Planning and Control for Robotic Tasks with a Human-in-the-Loop

Von der Fakultät Konstruktions-, Produktions- und Fahrzeugtechnik der Universität Stuttgart zur Erlangung der Würde eines Doktor-Ingenieurs (Dr.-Ing.) genehmigte Abhandlung

Vorgelegt von

Carlo Masone

aus Rom (Italien)

Hauptberichter:	Prof. Dr.-Ing. Frank Allgöwer
Mitberichter:	Prof. Dr. Heinrich H. Bülthoff
	Prof. Dr. Cristian Secchi

Tag der mündlichen Prüfung: 16. Juli 2014

Institut für Systemtheorie und Regelungstechnik

Universität Stuttgart

2014

Acknowledgements

This thesis is the only final step of a long journey full of challenges and hard work, which would have not been possible without the support of many people who accompanied me in the past few years.

Firstly, I want to express my gratitude to Prof. Dr. Heinrich H. Bülthoff, for stimulating me in my research and for allowing me to work in his group at MPI during these years. He created a great research environment where I was able to meet brilliant colleagues and precious friends.
Along similar lines I want to thank Prof. Dr. Frank Allgöwer, for accepting me in his group within the prestigious university of Stuttgart, for co-advising my thesis and for giving me invaluable help.
I also want to thank Prof. Dr. Cristian Secchi, for showing interest in my work and for accepting to co-advise my thesis.

My most sincere gratitude goes to Dr. Paolo Robuffo Giordano and Dr. Antonio Franchi, who supervised me during the years of my doctorate. They have both given me constant guidance and have been always available for discussions, on days, nights and weekends. I will cherish the many things that I have learned from them during these years.

Finally, I want to thank the great colleagues that I was lucky to work with: Volker Grabe, Johannes Lächele, Maria Lächele, Thomas Nestmeyer, Martin Riedel, Markus Ryll, Burak Yüksel. They gave me technical help in the many days and nights we worked side-by-side and, most importantly, they have been truly good friends.

Tübingen, July 2014
Carlo Masone

To my wife, Paolina.

Table of Contents

Notation

Throughout the different chapters of this thesis the notation has been kept as uniform and consistent to the literature as possible, even though the problems discussed are various. This section briefly presents the guidelines for the notation that is used in the thesis, in order to provide an easy-to-access summary for the reader's convenience. Nevertheless, the symbols that are here described will be recalled whenever necessary. Moreover, the symbols/notation that are used only in specific passages will be introduced only when needed during the thesis.

Scalars, vectors and matrices Scalars are denoted with normal weighted characters, e.g., $k \in \mathbb{R}$ or $\alpha \in SO(1)$. Vectors are indicated with boldfaced symbols, e.g., $\boldsymbol{v} \in \mathbb{R}^3$ or $\boldsymbol{\beta} \in \mathbb{R}^2$. Matrices are denoted with capital normal weighted symbols, e.g., $M \in \mathbb{R}^{3\times 2}$, and the range space of a matrix M is denoted as $\mathfrak{R}(M)$. Special vectors are the column vector of zeros, i.e., $\mathbf{0}_m \in \mathbb{R}^{m\times 1}$, and column vector of ones, i.e., $\mathbf{1}_m \in \mathbb{R}^{m\times 1}$. A special notation is also reserved to the identity matrix, denoted as $I_m \in \mathbb{R}^{m\times m}$, and the null matrix, written as $\mathbf{0}_{m\times m} \in \mathbb{R}^{m\times m}$. Unit vectors are denoted as vectors with the symbol $\hat{\cdot}$, e.g., $\hat{\boldsymbol{v}} \in \mathbb{S}^2$. The notation $[\boldsymbol{v}]_\wedge \in so(3)$ is used to denote the skew-symmetric matrix associated to a vector $\boldsymbol{v} = (v_1\, v_2\, v_3)^T \in \mathbb{R}^3$, i.e.,

$$[\boldsymbol{v}]_\wedge = \begin{pmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{pmatrix}. \tag{0.1}$$

Conversely, the notation $[S]_{\vee,3} \in \mathbb{R}^3$ denotes the vector that is extracted from a skew-symmetric matrix S.

Frames of reference and transformations A frame of reference in the Cartesian space is denoted as $\mathcal{F} : \{O; \vec{X}; \vec{Y}; \vec{Z}\}$, where $O \in \mathbb{R}^3$ is the origin of the frame and $\vec{X}, \vec{Y}, \vec{Z} \in \mathbb{R}^3$ are the axis of an orthonormal basis. The notation for the origin and axes of a frame is deliberately chosen different from standard vectors in order make clear that they define a frame. A

vector expressed in a specific frame of reference, e.g., frame $\mathcal{F}_{\mathcal{W}}$, is generally written with the indicator of the frame as pre-superscript, e.g., ${}^{\mathcal{W}}\boldsymbol{v}$. The rotation matrix of a frame $\mathcal{F}_{\mathcal{A}}$ with respect to another frame $\mathcal{F}_{\mathcal{W}}$ is written as ${}^{\mathcal{W}}R_{\mathcal{A}} \in SO(3)$. For the well known properties of rotation matrix, it is also ${}^{\mathcal{W}}R_{\mathcal{A}}^{T} = {}^{\mathcal{W}}R_{\mathcal{A}}^{-1} = {}^{\mathcal{A}}R_{\mathcal{W}}$. The canonical rotation matrices around the frame axes are indicated as $R_{\vec{X}}(\cdot)$, $R_{\vec{Y}}(\cdot)$ and $R_{\vec{Z}}(\cdot)$.

Abstract

The design of robotic tasks with a joint interaction with a human user (*human-in-the-loop*) is currently a highly popular topic in robotics research. One of the main reasons of interest is the possibility of combining the skills of both humans and robots to successfully perform complex tasks. In particular:

- Robots are extremely capable at autonomously executing specific and repetitive tasks, with great speed and precision, and they can operate in environments that are dangerous for a human operator.
- With respect to robots, humans possess far superior cognitive capabilities and world awareness which allow them to tackle applications that involve unstructured environments or require taking difficult and quick decisions.

The co-participation of humans and robots to a task can also arise from other reasons, such as an implicit constraint of the task itself (wearable robots, motion simulators) or safety regulations that require a human to supervise the activity of robotic workers.

In view of these considerations, shared control (between human and robot) is a promising (and, in some fields, consolidated) approach to address a number of robotics applications. However, there are several open questions and challenges regarding the design of shared control architectures, such as choosing the role of the human in the task, devising suitable command interfaces and feedback algorithms that increase the situation awareness of the operator, and coping with the unpredictable signals or decisions from the human user.

In this doctoral thesis it is presented a study of some novel robotic tasks involving human-robot interaction. The original shared control architectures developed for these tasks illustrate several novel solutions to the aforementioned questions. Furthermore, the tasks considered in the thesis span various possibilities for the typical characteristics of shared control architectures in robotics, i.e.:

- the role of the human operator in the shared task and his/her interaction with the robot(s);
- the typology and number of robots that participate to the shared task;
- the feedback returned to the human operator.

