+$, quantifying the innovation on the target position estimate with respect to the previous iteration, is below a given threshold $\epsilon \in \mathbb{R}^+$ (line 6).

At this point the need of a fully connected graph should be clear. Algorithm 1 has a centralized structure and requires to collect data from all the agent

Algorithm 1: Iterative least square algorithm.

Data: $\hat{\mathbf{p}}_{t,0}$, $\widetilde{\mathbf{b}}_S$, ϵ, $\mathbf{W}$
Result: $\hat{\mathbf{p}}_t$

1 $\hat{\mathbf{p}}_t \leftarrow \hat{\mathbf{p}}_{t,0}$;
2 **do**
3 $\quad \Delta\widetilde{\mathbf{b}} \leftarrow \widetilde{\mathbf{b}}_S - \mathbf{F}_S(\hat{\mathbf{p}}_t)$;
4 $\quad \Delta\hat{\mathbf{p}}_t \leftarrow \left(\mathbf{F}_S(\hat{\mathbf{p}}_t)^\top \mathbf{W} \mathbf{F}_S(\hat{\mathbf{p}}_t)\right)^{-1} \mathbf{F}_S(\hat{\mathbf{p}}_t)^\top \mathbf{W} \Delta\widetilde{\mathbf{b}}$;
5 $\quad \hat{\mathbf{p}}_t \leftarrow \hat{\mathbf{p}}_t + \Delta\hat{\mathbf{p}}_t$;
6 **while** $\|\Delta\hat{\mathbf{p}}_t\| > \epsilon$;

at each iteration. The easiest way to achieve this requirement is to impose the use of fully connected communication graph. While this condition is definitively sufficient, it is necessary. Future works could consider using partially connected communication graph among the seekers and exploit distributed message passing techniques to share the data among the agents.

Remark 3.1. In the common bearing rigidity framework, the capability of recovering the formation shape depends upon the bearing rigidity condition. This is not the case as it is assumed that the seekers know in advance their position. In this way the bearing based formation localization problem boils down to a single agent (the target) localization task which can be theoretically solved by two non-collinear bearing measurements.

3.3.2 Algorithm initialization

The goodness of the target position estimate outputted from algorithm 1 depends on its initialization (line 1).

If any prior information is available, e.g., in correspondence to the first measurements acquisition, a suitable initial estimate guess $\hat{\mathbf{p}}_{t,0} \in \mathbb{R}^3$ is the point that minimizes the distance from the lines passing through $\mathbf{p}_i$ and directed along $\widetilde{\mathbf{b}}_{it}, \forall i \in S$. Introducing the seeker agents position vector $\mathbf{p}_S = \left[\mathbf{p}_1^\top \dots \mathbf{p}_n^\top\right]^\top \in \mathbb{R}^{3n}$, it results that

$$\hat{\mathbf{p}}_{t,0} = \underset{\mathbf{p}\in\mathbb{R}^3}{\operatorname{argmin}}\, d(\mathbf{p} \mid \mathbf{p}_S, \widetilde{\mathbf{b}}_S), \tag{3.13}$$

where $d(\cdot \mid \mathbf{p}_S, \widetilde{\mathbf{b}}_S) : \mathbb{R}^3 \to \mathbb{R}$ is the cumulative distance function given the seeker agents position and bearing measurements, namely it is

$$d(\mathbf{p} \mid \mathbf{p}_S, \widetilde{\mathbf{b}}_S) = \frac{1}{2} \sum_{i=1}^{n} (\mathbf{p}_i - \mathbf{p})^\top \mathbf{P}(\widetilde{\mathbf{b}}_{it})(\mathbf{p}_i - \mathbf{p}). \tag{3.14}$$

Observe that the function (3.14) is convex and quadratic with respect to the argument $\mathbf{p}$, thus the global minimum for the problem (3.13) can be computed as

$$\hat{\mathbf{p}}_{t,0} = \left(\mathbf{P}_S(\widetilde{\mathbf{b}}_S)^\top \mathbf{P}_S(\widetilde{\mathbf{b}}_S)\right)^{-1} \mathbf{P}_S(\widetilde{\mathbf{b}}_S)\mathbf{p}_S, \tag{3.15}$$

where $\mathbf{P}_S(\cdot) : \mathbb{S}^{2^n} \to \mathbb{R}^{3n \times 3}$ is the stacked projectors operator, so that it is $\mathbf{P}_S(\widetilde{\mathbf{b}}_S) = \begin{bmatrix} \mathbf{P}(\widetilde{\mathbf{b}}_{1t})^\top & \dots & \mathbf{P}(\widetilde{\mathbf{b}}_{nt})^\top \end{bmatrix}^\top$. Thus, exploiting the idempotent property of $\mathbf{P}_S(\cdot)$, the vector $\hat{\mathbf{p}}_{t,0}$ can be retrieved as

$$\hat{\mathbf{p}}_{t,0} = \mathbf{A}^{-1}\mathbf{y} \tag{3.16}$$

with matrix $\mathbf{A} \in \mathbb{R}^{3\times3}$ and vector $\mathbf{y} \in \mathbb{R}^3$ defined as

$$\mathbf{A} = \sum_{i=1}^{n} \mathbf{P}(\widetilde{\mathbf{b}}_{it}) \quad \text{and} \quad \mathbf{y} = \sum_{i=1}^{n} \mathbf{P}(\widetilde{\mathbf{b}}_{it})\mathbf{p}_i. \tag{3.17}$$

Specifically, the existence of the solution (3.16) is guaranteed by the full rankness of the matrix $\mathbf{A}$: this condition holds when at least two bearing measurements are not collinear.

When a target position estimation is already available, e.g., based on some previous measurements set, the algorithm 1 can be initialized by exploiting this information. In this case, it is, e.g., $\hat{\mathbf{p}}_{t,0}(t+T) = \hat{\mathbf{p}}_t(t)$.

Remark 3.2. Addressing problem (3.13) implies the computation of a target position estimation which turns out to be a suboptimal solution for the target localization task formalized as in (3.11). This is due to the fact that the minimization problem (3.11) involves the notion of chordal distance over $\mathbb{S}^2$, whereas (3.13) is based on the line distance. Accounting for figure 3.1, (3.16) is computed by minimizing the (orange) projection along the direction of $\widetilde{\mathbf{b}}_{it}$, whereas the solution of (3.11) is determined by minimizing the norm of the (green) vector $\mathbf{n}_i$.

3.3.3 Localization uncertainty

The covariance matrix associated to the target position estimate $\hat{\mathbf{p}}_t$ outputted from algorithm 1 can be approximated using the linear regression theory. Accordingly, it holds that

$$\begin{aligned}\boldsymbol{\Sigma}_{\hat{\mathbf{p}}_t} = &\left((\mathbf{F}_S(\hat{\mathbf{p}}_t)^\top \mathbf{W} \mathbf{F}_S(\hat{\mathbf{p}}_t))^{-1} \mathbf{F}_S(\hat{\mathbf{p}}_t)^\top \mathbf{W}\right) \boldsymbol{\Sigma}_{\mathbf{n}_S} \\ &\left((\mathbf{F}_S(\hat{\mathbf{p}}_t)^\top \mathbf{W} \mathbf{F}_S(\hat{\mathbf{p}}_t))^{-1} \mathbf{F}_S(\hat{\mathbf{p}}_t)^\top \mathbf{W}\right)^\top,\end{aligned} \tag{3.18}$$

where $\boldsymbol{\Sigma}_{\mathbf{n}_S} \in \mathbb{R}^{3n \times 3n}$ is the positive definite covariance matrix of the noise vector $\mathbf{n}_S$. The validity of (3.18) is confirmed by the numerical results in section 3.5.1.

In the particular case wherein $\boldsymbol{\Sigma}_{\mathbf{n}_S}$ is known with a certain level of confidence, it is suitable to select the weight matrix as $\mathbf{W} = \boldsymbol{\Sigma}_{\mathbf{n}_S}^{-1}$ in order to have

$$\boldsymbol{\Sigma}_{\hat{\mathbf{p}}_t} = \left(\mathbf{F}_S(\hat{\mathbf{p}}_t)^\top \boldsymbol{\Sigma}_{\mathbf{n}_S}^{-1} \mathbf{F}_S(\hat{\mathbf{p}}_t)\right)^{-1}. \tag{3.19}$$

In this way, the target position estimate covariance matrix $\boldsymbol{\Sigma}_{\hat{\mathbf{p}}_t}$ results to be dependent on the noise covariance matrix $\boldsymbol{\Sigma}_{\mathbf{n}_S}$ and on the matrix $\mathbf{F}_S(\hat{\mathbf{p}}_t)$ which summarizes the relative position between the seeker agents and the target based on its current position estimate. This fact supports the employment of an active-sensing approach in the design of the seeker agents controller: the idea is to steer the seeker agents in order to minimize the uncertainty on the estimated target position. Finally, note that the eq. (3.19) is the inverse of the *symmetric* bearing rigidity matrix weighted by $\boldsymbol{\Sigma}_{\mathbf{n}_S}^{-1}$.

3.4 Active sense control approach

To develop an ad-hoc active-sensing control approach for the seeker agents group, the fact that the covariance matrix $\boldsymbol{\Sigma}_{\hat{\mathbf{p}}_t}$ is a function of the seeker agents position $\mathbf{p}_S$ (i.e., $\boldsymbol{\Sigma}_{\hat{\mathbf{p}}_t} = \boldsymbol{\Sigma}_{\hat{\mathbf{p}}_t}(\mathbf{p}_S)$) and its determinant constitutes a measure of the volume of the uncertainty ellipsoid associated to the target position estimate $\hat{\mathbf{p}}_t$ is exploited. Thus, given that $\det(\mathbf{M}^{-1}) = \det(\mathbf{M})^{-1}$

for any nonsingular matrix $\mathbf{M}$, an active-sense control law based on the maximization of the estimation reward

$$J(\mathbf{p}_S) = \det\left((\boldsymbol{\Sigma}_{\hat{\mathbf{p}}_t}(\mathbf{p}_S))^{-1}\right) \tag{3.20a}$$

$$= \det(\mathbf{F}_S(\hat{\mathbf{p}}_t)^\top \Sigma_{\mathbf{n}_S}^{-1} \mathbf{F}_S(\hat{\mathbf{p}}_t)), \tag{3.20b}$$

relying also upon the weight matrix selection $\mathbf{W} = \boldsymbol{\Sigma}_{\mathbf{n}_S}^{-1}$, is designed.

The cost function (3.20) is unbounded from above and it has a singularity when $\mathbf{p}_i \to \hat{\mathbf{p}}_t$ for at least a seeker agent (in this case, it holds that $J(\mathbf{p}_S) \to +\infty$). On the other hand, the uncertainty on the target position estimate is intuitively reduced when the seeker agents approach the target itself.

Now, under the requirement of designing the control input $\mathbf{u}_i$ of any i-th seeker agent in order to guarantee that $\mathbf{p}_i \neq \mathbf{p}_t$, the minimization of the estimation reward (3.20) is addressed in a constrained framework, imposing all seeker agents to move while maintaining their estimated distance with respect to the target. In light of the assumed single integrator dynamic model (3.1), the given constraint can be fulfilled by adopting a projected gradient ascend control law. Formally, for any i-th seeker agent, the control input is computed as

$$\mathbf{u}_i = k\, \mathbf{P}(\hat{\mathbf{b}}_{it}) \mathbf{J}(\mathbf{p}_S) \quad \text{with} \tag{3.21a}$$

$$\hat{\mathbf{b}}_{it} = \frac{\hat{\mathbf{p}}_t - \mathbf{p}_i}{\|\hat{\mathbf{p}}_t - \mathbf{p}_i\|}, \quad \mathbf{J}(\mathbf{p}_S) = \frac{\partial J(\mathbf{p}_S)}{\partial \mathbf{p}_i}, \tag{3.21b}$$

where $k \in \mathbb{R}^+$ is a tunable gain parameter and $\hat{\mathbf{b}}_{it} \in \mathbb{S}^2$ in (3.21) represents the estimated bearing of the i-th seeker agent with respect to the target estimated position $\hat{\mathbf{p}}_t$. Thus, the projector operator $\mathbf{P}(\hat{\mathbf{b}}_{it}) \in \mathbb{R}^{3\times 3}$ ensures that the i-th seeker agent moves on the sphere centered in the estimated target position with radius $\hat{d}_{it} = \|\hat{\mathbf{p}}_t - \mathbf{p}_i\| \in \mathbb{R}$. Note that the Jacobian matrix $\mathbf{J}(\mathbf{p}_S) \in \mathbb{R}^{3\times 3}$ in (3.21) can be computed as the sum of two terms, namely

$$\mathbf{J}(\mathbf{p}_S) = \frac{\partial J(\mathbf{p}_S)}{\partial \hat{\mathbf{b}}_{it}} \frac{\partial \hat{\mathbf{b}}_{it}}{\partial \mathbf{p}_i} + \frac{\partial J(\mathbf{p}_S)}{\partial \hat{d}_{it}} \frac{\partial \hat{d}_{it}}{\partial \mathbf{p}_i}. \tag{3.22}$$

One can verify that the first addendum in (3.22) belongs to the column space of the matrix $\mathbf{P}(\hat{\mathbf{b}}_{it})$, i.e., to $\mathrm{Im}(\mathbf{P}(\hat{\mathbf{b}}_{it}))$, while the second addendum is

contained in its null space, i.e., $\ker(\mathbf{P}(\hat{\mathbf{b}}_{it}))$. Then, the control input (3.21) can be equivalently rewritten as

$$\mathbf{u}_i = k \, \frac{\partial J(\mathbf{p}_S)}{\partial \hat{\mathbf{b}}_{it}} \frac{\partial \hat{\mathbf{b}}_{it}}{\partial \mathbf{p}_i}. \tag{3.23}$$

By accounting for the case wherein the noise vectors $\mathbf{n}_1, \dots \mathbf{n}_n$ are uncorrelated and characterized by diagonal covariance matrices, namely when $\mathbf{\Sigma}_{\mathbf{n}_S} = \mathrm{diag}(\mathbf{\Sigma}_{\mathbf{n}_1}, \dots \mathbf{\Sigma}_{\mathbf{n}_n})$ and $\mathbf{\Sigma}_{\mathbf{n}_i} = \sigma_i^2 \mathbf{I}_3$ for all $i \in \mathcal{S}$, it is possible to express the control input (3.21) in a more convenient closed form. Under these conditions, indeed, the matrix $\mathbf{\Sigma}_{\hat{\mathbf{p}}_t}^{-1}$ results to be

$$\mathbf{\Sigma}_{\hat{\mathbf{p}}_t}^{-1} = \sum_{i=1}^{n} \frac{1}{\sigma_i^2} \frac{1}{\hat{d}_{it}^2} \mathbf{P}(\hat{\mathbf{b}}_{it}), \tag{3.24}$$

and it can be interpreted as a weighted version of the matrix $\mathbf{A}$ in (3.17). Note also that $\hat{d}_{it}$ and $\hat{\mathbf{b}}_{it}$ in (3.24) can be seen as independent variables that account for the radial and tangent direction of the gradient of the estimation reward (3.20). Hence, $J(\mathbf{p}_S)$ is trivially maximized by reducing the estimated distance of the seeker agents with respect to the target or by relocating the seeker agents. This observation is supported by the numerical results provided in section 3.5.1.

It is important to understand that there is no unique possible choice of the reward function. For example, minimizing the biggest eigenvalue of $\mathbf{\Sigma}_{\hat{\mathbf{p}}_t}$ while ensuring that the matrix remains positive definite, or minimizing its trace (again, ensuring that it remains positive definite) could be reasonable choices.

3.5 Validation

Firstly, the validity of the approximation (3.18) of the target position estimate covariance matrix has to be asserted. Then, the performance of both the outlined WLS-based target position observer and active-sensing controller will be investigated.

3.5.1 Model Validation: localization uncertainty

The expression (3.18) of the target position estimate covariance matrix comes from the assumption that $\mathbf{n}_S$ approximates a Gaussian random vector having zero mean and a certain covariance matrix $\mathbf{\Sigma}_{\mathbf{n}_S}$. Accounting for (3.9), it can be observed that any i-th component $\mathbf{n}_i, i \in \mathcal{S}$ of the vector $\mathbf{n}_S$ results from the sum of two orthogonal terms. The first term is a downscaled version of the vector $\mathbf{v}_i$, hence the Gaussian-based modeling turns out to be fairly adequate, while this is not valid for the second term of (3.9) which entails the corresponding measurement $\widetilde{\mathbf{b}}_{it}$ to belong to the unit sphere $\mathbb{S}^2$.

Given these premises, remark that the performance of algorithm 1 rests upon the selection $\mathbf{W} = \mathbf{\Sigma}_{\mathbf{n}_S}^{-1}$, hence it is necessary to ensure a good knowledge of the covariance matrix of the vector $\mathbf{n}_S$ whose components are supposed to be uncorrelated and characterized by diagonal covariance matrices. In the following, thus, the focus is placed on the effectiveness of the assumed approximation

$$\mathbf{\Sigma}_{\mathbf{n}_S} \approx \mathbf{\Sigma}_{\bar{\mathbf{v}}_S} \tag{3.25}$$

where $\mathbf{\Sigma}_{\bar{\mathbf{v}}_S} = \operatorname{diag}\left(\mathbf{\Sigma}_{\bar{\mathbf{v}}_i}\right) \in \mathbb{R}^{3n \times 3n}$ being $\mathbf{\Sigma}_{\bar{\mathbf{v}}_i}$ the design parameter introduced in (3.4).

To assess (3.25), $N = 1000$ tests are performed simulating the target localization task in correspondence to three different (initial) displacements of the $n = 3$ seeker agents. Hereafter, S_a, S_b and S_c are introduced to denote the three considered scenarios, ranging from a more clustered to a well-spaced configuration, and the average angle $\theta_M \in [0, \pi]$ between each pair of bearings is used as an index of the closeness of the seeker agents group. Formally, this is

$$\theta_M = \frac{1}{n(n-1)/2} \sum_{i,j \in \mathcal{S}, i \neq j} \arcsin\left(\left\|\mathbf{b}_{it} \otimes \mathbf{b}_{jt}\right\|\right), \tag{3.26}$$

with $\cdot \otimes \cdot$ the cross product operator. In all the conducted tests, the target position is always equal to $\mathbf{p}_t = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^\top \mathrm{m}$. Moreover, any bearing measurement is generated according to (3.3) with different and independent noise profile at each test. Specifically, the covariance matrix of the corresponding perturbation $\mathbf{v}_i$ is computed by exploiting (3.4) and setting $\mathbf{\Sigma}_{\bar{\mathbf{v}}_i} = (\pi/180)\,\mathbf{I}_3$.

For the analysis, the collection $\left\{\hat{\mathbf{p}}_{t,k}\right\}_{k=1}^{N}$ of the (independent) estimates of the target position is considered. The estimates are outputted from algorithm 1 whose stopping condition is regulated by the parameter $\epsilon = 10^{-4}\,\mathrm{m}$. Then, the empirical target position estimate covariance matrix $\hat{\mathbf{\Sigma}}_{\hat{\mathbf{p}}_t} \in \mathbb{R}^{3\times 3}$ can determined as

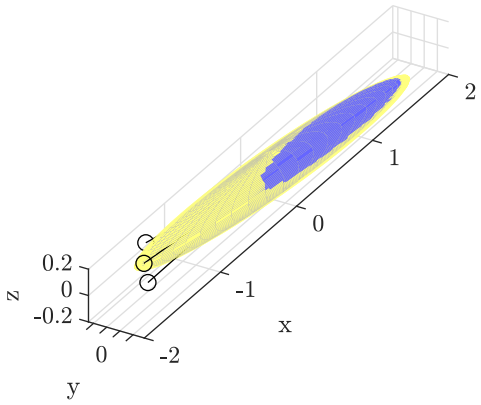
$$\hat{\mathbf{\Sigma}}_{\hat{\mathbf{p}}_t} = \frac{1}{N-1}\sum_{k=1}^{N}(\hat{\mathbf{p}}_{t,k} - \bar{\hat{\mathbf{p}}}_t)(\hat{\mathbf{p}}_{t,k} - \bar{\hat{\mathbf{p}}}_t)^{\top}, \tag{3.27}$$

where $\bar{\hat{\mathbf{p}}}_t \in \mathbb{R}^3$ indicates the empirical target position estimate mean, namely it is $\bar{\hat{\mathbf{p}}}_t = 1/N \sum_{k=1}^{N} \hat{\mathbf{p}}_{t,k}$.