Deutsche Kurzfassung

Planung und Steuerung von Roboter-Mensch Systemen

Das Design von Aufgaben in der Robotik in Kooperation mit einer Bedienperson (*human-in-the-loop*) ist zur Zeit ein sehr weitverbreitetes Problem in der Forschung. Eine der Hauptgründe dieses Interesses ist die Möglichkeit, die Stärken sowohl des Menschen, als auch der Roboter zu kombinieren, um erfolgreich komplexe Aufgaben zu erfüllen Im Einzelnen:

- Roboter sind extrem gut darin, spezifische und sich wiederholende Aufgaben mit großer Geschwindigkeit und Präzision auszuführen und können in Umgebungen arbeiten, die gefährlich für einen Arbeiter sind.

- Im Gegensatz zu Robotern besitzen Menschen weit bessere kognitive Fähigkeiten und ein Bewusstsein für ihre Umgebung, was ihnen erlaubt, Aufgaben in einer unstrukturierte Umwelt auszuführen, welche schwierige und schnelle Entscheidungen voraussetzen.

Gemeinsames Mitwirken von Menschen und Robotern an einer Aufgabe kann auch andere Gründe haben, wie zum Beispiel eine implizite Einschränkung der Aufgabe selbst (am Körper tragbare Roboter, Bewegungssimulatoren) oder Sicherheitsbestimmungen, die vorschreiben, dass ein Mensch die Aktivitäten eines Roboters überwacht.

In Bezug auf diese Überlegungen ist die gemeinsame Kontrolle (zwischen Mensch und Roboter) ein vielversprechender (und, in einigen Gebieten, fundierter) Ansatz, eine Großzahl von Anwendungen in der Robotik anzugehen. Jedoch gibt es verschiedene offene Fragen und Herausforderungen, die das Design der gemeinsamen Kontrollarchitektur betreffen, wie beispielsweise die Rolle des Menschen an der Aufgabe, die Bestimmung geeigneter Benutzeroberflächen, die das Bewusstsein des Bedieners für die Situation

erweitern und Feedbackalgorithmen, die mit unvorhersehbaren Signalen und Entscheidungen des menschlichen Benutzers umgehen können.

In dieser Doktorarbeit wird eine Studie präsentiert, die neuartige Anwendungen in der Mensch-Maschine-Interaktion untersucht. Die einzigartigen geteilten Kontrollarchitekturen, die für diese Arbeit entworfen wurden, zeigen verschiedene neuartige Lösungen der vorhergehenden Fragen. Zusätzlich spannen die Anwendungen, die in dieser Arbeit berücksichtigt wurden, verschiedenste Möglichkeiten für die typischen Charakteristiken geteilter Kontrollarchitekturen in der Robotik:

- Die Rolle der Bedienperson in der geteilten Aufgabe und seine/ihre Interaktion mit den Robotern;
- Den Typ und die Anzahl der Roboter die an der geteilten Aufgabe teilnehmen;
- Das Feedback, das an die Bedienperson gegeben wird.

List of Abbreviations

2D/3D Two/Three dimensional

ATC Air Traffic Control

CMS CyberMotion Simulator

DoF Degrees of Freedom

FoV Field of View

HLC High-Level Controller

HRI Human-Robot Interaction

IMU Inertial Measurement Unit

LLC Low-Level Controller

MCA Motion Cueing Algorithm

pHRi physical Human-Robot Interaction

PoI Points of Interest

PSPM Passive Set-Position Modulation

RPY Roll-Pitch-Yaw

RTHL Robotic Task with a Human-in-the-Loop

SA Situation Awareness

TP Task Priority

UAV Unmanned Aerial Vehicle

UI User Interface

Chapter 1

Introduction

1.1 Robotic Tasks with a Human-in-the-Loop

Human-Robot Interaction (HRI) is a branch of robotics whose mission, according to Goodrich and Schultz (2007), is "to understand and shape the interactions between one or more humans and one or more robots". As this statement suggests, HRI is a very broad field of research (the interested reader is referred to Goodrich and Schultz (2007); Thrun (2004) for a wide overview), but it is also quite young in comparison to other fields of robotics and rapidly growing. The first conference dedicated entirely to this subject started in 1992: the IEEE International Symposium on Robot & Human Interactive Communication (RoMan). Nowadays, HRI has become one of the most relevant fields of robotics, as proven by the numerous dedicated scientific meetings that have emerged in the last decade (e.g., the ACM/IEEE International Conference on Human-Robot Interaction (HRI conference) or the workshop on Human Friendly Robotics (HFR). The importance of HRI is further confirmed by several European projects: PHRIENDS[1] (2006-2009), CHRIS[2] (2008-2012), SAPHARI[3] (2011-2015).

A major topic within HRI is the study of *robotic tasks with a human-in-the-loop* (RTHL), i.e., tasks that are cooperatively executed by humans and robots, in contrast to robotic tasks in which people are obstacles or *bystanders* (Scholtz and Bahrami (2003)) in the environment (e.g., autonomous vehicles driving in an urban environment). An example of RTHL is the cooperative handling and transportation of objects presented by Kosuge and Hirata (2004). In order to comprehend the variety of

[1] Physical Human-Robot Interaction: Dependability and Safety, `http://www.phriends.eu`

[2] Cooperative Human Robot Interaction Systems, `http://www.chrisfp7.eu`

[3] Safe and Autonomous Physical Human-Aware Robot Interaction, `http://www.saphari.eu`

RTHL, the next section will describe few application domains and research fields that involve or relate to the cooperation of humans and robots. This brief overview, is helpful not only to understand the motivations for researching human-robot co-partecipation to a task, but also to identify several characteristics and open challenges of RTHL that will be addressed during the thesis.

.

1.2 Application Domains of RTHL and Related Fields

Field applications The domain of field applications pertains tasks that take place in unstructured, unconstrained and dynamic environments, typically outdoors in the natural world. This is a broad domain, spanning the topics of environmental monitoring, construction, forestry, agriculture, intelligent highways, search and rescue, etc. The overlap between RTHL (or more in general HRI) and field robotics stems from the limited cognitive and processing capabilities of current robots: at present, even the state of the art robotic systems are not yet capable of fully autonomously coping with the complexity and unpredictability of the natural world, therefore humans are generally required to control or supervise the operations. For example, after the Tohoku earthquake and tsunami remotely operated underwater vehicles were used to inspect critical infrastructures by Murphy et al. (2011).

Among field applications (see Yuta et al. (2006)), urban search and rescue (USAR) (Murphy et al. (2008)) is often taken as an exemplary study of RTHL, because it includes many of the problematics of human-robot interactions, such as user interfaces, sensing, evaluation of human-factors, etc. For this reason USAR has also been suggested by a DARPA/NSF study from Burke et al. (2004a) as one of the great challenges for human-robot interaction, and USAR related competitions are organized at important robotics conferences in order to promote and study important questions of RTHL. A detailed overview of RTHL in rescue robotics is presented by Murphy (2004a).