Figure 3.2 clears up the relation between the covariance matrix $\mathbf{\Sigma}_{\hat{\mathbf{p}}_t}$ computed as in (3.19) (hereafter termed *theoretical covariance*) and its empirical counterpart (3.27) (hereafter termed *empirical covariance*), by reporting the ellipsoidal covariance representation in correspondence to the three considered target localization scenarios. The results in figure 3.2 lead to two main observations. First, the theoretical covariance over-approximates the empirical one, ensuring a conservative target position estimation. Second, the difference between the two covariances reduces when the seeker agents are more spread (see figure 3.2c as compared to figure 3.2a, taking into account the different scales of the axes). This last point will be better discussed in the rest of the section. Nonetheless, it is possible to conclude that since the theoretical covariance is a good approximation of the empirical, especially in the more probable case in which the seeker agents are not all clustered together, the soundness of assumption (3.25) is guaranteed.

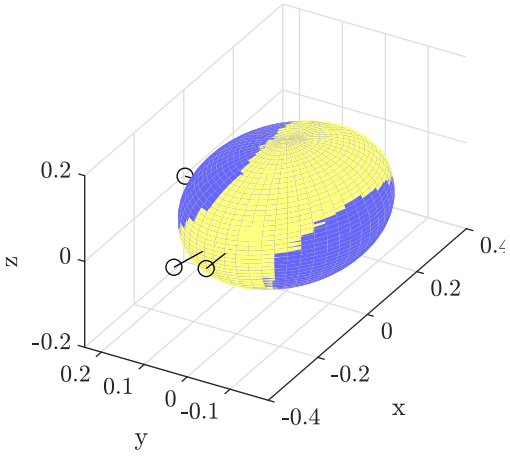
3.5.2 Solution Validation: observer performance

The tests described in the previous section provide also some insights into the performance of the proposed target position estimation method, i.e., of algorithm 1. Indeed, the empirical mean $\bar{\hat{\mathbf{p}}}_t$ of the estimated target position turns out to be biased for clustered configurations of seeker agents ($\bar{\hat{\mathbf{p}}}_t = \begin{bmatrix}0.12 & 0 & 0\end{bmatrix}\mathrm{m}$ in correspondence to to figure 3.2a). In addition, the clustered seeker agents configuration leads to a high volume of uncertainty and also a low estimation reward, while the well-spaced configuration (figure 3.2c) ensures greater accuracy. This is justified by the fact that the volume of the covariance ellip-



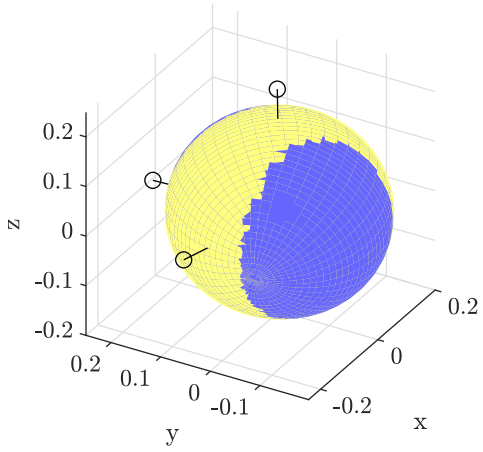
(a) clustered seeker agents configuration

$\mathbf{p}_S = [-15\ 0\ 0\quad -15\ 3\ 0\quad -15\ 0\ 1]$ m
$\theta_M = 0.16$ rad
$\bar{\hat{\mathbf{p}}}_t = [0.12\ 0\ 0]$ m
$J(\mathbf{p}_S) = 0.755 \times 10^3, \quad c_\Sigma = 105$



(b) spaced seeker agents configuration

$\mathbf{p}_S = [0\ 15\ 0\quad -15\ 3\ 0\quad -15\ 0\ 1]$ m
$\theta_M = 1.05$ rad
$\bar{\hat{\mathbf{p}}}_t = [0\ 0\ 0]$ m
$J(\mathbf{p}_S) = 18 \times 10^3, \quad c_\Sigma = 3$



(c) well-spaced seeker agents configuration

$\mathbf{p}_S = [0\ 15\ 0\quad -15\ 3\ 0\quad 0\ 0\ 15]$ m
$\theta_M = 1.47$ rad
$\bar{\hat{\mathbf{p}}}_t = [0\ 0\ 0]$ m
$J(\mathbf{p}_S) = 24 \times 10^3, \quad c_\Sigma = 1.2$

Figure 3.2: Ellipsoids representing the theoretical $\mathbf{\Sigma}_{\hat{\mathbf{p}}_t}$ (yellow) and the empirical $\hat{\mathbf{\Sigma}}_{\hat{\mathbf{p}}_t}$ (blue) covariance matrices in correspondence to different seeker agents configurations. The bearing measurements are marked in black. Note that the scale in (a) is one order of magnitude larger than (b) and (c).

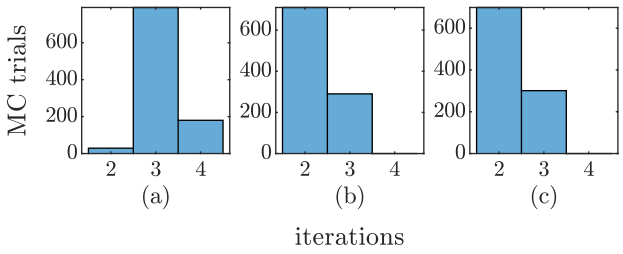


Figure 3.3: Number of iterations of algorithm 1.

soids strictly depends on the position $\mathbf{p}_S$ of the seeker agents, but also on their relative position with respect to the target according to (3.24). In the three considered scenarios, the seeker agents are roughly placed at $15\,\mathrm{m}$ from the target but the direction of the bearings varies. Intuitively, any bearing identifies an infinite set of target position estimates lying on the line passing through the corresponding seeker agent position. To resolve the correct distance, thus, at least another not collinear bearing is needed and, in particular, the more the second bearing is orthogonal to the first one, the greater the information provided. Lastly, it can be observed that the well-spaced seeker agents configuration S_c yields also a covariance ellipsoid less flattened at the poles: in this case, the condition number of the theoretical covariance $c_\Sigma \in \mathbb{R}$ approaches its minimum value. Note that when $c_\Sigma = 1$ the covariance ellipsoid is a sphere, in this case, the localization uncertainty is favorably uniformly distributed on the three position components.

Figure 3.3 shows that the number of iterations required by algorithm 1 to fulfill the stopping condition is no greater than four in all the considered scenarios. This implies the goodness of the adopted initialization strategy (section 3.3.1).

3.5.3 Solution Validation: controller performance

To investigate the performance of the outlined active-sensing control approach, the target localization scenario wherein the target position estimation poorly performs is taken into account. Thus, the attention is focused on the effects of the application of the control law (3.23) to any seeker agent composing the clustered configuration (figure 3.2a). In detail, the bearing measurements acquisition period is set to $T = 0.1\,\mathrm{s}$ and the control gain to $k = 0.002$.

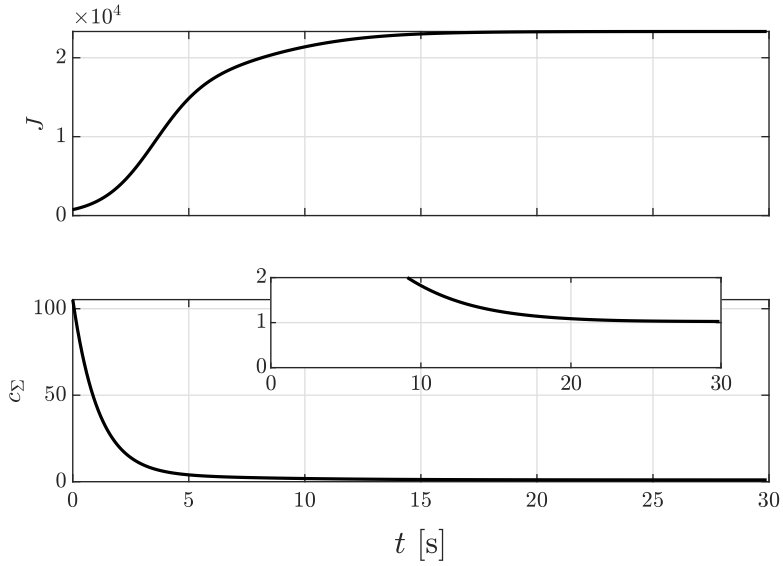


Figure 3.4: Estimation reward and condition number of the (theoretical) covariance matrix.

Figure 3.4 reports the trend of the estimation reward $J(\mathbf{p}_S)$ and of the condition number of the covariance matrix $\boldsymbol{\Sigma}_{\hat{\mathbf{p}}_t}$. Note that $J(\mathbf{p}_S)$ reaches the steady-state value $\sim 23 \times 10^3$ in almost $15\,\mathrm{s}$, concurrently the condition number converges to 1.02 meaning that the covariance ellipsoid approximates a sphere. The trajectories of the seeker agents are finally depicted in figure 3.5: starting from a clustered configuration, the seeker agents are steered in some new positions such that the resulting bearings are almost orthogonal among them. Observe also that the proposed control law (3.21) should guarantee that the seeker agents maintain constant the distance from the current target estimation. This is not ensured in the practical discrete implementation, however, such a distance can not decrease since any seeker agent velocity vector turns out to be tangential to the bearing with respect to the target.

To conclude, this chapter shows how multi-agent systems can be exploited to achieve high level goals such as target localization and how active-sense frameworks can increase the performances of sensing algorithms by intelligently controlling the agents. Once again, the usefulness of bearing sensing techniques has been highlighted. A few final notes are in order though,

- The target was assumed to be *static*, this limits the practical applications of the proposed approach. Nevertheless, as the WLS method is static too, in the sense that it is not a dynamic localization algorithm and the previous estimate is only used to initialize the new run, the target

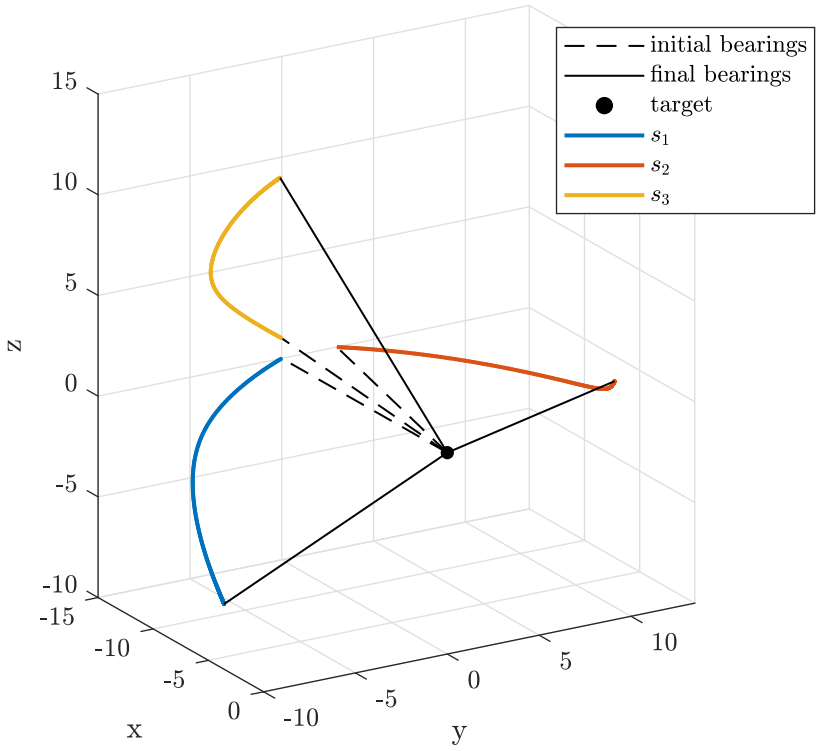


Figure 3.5: seekers trajectories.

position estimation phase is not supposed to be influenced by moving targets. On the other hand, the active-sense controller relies on the static target to guarantee that the seeker-target distances are kept constant.

- As was mentioned above, the localization algorithm is not dynamic. A better estimator should exploit the underlying target dynamics to improve its performances.

- The seeker agents formation is assumed to know its relative positions without the need of any inter-agent measurement and to be able to have complete communication graph. In a more realistic scenario distributed localization and active sense control should be adopted, in combination with inter-seekers sensing. This would allow to relax the complete sensing graph assumption.

4 Tilting quadrotor

In chapter 2 bearing rigidity theory was used to control an heterogeneous formation having different actuation capabilities. In chapter 3 the task of localizing an uncooperative target using a group of seeker and measuring only bearing vectors has been addressed. From an actuation and sensing point of view, in the simulative validation of chapter 2 the greatest degrees of freedom are employed, i.e. fully actuated UAVs. In that way, it is actually possible to apply thrusts and torques on the whole $\mathbb{R}^6$. At the same time, the bearing measurements are acquired in the local frames of the agents. On the other hand, the problem setup of chapter 3 adopts simple points in $\mathbb{R}^3$ to model the agents, resorting on the absolute communication capabilities among the seekers to retrieve the underlying agents relative orientations.

At the same time, in practical application, it is difficult to have either a *proper* fully actuated UAVs or a communication infrastructure that verifies the assumptions of chapter 3. This chapter tries to bridge the gap between complex and inefficient fully actuated UAVs and standard coplanar quadrotors, proposing a new low level control law for *tilting* quadrotors.

4.1 Introduction

Quadrotors are probably the most widespread platforms among Unmanned Aerial Vehicles (UAVs). Thanks to their low cost, high agility and flexibility, they have gained increasing interest in recent years and they are currently employed in a large variety of robotics applications, such as surveillance (Semsch et al., 2009), monitoring (Ren et al., 2019), search and rescue (Scherer et al., 2015), mapping (Christiansen et al., 2017) and data collection (Liu et al., 2018), only to mention a few.

Standard quadrotors are under-actuated systems, with a tight coupling between the translational and the rotational dynamics. This prevents their application in interactive tasks requiring the drone to exchange forces in

arbitrary directions, or in applications requiring challenging flights, with decoupled position and rotation trajectories. In multi-agent scenarios, with inter-agent bearing sensing capabilities, they are normally used for formation embedded in $\bar{\mathcal{D}} = (\mathbb{R}^3 \times \mathbb{S}^1)^n$ as their four cdofs correspond to three dimensional position in the space and heading (yaw) angle. This way of expressing the platform configuration is sufficient to model their *static* behavior or their equilibrium points. However, it falls short to encode their dynamic properties, as planar quadrotors need to tilt and roll in order to acquire any kind of horizontal acceleration. Consequently, a bearing sensor rigidly attached to a coplanar quadrotor would tilt and roll too, and its field of view would end up moving constantly.

The *1-dof tilting quadrotor* design (Ryll et al., 2014), hereafter denoted as tilting-quad, is a modification of the conventional quadrotor in which each propeller, instead of being fixed along the vertical direction, is allowed to tilt about a predefined axis, typically actuated by a servomotor. The presence of 8 control inputs, namely 4 for the tilting angles and 4 for the spinning rates of the propellers, makes the tilting-quad an over-actuated platform, capable also of decoupling translation and rotation. This design, however, presents new challenges from a control point of view. Indeed, the complexity of the system dynamics is increased by the presence of five interconnected moving bodies, namely the 4 rotors and the main frame, which introduces additional nonlinear couplings. Moreover, its inherent over-actuation requires the development of suitable allocation methods to efficiently exploit the actuation redundancy.

Several strategies have been proposed in the literature to deal with the control of tilting platforms. Focusing on trajectory tracking problems, proposed solutions rely on PID based controllers with control allocators (Bin Junaid et al., 2018; Oosedo et al., 2015), LQR techniques (Öner et al., 2008), H_∞ control (Raffo et al., 2011), geometric control (Invernizzi and Lovera, 2017) and back-stepping (Saif et al., 2018). These methods, however, are based on strong assumptions about the model, such as small angle approximation for system linearization, which heavily limits the set of feasible trajectories. Promising results have been obtained with Feedback Linearization (FL): in (Ryll et al., 2014), a dynamical FL strategy based on a simplified nonlinear model of the tilting-quad is developed. This approach has proved to obtain

accurate tracking performance of both positions and orientations, and also a certain degree of robustness against unmodeled dynamics. However, since it does not consider saturation constraints, its performance could degrade with fast trajectories that require the system to operate close to the physical limits.

In recent years, Model Predictive Control (MPC) (Rawlings et al., 2017) has gained popularity in aerial robotics, thanks both to its predicting nature and its ability to handle constraints. In the context of UAVs, however, due to the high computational burden it requires, MPC has typically been used as a trajectory planner rather than as a real-time controller acting at the motor-level (Pozzan et al., 2022; Ganga and Dharmana, 2017). In (Bicego et al., 2020), instead, a complete Nonlinear MPC (NMPC) framework is exploited both as optimal trajectory planner and tracking controller also for a tilting platform, but considering the same tilting angle for all the propellers. In (Jacquet et al., 2020), a perception-constrained motor-level NMPC framework has been developed also for over-actuated platforms with fixed propellers, such as the *Tilt-Hex* (Ryll et al., 2017).

In this chapter, a motor-level NMPC controller is proposed for the tilting-quad design introduced by (Ryll et al., 2014). Such controller provides directly the low-level inputs, namely the tilting velocities and the spinning acceleration of the propellers. This strategy has two main advantages: i) the low-level constraints are inherently considered in the control problem; ii) the control allocation is solved inside the NMPC optimization step allowing considering energy consumption minimization and/or tilting angle penalization. As it acts at the motor level, the proposed solution has also to cope with strong real-time constraints. To this aim, a fast and efficient NMPC implementation, based on (Chen et al., 2019b) is considered.

To stress its advantages, the proposed controller is tested on a realistic simulative setup, considering fast and challenging trajectories of increasing complexity. Results show that the proposed NMPC controller performs better than the state-of-the-art FL strategy proposed in (Ryll et al., 2014), providing both accurate tracking performance and robustness to unmodeled dynamics, even when the desired trajectory approaches the physical limits of the system.

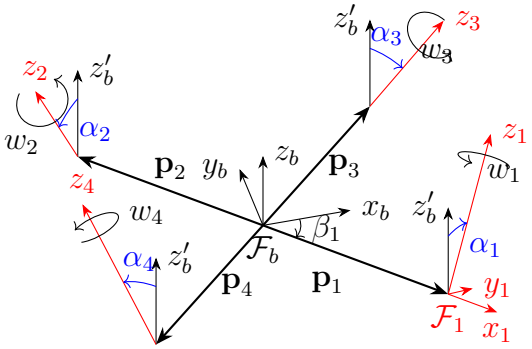


Figure 4.1: Reference frames for a tilting quadrotor.

The remainder of the chapter unfolds as follows. In section 4.2 an accurate model of the tilting quadrotor is reviewed; section 4.3 presents the proposed NMPC scheme; In section 4.4 the performed experiments are described.

4.2 Modeling

In this section, an accurate mathematical model for a tilting quadrotor is presented together with its simplified version suitable for the model predictive control scheme.