Unmanned Aerial Vehicles Among mobile robots, Unmanned Aerial Vehicles (UAVs) deserve a special mention as a blooming area of application

for RTHL. UAVs can be roughly defined as autonomously or remotely piloted vehicles that move in three dimensions and possess 6 degrees of freedom (DoF). Thanks to their mobility and ability to quickly cover large areas, UAVs have allowed to greatly expand the range of applications in the field domain. For example, teams of UAVs can be extremely valuable for border patrolling and surveillance tasks (see Girard et al. (2004)) or for environmental monitoring such as forest fire monitoring (see Casbeer et al. (2005)) or pollution monitoring (see Kristiansen et al. (2012)). In recent news, the importance of UAVs for monitoring missions has also been recognized by the International Atomic Energy Agency[4]. The great interest in UAVs research is also been confirmed by several European projects, such as sFly[5] (2009-2011), AIRobots[6] (2010-2013), ARCAS[7] (2011-2015) and ICARUS[8] (2012-2016).

The overlap with RTHL is due to the fact that in most missions UAVs are remotely operated for several reasons, such as the complexity of the task or the lack of predetermined flight plans (see Quigley et al. (2004)). However, the main reason for remote teleoperation of UAVs is safety. UAVs are deemed very risky, especially for urban missions, because they can fall to the ground in case of failure. For this reason, there are often strict safety regulations that mandate the presence at all times of human supervisors or pilots. The dichotomy between the need for human pilots and the distance from the vehicles has recently prompted the adoption of a bilateral teleoperation paradigms to command UAVs, in order to increase telepresence and situation awareness, see e.g., Lam et al. (2006a,b).

Physical Human-Robot Interaction Physical human-robot interaction (pHRI) is branch of HRI that studies tasks involving a direct contact between human and robot. Perhaps, the main focus of pHRI is the question of safety, for example in the case of accidental collisions of a robot with a person as shown by Haddadin et al. (2008). However, the physical contact between human and robot could also be an implicit constraint or the goal itself of a RTHL. A clear example of this kind of scenario is provided by assistive and medical robots that are designed to support injured or disabled people during rehabilitation (see Jungwon et al. (2010)). Another example

[4] http://www.iaea.org/newscenter/news/2013/aerialvehicles.html
[5] http://www.sfly.org
[6] http://airobots.ing.unibo.it
[7] http://www.arcas-project.eu
[8] http://www.fp7-icarus.eu

of RTHL involving physical interaction is the case of motion simulators, in which a person is situated onboard moving robotic platforms that are used to create the perception of a desired motion, such as the motion of a vehicle.

Aviation Even though aviation does not involve RTHL and the literature from these communities is completely disjoint, these two fields share several problems and questions. For instance, the question of designing human-machine interfaces that simplify pilots' control and increase their situation awareness has been for a long time a research topic in aviation. An example of such interfaces are tunnel-in-the-sky displays that have been designed to give the pilot a better understanding of the effect of his/her control on the trajectory of the aircraft (see Mulder et al. (1999a,b)). Another element of similarity with RTHL is the question of how many vehicles (or robots) a person can command or supervise. In aviation, this question has been raised by the problem of air traffic control (ATC), which requires a person to handle the traffic of multiple aircrafts. A possible point of convergence between RTHL and aviation could be offered by the research on personal aerial transportation systems, pursued for example within the European project myCopter[9] (2011-2014). Realizing a mainstream personal aerial vehicle requires in fact to study the question of balancing autonomy and manual control, which is central in RTHL.

Telerobotics and Haptics Telerobotics is one of the earliest fields of research in robotics and it is also a major contributor to RTHL. The connection with RTHL is easily explained by the very definition of telerobotics, which, according to Niemeyer et al. (2008), "... is generally understood to refer to robotics with a human operator in control or human-in-the-loop. Any high level, planning or cognitive decisions are made by the human user, while the robot is responsible for their mechanical implementation. In essence, the *brain* is removed or distant from the *body*". The key concept of telerobotics is indeed the distance of the user from the robot, which in the RTHL applications could derive from the the nature of the robot itself, as seen in the case of UAVs, or to the presence of dangers in the environment.

The distance between human operator and robot raises the issue of letting the human operator perceive adequately the remote environment and the robot. To this purpose, the human-robot interaction can be integrated

[9] `http://www.mycopter.eu`

with an informative feedback, i.e., suitable cues are provided to the human co-operator in order to let her/him 'feel' the remote robot(s) and site (see Hokayem and Spong (2006)). Haptic cues in particular appear to be a very effective way to convey this kind of information. On one side, it was experimentally proven that in many cases this 'bilateral' interaction increases the situational awareness and consequently the performance of the operator in the execution of the task, see, e.g., Abbink et al. (2011, 2012); Boessenkool et al. (2013); Lam et al. (2009); Wildenbeest et al. (2013). On the other hand, haptic feedback has the advantage of requiring little bandwith in comparison to video streaming, thus making it the ideal solution for many scenarios such as remote control of underwater vehicles shown by Murphy et al. (2011) or intercontinental control of mobile robots over the internet shown by Riedel et al. (2012).

1.3 Motivations of RTHL

Summarizing from the discussion of Sec. 1.2, there are several motivations for considering RTHL:

- The joint execution of a task allows to combine the strengths of both humans and robots:
  - Robots are extremely capable at autonomously executing specific and repetitive tasks, with great speed and precision, and they can operate in environments that are dangerous for a human operator.
  - Humans possess unmatched cognitive capabilities and world awareness which allow them to tackle applications that involve unstructured environments or require taking difficult and quick decisions.

  This is for instance the case of field robotics and USAR, in which robots are used to reach impervious and dangerous locations while human operators grant the cognitive capabilities to cope with unstructured and unpredictable environments.

- The co-participation of humans and robots can be an implicit constraint of the task or the goal itself. This is for instance the case of assistive robots for supporting injured patients during their physical recovery or for human augmentation. Another example of an

application in which the interaction of the robot with the user is the primary objective are motion simulators, i.e., robotic platforms that are designed to move the user to simulate the motion of a vehicle.

- Safety regulations can mandate the presence at all times of a person in charge of supervising and, if necessary, take direct control of the robotic workers. For example, this is a common requirement in applications with UAVs, in particular when flying in civil airspace such as over urban or populated areas.

1.4 Characteristics and Challenges of RTHL

The numerous application domains and research fields referred in Sec. 1.2 also suggest that the cooperation of humans and robots can take various forms and present many different problems, such as safety in physical human-robot interactions, social and ethical implications, control, etc. Disregarding the ethical and social implications as well as safety issues, the discussion hereinafter will focus on the control aspect of RTHL, addressing in particular the few questions and aspects that are introduced in the following.

1.4.1 Autonomy

Autonomy is a key element in the design RTHL as well as one of the reasons leading to its emergence. For a long time since the early years of robotics, robotic tasks have been confined to an industrial setting (see Hägele et al. (2008)), in which the robots operate in workcells where people are not allowed and that are engineered to maximize productivity and guarantee repeatability. For example, automatic transportation of goods in manufacturing centres is done along fixed paths that require painted routes or embedded wires and magnets for motion guidance. In general, since work cells are engineered ad hoc for a specific task, industrial robots do not need to have extensive perception of their surroundings nor advanced intelligence.

What made possible the shift towards the application domains mentioned in Sec. 1.2, and therefore RTHL, is the increased autonomy of the robots prompted by the technological advancements made in several fields, such as sensing, estimation, artificial intelligence, etc.. In order to better understand its implications in RTHL, it is important to define more precisely the concept

of autonomy. In some cases it is considered as the capacity of a robot to react to variations in the environment or as the mean time between failures. An operative definition of autonomy is expressed by the neglect tolerance introduced by Crandall et al. (2005): high autonomy corresponds to a robot that can operate without interactions, i.e., a robot that is neglected. However, these definitions do not capture the essence of human-robot cooperation which is the key in RTHL. A more fitting interpretation is found in Sheridan and Verplanck (1978) (albeit for human-computer interaction) with the notion of levels of autonomy: the levels of autonomy define a scale in which, at one extreme the human executes the task and at the other extreme the computer (or robot) does the job alone.

Following this interpretation, it is clear that the choice of the autonomy of the robots in a RTHL is critical because it affects the role of the human operator in the task. One problem in this regard is that in many real world applications robots often have little autonomy and they largely rely on the constant supervision of human operators, who therefore are required high workload and commitment. Therefore, finding viable ways to increase the autonomy of the robots in the cooperation with humans, and thus elevate the role of operator at a higher-level, is a key challenge of RTHL.

1.4.2 Human-to-robot ratio

RTHL does not only involve interactions between a single human and a single robot. On the contrary, in many of the application domains cited in Sec. 1.1 multiple humans are assigned to a single robot. For example, in missions with UAVs there are typically two operators per vehicle: one acting as a pilot, the other as a specialist in charge of interpreting sensor information (see Drury et al. (2006); McCarley and Wickens (2005)). The number of operators might even increase due to the complexity of the mission or for safety reasons. Similarly, in search and rescue it was proven by Burke et al. (2004b) that performance with ground robots increases sensibly when two operators work together to control each robot.

Clearly, the need for a crew of multiple people per robot is not a problem for missions with a single robot (or few robots), however it becomes a huge limitation for tasks involving large teams or swarms of robots. Therefore, finding approaches to reduce the human-to-robot ratio is a mandatory condition to make multi-robot systems viable in RTHL and it is recognized as major challenge of HRI.

1.4.3 User interfaces

User interfaces (UIs) are a fundamental component of RTHL that allow for a bidirectional exchange of information:

- commands or instructions are communicated by the operator to the robots;
- a feedback regarding the status of the task or the environment is computed from the robots sensing and internal state and it is rendered back to the user.

As a consequence of this bidirectional flow of information, the challenge with UIs is twofold.

The first issue regards the complexity of the command interface. Control interfaces in robotics applications generally are not very intuitive, because of the inherent complexity of the task or of the robot. The drawback of difficult interfaces is that the operators must undergo a training before being able to operate the robot, and it might be necessary to rehearse it before every mission. This is a problem because in some cases is not possible to constantly have a trained team of operators, for instance in the case of rescue robots that are rarely used. As a practical example, a robot could not be used after the World Trade Center disaster because of the complexity of its interface (see Murphy (2004b)). In view of these considerations, a key issue in RTHL is to design UIs with command interfaces that are easy to use and that provide a suitable level of abstraction to implement high-level directives, such as formation behaviour control and motion planning.

The second issue regards the design of the feedback according to some human metrics (see Murphy and Schreckenghost (2013) for several human metrics in HRI). For instance, when the operator interacts with the robot(s) from a remote location it is critical that the feedback provides a good *situation awareness* (SA). SA is one of several human metrics used in HRI, as shown by Murphy and Schreckenghost (2013), and it is defined in Endsley (1988) as "the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future". Namely, SA can be seen as the pilot's model at a certain point in time of the environment where the task takes place. This model affects the pilot's decisions and actions, therefore it must be accurate for the success of a mission. However, if the interface provides too much information it can be difficult and taxing

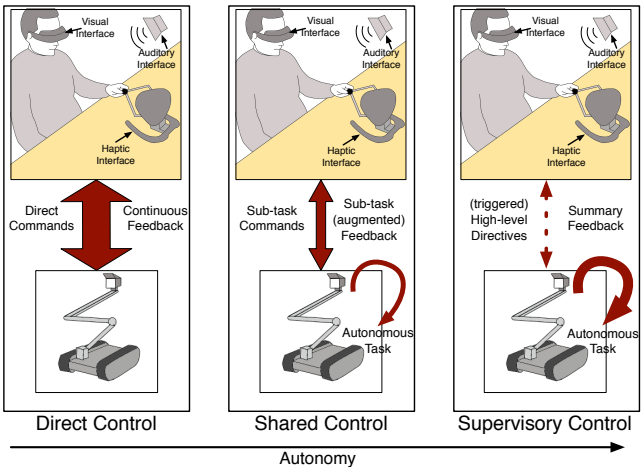


Figure 1.1: Different control paradigms for various levels of autonomy (Inspired by Fig. 31.10 in Niemeyer et al. (2008)).

for the pilot to parse and comprehend all the data. For example, evaluation studies in rescue robotics show that the visual channel of an operator is often overloaded, thus causing fatigue (see Murphy et al. (2008)). For this reason, the use of other feedback modalities besides visual feedback is an active topic of research. In this regard, the realization of realistic immersion in virtual environments is another active topic of research in robotics-related fields (see Hettinger and Haas (2009)), which aims at using multiple sensory cues (e.g., visual and haptic) not only to enhance the operator's SA of remote locations, but also to train him/her in specific tasks (see Bicchi et al. (2008); Hokayem and Spong (2006)).

1.4.4 Shared Control

In Sec. 1.4.1 it has been discussed how the level of autonomy in a RTHL affects the role of the operator. However, the level of autonomy influences also the control architecture of the system (see Fig. 1.1). As shown in Niemeyer et al. (2008), the two extremes, i.e., no autonomy or complete autonomy, correspond to the paradigms of *direct control* and *supervisory control*, respectively.

Direct control is a paradigm that involves no autonomy nor intelligence

from the robot: the human operator is in charge of directly piloting the robot, commanding, e.g., a thrust, velocity or position. At the same time, a continuous feedback is necessary to let the pilot understand how the robot is moving or what is the state of the environment. In short, the robot can be considered as a mere extension of the user's body. Furthermore, since the user executes the task through the robot, this approach requires a continuous and high exchange of information.

On the other hand, supervisory control is a paradigm that involves the (almost) complete autonomy of the robot. The definition of supervisory control was introduced by Ferrell and Sheridan (1967) as an analogy to a hierarchical staff: at the high-level, the supervisor (human) only gives directives to the subordinates (robots) who are then delegated the execution of the task. A more detailed description of this approach is found in Sheridan (1992). Contrary to direct control, this approach is characterized by a slow (or intermittent) and modest information exchange, since the robot autonomously performs the task. In fact, the human operator gives directives to the robot only when it is necessary to modify the currently executed task (e.g., to change the goal of a planning algorithm) or at the occurrence of trigger events (e.g., when it is required a critical decision or the current job is completed). Also the feedback is provided when necessary (e.g., triggered by events) and it informs the user about the overall status of the task.