Taking inspiration from (Ryll et al., 2014), the tilting-quad can be modeled as a constrained multi-body system in which four motors and propellers are linked to the main body by revolute joints. Six reference frames are defined: $\mathcal{F}_o$ is the fixed, inertial frame; $\mathcal{F}_B$ (*body* frame) is the frame rigidly attached to the quadrotor center of mass (CoM) and $\mathcal{F}_i$, $i = 1, \dots 4$ (*motor* frames) are four frames rigidly attached to the CoM of the motors as shown in figure 4.1. Let β_i, with $i = 1, \dots 4$ be the angles that define the rotation of the tilting planes of the propellers around the z-axis of the *body* frame. Then, the relative orientations between the *motor* frames and the *body* frame are completely characterized by the rotation matrices

$$ {}^{B}\mathbf{R}_i = \mathbf{R}_Z(\beta_i)\mathbf{R}_X(\alpha_i) \in SO(3), \quad i = 1, \dots 4 \tag{4.1} $$

where $\alpha_i \in \mathbb{R}$ is the i-th propeller tilting angle w.r.t. the vertical position, i.e., the angle between z_i and z_b'. Eqs. (4.1) impose the geometric constraints

among the quadrotor frames, which can be differentiated in order to obtain the constraints on the angular velocities and angular accelerations:

$$\boldsymbol{\omega}_i = {}^B\mathbf{R}_i^\top \boldsymbol{\omega}_B + \begin{bmatrix} \dot{\alpha}_i & 0 & w_i \end{bmatrix}^\top \tag{4.2}$$

$$\dot{\boldsymbol{\omega}}_i = {}^B\mathbf{R}_i^\top \dot{\boldsymbol{\omega}}_B + {}^B\dot{\mathbf{R}}_i^\top \boldsymbol{\omega}_B + \begin{bmatrix} \ddot{\alpha}_i & 0 & \dot{w}_i \end{bmatrix}^\top \tag{4.3}$$

where $\boldsymbol{\omega}_i \in \mathbb{R}^3$ is the propeller angular rate in the motor frame, $\boldsymbol{\omega}_B \in \mathbb{R}^3$ is the quadrotor angular rate in the body frame and $w_i \in \mathbb{R}$, $i \in 1 \dots 4$ are the propeller spinning rates and

$${}^B\dot{\mathbf{R}}_i = {}^B\mathbf{R}_i \begin{bmatrix} \dot{\alpha}_i \\ 0 \\ 0 \end{bmatrix}_\times \tag{4.4}$$

is the standard time derivative of a rotation matrix as a function of its non-inertial angular rate. The i-th propeller angular dynamics expressed in $\mathcal{F}_i$ has expression

$$\boldsymbol{\tau}_i = \mathbf{J}_i \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \mathbf{J}_i \boldsymbol{\omega}_i - \boldsymbol{\tau}_{i,ext} \tag{4.5}$$

where $\mathbf{J}_i \in \mathbb{R}^{3\times3}$ is the i-th rotor inertia matrix, $\boldsymbol{\tau}_{i,ext}$ is the drag torque due to the propellers

$$\boldsymbol{\tau}_{i,ext} = \begin{bmatrix} 0 & 0 & -k_m w_i |w_i| \end{bmatrix}^\top \tag{4.6}$$

and $k_m > 0$ is the propeller drag coefficient. Be $\mathbf{e}_i \in \mathbb{R}^3$ the i-th vector of the canonical base, then in eq (4.5), $\tau_{i,x} = \mathbf{e}_1^\top \boldsymbol{\tau}_i$ is the motor tilting torque, $\tau_{i,y} = \mathbf{e}_2^\top \boldsymbol{\tau}_i$ is the reaction torque of the geometric constraint and $\tau_{i,z} = \mathbf{e}_3^\top \boldsymbol{\tau}_i$ is the motor spinning torque. On the other hand, the main body rotational dynamics takes the form

$$\boldsymbol{\tau}_B = \mathbf{J}_B \dot{\boldsymbol{\omega}}_B + \boldsymbol{\omega}_B \times \mathbf{J}_B \boldsymbol{\omega}_B + \sum_{i=1}^{4} {}^B\mathbf{R}_i \boldsymbol{\tau}_i \tag{4.7}$$

where $\mathbf{J}_B \in \mathbb{R}^{3\times3}$ is the main body inertia and $\boldsymbol{\tau}_B \in \mathbb{R}^3$ is the total external torque acting on the body, which accounts for the torque generated by the thrust of the propellers, namely

$$\boldsymbol{\tau}_B = \sum_{i=1}^{4} \mathbf{p}_i \times {}^B\mathbf{R}_i \mathbf{T}_i \tag{4.8}$$

$$\mathbf{T}_i = \begin{bmatrix} 0 & 0 & k_{t,i} w_i |w_i| \end{bmatrix}^\top \tag{4.9}$$

with $\mathbf{p}_i \in \mathbb{R}^3$ the position of the motor frames w.r.t. the body frame and $k_{t,i} \in \mathbb{R}$ the i-th propeller thrust constant, which is positive for counter-clock-wise propellers and negative for clock-wise ones. The complete dynamics of a tilted quadrotor is finally obtained taking into account the the orientation kinematics and the dynamics of the CoM, which are respectively

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{M}(\mathbf{q}_i)\boldsymbol{\omega}_B, \tag{4.10}$$

$$\ddot{\mathbf{p}} = -\mathbf{g} + \frac{1}{m_b}\mathbf{R}(\mathbf{q}) \sum_{i=1}^{4} {}^B\mathbf{R}_i \mathbf{T}_i, \tag{4.11}$$

where the quaternion $\mathbf{q} \in \mathbb{S}^3$ is used to represent the orientation of the body frame w.r.t $\mathcal{F}_W$. In (4.11), $\mathbf{p} \in \mathbb{R}^3$ is the position of the CoM of the quadrotor in the world frame, and $m_b > 0$ is total quadrotor mass.

Combining the equations above, the non-linear dynamics $\dot{\mathbf{x}}_a = \mathbf{f}_a(\mathbf{x}_a, \mathbf{u}_a)$ of a tilting-quad is retrieved. The state $\mathbf{x}_a = [\mathbf{p}^\top\ \mathbf{q}^\top\ \mathbf{v}^\top\ \boldsymbol{\omega}_B^\top\ \boldsymbol{\alpha}^\top\ \dot{\boldsymbol{\alpha}}^\top\ \mathbf{w}^\top]^\top$ consists of the body CoM position and body orientation, its linear and angular velocities, tilting angles $\boldsymbol{\alpha} = [\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4]^\top$ and rates $\dot{\boldsymbol{\alpha}} = [\dot{\alpha}_1\ \dot{\alpha}_2\ \dot{\alpha}_3\ \dot{\alpha}_4]^\top$ and spinning rates $\mathbf{w} = [w_1\ w_2\ w_3\ w_4]^\top$. The subscript a is used to denote the *accurate* model. Regarding the input $\mathbf{u}_a = [\boldsymbol{\tau}_x^\top\ \boldsymbol{\tau}_z^\top]^\top$, it consists of the tilting, $\boldsymbol{\tau}_x = [\tau_{1,x}\ \tau_{2,x}\ \tau_{3,x}\ \tau_{4,x}]^\top$, and spinning, $\boldsymbol{\tau}_z = [\tau_{1,z}\ \tau_{2,z}\ \tau_{3,z}\ \tau_{4,z}]^\top$, torques. Note that, contrary to the common practice in coplanar quadrotor modeling, here the propeller spinning rates w_i can either be positive or negative. This does not violate the *unidirectional* assumption though. Clock-wise propeller will always have negative w_i while counter clock-wise ones will keep positive w_i. To retrieve the close form expression of $\dot{\mathbf{x}}_a$ it is necessary to expand the constraints imposed by eq (4.3). Each body-propeller joint enforces the constraint

$$0 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \left(\dot{\boldsymbol{\omega}}_{P_i} - {}^B\mathbf{R}_i^\top \dot{\boldsymbol{\omega}}_B - {}^B\dot{\mathbf{R}}_i^\top \boldsymbol{\omega}_B \right), \tag{4.12}$$

stating that the y-axis angular acceleration of $\mathcal{F}_B$ w.r.t $\mathcal{F}_{P_i}$ in $\mathcal{F}_{P_i}$ must be zero. This constraint is enforced by the second component of $\boldsymbol{\tau}_{P_i}$. In order to do so the *constrained* state variable $\boldsymbol{\omega}_{P_i}$ must be written as a function of all the other variables. The procedure is described in the following:

1. The i-th frame angular velocity $\boldsymbol{\omega}_i$ in eq (4.3) is substituted into eq (4.5) giving

$$\boldsymbol{\tau}_i = \mathbf{J}_i \left({}^B\mathbf{R}_i^\top \dot{\boldsymbol{\omega}}_B + {}^B\dot{\mathbf{R}}_i^\top \boldsymbol{\omega}_B + \begin{bmatrix} \ddot{\alpha}_i & 0 & \dot{w}_i \end{bmatrix}^\top \right) + \boldsymbol{\omega}_i \times \mathbf{J}_i \boldsymbol{\omega}_i - \boldsymbol{\tau}_{ext_i}. \quad (4.13)$$

2. Then $\dot{\boldsymbol{\omega}}_B$ can be removed exploiting eq (4.7) and thus obtaining

$$\boldsymbol{\tau}_i = \mathbf{J}_i \left({}^B\mathbf{R}_i^\top \mathbf{J}_B^{-1} \left(\boldsymbol{\tau}_B - \boldsymbol{\omega}_B \times \mathbf{J}_B \boldsymbol{\omega}_B - \sum_{j=1}^{4} {}^B\mathbf{R}_j \boldsymbol{\tau}_j \right) + {}^B\dot{\mathbf{R}}_i^\top \boldsymbol{\omega}_B + \begin{bmatrix} \ddot{\alpha}_i \\ 0 \\ \dot{w}_i \end{bmatrix} \right) + \boldsymbol{\omega}_i \times \mathbf{J}_i \boldsymbol{\omega}_i - \boldsymbol{\tau}_{ext_i}. \quad (4.14)$$

3. It is then possible to group the terms containing the i-th frame torques $\boldsymbol{\tau}_i$, $i \in \{1, \dots 4\}$:

$$\mathbf{0} = -\boldsymbol{\tau}_i - \sum_{j=1}^{4} \mathbf{J}_i \, {}^B\mathbf{R}_i^\top \mathbf{J}_B^{-1} \, {}^B\mathbf{R}_j \boldsymbol{\tau}_j + \mathbf{J}_i \begin{bmatrix} \ddot{\alpha}_i \\ 0 \\ \dot{w}_i \end{bmatrix} + \mathbf{c}_i, \quad (4.15)$$

where

$$\mathbf{c}_i = \mathbf{J}_i \, {}^B\mathbf{R}_i^\top \mathbf{J}_B^{-1} \left(\boldsymbol{\tau}_B - \boldsymbol{\omega}_B \times \mathbf{J}_B \boldsymbol{\omega}_B \right) + \mathbf{J}_i \, {}^B\dot{\mathbf{R}}_i^\top \boldsymbol{\omega}_B + \boldsymbol{\omega}_i \times \mathbf{J}_i \boldsymbol{\omega}_i - \boldsymbol{\tau}_{ext_i}, \quad (4.16)$$

takes into account all other terms. Its value, if the tilting quadrotor state $\mathbf{x}$ is known, is fully determined exploiting eqs (4.3, 4.4, 4.6, 4.8).

4. At this point it is possible to rewrite eq (4.15) highlighting $\ddot{\alpha}_i$ and $\dot{w}_i$:

$$\begin{bmatrix} \ddot{\alpha}_i \\ 0 \\ \dot{w}_i \end{bmatrix} = \mathbf{J}_i^{-1} \left(\boldsymbol{\tau}_i + \sum_{j=1}^{4} \mathbf{J}_i \, {}^B\mathbf{R}_i^\top \mathbf{J}_B^{-1} \, {}^B\mathbf{R}_j \boldsymbol{\tau}_j - \mathbf{c}_i \right). \tag{4.17}$$

5. Now there are $N = 4$ matrix equations (one for each propeller) that provide $\ddot{\boldsymbol{\alpha}}$ and $\dot{\mathbf{w}}$ given the inputs $\boldsymbol{\tau}_i$, $i = 1, \dots 4$. At the same time, each equation introduces a constraint on the second components of $\boldsymbol{\tau}_i$. Those have the physical meaning for the reaction torques of the motor joints, which must cancel any attempt to tilt the propeller frames along their local y-axes. In light of this, the constraint matrix equation has linear expression in the form $\mathbf{0} = \mathbf{A}\boldsymbol{\tau}_y + \mathbf{b}$, where $\boldsymbol{\tau}_y = [\tau_{1,y}, \dots \tau_{4,y}]^\top$.

6. To retrieve $\mathbf{A} \in \mathbb{R}^{4\times4}$ and $\mathbf{b} \in \mathbb{R}^4$, first separate the known and unknown parts of $\boldsymbol{\tau}_i$ and then restrict the analysis on the second row:

$$0 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \mathbf{J}_i^{-1} \left(\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tau_{P_i,y} + \begin{bmatrix} \tau_{P_i,x} \\ 0 \\ \tau_{P_i,z} \end{bmatrix} + \sum_{j=1}^{4} \mathbf{J}_i \, {}^B\mathbf{R}_i^\top \mathbf{J}_B^{-1} \, {}^B\mathbf{R}_j \left(\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tau_{P_j,y} + \begin{bmatrix} \tau_{P_j,x} \\ 0 \\ \tau_{P_j,z} \end{bmatrix} \right) - \mathbf{c}_i \right). \tag{4.18}$$

7. The known term b_i of $\mathbf{b} = [b_1 \; b_2 \; b_3 \; b_4]^\top$ are

$$b_i = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \left(\mathbf{J}_i^{-1} \left(\begin{bmatrix} \tau_{P_i,x} \\ 0 \\ \tau_{P_i,z} \end{bmatrix} - \mathbf{c}_i \right) + \sum_{j=1}^{4} {}^B\mathbf{R}_i^\top \mathbf{J}_B^{-1} \, {}^B\mathbf{R}_j \begin{bmatrix} \tau_{P_j,x} \\ 0 \\ \tau_{P_j,z} \end{bmatrix} \right), \tag{4.19}$$

while the i,j-cell of $\mathbf{A}$ has expression

$$a_{i,i} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \mathbf{J}_i^{-1} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} {}^B\mathbf{R}_i^\top \mathbf{J}_B^{-1} \, {}^B\mathbf{R}_i \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \tag{4.20}$$

for $j = i$ and

$$a_{i,j} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} {}^{B}\mathbf{R}_i^{\top} \mathbf{J}_B^{-1} {}^{B}\mathbf{R}_j \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \tag{4.21}$$

for $j \neq i$.

At this point, to evaluate $\dot{\mathbf{x}}_a$ given the current state $\mathbf{x}_a$ and the current inputs $\mathbf{u}_a$, it is necessary to solve the constraint equations to retrieve $\boldsymbol{\tau}_y$, after that combining $\mathbf{u}$ and $\boldsymbol{\tau}_y$ yields $\ddot{\alpha}$ and $\dot{w}$. The rest of $\dot{\mathbf{x}}_a$ directly follows.

This model will be used in section 4.4 to evaluate the performance of the proposed control architecture, but it is important to notice that two major simplifications remain: no air friction is considered and no propellers coupling is considered.

To complete the tilting quadrotor model, it remains to address the actuation devices, namely the motor driving the propellers and the servomotors driving the tilting mechanisms, which provide $\boldsymbol{\tau}_z$ and $\boldsymbol{\tau}_x$.

Propellers are normally driven by brushless DC (BLDC) motors; these devices are paired with dedicated *electronic speed controller* (ESC). Therefore, the quadrotor flight controller only needs to provide appropriate spinning rate setpoints while the close loop response of ESC, BLDC and propeller can be approximated by a low-pass filter whose bandwidth is larger than the one of the main quadrotor body. The same applies for the tilting control, with the only difference that BLDC motors are usually replaced by high precision, high torque DC servomotors as the the accuracy in tracking the desired tilting angle and tilting rate is critical. The low pass filter approximation holds because, under realistic assumptions (i.e. $\mathbf{J}_i \ll \mathbf{J}_B$), the *combined* dynamics of $\boldsymbol{\omega}_B$, eq (4.5), and $\boldsymbol{\omega}_i$, eq (4.7), compose four coupled *two-time scale dynamical systems*. Section 4.3.2 will better explain this concept.

In the end, by simplifying the fast dynamics of the propellers and tilting mechanisms and neglecting the second order term $\boldsymbol{\omega}_B \times \mathbf{J}\boldsymbol{\omega}_B$, it is possible to actuate the model directly in terms of tilting and spinning rates, $\dot{\boldsymbol{\alpha}}, \mathbf{w}$, as long as high-frequency super low-level controllers, the one embedded in the ESCs and in the servomotors, are correctly adopted. In this scenario, the

new simplified state is $\mathbf{x}_s = [\mathbf{p}^\top \ \mathbf{q}^\top \ \mathbf{v}^\top \ \boldsymbol{\omega}_B^\top \ \boldsymbol{\alpha}^\top]^\top$, the new simplified input is $\mathbf{u}_s = [\dot{\boldsymbol{\alpha}}^\top \ \mathbf{w}^\top]^\top$ and the model equations are given combining (4.10, 4.11) with the simplified angular dynamics

$$\boldsymbol{\tau}_B = \mathbf{J}_B \dot{\boldsymbol{\omega}}_B - \sum_{i=1}^{4} {}^B\mathbf{R}_i \boldsymbol{\tau}_{i,ext}, \tag{4.22}$$

$$\dot{\mathbf{x}}_s = \mathbf{f}_s(\mathbf{x}_s, \mathbf{u}_s) = \begin{bmatrix} \mathbf{v} \\ \frac{1}{2}\mathbf{M}(\mathbf{q}_i)\boldsymbol{\omega}_B \\ -\mathbf{g} + \frac{1}{m_b}\mathbf{R}(\mathbf{q}) \sum_{i=1}^{4} {}^B\mathbf{R}_i \mathbf{T}_i \\ \mathbf{J}_B^{-1}(\boldsymbol{\tau}_B + \sum_{i=1}^{4} {}^B\mathbf{R}_i \boldsymbol{\tau}_{i,ext}) \\ \dot{\boldsymbol{\alpha}} \end{bmatrix} \tag{4.23}$$

This simplification brings tow major contributions:

- It is reasonable and add feasibility in the proposed control scheme: the ESC controller act on the current loop of the motor, these loops can easily have to run above $10\,\mathrm{kHz}$. Designing an NMPC with such a small sampling time that is also able to predict the full dynamics of the quadrotor is not possible.

- The simplified model completely ignores the low pass filters on tilting and spinning rate, at the point that the propeller and tilting mechanism dynamics is neglected and only the first order kinematics remains. The modeling error introduced by these assumptions makes the analysis performed in simulation more meaningful.

4.2.1 Actuation analysis and comparison with tilted solutions

At the beginning of this chapter, the interest in studying tilting quadrotor was motivated claiming that they can bridge between efficient but under-actuated coplanar quadrotors and fully-actuated but inefficient titled multirotors. That claim is now discussed. Inspecting the simplified model (4.23), and recalling the actuation properties discussed in chapter 1, a tilting quadrotor has *dynamic* actuation matrix $\mathbf{F}(\mathbf{x}) \in \mathbb{R}^{6\times4}$, where only the *square* of the propeller spinning rates have been considered as inputs. Indeed, the tilting

rates $\dot{\boldsymbol{\alpha}}$ do not contribute in the first order linearization of the dynamics and for this reason they have been omitted. More importantly in the body frame of the quadrotor, the allocation matrix actually depends only on the tilting angles $\boldsymbol{\alpha}$

$$\mathbf{F}(\boldsymbol{\alpha}) = \begin{bmatrix} \dots & \frac{k_t}{m_b}\,{}^{B}\mathbf{R}_i\mathbf{e}_3 & \dots \\ \dots & \mathbf{J}_B^{-1}\left(k_t\mathbf{p}_i \times {}^{B}\mathbf{R}_i\mathbf{e}_3 - k_m\,{}^{B}\mathbf{R}_i\mathbf{e}_3\right) & \dots \end{bmatrix}. \tag{4.24}$$

Recall that the allocation matrix defines the possible linear and angular accelerations,

$$\begin{bmatrix} \dot{\mathbf{v}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \mathbf{F}(\boldsymbol{\alpha}) \begin{bmatrix} w_1^2 \\ w_2^2 \\ w_3^2 \\ w_4^2 \end{bmatrix} \tag{4.25}$$

While $\operatorname{rk}\mathbf{F}(\boldsymbol{\alpha}) = 4$, i.e. the tilting quadrotor is not fully-actuated, the image of the allocation matrix can be modified tuning the tilting angles $\boldsymbol{\alpha}$. In principle, this allows to set-up an optimization problem that finds the tilting configuration minimizing the control effort needed to generate a desired linear and angular acceleration vector,

$$\begin{aligned} &\underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \|\bar{\mathbf{u}}\|^2 \\ &s.t. \quad \begin{bmatrix} \mathbf{a}_t \\ \mathbf{a}_r \end{bmatrix} = \mathbf{F}(\boldsymbol{\alpha})\bar{\mathbf{u}}, \\ &\alpha_i \in \{\alpha_{min},\ \alpha_{max}\}, \\ &\bar{u}_i \in \{\bar{u}_{min},\ \bar{u}_{max}\}, \end{aligned} \tag{4.26}$$

where $\bar{\mathbf{u}} = [w_1^2,\ w_2^2,\ w_3^2,\ w_4^2]^\top$ are the square of the propellers spinning rate, $\mathbf{a}_t, \mathbf{a}_r \in \mathbb{R}^3$ are the desired linear and angular acceleration, respectively and $\alpha_{min,max}$ and $\bar{u}_{min,max}$ are the lower and upper bounds for the tilting angles and the input. Solving eq (4.26) allows to find the tilting configuration that realizes the desired acceleration with the minimum effort, i.e. the most efficient configuration. Therefore, as long as the actuation limits are not violated, any *feasible* $[\mathbf{a}_t^\top\ \mathbf{a}_r^\top]^\top$ can be optimally realized. The feasibility is mostly imposed by the finite, non negative, input (the propellers are unidirectional and cannot reverse their spinning direction) and by the limit on the tilting angles. For example, common tilting-quadrotor builds limits their tilting to $\pm 60°$, both for mechanical reasons and for aerodynamical

ones, as higher tilting angles would eventually cause strong, non negligible, coupling effects. Finally, generating negative thrust on the body frame vertical axis is impossible.