The two control architectures just described, direct and supervisory, apply to cases in which the task is executed (almost) completely by the operator or by the robot. The co-participation of a human and a robot to a task requires a different architecture in which the intelligence of the system is shared between the two, i.e., a *shared control* paradigm. Shared control can be regarded as a compromise between direct and supervisory control, because the operator directly and continuously controls only part of the task leaving the rest to the robot. For example, in a multi-robot application the user might command global behaviours of the group of robot and receive a feedback on the state of the overall group, while individual control of each robot is left to the autonomous part of the system.

Shared control is adopted in several fields, such as space applications, where the large time delays require some autonomy of the robot (see Hirzinger et al. (1994)), or surgical procedures, where the robot is required to autonomously compensate for the patients movement in order to facilitate the job of the surgeon (see Ortmaier et al. (2005)). One advantage of the this paradigm is that the autonomy of the robot can be used to augment the

feedback provided to the operator in order to assist him/her. An example of augmented feedback is the use of *virtual fixtures* (see Rosenberg (1993)), i.e., guides that are superimposed to the standard feedback (e.g., visual or haptic) to give an indication or even to constrain the control inputs of the operator. The benefit of using such guides is well explained in Abbott et al. (2007) with a similitude: "A straight line drawn by a human with the help of a ruler is drawn faster and straighter than a line drawn freehand. Similarly, a robot can apply forces or positions to a human operator to help him or her draw a straight line. However, a robot (or haptic device) has the additional ability to provide assistance of varying type, level and geometry." On the other hand, a challenge that arises with shared control architectures is the fact that the unpredictable commands given (online) by the human operator to execute his/her subtask can also affect the subtask executed autonomously by the robot.

In light of these considerations, the design of suitable shared control architectures is a major theme of research in RTHL.

1.5 Objectives and Outline of the Thesis

The discussion so far has given a glimpse of the importance and broadness of RTHL and it has illustrated several aspects and challenges that are characteristic of tasks with a human-in-the-loop. The goal of this Ph.D. thesis is to study few specific problems that well illustrate the main traits and variety of RTHL and, by solving these problems, provide also an insight regarding the aspects and challenges that have been discussed in Sec. 1.4. Summarizing, the contributions are:

- *Autonomy:* The problems studied exhibit different roles for the operator and increasing levels of autonomy, from an almost direct control architecture for a robotic manipulator to the shared control of mobile robots in which the autonomy of the robots is exploited to reduce the workload of the user.
- *Shared control:* Various shared control architectures have been developed to tackle the problems and they also illustrate different approaches for coping with the unpredictable human commands.
- *User Interface:* The user interfaces that have been developed span various possibilities for the typical RTHL, by illustrating different designs of command interfaces and feedback (vestibular, visual, haptic).

Furthermore, they extend the traditional interfaces for shared control architectures in robotics to formation control and path planning.

- *Human-to-robot ratio:* One of the problems studied regards a multi-robot scenario. By increasing the autonomy of the robots it has been designed a shared control framework in which a single person can easily control a team of robots.

The rest of the thesis is organized as follows:

- In Chap. 2 it is modelled a novel actuated cabin to be used for a motion simulator together with an anthropomorphic arm and then it is designed the control algorithm of the overall simulator with a pilot in the loop.
- In Chap. 3 it is developed a control architecture for a group of unmanned aerial vehicles (UAVs) with bound formation and a human-in-the-loop that steers the overall swarm of robots.
- In Chap. 4 it is designed a framework in which the motion planning of a mobile robot is shared between a human operator and the robot itself.
- Chapter 5 contains a summary of the contributions of the previous chapters and a discussion of the implications on RTHL that can be derived from the projects presented.

Chapter 2

Design and control of a novel motion simulator

The discussion presented in this chapter is based upon the work published in Masone et al. (2011) and presented in Masone (2011).

2.1 Introduction

Motion simulators are devices designed with the purpose of giving to the user the realistic perception of being in a moving vehicle. The means to achieve this goal is providing the user with the same sensor cues that would be experienced on the real vehicle, or similar sensor cues that create the *illusion* of a realistic motion. Most simulators rely only on visual cues to create the sensation of motion, also because it is suggested that drivers mostly use visual information to follow the road (see Sivak (1996)). However, it was proven by Groen and Bles (2004) that the vestibular cues provided by a moving platform positively contribute to the immersion and feeling of realistic motion, thus making the development of such platforms an important topic of research in robotics. The characteristics of a motion simulator as RTHL are shown in the scheme of Fig. 3.1, in which:

Human: The human is onboard the motion platform and drives a model of a vehicle which then produces a trajectory for the robot that gives a similar sensation of motion.

Robot: The robot (motion platform) has to track the desired trajectory as accurately as possible, but it can deviate from it in order to account for its physical constraints (e.g., joint limits).

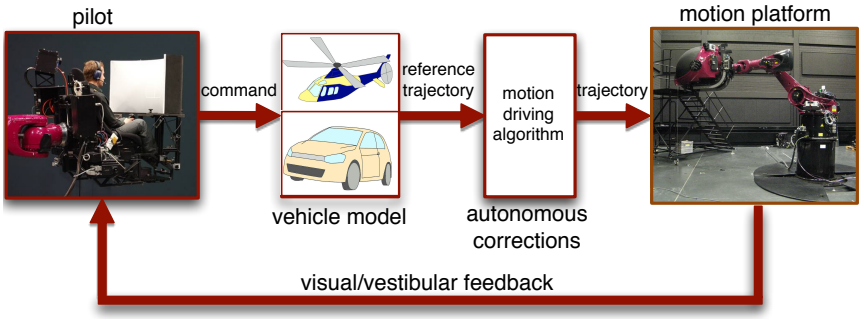


Figure 2.1: Scheme of the motion simulator.

Feedback: The vestibular feedback provided to the operator is given by the self motion of the platform. The realization of a proper feedback is the goal of the simulator.

In this chapter it will be discussed the design of a novel moving cabin to be equipped on an robotic arm platform to improve accuracy in rendering the desired feedback. The redundancy granted by this novel 7-th joint is exploited to optimize the configuration of the robot and avoid singularities/joint limits, with the latter being particularly critical for the reproduction of the desired motion cues. It is also described the control architecture that has been used for the extended system (actuated cabin + robotic arm), which well illustrates the challenge of coping with the unpredictable commands given online by the human-in-the-loop.

2.1.1 Related Works

Historically, the wide diffusion of motion simulators dates back to 1965 with the Gough-Stewart platform (more commonly known as Stewart platform or hexapod, see Stewart (1965)) and the first studies were carried out in the field of aeronautics at NASA in the 70's, see Conrad and Schmidt (1971); Dieudonne et al. (1972). Indeed, aviation is the leading application for motion simulators, in particular for the training of pilots (see Burki-Cohen et al. (2001)). A motion simulator in fact allows to reproduce unexpected and risky situations in a controlled and repeatable way and, most importantly, in complete safety for the crew and for the expensive aircrafts.

For example, within the European project SUPRA[1] motion simulators have been used to reproduce the dangerous scenario of upset recovery with aircrafts. Nowadays, motion simulators are also used for other purposes, for instance for psychophysical research to evaluate human factors and to study how humans perceive linear and rotational motions, see e.g., Nieuwenhuizen and Bülthoff (2013); Soyka et al. (2011, 2012). A growing application field is the entertainment industry (discussed in EUROP (2009)), where robotic motion platforms are used as rides or roller-coasters in amusement parks. Motion platforms are also a fundamental tool in the automotive industry, for collecting data in the development and evaluation of new vehicles. As a reflection of this wide range of applications, motion simulators have been specialized to simulate different typologies of vehicles, such as aircrafts, helicopters (see Beykirch et al. (2007)), cars (see Robuffo Giordano et al. (2010c); Siuikat (2005)), motorcycles (see Avizzano et al. (2000); Ferrazzin et al. (1999); Nehaoua et al. (2011)), sailing boats (see Avizzano et al. (2010)) and even space flights (see Heindl et al. (2006)).

Despite their success in many different applications, motion simulators are still largely researched to overcome the problem of the dissimilarity between the motion envelope of a real vehicle and that of a motion platform. The limitations of a motion platform (workspace, velocities, accelerations, etc.) make it impossible to exactly reproduce any motion of a vehicle that is moving in a large (virtually unlimited) space. For example, it is clear that a simulator cannot repeat the same motion of a car driven along a very long straight street, because at some point the platform would reach its joint limits. Furthermore, if a joint reaches its limit during the simulation, the reproduction of the desired motion can be heavily distorted thus inducing false cues on the pilot. In order to cope with this limitation, research has mainly taken two directions:

- The first approach is to develop motion cueing algorithms (MCA) that attempt to generate a trajectory within the limited workspace of the robot but with the same *perception* of the real trajectory of the vehicle. MCAs typically use two approaches, i) motion scaling (see Berthoz et al. (2013); Schwarz (2007)), and ii) reorientation of the gravity vector to create the illusions of sustained accelerations. This second technique is known in literature as 'tilt coordination' due to its reference to flight (see Grant and Reid (1997)).

[1] `http://www.supra.aero`

- The second approach is to develop novel motion platforms specialized for specific tasks or with a larger workspace in comparison to classic Stewart platforms. Examples of unique architectures are the NASA vertical simulator (see Beard et al. (2013)) which allows to have 60ft of vertical motion thus excelling at simulating aerospace vehicles, or the Desdemona simulator (see Bles and Groen (2009); Wentink et al. (2005)) which uses a gimbaled system and can rotate around a central axis to reproduce centrifugal motion with g-loads up to $3g$. The use of linear rails is another common strategy for extending the workspace of a Stewart platform. This approach is used in the most advanced motions simulators that are operative in the automotive industry, such as the Daimler-Benz simulator (see Käding and Hoffmeyer (1995)), the Toyota simulator[2] or the Renault ULTIMATE[3]. However, these solutions for the automotive industry are very expensive and require large spaces and infrastructures, thus they are not viable for other applications.

Along the lines of developing novel motion platforms, serial manipulators have recently gained attention in the research community. The underlying idea is to attach a seat or a cabin to the end-effector of an industrial manipulator and move it (and the passenger onboard) together with the robot. In comparison to parallel actuation systems (such as classic Stewart platforms), serial architectures have inferior load capability, stiffness and bandwidth but the great advantage of a much larger dexterity envelope. In particular, by using a 6 DoF anthropomorphic manipulator such as the KUKA Robocoaster[4], the seat at the end-effector of a serial manipulator can be moved along complex trajectories and reach any attitude. The seat could even be placed upside down in order to simulate complex aerial manoeuvres such as inverted flight (see Bellman et al. (2007)) or sustained negative vertical accelerations. Another reason for using industrial-like serial manipulators, is the fact that, being mass produced, they are cheaper than Stewart platforms or other specialised design.

Since the adaptation of anthropomorphic manipulators as motion simulators is a recent concept, previous works in literature are mainly exploratory studies that use only a subset of the DoF of the robot (see e.g., Beykirch

[2] http://www.toyota-global.com/innovation/safety_technology/safety_measurements/driving_simulator.html

[3] http://www.experts.renault.com/kemeny/projects/ultimate/

[4] A modified KUKA KR500 that is certified for mounting of a passenger seat/cabin, http://www.kuka-entertainment.com/en/products/robocoaster/

et al. (2008); Nusseck et al. (2008); Pretto et al. (2009)) or that work without a human-in-the-loop (see e.g., Bellman et al. (2007); Pollini et al. (2008)). A complete control framework for simulating general vehicle dynamics by using a 6 DoF anthropomorphic robot arm was recently presented by Robuffo Giordano et al. (2010b,c) under the name CyberMotion Simulator (CMS), and it was also demonstrated at the ILA Berlin air show 2010[5] for a passive (without pilot) helicopter simulation.

The work presented in this chapter represents a step further in the direction of developing more flexible motion simulators based on industrial manipulators. It is important to point out that the novel CMS cabin represents the first design of an actuated cabin that can be equipped on a serial manipulator used as a motion simulator. Another cabin design was presented by DLR (Bellmann et al. (2011a,b)) at the same time as the CMS cabin (Masone et al. (2011))). The DLR cabin can only be connected to the end-effector of the serial manipulator in a few *fixed* and therefore is not as flexible as a CMS cabin and it does not provide an additional degree of freedom to the simulator.

2.2 Preliminaries

The novel actuated cabin consists of two main parts, a gondola and an actuation system. The gondola (illustrated in Fig. 2.2) is a $1.6 \times 1.8 \times 1.9$ m closed shell that encapsulates the seat (and the pilot). The closed shell design allows to eliminate the unwanted sensory cues that come from the external environment, thus improving the simulated experience. The dome of the gondola also plays the role of a large screen which, together with two projectors mounted at the back of the cabin, allows to have a stereo visualisation and a larger field of view in comparison to a traditional monitor. Moreover, the gondola offers a great flexibility in terms of the input devices it can carry. For example, it has been used with i) a full force-feedback helicopter control device, which includes cyclic stick, collective stick and actuated pedals, and ii) a driving setup, composed by a force-feedback steering wheel and pedals module. The possibility of having interchangeable input devices makes the cabin suitable for the simulation of different typologies of vehicles. The communication between the onboard

[5] http://tuebingen.mpg.de/en/kybernetik-neuigkeiten/detail.html?tx_ttnews%5Btt_news%5D=116&tx_ttnews%5BbackPid%5D=118&cHash=02d983541b49f603f4e14e2dad6d705c

control electronics and the control computer is done via a controller area network (CAN) bus running at 1 kHz.

The actuation system is composed of several parts, as illustrated in Fig. 2.2. A rigid flange (blue part in Fig. 2.2) is attached at the end-effector of the manipulator and it mounts two servomotors capable of 328 Nm output torque[6]. The servomotors set in motion a system of gears ending with two gears per side (*pivots*) that connect to metal rails. The connection to the cabin is provided by two identical and parallel rails (green parts in Fig. 2.2) that are rigidly fixed at the back of the gondola. By connecting the flange and pivots (red parts in Fig. 2.2) to the rails, the servomotors can move the cabin thus providing an additional DoF for the simulator.