On the other hand, the tilted multicopter are intrinsically fully actuated platforms (as long as a non-degenerate tilted configuration is chosen and at least 6 rotors are adopted). Tilted multicopters can be seen and tilting platform with fixed tilting angles α. This greatly reduce the mechanical and control complexity but, at the same time, reduces the flexibility of the devices. The tilt angles α are a design parameter for a tilted multicopter and the designer must chose them weighting maneuverability and efficiency. The former measures the amount of achievable linear and angular accelerations and can be measured with, for instance, the smallest singular value of the allocation matrix. Indeed, that value represents how difficult is to apply the worst possible combination of linear and angular accelerations. If the value is zero, the tilted platform is ill-designed and under-actuated. The latter index, instead, represents how much of the input energy is wasted internally to cancel the undesired torques and thrusts in order to obtain the desired ones. This value is not fixed, but it changes depending on the trim point of the multicopter. For instance, if the UAV mostly has to hover horizontally, then the coplanar configuration has the absolute best efficiency. On the other hand that is one of the ill-conditioned configurations. If the drone instead has to hover in a tilted attitude, then the most efficient configuration will be different. Lastly, it is important to remark that the tilted multicopter can change its acceleration profile by just adjusting the propeller spinning rates while in most of the cases, a tilting quadrotor need to update its tilting angles, which in general introduces non negligible delays.

In conclusion, a tilted multicopter can be designed to be incredibly agile, with the capability of generating thrust and torques in any arbitrary direction, or it can be designed to be efficient, but it cannot be both things at the same time. On the other hand, a tilting quadrotor, at the cost of an increase of mechanical and control complexity, can achieve both goals by reorienting its spinning axes.

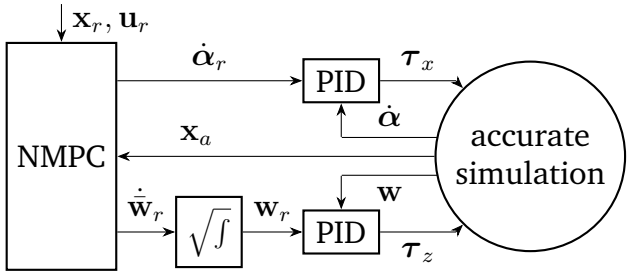


Figure 4.2: Control architecture.

4.3 Control architecture

With the advantages of tilting quadrotor over other configuration well asserted, the next step consist on presenting the proposed control law. The main goal is to track fast and possibly unfeasible trajectories. To do so, the proposed control architecture, depicted in figure 4.2, has two main components, discussed in the next two subsections.

4.3.1 Nonlinear model predictive controller

The core of the controller is composed of a NMPC scheme that solves the Nonlinear Programming problem (NLP) obtained by discretization of the Optimal Control problem (OCP) of interest. Given N shooting intervals and the evenly spaced sampling instants t_k, $k = 0, \dots N$, the NLP takes the general form (Chen et al., 2019b)

$$\min_{\mathbf{x}_k, \mathbf{u}_k} \sum_{k=0}^{N-1} \frac{1}{2}\|\mathbf{h}_k(\mathbf{x}_k, \mathbf{u}_k)\|^2_{\mathbf{W}} + \frac{1}{2}\|\mathbf{h}_N(\mathbf{x}_N)\|^2_{\mathbf{W}_N} \tag{4.27a}$$

$$s.t.\ 0 = \mathbf{x}_0 - \hat{\mathbf{x}}_{mpc,0}, \tag{4.27b}$$

$$0 = \mathbf{x}_{k+1} - \boldsymbol{\phi}(\mathbf{x}_k, \mathbf{u}_k),\ k = 0, \dots N-1, \tag{4.27c}$$

$$\underline{\mathbf{r}}_k \leq \mathbf{r}_k(\mathbf{x}_k, \mathbf{u}_k) \leq \bar{\mathbf{r}}_k,\ k = 0, \dots N-1, \tag{4.27d}$$

$$\underline{\mathbf{r}}_N \leq \mathbf{r}_N(\mathbf{x}_N) \leq \bar{\mathbf{r}}_N, \tag{4.27e}$$

where $\hat{\mathbf{x}}_{mpc,0}$ is the measurement of the current state, while $\boldsymbol{\phi}(\mathbf{x}_k, \mathbf{u}_k)$ is the numerical integrator that solves the initial value problem

$$\dot{\mathbf{x}}_{mpc}(t) = \mathbf{f}_{mpc}(\mathbf{x}_{mpc}(t), \mathbf{u}_{mpc}(t)), \quad \mathbf{x}_{mpc}(0) = \mathbf{x}_k \tag{4.28}$$

at time t_{k+1} with $\mathbf{u}_{mpc}(t)$ piece-wise constant in the sampling instants. $\mathbf{h}_k(\cdot,\cdot)$ and $\mathbf{h}_N(\cdot)$ are the output and final stage output function, respectively. $\mathbf{r}_k(\mathbf{x}_k,\mathbf{u}_k)$ and $\mathbf{r}_N(\mathbf{x}_N)$ are the non-linear constraints. For the simplified tilting quadrotor model (4.10, 4.11, 4.22) an *integrated*-NMPC is proposed, with the integral component placed on the spinning rates. Therefore, the differential equation (4.28) has expression

$$\dot{\mathbf{x}}_{mpc} = \begin{bmatrix} \dot{\mathbf{x}}_s \\ \dot{\bar{\mathbf{w}}} \end{bmatrix} = \mathbf{f}_{mpc}(\mathbf{x}_{mpc},\mathbf{u}_{mpc}) = \begin{bmatrix} \mathbf{v} \\ \frac{1}{2}\mathbf{M}(\mathbf{q}_i)\boldsymbol{\omega}_B \\ -\mathbf{g} + \frac{1}{m_b}\mathbf{R}(\mathbf{q})\sum_{i=1}^{4} {}^B\mathbf{R}_i\mathbf{T}_i \\ \mathbf{J}_B^{-1}(\boldsymbol{\tau}_B + \sum_{i=1}^{4} {}^B\mathbf{R}_i\boldsymbol{\tau}_{i,ext}) \\ \dot{\boldsymbol{\alpha}}_r \\ \dot{\bar{\mathbf{w}}}_r \end{bmatrix}, \tag{4.29}$$

with $\mathbf{x}_{mpc} = [\mathbf{p}^\top\ \mathbf{q}^\top\ \mathbf{v}^\top\ \boldsymbol{\omega}_B^\top\ \boldsymbol{\alpha}^\top\ \bar{\mathbf{w}}^\top]^\top$ the NMPC state and $\mathbf{u}_{mpc} = [\dot{\boldsymbol{\alpha}}_r^\top\ \dot{\bar{\mathbf{w}}}_r^\top]^\top$ the NMPC input. $\dot{\boldsymbol{\alpha}}_r, \dot{\bar{\mathbf{w}}}_r$ are the tilting speed reference and the spinning acceleration reference, respectively. In order to reduce the NMPC computational burden, the following transformations have been applied to the input-state representation:

1. Firstly, the square of the propeller spinning rate $\bar{w}_i' = w_i^2$ is adopted, which leads to a linear dynamics in the input vector, indeed, inspecting eqs. (4.6) and (4.9), it results

$$\boldsymbol{\tau}_{i,ext} = \begin{bmatrix} 0 & 0 & -\bar{k}_m'\bar{w}_i' \end{bmatrix}^\top \qquad \mathbf{T}_i = \begin{bmatrix} 0 & 0 & \bar{k}_t'\bar{w}_i' \end{bmatrix}^\top \tag{4.30}$$

   where $\bar{k}_m'$ and $\bar{k}_t'$ are the adjusted drag and thrust coefficients. In almost all practical application *unidirectional propellers* are employed and the propeller thrust is always positive along the positive local z-semiaxes. This implies that k_t' is always positive while k_m' is positive for counter clock-wise propeller and negative for clock-wise ones.

2. $\bar{w}_i'$ is normalized, $\bar{w}_i = \bar{w}_i'/w_{max}$, with w_{max} the higher achievable spinning rate, in order to not have a multi-scale dynamics equation and improve numerical stability. The proper scaling is then taken into account in the updated thrust and torque constants: $\bar{k}_m = w_{max}\bar{k}_m'$ and $\bar{k}_t = w_{max}\bar{k}_t'$.

Regarding the model state constraints (4.27d) and (4.27e), the tilting angles and the spinning rates are lower and upper bounded and the same applies to all the inputs. In accordance to the receding horizon framework, at each NMPC iteration, of the whole optimal input sequence $\mathbf{u}^*_{mpx}$, only the first element is sent to the low level controllers. While $\dot{\boldsymbol{\alpha}}_r$ can be directly pushed down the control pipeline, $\dot{\tilde{\mathbf{w}}}_r$ firstly has to go through an integrator, and then it needs to be denormalized and its squared root extracted, this gives the required spinning rate setpoint $\mathbf{w}_r$ for the low-level controllers.

4.3.2 Low-level controllers

The low-level controllers block simulate the motor ESCs and it is responsible for bridging the NMPC commands $\dot{\boldsymbol{\alpha}}_r$ and $\mathbf{w}_r$ into motor tilting torques $\boldsymbol{\tau}_x$ and motor spinning torques $\boldsymbol{\tau}_z$. This is achieved by a pair of PID controllers for each propeller, which run at a much higher frequency than the NMPC in order to stabilize the fast dynamics of the motors. It is important to remark that the propellers dynamics is highly coupled through the main body one. Indeed, for each propeller, substituting (4.2) and (4.3) into (4.5) and denoting with $\mathbf{x}_i = [\dot{\alpha}_i \; 0 \; w_i]^\top$ the state of the i-th propeller, yields

$$\mathbf{J}_i \dot{\mathbf{x}}_i = -\mathbf{J}_i \left({}^B\mathbf{R}_i^\top \dot{\boldsymbol{\omega}}_B + {}^B\mathbf{R}_i \begin{bmatrix} \dot{\alpha}_i \\ 0 \\ 0 \end{bmatrix}_\times \boldsymbol{\omega}_B \right) + \boldsymbol{\tau}_{ext_i} + \boldsymbol{\tau}_i - \left({}^B\mathbf{R}_i^\top \boldsymbol{\omega}_B + \mathbf{x}_i \right) \times \mathbf{J}_i \left({}^B\mathbf{R}_i^\top \boldsymbol{\omega}_B + \mathbf{x}_i \right), \quad (4.31)$$

which in first-order approximation represents a double integrator dynamics for α_i and a single integrator dynamics for w_i subject to disturbances coming from the main body dynamics $\dot{\boldsymbol{\omega}}_B$, $\boldsymbol{\omega}_B$ and the drag torque $\boldsymbol{\tau}_{ext_i}$. Therefore, standard PIDs are expected to perform properly only under the assumption that these disturbances remain limited. This constraint is already satisfied for $\boldsymbol{\tau}_{ext_i}$, since it is a function of w_i and it is reasonable to assume that $\dot{\boldsymbol{\omega}}_B$, $\boldsymbol{\omega}_B$ remain bounded also when dealing with aggressive maneuvers in real-world scenarios. The PIDs emulate the pairs ESC+BLDC motor and servomotor, for the spinning and tilting branches, respectively. Their inputs are the the spinning rate errors $e_{w,i} = w_r - w_i$ and the the tilting rate error $e_{\alpha,i} = \dot{\alpha}_r - \dot{\alpha}_i$.

Table 4.1: Accurate model parameters

Parameter	Value
m_b	$2.62\,\mathrm{kg}$
$\mathbf{J}_B$	$\mathrm{diag}([0.03\ 0.03\ 0.06])\ \mathrm{kgm}^2$
k_t	$1.76 \times 10^{-5}\,\mathrm{N/s}^2$
k_m/k_t	$0.05\,\mathrm{m}$
$\mathbf{J}_i$	$\mathrm{diag}([0.135\ 0.135\ 0.26]) \times 10^{-4}\,\mathrm{kgm}^2$

Table 4.2: Accurate model limit values

Variable	Range
$\lvert w \rvert$	$[100, 10\,000]$ rpm
α_i	$[-1.04, 1.04]$ rad
$\dot{\alpha}_i$	$[-6, 6]$ rad/s
$\tau_{i,x}$	$[-0.1, 0.1]$ Nm
$\tau_{i,z}$	$[-1, 1]$ Nm

4.4 Simulation results

In this section, the proposed controller is evaluated in a simulative setup, obtained by implementing in Simulink® the accurate model derived in section 4.2.[1] The NMPC controller is compared with a state-of-the-art FL control scheme on different trajectories with increasing complexity. Both controllers exploit the simplified model described by eq (4.23), initially assuming perfect knowledge of the model parameters and state vector.

4.4.1 Model parametrization

In order to have realistic simulations, the high-accuracy model parameters have been chosen to mimic as much as possible those of a real drone. They are listed in tables 4.1 and 4.2. Specifically, a $40\,\mathrm{cm}$-wide X-shaped quadrotor

[1] The code is available at https://github.com/sparcs-unipd/NMPC-tilting-quadrotor.

Table 4.3: NMPC weight matrices

Weight	Value	Weight	Value
$\mathbf{W}_p$	$5\ \mathbf{I}_3$	$\mathbf{W}_v$	$0.5\ \mathbf{I}_3$
$\mathbf{W}_q$	$\mathbf{I}_3$	$\mathbf{W}_\omega$	$10\ \mathbf{I}_3$
$\mathbf{W}_\alpha$	$1 \times 10^{-2}\ \mathbf{I}_3$	$\mathbf{W}_{\dot{\alpha}}$	$1 \times 10^{-3}\ \mathbf{I}_3$
$\mathbf{W}_{\bar{w}}$	$1 \times 10^{-3}\ \mathbf{I}_3$	$\mathbf{W}_{\dot{\bar{w}}}$	$1 \times 10^{-1}\mathbf{I}_3$

layout has been adopted with tilting axes parallel to the motor arms, which gives

$$\mathbf{p}_i = 20\,\mathrm{cm} \begin{bmatrix} \cos\beta_i \\ \sin\beta_i \\ 0 \end{bmatrix} \tag{4.32}$$

with $\beta_1 = -\frac{\pi}{4}$, $\beta_2 = \frac{3\pi}{4}$, $\beta_3 = \frac{\pi}{4}$, $\beta_4 = -\frac{3\pi}{4}$. The first and second propellers are counter-clockwise while the third and fourth are clockwise. In the simulation, the accurate model has been integrated using *Runge-Kutta 4* with step size $T_s = 1\,\mathrm{ms}$. Regarding the low-level PID controllers, they are obtained by implementing eight independent the continuous time transfer functions in standard form

$$P + I\frac{1}{s} + D\frac{N_d}{1 + N_d\frac{1}{s}} \tag{4.33}$$

with $P = 0.01$, $I = 0.1$, $D = 10^{-5}$, and $N_d = 100$.

4.4.2 NMPC implementation

The NMPC scheme presented in Section 4.3.1 has been implemented exploiting MATMPC (Chen et al., 2019b). The simplified non-linear dynamics of the tilting-quad (4.29) is discretized with step size $T_{s,MPC} = 10\,\mathrm{ms}$ using an explicit *Runge-Kutta 4* integrator. CASADI (Andersson et al., 2019) automatically performs offline the differentiation of the model equations and the *interior point method* HPIPM_SPARSE is used to solve the final sequential quadratic programming problem. This latter choice has been experimentally validated: it is faster than the other available solvers and it is sufficiently robust for the considered model, i.e., no numerical artifact are visible in simulation. MATMPC implements a real-time iteration (RTI) scheme, in which the sensitivities are adaptively updated online and only one SQP iteration is performed. This allows to achieve a real-time feasible implementation. The

cost function in (4.27a) is a quadratic form based on the difference between the predicted state and input and the desired ones, namely

$$\mathbf{h}_k = \begin{bmatrix} \mathbf{x}_k - \mathbf{x}_{k,r} \\ \mathbf{u}_k - \mathbf{u}_{k,r} \end{bmatrix},$$

while the final cost is just the weighted square of the difference between the final predicted state and the final desired one $\mathbf{h}_N = \mathbf{x}_N - \mathbf{x}_{N,r}$.[2] The weight matrices have structure $\mathbf{W} = \text{diag}(\mathbf{W}_p, \mathbf{W}_q, \mathbf{W}_v, \mathbf{W}_\omega, \mathbf{W}_\alpha, \mathbf{W}_{\bar{w}}, \mathbf{W}_{\dot{\bar{w}}}, \mathbf{W}_{\dot{\alpha}})$ and $\mathbf{W}_N = \text{diag}(\mathbf{W}_p, \mathbf{W}_q, \mathbf{W}_v, \mathbf{W}_\omega, \mathbf{W}_\alpha, \mathbf{W}_{\bar{w}})$ and their values are reported in table 4.3. Finally, the prediction horizon has length $N = 30$ samples.

4.4.3 Benchmark trajectories

The controller is tested on infinity-shaped planar trajectories with a constant attitude

$$\mathbf{p}_r(t) = \begin{bmatrix} a_x \sin(\omega t) \\ a_y \sin(2\omega t) \\ 0 \end{bmatrix} + \mathbf{s}(t), \quad \omega = 1\,\text{rad/s} \tag{4.34}$$

$$\mathbf{q}_r(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^\top, \tag{4.35}$$

that are parametrized by the x and y amplitude a_x, a_y, respectively. $\mathbf{s}(t) = \mathbf{s}_1 e^{-t} + \mathbf{s}_2 e^{-2t} + \mathbf{s}_3 e^{-3t}$ is an exponentially decaying smoothing function that ensures zero initial condition for the position, velocity and acceleration references. The final NMPC state and input reference trajectories result $\mathbf{x}_r = [\mathbf{p}_r^\top \ \mathbf{q}_r^\top \ \mathbf{v}_r^\top \ \boldsymbol{\omega}_{B,r}^\top \ \boldsymbol{\alpha}_r^\top \ \bar{\mathbf{w}}_r^\top]^\top$ and $\mathbf{u} = [\dot{\boldsymbol{\alpha}}_r^\top \ \dot{\bar{\mathbf{w}}}_r^\top]^\top$, with $\mathbf{v}_r, \boldsymbol{\omega}_{B,r}$ the analytical derivatives of $\mathbf{p}_r, \mathbf{q}_r$ and all the other values fixed to zero. In such a way the NMPC cost will prefer to

- not deviate too much from the coplanar configuration: $\boldsymbol{\alpha}_r = \mathbf{0}$;
- minimize the spinning rate, i.e,. the energy consumption of the tilting quadrotor: $\bar{\mathbf{w}}_r = \mathbf{0}$;
- not change tilting angles or spinning rate too fast: $\dot{\boldsymbol{\alpha}}_r = \dot{\bar{\mathbf{w}}}_r = \mathbf{0}$.

[2] Notice that with abuse of notation, the quaternion error $\mathbf{q}_k - \mathbf{q}_{k,r}$ is computed as the imaginary part of $\mathbf{q}_k \circ \mathbf{q}_{k,r}^{-1}$.

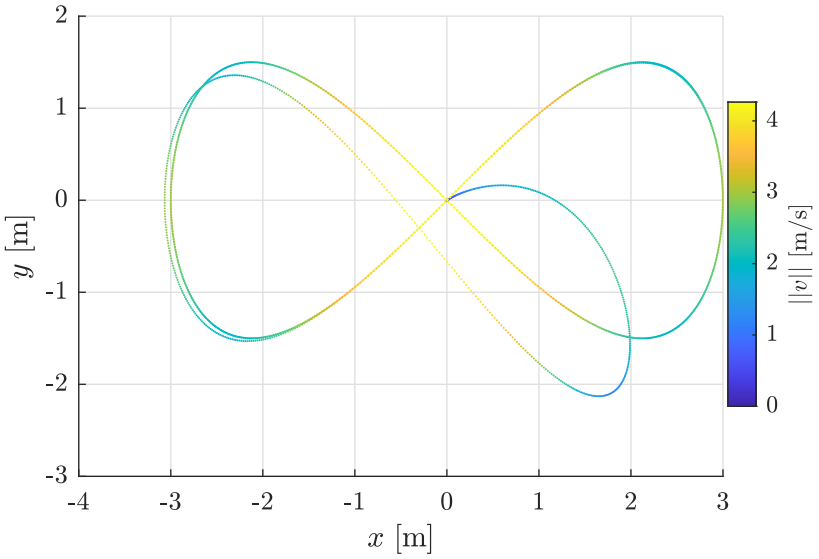


Figure 4.3: Exemplifying realization of the desired trajectory on the xy-plane. It starts from $(0, 0)$ with zero initial velocity and acceleration and, after a transient, it syncs with the infinity-shaped trajectory.

The effect with the greatest weight is selected by tuning $\mathbf{W}_{\alpha}, \mathbf{W}_{\bar{w}}, \mathbf{W}_{\dot{\bar{w}}}$ and $\mathbf{W}_{\dot{\alpha}}$). A possible trajectory realization is shown in figure 4.3 with $a_x = 3\,\mathrm{m}, a_y = 1.5\,\mathrm{m}$. Table 4.4, instead, reports details of the six chosen trajectories with increasing performance level (required tracking speed) together with their maximum velocity $v_{r,max}$ and acceleration $a_{r,max}$. The infinity-shaped trajectory with constant attitude has been chosen to stress the thrust-torque decoupling capabilities of the tilting quadrotor.

Table 4.4: Trajectories characterization, a_x, a_y, levels of difficulty $v_{r,max}, a_{r,max}$ and controllers performance indexes $\hat{e}_p, \hat{e}_\vartheta$. NMPCe refers to the NMPC scheme with $N = 60$ (extended) shooting intervals.

trj id	**1**		**2**		**3**		**4**		**5**		**6**		
a_x [m]	2		3		4		5		6		7		
a_y [m]	1		1.5		2		2.5		3		3.5		
$v_{r,max}[\mathrm{m/s}]$	2.8		4.3		5.7		7.1		8.5		9.9		
$a_{r,max}[\mathrm{m/s^2}]$	4.3		6.45		8.6		10.8		12.9		15		
controller	**NMPC**	**FL**	**NMPC**	**FL**	**NMPC**	**FL**	**NMPC**	**FL**	**NMPC**	**FL**	**NMPC**	**FL**	**NMPCe**
$\hat{e}_p$ [cm]	1.5	0.9	3	2.5	7.9	-	17	-	58	-	-	-	10
$\hat{e}_\vartheta$ [mrad]	7.9	7	11.4	90	25.5	-	42.5	-	33.7	-	-	-	216

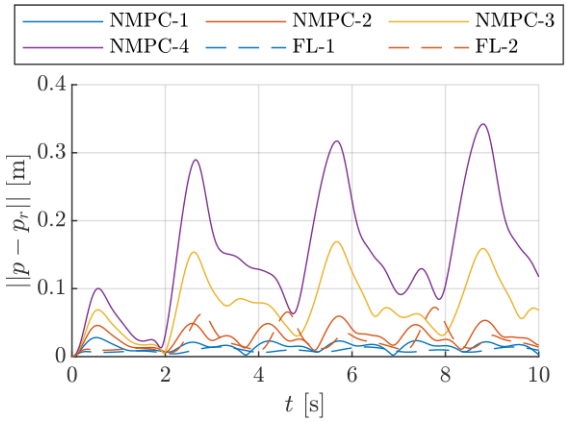


Figure 4.4: Position error performance of the two controllers. The NMPC (solid line) is able to track all four trajectories while the FL (dashed line) is limited to the first two.

4.4.4 FL comparison controller

The FL controller proposed in (Ryll et al., 2014) is used as the baseline for performance evaluation. This controller uses the same internal model of the MPC with the only difference being that the propeller spinning rates are not normalized. Introducing the third derivative of the quadrotor position and the second derivative of its angular velocity the dynamics can be written as

$$\begin{bmatrix} \dddot{\mathbf{p}} \\ \ddot{\boldsymbol{\omega}} \end{bmatrix} = \mathbf{A}(\boldsymbol{\alpha}, \bar{\mathbf{w}}') \begin{bmatrix} \dot{\bar{\mathbf{w}}}' \\ \dot{\boldsymbol{\alpha}} \end{bmatrix} + \mathbf{b}(\boldsymbol{\alpha}, \bar{\mathbf{w}}', \boldsymbol{\omega}_B) \tag{4.36}$$

with $\mathbf{A}(\boldsymbol{\alpha}, \bar{\mathbf{w}}') \in \mathbb{R}^{6\times 8}$ full rank as long as $\bar{\mathbf{w}}' \neq 0$. Then the FL controller can be applied

$$\begin{bmatrix} \dot{\bar{\mathbf{w}}}' \\ \dot{\boldsymbol{\alpha}} \end{bmatrix} = \mathbf{A}^{\dagger} \left(\begin{bmatrix} \dddot{\mathbf{p}}_c \\ \ddot{\boldsymbol{\omega}}_c \end{bmatrix} - \mathbf{b} \right) + (\mathbf{I}_8 - \mathbf{A}^{\dagger}\mathbf{A})\mathbf{z} \tag{4.37}$$

where $\mathbf{A}^{\dagger} \in \mathbb{R}^{8\times 6}$ is the Moore-Penrose pseudoinverse of $\mathbf{A}$ and $\dddot{\mathbf{p}}_c, \ddot{\boldsymbol{\omega}}_c$ are the new control inputs of the linearized system which can be obtained by a stabilizing static state feedback. The system overactuation $(\mathbf{I}_8 - \mathbf{A}^{\dagger}\mathbf{A}) \neq 0$ is exploited in the additional input $\mathbf{z} \in \mathbb{R}^8$ to steer $\dot{\bar{\mathbf{w}}}' and \dot{\boldsymbol{\alpha}}$ acting on the

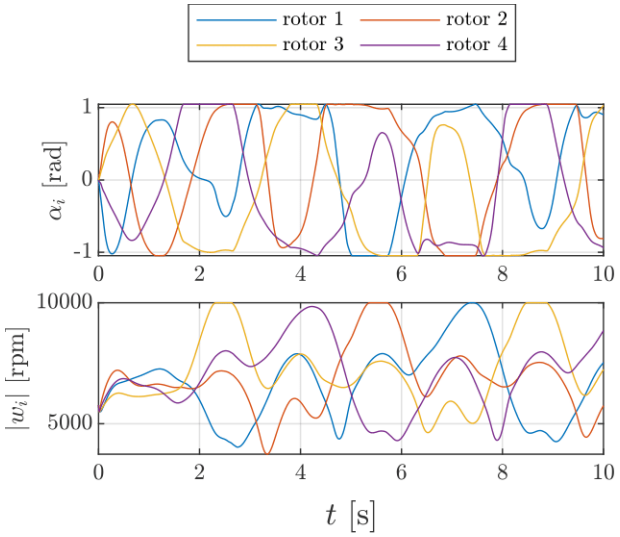


Figure 4.5: Tilting angles and spinning rates computed by the NMPC for the fourth trajectory. Heavy saturation is observable (see values in table 4.2).

non-empty kernel of $\mathbf{A}$. Here $\mathbf{z}$ is obtained as the negative gradient of the potential $H(\mathbf{w}) = \sum_{i=1}^{4} h(w_i)$,

$$h(w_i) = \begin{cases} k_{h_1} \tan^2(\gamma_1 |w_i| + \gamma_2) & w_{min} < |w_i| \leq w_{rest}, \\ k_{h_2} (|w_i| - w_{rest})^2 - \frac{(|w_i| - w_{rest})^2}{|w_i| - w_{max}} & |w_i| > w_{rest}, \end{cases} \tag{4.38}$$

with $\gamma_1 = \frac{\pi}{2(w_{rest} - w_{min})}$, $\gamma_2 = -\gamma_1 w_{rest}$ and $k_1 > 0$, $k_2 > 0$ tuning gains. In eq (4.38), w_{min} is the minimum allowed spinning rate (*hard* constraint), $w_{rest} > w_{min}$ is instead the "rest" speed, for example the one required for hovering. Finally, $w_{max} > w_{rest}$ it the maximum propeller speed. Such a cost function is meant to keep the propeller speed inside the operative range with *asymmetric* profile. The cost increases quadratically for rates above w_{rest} while it behaves as a hard barrier function as w_i approaches w_{min}.

4.4.5 Discussion

The simulations results, in terms of average position error $\hat{e}_p$ and average orientation error $\hat{e}_\vartheta$, are reported in table 4.4. If the controller was unable to track the relative trajectory, '-' is used. The behavior of the position error for some trajectories is also shown in figure 4.4. The NMPC control scheme is able to track with negligible error up to the fourth trajectory and to stabilize the system up to the fifth one. On the other hand, the FL controller is only

Table 4.5: Average position and attitude error in presence of parameter uncertainties on the second trajectory.

		$\hat{e}_p$ [cm]		$\hat{e}_\vartheta$ [mrad]	
		NMPC	**FL**	**NMPC**	**FL**
m_b	$+5\,\%$	9.7	2.9	12	81
	$+10\,\%$	18	4.0	11	71
	$+15\,\%$	26	5.5	11	63
	$+20\,\%$	33	7.2	10	56
k_m	$+5\,\%$	3.4	2.7	17.4	104
	$+10\,\%$	3.7	-	29.2	-
	$+15\,\%$	4.2	-	42	-
	$+20\,\%$	8.9	-	59	-

able to stabilize and track the first two trajectories. The reason is that the FL control (4.37) is not able to cope well with input and state saturations and even if the corrective term $\mathbf{z}$ is employed, it is not trivial how to choose it to remain inside the feasibility sets. On the other hand, the predictive abilities of the NMPC allow it to keep the trajectory even when the setpoints require the saturation of the actuators as shown in figure 4.5.

As perfect knowledge of the model parameters is practically impossible, robustness of the proposed NMPC controller is evaluated on the second trajectory for different values of quadrotor mass m_b and thrust coefficient k_m. The cumulative results are reported in table 4.5 while the behavior of the tracking error under different thrust coefficients is shown in figure 4.6 Even with $20\,\%$ error on the value of m_b, both controllers are able to stabilise and track the trajectory. The FL performs better in terms of position error while the NMPC has better results in keeping the desired attitude. On the other hand, even small variations on the motor thrust constant k_m leads to the instability of the FL controller while the *integral* action of the NMPC efficiently compensates for them.

A few final remarks a due. This chapter proposed a Nonlinear Model Predictive Control approach for the motor-level control of tilting quadrotor as an interesting alternative to coplanar and tilted multicopters. The simulations results are promising and show how NMPC is the right tool to tackle the intrinsic complexity of the tilting quadrotor and even under challenging tra-

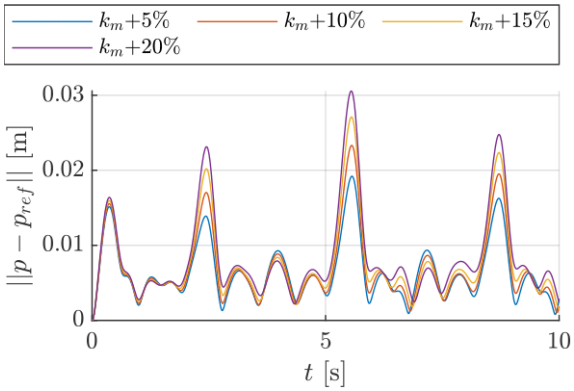


Figure 4.6: Position error e_p of the NMPC when the motor thrust constant k_m is not perfectly known.

jectories. Nevertheless, tuning and real time implementation issues remains; the last column of table 4.4 show that even the most challenging trajectory was tracked if and *extended* (NMPC*e*) prediction horizon of $N = 50$ samples was adopted. The additional computational burden makes this solution not feasible to be implemented onboard as the computational time would exceed the maximum allowed. On one hand, non-uniform sampling grids could be adopted, or better NMPC toolboxes could be tested, such as *acados*. The next chapter is for this reason entirely devolved to the implementation problem of NMPC algorithms for heterogeneous multi-agent system.

5 Bearing based autonomous landing

While in the last chapter an NMPC controller for a tiling quadrotor was proposed to address the limitations of standard planar quadrotors and the complexity and inefficiency of tilted multicopters, in this last chapter a multi agent application, namely the autonomous landing of a drone on a moving platform, is addressed together with experimental validation.

5.1 Introduction

In addition to the specific mission accomplishment, UAVs must be able to perform take-off and landing autonomously. Landing accuracy, in particular, is a crucial aspect of the autonomous vehicles management, as they often need to reach predetermined points for returning to base, payload unloading/loading, recharging, and others. In recent years, there has been a focus on the problem of landing on moving targets, which yields multiple potential advantageous features, such as recharging the UAV at mobile stations rather than fixed, in long-distance missions, or landing on a moving supporting vehicles in emergency scenarios (Grlj et al., 2022).

In this chapter, an hybrid architecture, composed by an high level Nonlinear Model Predictive Control (NMPC) and low level PID controllers, is exploited to perform the autonomous landing of a UAV on a moving target. Differently from the previous chapter, here then NMPC will not directly control the propeller spinning rates as that will be the role of the low level PID controllers. The NMPC will operate at an higher level, reducing its computational burden and allowing of larger sampling times. In the considered scenario, no information is exchanged between the aerial vehicle and the landing platform, and only bearing and elevation measurements are acquired. In particular, no GNSS or other absolute positioning devices (for outdoor or indoor scenarios) are adopted. Compared with chapter 4, the focus here is not on the control

of a challenging platform but instead it is about showing how NMPC can be used as trajectory generator for a multi agent application and how this affects the transition from simulation to real experiment.

Related works - Several contributions have been proposed in the automation and robotics literature dealing with the landing of a UAV on a stationary target, illustrating different modeling, perception and control strategies: for instance, in (Li et al., 2019) the problem of landing is addressed by proposing an AprilTag vision-based approach, in (Wynn and McLain, 2019) the landing maneuver is tested in different light conditions (day and night), whereas in (Shi et al., 2019) a learning-based approach is used to perform a precise landing. Conversely, autonomous landing of a multi-rotor UAV on a moving area by estimating its relative position has been an increasingly popular research area of the last years. Indeed, many of these works use a global positioning system combined with vision techniques to accurately estimate the relative position between the UAV and the target (Xuan-Mung et al., 2020; Baca et al., 2019). However, in some scenarios the use of GNSS or similar systems is not feasible as they cannot be available or reliable due to uncovered areas, malicious attacks, urban structures, or faulty hardware. Common methods to avoid the employment of GNSS are based on the use of alternative sensors such as standard and/or depth cameras, lightweight LiDARs, rangefinders, or ultra-wideband sensors. Accordingly, several procedures based on PID loops to control the UAV dynamics and to accomplish the landing on a moving target have been presented, assessing the use of different types of sensors. With the focus on standard vision methodologies, we mention in this sense (Xuan-Mung et al., 2020), where a visual servoing approach is proposed basing again on the AprilTag system, and (Lin et al., 2017), where computer vision algorithms are used to detect the landing pad characterized by known features.

An alternative and overarching control approach, as introduced in chapter 4, is based on MPC, which allows the UAVs dynamics and actuation limits to be taken into account under a planning and control unified framework. With respect to the application scenario of interest, in (Mohammadi et al., 2020) a linear MPC is used to land on a mobile platform performing a straight trajectory, while in (Paris et al., 2020) the authors focus on the robustness issue in designing an MPC controller to land in wind conditions.

The solution proposed in this chapter is developed along this methodological direction and to this aim the rest of this work is structured as follows: section 5.2 models the considered UAV platform, landing target, and their mutual interaction; section 5.3 presents the considered tracking and landing problem and introduces the adopted control architecture; section 5.4 discusses the experimental results.

5.2 Problem formulation and modeling

A standard, under-actuated coplanar quadrotor (Nascimento and Saska, 2019) tasked to track and to land on a moving platform (*target*) is considered. In order to do so, the UAV can exploit only bearing measurements obtained, for example, from a calibrated camera, its altitude from the ground, and internal sensor data coming from an Inertial Measurement Unit (IMU). In the following, UAV and target will be referred generically as *agents*, acting in the given scenario.

5.2.1 Quadrotor UAV model

To derive the dynamic model of the quadrotor the *Newton-Euler* formalism is applied in the same way it was used in chapter 4. Let $\mathcal{F}_o$ denote the common inertial frame and $\mathcal{F}_B$ the quadrotor body-fixed frame, whose origin O_B is at the center of mass (CoM) of the UAV. Moreover, let $\mathbf{p} \in \mathbb{R}^3$ and $\mathbf{v} \in \mathbb{R}^3$ be respectively, the position and the linear velocity of O_B in the inertial frame $\mathcal{F}_o$. Regarding the UAV attitude, the quaternion representation is once again used in order to benefit from its computational efficiency; $\mathbf{q} \in \mathbb{S}^3$ represents the orientation of $\mathcal{F}_B$ w.r.t. $\mathcal{F}_o$ while $\boldsymbol{\omega} \in \mathbb{R}^3$ represents the angular velocity of the body-fixed frame with respect to the inertial world frame expressed in $\mathcal{F}_B$.

The motion equations are derived considering the forces and torques that are generated by the propellers, which are all assumed to have parallel rotation axes $\mathbf{e}_3 = [0\ 0\ 1]^\top$ w.r.t. $\mathcal{F}_B$ and with unidirectional spinning motion, i.e. $w_i > 0,\ i = 1...4$, where w_i are the propellers spinning rates. Under this assumption, each propeller applies a thrust force $f_i = k_t w_i^2$ and a drag torque $\tau_i =$

$k_m w_i^2$ along $\mathbf{e}_3$; k_t and k_m are respectively the thrust and drag aerodynamic coefficients as in chapter 4. Combining all propeller contributions it is possible to determine the total vertical thrust force and total steering moment acting on the CoM of the quadrotor w.r.t. the body-fixed frame,

$$f_z = \sum_{i=1}^{4} k_t w_i^2, \tag{5.1}$$

$$\boldsymbol{\tau} = \sum_{i=1}^{4} (k_t \mathbf{p}_i \times \mathbf{e}_3 + k_m \mathbf{e}_3) w_i^2, \tag{5.2}$$

where $\mathbf{p}_i$ is the position of the i-th propeller in body-fixed frame and $\times$ is the outer-product between vectors. The overall quadrotor model is hence described by

$$\begin{aligned} \dot{\mathbf{p}} &= \mathbf{v} \\ m\dot{\mathbf{v}} &= f_z \mathbf{R}(\mathbf{q}) \mathbf{e}_3 - m\mathbf{g} \\ \dot{\mathbf{q}} &= 1/2\, \mathbf{M}(\mathbf{q}) \boldsymbol{\omega}_B \\ \mathbf{J}\dot{\boldsymbol{\omega}}_B &= \boldsymbol{\tau} - \boldsymbol{\omega}_B \times \mathbf{J}_B \boldsymbol{\omega}_B \end{aligned} \tag{5.3}$$

where $m > 0$ is the quadrotor mass and $\mathbf{J}_B \in \mathbb{R}^{3\times 3}$ is the inertia matrix with respect to the body frame $\mathcal{F}_B$. Once the geometrical and aerodynamic parameters of the quadrotor are known (by analysis or by estimation) it is possible to design an invertible map (*mixer*) that translates the input vector of the UAV from $[w_1^2\ w_2^2\ w_3^2\ w_4^2]^\top \in \mathbb{R}^4_+$ to $[f_z\ \boldsymbol{\tau}]^\top \in \mathbb{R}^4$. Indeed combining eqs. (5.1) and (5.2) yields

$$\begin{bmatrix} f_z \\ \boldsymbol{\tau} \end{bmatrix} = \underbrace{\begin{bmatrix} \begin{bmatrix} k_t & k_t & k_t & k_t \end{bmatrix} \\ -k_t \left[\mathbf{e}_z\right]_\times \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix} + \begin{bmatrix} k_m & k_m & k_m & k_m \end{bmatrix} \mathbf{e}_3 \end{bmatrix}}_{=:\mathbf{F}} \begin{bmatrix} w_1^2 \\ w_2^2 \\ w_3^2 \\ w_4^2 \end{bmatrix}, \tag{5.4}$$

and therefore $[w_1^2\ w_2^2\ w_3^2\ w_4^2]^\top = \mathbf{F}^{-1} \begin{bmatrix} f_z \\ \boldsymbol{\tau} \end{bmatrix}$. Unless degenerate configurations are chosen, i.e. $\mathbf{p}_i = a\,\mathbf{p}_j$, $i \neq j$ and $a > 0$, then *almost* any positive thrust and three dimensional torque can be allocated with a suitable combination of square propeller speeds. Note that, eq (5.3) could also be derived imposing $\boldsymbol{\alpha} = \mathbf{0}$ in eq (4.23) at page 62.