The control architecture for the overall 7 DoF CMS (6 DoF robot + cabin) is organized in two levels. The low-level controller (LLC) accepts joint increment commands that are implemented at a fast rate, and it returns the measured joint configuration with an output rate of $T_s = 0.012$ s. The inner structure of the LLC is not accessible due to the agreement with KUKA (manufacturer of the manipulator that is used in the CMS), therefore the high-level controller (HLC) is designed disregarding any dynamical issue and considering joint velocities as actual control inputs, as it is classically done within the *kinematic control* framework (see Chiacchio et al. (1991)). Formally, let $\boldsymbol{q} = [\boldsymbol{q}_M^T \; q_C]^T \in \mathbb{R}^7$ be the joint configuration vector of the complete 7 DoF CMS, where $\boldsymbol{q}_M = [q_1 \; \dots \; q_6] \in \mathbb{R}^6$ is the joint vector of the original KUKA Robocoaster and $q_C \in \mathbb{R}$ is the parameter describing the configuration of the cabin, which is hereinafter treated as a seventh joint. The LLC accepts joint increments $\Delta\boldsymbol{q}_k = \boldsymbol{q}(t_k + 1) - \boldsymbol{q}(t_k)$ and returns the measured joint configuration $\boldsymbol{q}(t_k)$ with the output rate T_s. From the point of view of the HLC, the joint motion is modelled as a single integrator

$$\dot{\boldsymbol{q}} = \boldsymbol{u}, \tag{2.1}$$

where $u \in \mathbb{R}^7$ is the commanded joint velocity. The HLC scheme presented in Sec. 2.4 is built on top of (2.1).

Finally, the joint limitations (joint range, maximum joint velocity and maximum joint acceleration) of the 7 DoF CMS are reported in Tab. 2.1.

[6] In order to reduce the weight of the cabin one servomotor has been later removed. However performance was not affected since the second motor was present for redundancy.

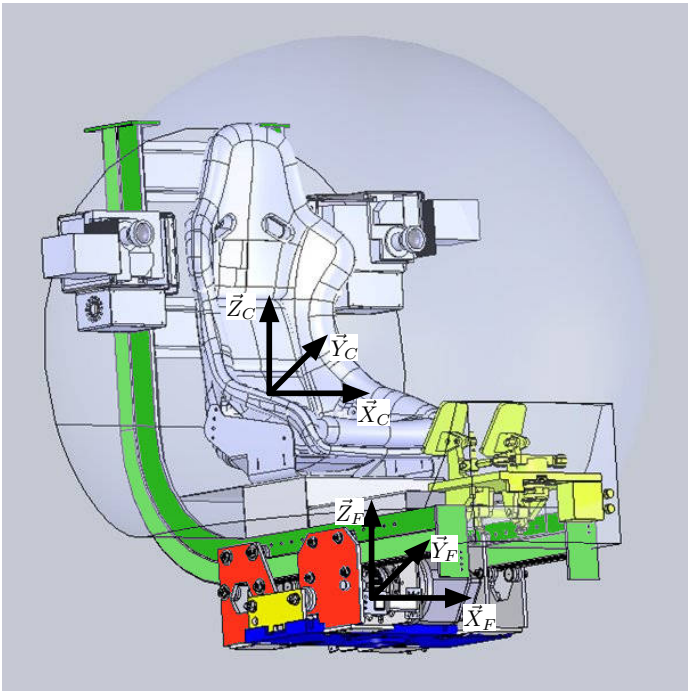


Figure 2.2: Details of the novel actuated cabin for the CyberMotion Simulator.

Joint q	q_{min}	q_{max}	$\dot{q}_{max}$	$\ddot{q}_{max}$
q_1	$-130°$	$130°$	$69°/s$	$98°/s^2$
q_2	$-128°$	$-48°$	$57°/s$	$70°/s^2$
q_3	$-45°$	$92°$	$69°/s$	$128°/s^2$
q_4	$-180°$	$180°$	$76°/s$	$33°/s^2$
q_5	$-58°$	$58°$	$76°/s$	$95°/s^2$
q_6	$-180°$	$180°$	$120°/s$	$77°/s^2$
q_C	$0m$	$1.7317m$	$0.34m/s$	$0.6m/s^2$

Table 2.1: Joint range, velocity and acceleration limits of the 7-DOF manipulator with actuated cabin. Entries q_1 to q_6 are expressed in degrees, q_7 in meters.

2.3 Cabin Kinematics

In order to design the HLC for the 7 DoF CMS it is necessary to provide a kinematic description of the new joint, i.e., a description of the pose and velocity of the cabin w.r.t. the robotic manipulator and parameterised by a suitable configuration variable q_C. For this purpose, two frames of reference are introduced (see Fig. 2.2). The first frame, $\mathcal{F}_F : \{O_F; \vec{X}_F; \vec{Y}_F; \vec{Z}_F\}$, is fixed to the center of the flange and such that $\vec{Z}_F$ is perpendicular to the flange itself, and $\vec{X}_F$ and $\vec{Y}_F$ are respectively perpendicular and parallel to the rotation axes of the two engines. The second frame, $\mathcal{F}_C : \{O_C; \vec{X}_C; \vec{Y}_C; \vec{Z}_C\}$, is fixed to the gondola and such that $\vec{X}_C$ and $\vec{Z}_C$ are aligned with the gondola forward/upwards direction, respectively, while $\vec{Y}_C$ is parallel to $\vec{Y}_F$. Note that with this choice the direction $\vec{Y}_C$ is perpendicular to the rails of the actuation system, therefore no motion is allowed in this direction. Namely, the displacement between O_C and O_F along $\vec{Y}_C$ is fixed and, without loss of generality, equal to zero.

Since there is no relative motion between the two frames in the direction $\vec{Y}_C \equiv \vec{Y}_F$, the analysis of the kinematics of the cabin can be conducted on the plane Σ spanned by $\{\vec{X}_F, \vec{Z}_F\}$. In light of this consideration, the formulation hereinafter will refer to the sketch in Fig. 2.3, which depicts the simplified projection of the rails, pivots and flange on Σ. The projection

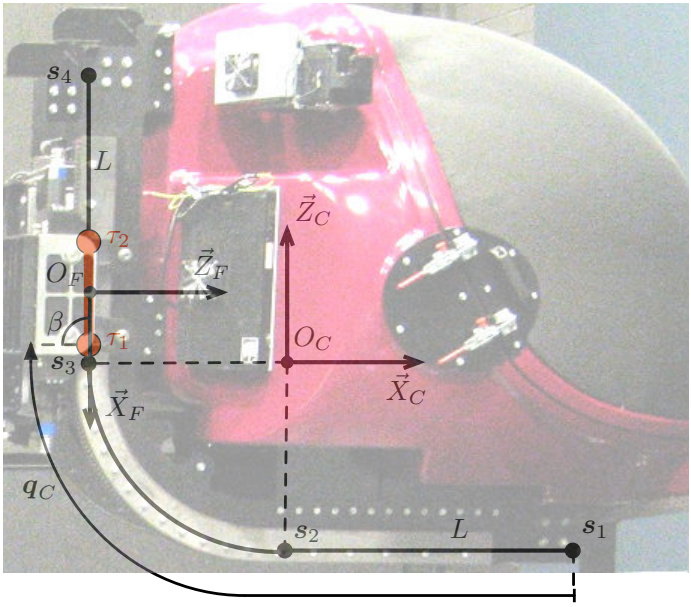


Figure 2.3: Simplified sagittal view of the cabin, obtained by projection on the plane $\Sigma = \text{span}\{\vec{X}_F,\ \vec{Z}_F\}$