5.2.2 Target model

The target platform is modeled as a massless point subject to an unknown but bounded acceleration $\mathbf{a}_t = [a_{t,x}\ a_{t,y}\ 0]^\top \in \mathbb{R}^3$. Therefore, its dynamical equations are

$$\begin{aligned} \dot{\mathbf{p}}_t &= \mathbf{v}_t, \\ \dot{\mathbf{v}}_t &= \mathbf{a}_t, \end{aligned} \tag{5.5}$$

where $\mathbf{p}_t, \mathbf{v}_t \in \mathbb{R}^3$ are the position and the velocity of the target platform in the inertial frame $\mathcal{F}_o$, respectively. The constraint imposing that the target platform moves only on the plane $\mathbf{e}_3^\top \mathbf{p}_t = 0$ is enforced by the initial conditions $p_{t,z}(0) = v_{t,z}(0) = 0$.

Observation. It is rather uncommon to have a target that can be modeled as an *omnidirectional* ground vehicle. However, unconstraining the target actuation properties stresses the UAV controller and allows to better evaluate its robustness.

5.2.3 Agents' interactions

In order to track and land over the target platform, the quadrotor needs to sense it. In particular, the quadrotor is capable of measuring the bearing vector $\mathbf{b} \in \mathbb{S}^2$ between its CoM and the target expressed in its local frame, namely

$$\mathbf{b} = \mathbf{R}(\mathbf{q})^\top \frac{\mathbf{p}_t - \mathbf{p}}{\|\mathbf{p}_t - \mathbf{p}\|} \in \mathbb{S}^2, \tag{5.6}$$

like the heterogeneous formations in chapter 2. Moreover, the UAV is capable of measuring its altitude w.r.t. the ground, i.e. $p_z > 0$. The former (bearing measurement) can be acquired using an optical camera while the latter (altitude measurement) can be taken by simple range sensors or barometers. All these sensing devices are nowadays standard add-on for quadrotors, with different realizations according to their indoor/outdoor use. The given quantities, stating the relations between the UAV and target agent, are schematically shown in figure 5.1.

It is important to notice that no data is exchanged between the UAV and the target: in particular, the target motion is unknown and the drone has

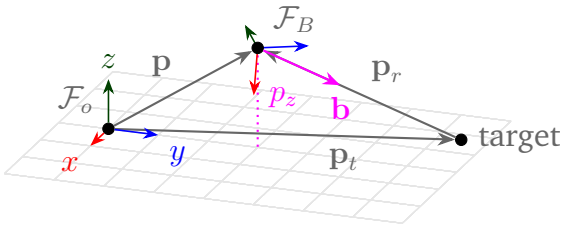


Figure 5.1: Quadrotor sensing: altitude p_z and bearing vector $\mathbf{b}$.

no direct information about the absolute target position or velocity. At the same time, the drone is not supposed to receive data coming from off-board sensing infrastructures (such as GNSS systems) about its own state.

Given the quadrotor and the target states, it is possible to introduce the *relative* position and velocity vectors as

$$\mathbf{p}_r = \mathbf{p} - \mathbf{p}_t \tag{5.7a}$$

$$\mathbf{v}_r = \mathbf{v} - \mathbf{v}_t \tag{5.7b}$$

and then, it is possible to define the tracking and landing control problems.

Problem 5.1 (tracking problem)**.** Given an unknown target dynamics as in (5.5), the tracking problem consists on finding a control law for (5.3) that ensures $\mathbf{v}_r = \mathbf{0}$ all the time.

Problem 5.2 (landing problem)**.** Given an unknown target dynamics as in (5.5), the landing problem consist on finding a control law for (5.3) that ensures $\mathbf{p}_r = \mathbf{0}$ asymptotically.

5.3 System overview

The control architecture has been designed taking into account common off-the-shelf quadrotor solutions in which a low level and high level microcontrollers (MCUs) are coupled together. Specifically, the former is responsible to interact with most of the embedded hardware, i.e. IMU, radio receiver and motor controllers, while the latter manages the low level information and exploits estimation, planning, and decision making algorithms to provide

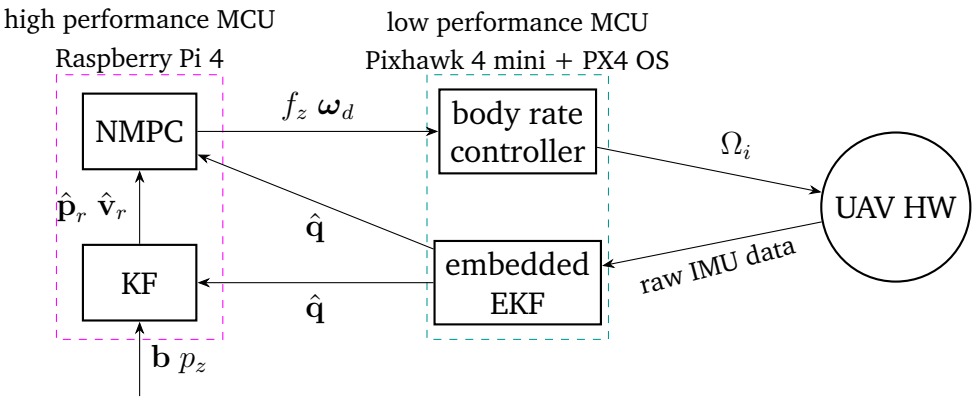


Figure 5.2: Quadrotor control architecture; the first control stage (in light blue) runs on the low level controller and interact with the drivers and sensors. The second control stage (in magenta) runs on the high level controller, provides setpoints and manages the inter-agent behavior.

the necessary setpoints to accomplish the desired tasks. In this perspective, the overall control scheme is shown in figure 5.2. First of all a *relative pose estimator* (Kalman Filter block, KF) uses the information coming from the measurements $\hat{\mathbf{q}} \in \mathbb{S}^3$, $\mathbf{b} \in \mathbb{S}^2$ and $p_z \in \mathbb{R}$ to get the estimate $\hat{\mathbf{p}}_r,\ \hat{\mathbf{v}}_r \in \mathbb{R}^3$ of the relative state. Then, a *Non Linear Model Predictive Control* (NMPC block) is adopted to drive the UAV dynamics such that these estimates converge towards the task specific planned values. The NMPC outputs the common thrust $f_z > 0$ as in (5.3) and the desired body angular rates $\boldsymbol{\omega}_d \in \mathbb{R}^3$, i.e., $\mathbf{u} = [f_z\ \boldsymbol{\omega}_d^\top]^\top$. A low level controller generates the required torque setpoints $\boldsymbol{\tau} \in \mathbb{R}^3$ in order to track $\boldsymbol{\omega}_d \in \mathbb{R}^3$ while the embedded Extended Kalman Filter (EKF) processes the data coming from the IMU and estimates the quadrotor attitude $\hat{\mathbf{q}} \in \mathbb{S}^3$.

5.3.1 Relative pose estimation

The relative pose estimator runs on the high performance microcontroller. It is composed by two blocks, the *relative positon decoder* and a *Kalman Filter*. By measuring the bearing vector $\mathbf{b} \in \mathbb{S}^2$ and the altitude p_z, with an estimate of the quadrotor attitude $\hat{\mathbf{q}} \in \mathbb{S}^3$ coming from the embedded EKF, it is possible to get an estimate of the relative position $\mathbf{p}'_r \in \mathbb{R}^3$ between the drone and the target as

$$\mathbf{p}'_r = p_z \frac{\mathbf{R}(\hat{\mathbf{q}})\mathbf{b}}{\mathbf{e}_3^\top \mathbf{R}(\hat{\mathbf{q}})\mathbf{b}}, \tag{5.8}$$

as long as p_z is non-zero. This quantity is then provided to a discrete-time KF in order to have a better estimate of the whole relative pose $(\hat{\mathbf{p}}_r,\ \hat{\mathbf{v}}_r)$. The adopted model in the filter has equations

$$\begin{aligned}\hat{\mathbf{p}}_r((k+1)T_s) &= \hat{\mathbf{p}}_r(k\,T_s) + T_s\hat{\mathbf{v}}_r(k\,T_s) + \mathbf{w}_p(k\,T_s)\\ \hat{\mathbf{v}}_r((k+1)T_s) &= \mathbf{v}_r(k\,T_s) + \mathbf{w}_v(k\,T_s)\\ \mathbf{p}'_r(k\,T_s) &= \hat{\mathbf{p}}_r(k\,T_s) + \mathbf{r}(k\,T_s)\end{aligned} \tag{5.9}$$

which represents a standard discrete-time double integrator subject to process $\mathbf{w}_p(k\,T_s) \sim \mathcal{N}(0, \mathbf{Q}_p)$, $\mathbf{w}_v(k\,T_s) \sim \mathcal{N}(0, \mathbf{Q}_v)$ and measurement $\mathbf{r}(k\,T_s) \sim \mathcal{N}(0, \mathbf{R})$ noises, where $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes a Gaussian vector with mean $\boldsymbol{\mu} \in \mathbb{R}^n$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$. This choice is motivated by the complete lack of knowledge about the dynamics of the target (from the quadrotor side), which is modeled as noise in the filter.

About the gaussianity of the measurement noise - the underlying assumption that $\mathbf{p}'_r$ is normally distributed with zero mean and covariance matrix $\mathbf{R}$ does not hold in general, but it is needed by the Kalman filter. Inspecting eq 5.8, three noisy component, the measured altitude p_z, the measured baring $\mathbf{b}$ and estimated attitude $\hat{\mathbf{q}}$ are nonlinearly combined together. Figure 5.3 shows the empirical distribution of $\mathbf{p}'_t = \mathbf{p} - \mathbf{p}'_r$, where $\mathbf{p}'_r$ comes from eq (5.8) assuming that the true landing platform is in $\mathbf{p} = [0\ 0\ 0]^\top$. Nine different cases are consider, each cases has the UAV placed at increasing distance to the target, $\mathbf{p} = [p_x\ 0\ p_z]^\top$, with vertical position fixes at $p_z = 5\,\mathrm{m}$ and $p_x = k\,2.5\,\mathrm{m}$, $k \in \{0, \dots 8\}$. In each case the UAV altitude and attitude are assumed to be perfectly known so that the only source of noise is due to the bearing measurements, which are perturbed with the same technique as in chapter 3 and standard deviation $\sigma_b = 5°$. For each case, $N = 1000$ independent noise realizations are collected. As it is expected from noise affecting bearing measurements, the more the UAV moves away from the landing platform, the more spread apart the estimated position are. On the other hand, the more the bearing vector deviates from the vertical direction (case $p_x = 0\,\mathrm{m}$) the less the distribution can be approximated with a Gaussian. In conclusion, the best performances and statistical properties are to be expected when the UAV is directly above the landing platform.

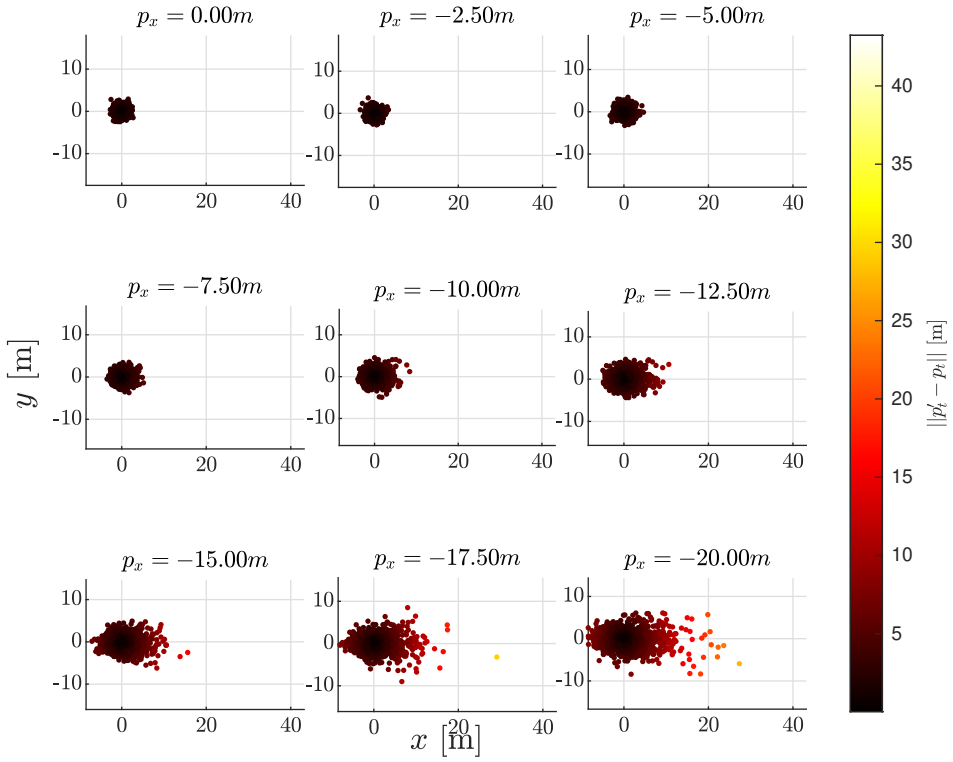


Figure 5.3: Empirical distribution of $\mathbf{p}'_t = \mathbf{p} - \mathbf{p}'_r$ for different UAV positions. The UAV altitude and attitude are noiseless while the bearing measurements are affected by Gaussian noise. As the UAV moves away from the target platform the distribution loses its Gaussian shape.

5.3.2 Nonlinear model predictive control

The continuous time NMPC approach is based on solving iteratively, at each time instant t, the same *Nonlinear Programming problem* as in eq (4.27) of chapter 4, with, of course, different model function, cost, and constraints. In particular, the MPC state is now composed by the relative state $\mathbf{p}_r$, $\mathbf{v}_r$ and by the quadrotor attitude, i.e., $\mathbf{x}_{mpc} = [\mathbf{p}_r^\top \ \mathbf{v}_r^\top \ \mathbf{q}^\top]^\top$ while MPC input is $\mathbf{u}_{mpc} = [f_z \ \boldsymbol{\omega}_d^\top]^\top$, this leads to the differential equation (4.28)

$$\dot{\mathbf{x}}_{mpc} = \begin{bmatrix} \dot{\mathbf{p}}_r \\ \dot{\mathbf{v}}_r \\ \dot{\mathbf{q}} \end{bmatrix} = \mathbf{f}_{mpc}(\mathbf{x}_{mpc}, \mathbf{u}_{mpc}) = \begin{bmatrix} \mathbf{v}_r \\ \frac{f_z}{m}\mathbf{R}(\mathbf{q})\mathbf{e}_3 - \mathbf{g} - \mathbf{a}_t \\ 1/2\,\mathbf{M}(\mathbf{q})\boldsymbol{\omega}_d \end{bmatrix} \tag{5.10}$$

noting, however, that during the prediction horizon the target velocity is assumed constant, hence $\mathbf{a}_t = \mathbf{0}$.

The cost function, instead, is designed to address the tracking and landing problems. The latter can be solved by making the controller command the quadrotor to a desired relative position $\mathbf{p}_r^d \in \mathbb{R}^3$ in the target reference frame. This task can be achieved by minimizing the relative position error $\mathbf{e}_\mathbf{p} = \mathbf{p}_r^d - \mathbf{p}_r$. In addition, the NMPC needs to accomplish the tracking task by penalizing the relative velocity error $\mathbf{e}_\mathbf{v} = \mathbf{v}_r^d - \mathbf{v}_r$. It is important to ensure that the norm of the quaternion remains unitary, so that the physical sense of representing a rotation is formally maintained. Indeed, the discretization of the quaternion dynamics does not ensure that $\mathbf{q}(t) \in \mathbb{S}^3 \ \forall t$. To tackle this issue the *soft constraint* approach has been chosen, therefore adding to the cost function a term proportional to $e_\mathbf{q} = \mathbf{q}^{-1}\mathbf{q} - 1$. In this way, there is no strict guarantee that the quadrotor attitude will remain consistently valid but, expecting its norm to drift slowly and applying the receding horizon paradigm, it is still safe to adopt a soft constraint strategy. Finally, a weight on the input is added; the NMPC should try to not deviate too much from a reference input $\mathbf{u}_{ref} = [mg \ 0 \ 0 \ 0]^\top \in \mathbb{R}^4$, where mg is the feedforward (known) term compensating the gravity, and the reference angular velocity is zero. This behavior is imposed by the cost components $e_{f_z} = f_z - mg$ and

$\mathbf{e}_{\mathbf{w}_d} = \mathbf{w}_d - \mathbf{0}$. Combining together all the aforementioned terms, the output and final stage output functions (4.27a) take expression

$$\begin{aligned} \mathbf{h}_k(\mathbf{x}_k, \mathbf{u}_k) &= \begin{bmatrix} \mathbf{e}_{\mathbf{p},k}^\top & \mathbf{e}_{\mathbf{v},k}^\top & e_{\mathbf{q},k} & e_{f_z,k} & \mathbf{e}_{\mathbf{w}_d,k}^\top \end{bmatrix}^\top \\ \mathbf{h}_N(\mathbf{x}_N) &= \begin{bmatrix} \mathbf{e}_{\mathbf{p},N}^\top & \mathbf{e}_{\mathbf{v},N}^\top & e_{\mathbf{q},N} \end{bmatrix}^\top \end{aligned} \tag{5.11}$$

where $\mathbf{h} = [\mathbf{e}_\mathbf{p}^\top \ \mathbf{e}_\mathbf{v}^\top \ e_\mathbf{q} \ e_{f_z} \ \mathbf{e}_{\mathbf{w}_d}^\top]^\top$ summarizes to all the discussed contribution errors.

Lastly, the physical limits of the system can and must be taken into account. For this application, only the output saturation is considered and therefore the NMPC constraints results

$$\begin{aligned} 0 \leq f_z &\leq f_z^{max} \\ w_{x,y}^{min} \leq w_{x,y} &\leq w_{x,y}^{max} \\ w_z^{min} \leq w_z &\leq w_z^{max}. \end{aligned} \tag{5.12}$$

One can argue that also state constraints should be enforced, such as limited roll and pitch angles. However, these limits are implicitly considered with the minimization of the cost function.

5.4 Performance assessment

In this section the implementation of the proposed tracking and landing controller is presented, together with the experimental setup and the obtained results.

5.4.1 Experimental setup

The experiments have been carried out at the *SPARCS laboratory* of the Department of Information Engineering of the University of Padova, Italy; the target consists of a $52\,\mathrm{cm} \times 55\,\mathrm{cm}$ platform mounted on four caster wheels and towed by a custom unicycle robot (turtlebot), as shown in figure 1.3 in page 7 and figure 1.1 in page 5. As for the drone, a QAV250 by HOLYBRO (hol) equipped with a Raspberry Pi 4 has been adopted. The low level

microcontroller is represented by a *Pixhawk 4 mini* in which *PX4 autopilot* (PX4) runs; the autopilot is responsible for the embedded EKF filtering the IMU measurements, the body rates controller, and the safety-related controls, such as failure detection and arming protection. On the other hand, the NMPC and the relative state KF algorithm (section 5.3.1) are deployed on the Raspberry Pi 4 and the *Robotic Operative System* (ROS) (ROS) is exploited to connect the two components. In particular, using a serial link and the *mavros* ROS package the internal states and variables of PX4 are made available to the Raspberry Pi; among them, there are the estimated attitude $\hat{\mathbf{q}} \in \mathbb{S}^3$ and the thrust and body rates setpoints, respectively $f_z > 0$ and $\boldsymbol{\omega}_d \in \mathbb{R}^3$.

A *Motion Capture System* (MOCAP) composed by ten *Vicon* infrared cameras provides ground-truth localization of target and UAV; furthermore, the MOCAP is used to mimic the bearing and altitude measurements, being the QAV250 not equipped with any camera or rangefinder.

5.4.2 Controller implementation

The controller is realized in SIMULINK inside a ROS node using the ROS toolbox, while the NMPC is implemented using the ACADO toolbox (Houska et al., 2011). The detailed node generation pipeline is the following:

1. By using the ACADO toolbox the MPC problem (4.27) is formulated directly as a MATLAB script.

2. The toolbox generates a standalone C code solving the MPC problem.

3. The code is then compiled in a MEX function added to the SIMULINK model that realizes the ROS node and the KF.

4. At this point it is possible to run the node inside SIMULINK, perform debugging and parameter tuning.

5. Finally, using the SIMULINK Coder, the model can be converted into a standalone ROS node and deployed inside the Raspberry Pi 4. Notice that the code of the MPC solver is not re-generated: the one generated by ACADO is directly added to the SIMULINK one.

This approach brings the advantages of rapid prototyping, because the high-level control design and tuning are performed in SIMULINK and, at the same time, yields high-efficiency, because the low level MPC routines are directly generated as C code by ACADO. Considering the MPC implementation, the objective function is minimized using the multiple shooting discretization, fixed step Runge-Kutta integrator and SQP (sequential quadratic programming) solver. The qpOASES library, which deals with *dense QP* problems, is used to solve the associated QP problem. These hyperparameters, together with the model equations (5.10), the cost function (5.11), the constraints structure (5.12) and the prediction horizon N and sampling time T_s are specified in phase 1. At the same time both the (discretized) cost matrices composing $\mathbf{W}$ in (5.11) (namely: $\mathbf{W}_{\mathbf{p}_r}$, $\mathbf{W}_{\mathbf{v}_r}$, $\mathbf{W}_{e_\mathbf{q}}$, $\mathbf{W}_{\mathbf{e}_{f_z}}$, $\mathbf{W}_{\mathbf{e}_\mathbf{w}}$) and the constraint values in (5.12) can be changed online, meaning that not only they can be tuned in SIMULINK without having to re-generate the NMPC solver, but they can be potentially modified in real time. This greatly speeds up the node deployment: if no model adjustment are needed, the NMPC solver needs to be generated, compiled as MEX file and compiled for the Raspberry Pi 4 only once.

For these experiment ACADO was preferred over MATMPC because it generates all MPC routines in pure C code that than be immediately compiled on the Raspberry Pi. On the other hand, MATMPC adopts and hybrid approach: it only generates C code for the low level routines, such as the model update equations, their sensitivities and the cost function evaluations. The solvers are also in pure C code, but those are interchangeable and independent add-ons. The rest of the code is written as MATLAB scripts and functions and this makes quite hard migrating it to a full standalone, C version.

5.4.3 Validation scenario definition

The experiments consist on the drone that, after a vertical takeoff at $p_{z,ref} = 2\,\mathrm{m}$, moves over the target, which initially is static, and starts the tracking procedure with $p_{z,ref}$ as target altitude. At this point, the target starts moving following a rounded rectangular trajectory, which is visible in figure 5.4 for the initial simulation and in figure 5.7 the the experimental validation. After an initial tracking phase, in which the drone tries to stay over the landing platform, it switches to landing mode towards the task completion. After an

Table 5.1: Simulation and experiment common parameters

quantity	value
N	50
T_s	$30\,\mathrm{ms}$
$w_{xy}^{max} = w_z^{max} = -w_{xy}^{min} = -w_z^{min}$	$1.5\,\mathrm{rads}^{-1}$
KF process noise covariance: position term	$0.01\,\mathrm{m}^2\,\mathbf{I}_3$
KF process noise covariance: velocity term	$1\,\mathrm{m}^2\,\mathbf{I}_3$
KF measurement noise covariance	$1 \times 10^{-6}\,\mathrm{m}^2\,\mathbf{I}_3$

Table 5.2: NMPC tuning parameters

	simulation	experiment
$f_z^{max}[\mathrm{N\,m}]$	20	15
$\mathbf{p}_r$-term $[\mathrm{m}^2]$	$\mathrm{diag}\{1,\,1,\,2\}$	$\mathrm{diag}\{4,\,4,\,8\}$
$\mathbf{v}_r$-term $[\mathrm{m}^2\mathrm{s}^{-2}]$	$\mathrm{diag}\{0.1,\,0.1,\,0.1\}$	$\mathrm{diag}\{0.2,\,0.2,\,0.5\}$
$e_{\mathbf{q}}$-term []	1	1
$\mathbf{e}_{\mathbf{w}}$-term $[\mathrm{rad}^2\mathrm{s}^{-2}]$	$\mathrm{diag}\{1,\,1,\,1\}$	$\mathrm{diag}\{1,\,1,\,1\}$
$\mathbf{e}_{f_z}$-term $[\mathrm{N}^2]$	0.1	0.1

initial tuning procedure, first performed using the robotic simulator GAZEBO, and then on the experimental hardware, the NMPC variable parameters have been found as reported in table 5.2. For the landing phase, the relative altitude reference $p_{r,z}$ is generated by a trajectory generator that applies a trapezoidal velocity profile with estimated landing time of $5\,\mathrm{s}$.

5.4.4 GAZEBO realistic simulations

There was no GAZEBO model for the QAV250 available at the time of writing, and it was not possible to create one performing an extensive parameters estimation campaign (inertia matrix and thrust and torque coefficients). Therefore, the model of the IRIS quadrotor has been adopted. The main difference between the two is the size: the QAV250 weights approximately $0.86\,\mathrm{kg}$ while the other weights $1.5\,\mathrm{kg}$, moreover the second is almost twice as wide, going from $25\,\mathrm{cm}$ of diagonal to $50\,\mathrm{cm}$. However, this is not a critical issue: the differences in dynamics are compensated by the action of the embedded low-level PIDs, which are tuned differently for the two drones. Moreover, the NMPC cost weight matrices have been tuned differently and their values are reported in table 5.2.

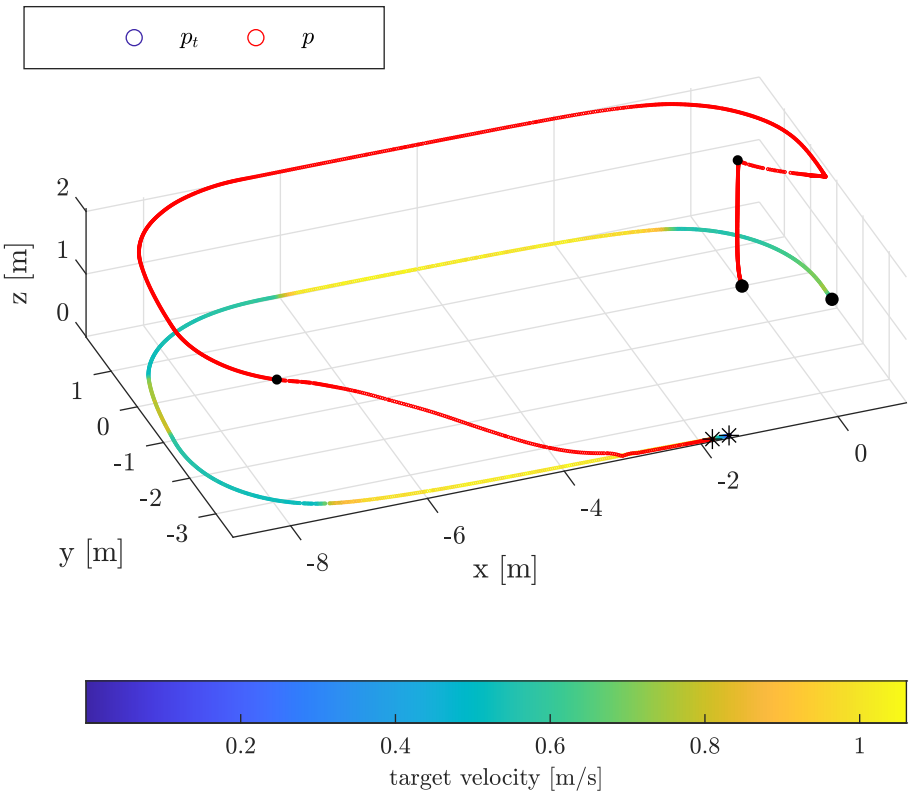


Figure 5.4: GAZEBO simulation. UAV and target trajectories: the big black circles represent the starting points, the two small circles along the drone trajectory represent the start of the tracking and landing phases, respectively; the stars represent the final positions.

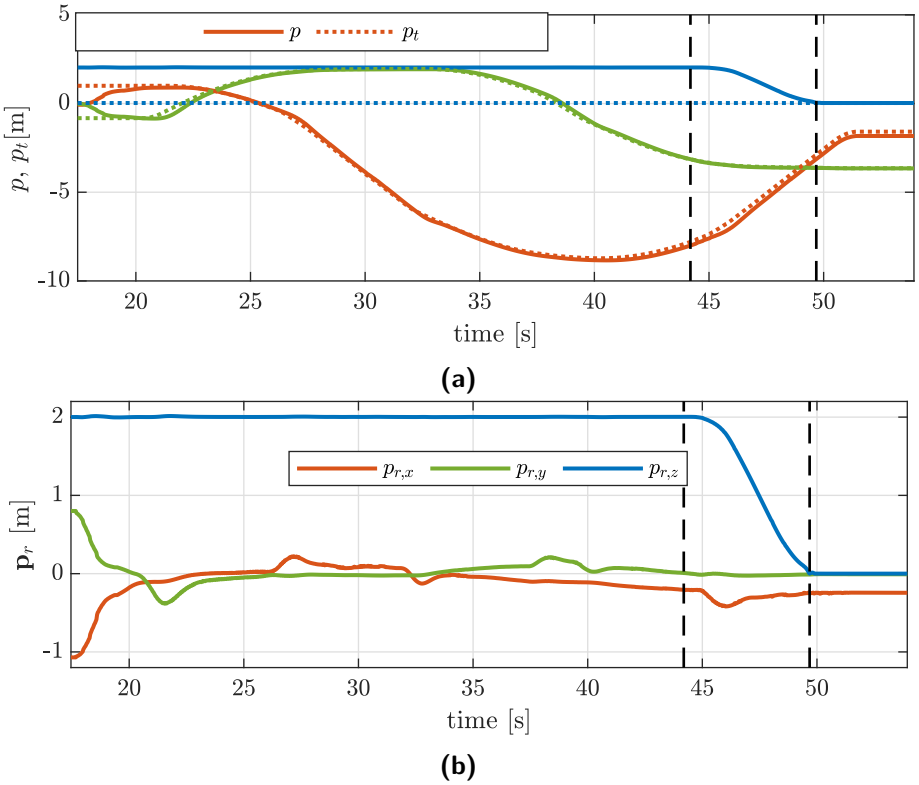


Figure 5.5: GAZEBO simulation. Absolute (a) and relative (b) position values: x-coordinate (red), y-coordinate (green), and z-coordinate (blue).

Figure 5.4 shows the UAV and target trajectories in simulation, together with the target velocity amplitude. The big black circles represent the starting positions while the first small black circle notifies the end of takeoff and the start of tracking, which takes place at $t_{track} = 17.46\,\mathrm{s}$. Once the drone is aligned, the target platform starts moving while the drone keeps tracking it. At $t_{landing} = 44.19\,\mathrm{s}$ the landing command is sent and the drone goes in landing mode (second black circle), which concludes at $t_{landed} = 49.68\,\mathrm{s}$ (black stars). The absolute position of the drone $\mathbf{p}$ and the landing platform $\mathbf{p}_t$ and the relative position $\mathbf{p}_r$ are shown in figure 5.5. It is possible to observe that, once the target platform is tracked (approximately at $t_{tracked} = 20\,\mathrm{s}$), the relative position error never exceeds $d_{max} = 0.4\,\mathrm{m}$ on the xy-plane, with the maximum reached at $21.54\,\mathrm{s}$ when the target shows the maximum acceleration. The segments with constant target velocity, even if above $1\,\mathrm{m/s}$ have much lower tracking error.

It is interesting to notice how the perfect knowledge of the GAZEBO simulation parameters leads to almost zero error in tracking the reference altitude and reference thrust level, as shown in figure 5.6a. Finally, in the same figure, it

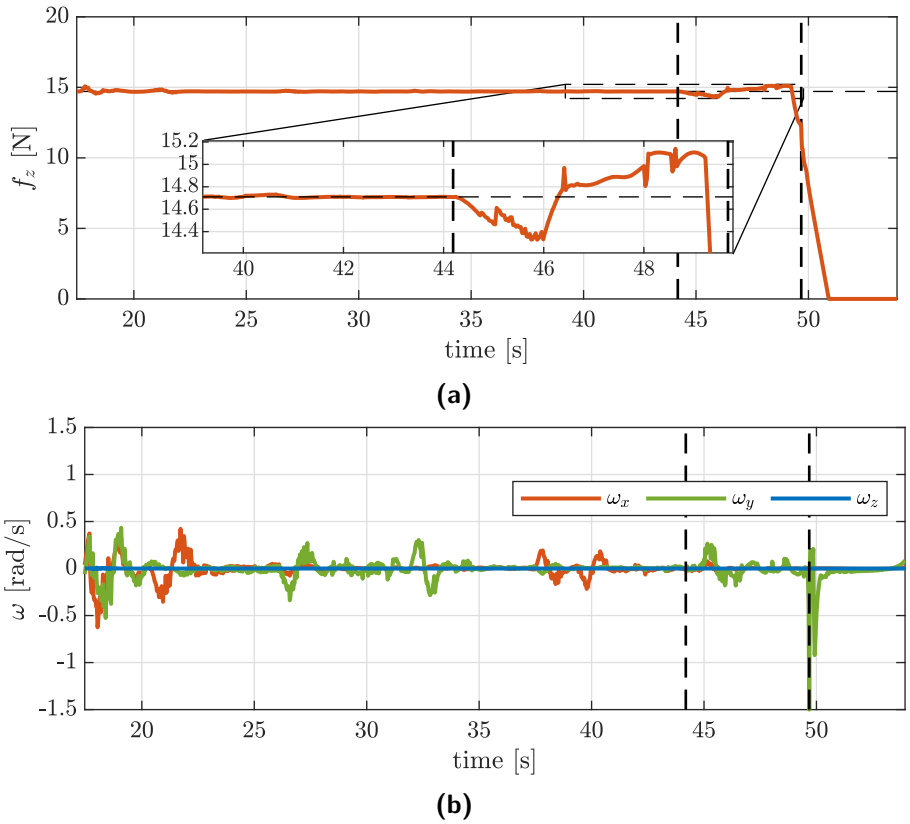


Figure 5.6: GAZEBO simulation. NMPC output values: common thrust (a) and body rate setpoint (b).

is possible to observe how the landing procedure is carried out; initially f_z is lowered to start the descent, then it is increased above the hovering level (dashed black line) to land with almost zero vertical velocity. The desired body rate setpoints $\boldsymbol{\omega}_r$ are instead reported in figure 5.6b. As the air friction is not modeled, the drone does not need to tilt once it reaches the desired horizontal velocity. This can be noticed form the segments in figure 5.6b having reference body rate fixed to zero; the corresponds to the parts of the trajectory where the velocity is constant.

5.4.5 Laboratory experiment

Figure 5.7 shows the trajectories of the QAV250 and the target; the maximum tracking distance on the xy-plane is less then $0.15\,\mathrm{m}$, as shown in figure 5.8 and the UAV safely lands on the target, with just $5\,\mathrm{cm}$ of misalignment. Comparing the Gazebo and the experimental trajectories it is evident how in the former situation all the unmodeled phenomena make the UAV slightly

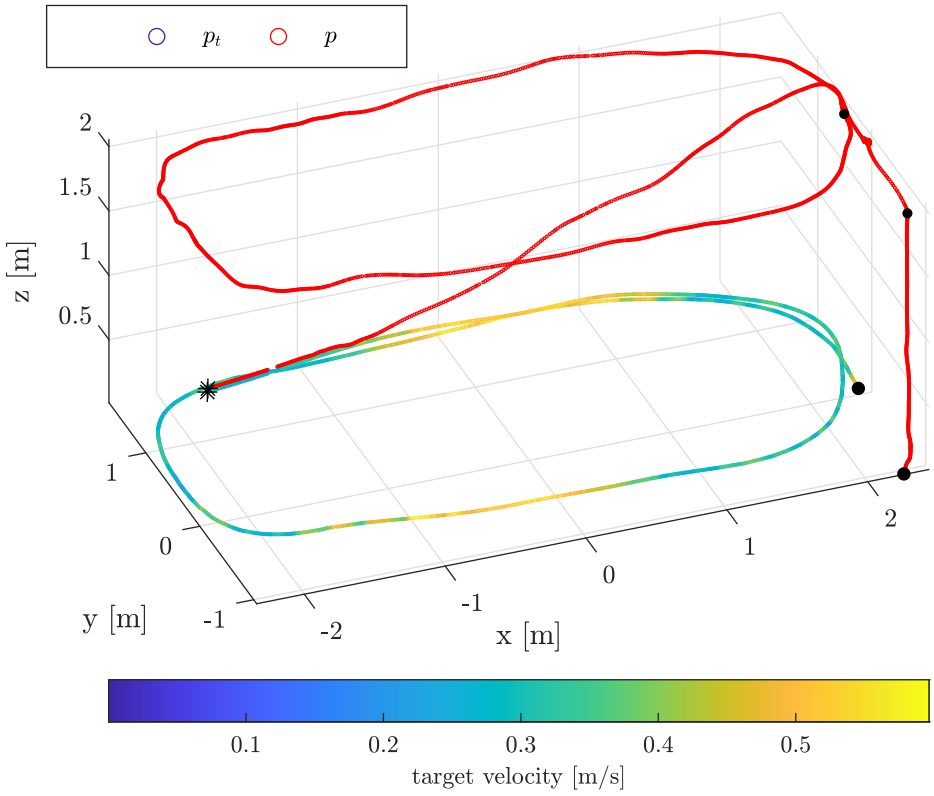


Figure 5.7: Laboratory experiment. UAV and target trajectories: the big black circles represents the starting points, the two small circles on the drone trajectory represents the start of the tracking and landing phases, respectively; the stars represent the final positions.

more unstable; indeed, the drone position $\mathbf{p}$ is more smooth in simulation which implies higher vibrations in the QAV250 attitude. Nevertheless, these results must take into account that the QAV250 is lighter than the IRIS model used in simulation and that, at the same time, a more aggressive NMPC tuning has been adopted for the laboratory experiment.

The biggest difference between the Gazebo simulation and the laboratory experiment is shown in figure 5.9a where one can observe that the NMPC has to request a desired common thrust $f_{z,des}$ $10\,\%$ higher than the nominal value. As the QAV250 mass was correctly measured, the issue arises from a mismatch between the propeller constant k_m used by the NMPC and the true one. Moreover, while the vehicle body rates $\boldsymbol{\omega}$ is internally controlled by the low level PIDs inside the Pixhawk, so that desired reference imposed by the NMPC is to be expected to be tracked with at most a small delay, the common thrust imposed in open loop and there is no feedback measuring it. This makes the propeller constants k_m and k_t critically important. However, a

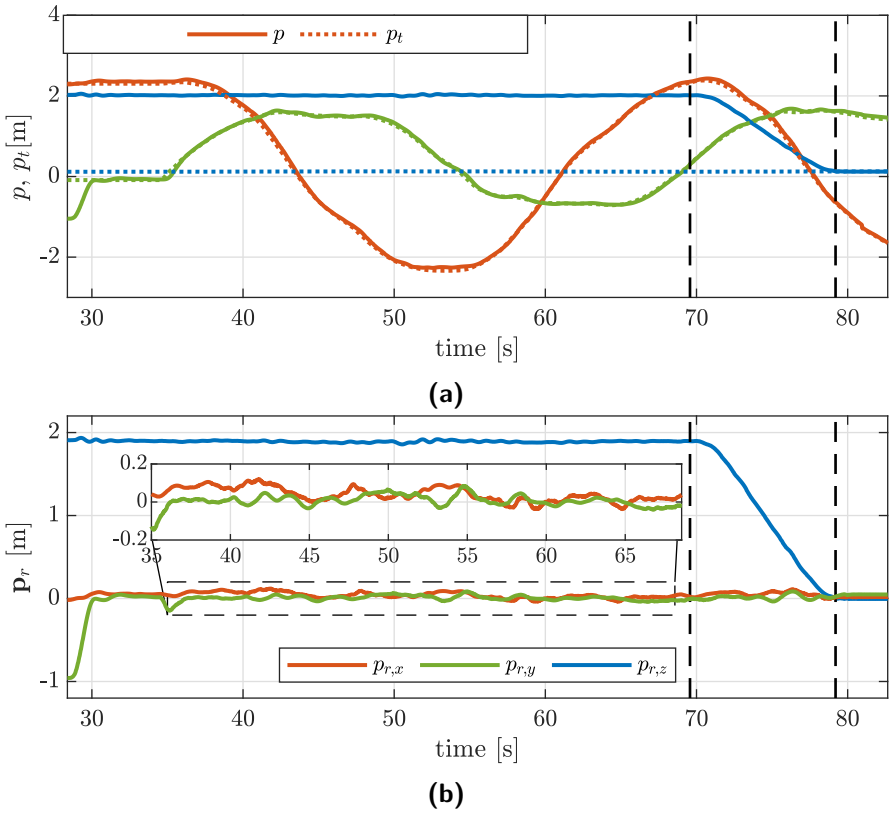
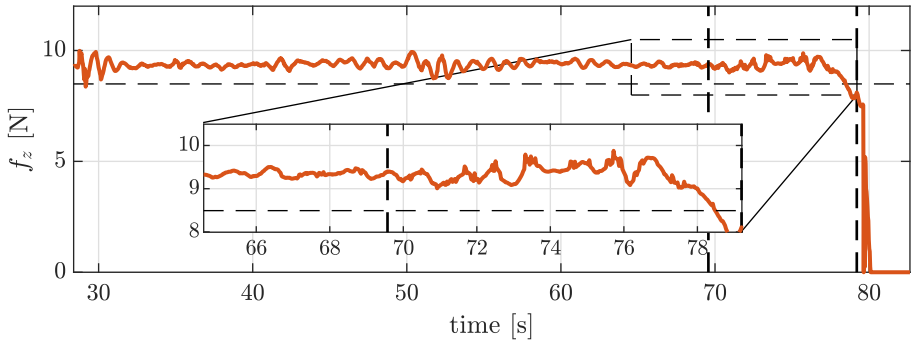
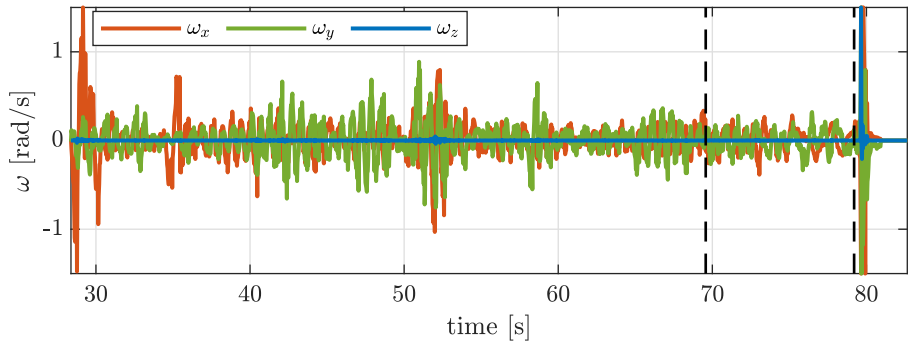


Figure 5.8: Laboratory experiment. Absolute (a) and relative (b) position values: x-coordinate (red), y-coordinate (green), and z-coordinate (blue).



(a)



(b)

Figure 5.9: Laboratory experiment. NMPC output values: common thrust (a) and body rate setpoint (b).

proper estimation of those values is operatively quite challenging as common ESCs applies open loop control even on the propeller spinning rates w_i, which is the affected by other parameters, first of all the battery level and even the ground effect appearing when flying close to the ground.

All these phenomena show their effects on the actual requested thrust and on the reached altitude. Indeed, inspecting figure 5.8b the quadrotor, even though it was tasked to keep an altitude of $2\,\mathrm{m}$ above the target, stops close to $1.9\,\mathrm{m}$. This is also due to the NMPC, which does not implement an integral action like the one used in chapter 4. In practice, this is not an issue during the tracking phase, but it complicates the landing procedure, as it is no more guaranteed the the drone will land with almost zero vertical velocity. One last proof of the model mismatches and of the aggressiveness of the controller is shown comparing figure 5.9b and 5.6b: on the laboratory experiment the required body rate is noisier during all the trajectory.

To conclude, this chapter showed how practical implementation of multi-agent control algorithms have do deal with,

- Sensor noises and disturbances, which are often non additive and non Gaussian, in particular when adopting bearing based approaches.
- Model uncertainties, which can never be fully eliminated and require robustness analysis.
- Limited computational resources, as the finite payload of the UAVs does not allow to put high performance computer onboard.

Nevertheless, all this issues can be tackled as long as the control law synthesis acknowledge them.

6 Conclusion

In this thesis different aspects and challenges related to heterogeneous multi-agent interaction have been addressed, starting from a control theoretical point of view, and concluding with implementation difficulties involved with practical applications. Inter-agent measurements, and in particular bearing-based measurement played the lead role in this work. Indeed, multi-agent interaction cannot safely rely on external, fragile groundtruth sensors but must be able, for enchanted flexibility and robustness, to only use measurements available between elements of the system. In addition, bearing-measurement were preferred over to distance-based methods for their increased simplicity.

Chapter 2 tackled the issue of heterogeneous formation control. In this scenario, multiple agents, each of them with potentially different actuation capabilities, had to be controlled such that the overall formation could reach a certain shape defined in terms of the inter-agent bearing measurements. The proposed solution was a revisited *heterogeneous* bearing rigidity formulation that allowed to embed the notion of agent actuation inside the bearing rigidity matrix. Consequently, it was possible to exploit a gradient descent control law to stir the shape error to zero. Alongside a formal proof of the local asymptotical stability of the proposed solution, Monte Carlo simulations showed that using the heterogeneous controller outperformed homogeneous based solutions coupled together with sub-formation shape matching algorithms. In light of those results, the natural extensions of this activity follow two parallel branches:

- Generalize toward generalized rigidity, where also the inter agent sensing can be heterogeneous and bearing, angle, and distance measurement could co-exist in the same framework.

- Further investigate the task of bearing based formation localization, in order to obtain a fully autonomous bearing based regulator.

In chapter 3, another application for bearing measurement, namely the target localization problem, was investigated. In this case, a group of agents or

seekers had to localize an uncooperative target using only the their bearing measurement with respect to it. Contrarily to the previous scenario, here the seekers were allowed to know in advance their own location and the attention was entirely reserved to the uncooperative target. The localization task has been solved using an iterative weighted least square approach combined with an efficient initialization scheme based on the line distance minimization problem. The iterative least square also provided a way to compute in close form an estimate of the localization error covariance whose accuracy was experimentally validated. The second contribution of this chapter was an active sense controller that moved the seekers in order to minimize the estimated localization error covariance while maintaining unchanged the estimated distance from the target. The result obtained in this activity depended on quite strong assumption on the communication and sensing capabilities of the seekers. Moreover, the developed active-sense scheme and localization algorithm were intrinsically centralized. I would be interesting to study how to relax these assumption, allowing partial sensing and communication capabilities and how to make the algorithm properly distributed.

Chaper 4 started recognizing the difficulties and challenges involved with acquiring bearing measurement and applying bearing based control laws on aerial vehicles and in particular multirotors. On one hand, coplanar quadrotors are quite simple to control but their highly coupled linear and rotational dynamics makes them poorly suited for rigidity tasks where bearing must be preserved. On the other hand, tilted multirotors are intrinsically decoupled but are also generally highly energy inefficient platform. For this reason, this chapter addressed the control of the tilting quadrotor, an interesting aerial platform that can dynamically adjust the trade-off between agility and efficiency. After a proper introduction and modeling of such platform, a motor-level Nonlinear Model Predictive Control solution was proposed and tested on an accurate simulation. Although accurate simulations were performed to evaluate the proposed solution, a proper experimental validation is still missing and it constitutes the direct extension of this research activity.

Lastly, chapter 5 investigates the problem of the autonomous landing of a coplanar quadrotor on a moving platform under the constraints that the quadrotor could only acquire the landing target bearing vector and it own

altitude w.r.t. the ground. In particular, the final goal of this activity was to deliver and experimental validation of the proposed solution. Therefore, contrarily to the previous chapters, the computational complexity of the algorithm, that eventually had to run on resource-constrained devices on the quadrotor, resulted to be a major issue. The solution was to employ a multi-stage controller, with the high level control that was provided by an NMPC scheme running on a Raspberry Pi 4 and efficiently implemented using the *acado* toolbox. The low level control was instead provided by the real-time flight controller, a Pixhawk 4 Mini. Accurate simulations performed in Gazebo initially verified the proposed solution before performing the final experiments. During the simulation phase, model mismatched were introduced and after having noticed that they did not affect the task performance, it was possible to conclude that the proposed solution was robust. Further development could involve outdoor tests, to evaluate the performance on high speed landing targets and the migration toward new NMPC toolbox, such as acados.

Bibliography

Holybro. URL https://holybro.com/collections/multicopter-kit/products/qav250-kit. last accessed, 2023/09.

Px4 autopilot. URL https://px4.io/. last accessed, 2023/09.

Ros. URL https://www.ros.org/. last accessed, 2023/09.

Ahn H.-S. *Formation control*. Springer, 2020.

Andersson J. A., Gillis J., Horn G., Rawlings J. B., and Diehl M. Casadi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11:1–36, 2019.

Aragues R., Carlone L., Calafiore G., and Sagues C. Multi-agent localization from noisy relative pose measurements. In *2011 IEEE International Conference on Robotics and Automation*, pages 364–369. IEEE, 2011.

Baca T., Hert D., Loianno G., Saska M., and Kumar V. Model predictive trajectory tracking and collision avoidance for reliable outdoor deployment of unmanned aerial vehicles. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6753–6760. IEEE, 2018.

Baca T., Stepan P., Spurny V., Hert D., Penicka R., Saska M., Thomas J., Loianno G., and Kumar V. Autonomous landing on a moving vehicle with an unmanned aerial vehicle. *Journal of Field Robotics*, 36(5):874–891, 2019.

Bicego D., Mazzetto J., Carli R., Farina M., and Franchi A. Nonlinear model predictive control with enhanced actuator model for multi-rotor

aerial vehicles with generic designs. *Journal of Intelligent & Robotic Systems*, 100:1213–1247, 2020.

Bin Junaid A., Diaz De Cerio Sanchez A., Betancor Bosch J., Vitzilaios N., and Zweiri Y. Design and implementation of a dual-axis tilting quadcopter. *Robotics*, 7(4):65, 2018.

Bishop A. N., Anderson B. D., Fidan B., Pathirana P. N., and Mao G. Bearing-only localization using geometrically constrained optimization. *IEEE Transactions on Aerospace and Electronic Systems*, 45(1):308–320, 2009.

Cao W., Zhang J., and Ren W. Leader–follower consensus of linear multi-agent systems with unknown external disturbances. *Systems & Control Letters*, 82:64–70, 2015.

Cao Y., Yu W., Ren W., and Chen G. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial informatics*, 9(1):427–438, 2012.

Chen K., Qi G., Li Y., and Sheng A. Target localization and multicircular circumnavigation with bearing-only measurements. *International Journal of Adaptive Control and Signal Processing*, 37(1):168–182, 2023.

Chen L., Cao M., and Li C. Bearing rigidity and formation stabilization for multiple rigid bodies in se (3). *Numerical algebra control and optimization*, 9(3):257–267, 2019a.

Chen Y., Bruschetta M., Picotti E., and Beghi A. Matmpc-a matlab based toolbox for real-time nonlinear model predictive control. In *2019 18th European Control Conference (ECC)*, pages 3365–3370. IEEE, 2019b.

Christiansen M. P., Laursen M. S., Jørgensen R. N., Skovsen S., and Gislum R. Designing and testing a uav mapping system for agricultural field surveying. *Sensors*, 17(12):2703, 2017.

Chun S. and Tian Y.-P. Multi-targets localization and elliptical circumnavigation by multi-agents using bearing-only measurements in two-dimensional

space. *International Journal of Robust and Nonlinear Control*, 30(8):3250–3268, 2020.

Dorri A., Kanhere S. S., and Jurdak R. Multi-agent systems: A survey. *Ieee Access*, 6:28573–28593, 2018.

Dou L., Song C., Wang X., Liu L., and Feng G. Target localization and enclosing control for networked mobile agents with bearing measurements. *Automatica*, 118:109022, 2020.

Drew D. S. Multi-agent systems for search and rescue applications. *Current Robotics Reports*, 2:189–200, 2021.

Ganga G. and Dharmana M. M. Mpc controller for trajectory tracking control of quadcopter. In *2017 International Conference on Circuit, Power and Computing Technologies (ICCPCT)*, pages 1–6. IEEE, 2017.

Grlj C. G., Krznar N., and Pranjić M. A decade of uav docking stations: A brief overview of mobile and fixed landing platforms. *Drones*, 6(1):17, 2022.

He S., Shin H.-S., and Tsourdos A. Trajectory optimization for target localization with bearing-only measurement. *IEEE Transactions on Robotics*, 35(3):653–668, 2019.

Houska B., Ferreau H. J., and Diehl M. Acado toolkit—an open-source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods*, 32(3):298–312, 2011.

Invernizzi D. and Lovera M. Geometric tracking control of a quadcopter tiltrotor uav. *IFAC-PapersOnLine*, 50(1):11565–11570, 2017.

Jacquet M., Corsini G., Bicego D., and Franchi A. Perception-constrained and motor-level nonlinear mpc for both underactuated and tilted-propeller uavs. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4301–4306. IEEE, 2020.

Kaplan L. M., Le Q., and Molnar N. Maximum likelihood methods for bearings-only target localization. In *2001 IEEE International Conference on*

Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221), volume 5, pages 3001–3004. IEEE, 2001.

Kia S. S., Van Scoy B., Cortes J., Freeman R. A., Lynch K. M., and Martinez S. Tutorial on dynamic average consensus: The problem, its applications, and the algorithms. *IEEE Control Systems Magazine*, 39(3):40–72, 2019.

Kovacina M. A., Palmer D., Yang G., and Vaidyanathan R. Multi-agent control algorithms for chemical cloud detection and mapping using unmanned air vehicles. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2782–2788. IEEE, 2002.

Li Z., Chen Y., Lu H., Wu H., and Cheng L. Uav autonomous landing technology based on apriltags vision positioning algorithm. In *2019 Chinese Control Conference (CCC)*, pages 8148–8153. IEEE, 2019.

Lin S., Garratt M. A., and Lambert A. J. Monocular vision-based real-time target recognition and tracking for autonomously landing an uav in a cluttered shipboard environment. *Autonomous Robots*, 41:881–901, 2017.

Lissandrini N., Verginis C. K., Roque P., Cenedese A., and Dimarogonas D. V. Decentralized nonlinear mpc for robust cooperative manipulation by heterogeneous aerial-ground robots. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1531–1536. IEEE, 2020.

Liu J., Wang X., Bai B., and Dai H. Age-optimal trajectory planning for uav-assisted data collection. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 553–558. IEEE, 2018.

Michieletto G. and Cenedese A. Formation control for fully actuated systems: a quaternion-based bearing rigidity approach. In *2019 18th European Control Conference (ECC)*, pages 107–112. IEEE, 2019.

Michieletto G., Cenedese A., and Zelazo D. A unified dissertation on bearing rigidity theory. *IEEE Transactions on Control of Network Systems*, 8 (4):1624–1636, 2021.

Michieletto G., Ryll M., and Franchi A. Fundamental actuation properties of multirotors: Force–moment decoupling and fail–safe robustness. *IEEE Transactions on Robotics*, 34(3):702–715, 2018.

Mohammadi A., Feng Y., Zhang C., Rawashdeh S., and Baek S. Vision-based autonomous landing using an mpc-controlled micro uav on a moving platform. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 771–780. IEEE, 2020.

Nascimento T. P. and Saska M. Position and attitude control of multi-rotor aerial vehicles: A survey. *Annual Reviews in Control*, 48:129–146, 2019.

Oh K.-K., Park M.-C., and Ahn H.-S. A survey of multi-agent formation control. *Automatica*, 53:424–440, 2015.

Olfati-Saber R. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on automatic control*, 51(3):401–420, 2006.

Öner K. T., Çetinsoy E., Ünel M., Akşit M. F., Kandemir I., and Gülez K. Dynamic model and control of a new quadrotor unmanned aerial vehicle with tilt-wing mechanism. 2008.

Oosedo A., Abiko S., Narasaki S., Kuno A., Konno A., and Uchiyama M. Flight control systems of a quad tilt rotor unmanned aerial vehicle for a large attitude change. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2326–2331. IEEE, 2015.

Ota J. Multi-agent robot systems as distributed autonomous systems. *Advanced engineering informatics*, 20(1):59–70, 2006.

Paris A., Lopez B. T., and How J. P. Dynamic landing of an autonomous quadrotor on a moving platform in turbulent wind conditions. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9577–9583. IEEE, 2020.

Pozzan B., Elaamery B., and Cenedese A. Non-linear model predictive control for autonomous landing of a uav on a moving platform. In *2022 IEEE Conference on Control Technology and Applications (CCTA)*, pages 1240–1245. IEEE, 2022.

Raffo G. V., Ortega M. G., and Rubio F. R. Nonlinear h_∞ controller for the quad-rotor helicopter with input coupling. *IFAC Proceedings Volumes*, 44 (1):13834–13839, 2011.

Rawlings J. B., Mayne D. Q., and Diehl M. *Model predictive control: theory, computation, and design*, volume 2. Nob Hill Publishing Madison, WI, 2017.

Ren H., Zhao Y., Xiao W., and Hu Z. A review of uav monitoring in mining areas: Current status and future perspectives. *International Journal of Coal Science & Technology*, 6:320–333, 2019.

Robin C. and Lacroix S. Multi-robot target detection and tracking: taxonomy and survey. *Autonomous Robots*, 40:729–760, 2016.

Ryll M., Bülthoff H. H., and Giordano P. R. A novel overactuated quadrotor unmanned aerial vehicle: Modeling, control, and experimental validation. *IEEE Transactions on Control Systems Technology*, 23(2):540–556, 2014.

Ryll M., Muscio G., Pierri F., Cataldi E., Antonelli G., Caccavale F., and Franchi A. 6d physical interaction with a fully actuated aerial robot. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5190–5195. IEEE, 2017.

Saif A.-W. A., Aliyu A., Dhaifallah M. A., and Elshafei M. Decentralized backstepping control of a quadrotor with tilted-rotor under wind gusts. *International Journal of Control, Automation and Systems*, 16:2458–2472, 2018.

Scherer J., Yahyanejad S., Hayat S., Yanmaz E., Andre T., Khan A., Vukadinovic V., Bettstetter C., Hellwagner H., and Rinner B. An autonomous multi-uav system for search and rescue. In *Proceedings of the first workshop on micro aerial vehicle networks, systems, and applications for civilian use*, pages 33–38, 2015.

Schiano F., Franchi A., Zelazo D., and Giordano P. R. A rigidity-based decentralized bearing formation controller for groups of quadrotor uavs. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5099–5106. IEEE, 2016.

Schiano F. and Giordano P. R. Bearing rigidity maintenance for formations of quadrotor uavs. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1467–1474. IEEE, 2017.

Semsch E., Jakob M., Pavlicek D., and Pechoucek M. Autonomous uav surveillance in complex urban environments. In *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, volume 2, pages 82–85. IEEE, 2009.

Shi G., Shi X., O'Connell M., Yu R., Azizzadenesheli K., Anandkumar A., Yue Y., and Chung S.-J. Neural lander: Stable drone landing control using learned dynamics. In *2019 international conference on robotics and automation (icra)*, pages 9784–9790. IEEE, 2019.

Stacey G. and Mahony R. The role of symmetry in rigidity analysis: A tool for network localization and formation control. *IEEE Transactions on Automatic Control*, 63(5):1313–1328, 2017.

Stansfield R. Statistical theory of df fixing. *Journal of the Institution of Electrical Engineers-Part IIIA: Radiocommunication*, 94(15):762–770, 1947.

Varotto L., Cenedese A., and Cavallaro A. Active sensing for search and tracking: A review. *arXiv preprint arXiv:2112.02381*, 2021.

Wynn J. S. and McLain T. W. Visual servoing with feed-forward for precision shipboard landing of an autonomous multirotor. In *2019 American Control Conference (ACC)*, pages 3928–3935. IEEE, 2019.

Xiao Y., Zhang N., Li J., Lou W., and Hou Y. T. Distributed consensus protocols and algorithms. *Blockchain for Distributed Systems Security*, 25: 40, 2019.

Xu S. Optimal sensor placement for target localization using hybrid rss, aoa and toa measurements. *IEEE Communications Letters*, 24(9):1966–1970, 2020.

Xuan-Mung N., Hong S. K., Nguyen N. P., Le T.-L., and others . Autonomous quadcopter precision landing onto a heaving platform: New method and experiment. *IEEE Access*, 8:167192–167202, 2020.

Zelazo D., Giordano P. R., and Franchi A. Bearing-only formation control using an se (2) rigidity theory. In *2015 54th ieee conference on decision and control (cdc)*, pages 6121–6126. IEEE, 2015.

Zhao S. and Zelazo D. Bearing rigidity theory and its applications for control and estimation of network systems: Life beyond distance rigidity. *IEEE Control Systems Magazine*, 39(2):66–83, 2019.