of the rails on Σ consists of two segments, $\widehat{s_1\, s_2}$, $\widehat{s_3\, s_4}$[7], having length L and connected by a quarter of a circle $\widehat{s_2\, s_3}$ having radius λ. Without loss of generality, the origin O_C of frame $\mathcal{F}_\mathcal{C}$ is placed at the center of $\widehat{s_2\, s_3}$ (see Fig. 2.3). The projection on Σ of the pivots, i.e., the gears in contact with the rails, is given by the two points τ_1 and τ_2. Finally, since the position of the pivots is fixed with respect to the flange, this is represented by the segment $\widehat{\tau_1\, \tau_2}$ (the red thick segment in Fig. 2.3) which, assuming negligible deformations, has a fixed length, i.e.,

$$\|\tau_1 - \tau_2\| = d. \tag{2.2}$$

The mechanical design of the cabin also implies that τ_1 and τ_2 are bound to move on $s_1\, \widehat{s_2\, s_3}\, s_4$. The physical parameters in this model are reported in Tab. 2.2.

[7]The symbol $\widehat{\ }$ is used to denote a segment or arc.

L	d	λ	θ
$0.570m$	$0.200m$	$0.504m$	$11.446°$

Table 2.2: Physical parameters of the cabin

With this setting, the relative orientation ${}^{F}R_{C} \in SO(3)$ from $\mathcal{F}_C$ to $\mathcal{F}_F$ is parameterized by a single angle β (see Fig. 2.3) as

$$ {}^{F}R_{C} = \begin{pmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{pmatrix}. \tag{2.3} $$

The translation vector from O_F to O_C is in the form[8] ${}^{F}\boldsymbol{p}_{C} = ({}^{F}p_{C,x}\ 0\ {}^{F}p_{C,z})^{T} \in \mathbb{R}^3$, where ${}^{F}p_{C,x}$ and ${}^{F}p_{C,z}$ are functions to be determined. Since O_F was chosen at the center of the flange, it follows that

$$ {}^{F}\boldsymbol{p}_{C} = -\,{}^{F}R_{C}\ {}^{C}\boldsymbol{p}_{F}, \tag{2.4} $$

where

$$ {}^{C}\boldsymbol{p}_{F} = {}^{C}O_{F} = \frac{{}^{C}\boldsymbol{p}_{1} + {}^{C}\boldsymbol{p}_{2}}{2}, \tag{2.5} $$

where ${}^{C}\boldsymbol{p}_{1}$ and ${}^{C}\boldsymbol{p}_{2}$ are the positions of τ_1 and τ_2 in $\mathcal{F}_C$.

Finally, the kinematic description of the joint must be parameterised as a function of a single configuration variable, q_C. The configuration variable is selected as the *arc length* of pivot τ_1 on $s_1\widehat{s_2\ s_3}\,s_4$. In particular, q_C is chosen such that $q_C = q_{C_{min}} = 0$ when $\tau_1 \equiv s_1$ and $q_C = q_{C_{max}} = \lambda\frac{\pi}{2} + 2L - d$ when $\tau_2 \equiv s_4$. Using this representation, in Sect. 2.3.1 it is discussed the expression of ${}^{F}R_{C}$ and ${}^{F}\boldsymbol{p}_{C}$ as functions of q_C.

2.3.1 Forward Kinematics

The forward kinematics for the new joint is solved by providing the expression of β in (2.3) and of ${}^{F}p_{C,x}$ and ${}^{F}p_{C,z}$ in (2.4) as functions of q_C. By visual inspection of Fig. 2.3, it is possible to identify the following cases for the cabin forward kinematics:

[8] Superscripts are used to indicate the frames where quantities are expressed.

Case $\mathbf{C_1}$: when τ_1 and τ_2 are both on a linear segment ($\widehat{s_1\, s_2}$ or $\widehat{s_3\, s_4}$), the cabin works as a prismatic (i.e., translational) joint. The frame $\mathcal{F}_C$ can only translate with respect to $\mathcal{F}_F$, while the relative orientation is constant.

Case $\mathbf{C_2}$: when τ_1 and τ_2 are both on the arc $\widehat{s_2\, s_3}$, the cabin acts as a revolute (i.e., rotational) joint. Owing to the placement of the two frames of reference, the position of O_C in $\mathcal{F}_F$ is constant and only the relative orientation between the two frames changes.

Case $\mathbf{C_3}$: when only τ_1 or τ_2 is on the arc $\widehat{s_2\, s_3}$, both the relative position and orientation between the $\mathcal{F}_C$ and $\mathcal{F}_F$ change. In this case the behaviour of the cabin differs from that of other standard joints (see Waldron and Schmiedeler (2008)).

The switch among these cases occurs when either τ_1 or τ_2 is coincident with s_2 or s_3. The corresponding *switching conditions* can be written in terms of q_C as

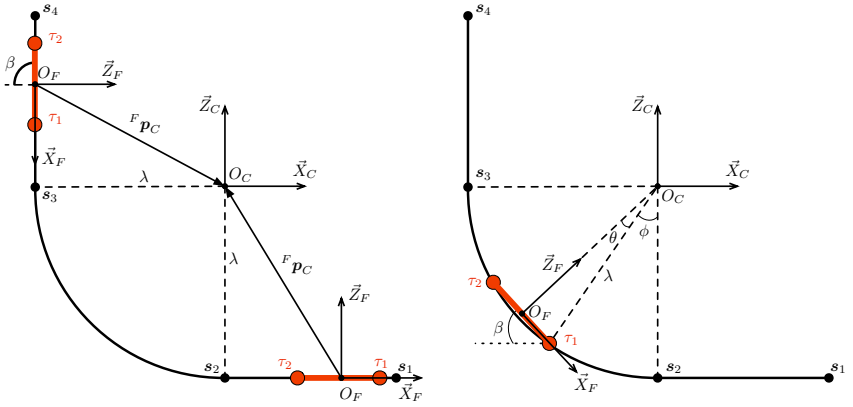
$$\begin{cases} \tau_2 \equiv s_2 & \Rightarrow & q_C = q_{C_1} = L - d, & C_1 \rightleftharpoons C_3 \\ \tau_1 \equiv s_2 & \Rightarrow & q_C = q_{C_2} = L, & C_3 \rightleftharpoons C_2 \\ \tau_2 \equiv s_3 & \Rightarrow & q_C = q_{C_3} = L + \lambda(\frac{\pi}{2} - 2\,\theta), & C_2 \rightleftharpoons C_3 \\ \tau_1 \equiv s_3 & \Rightarrow & q_C = q_{C_4} = L + \lambda\frac{\pi}{2}, & C_3 \rightleftharpoons C_1 \end{cases}, \tag{2.6}$$

where $\theta = \arcsin(\frac{d}{2\lambda})$ and $C_i \rightleftharpoons C_j$ indicates the transition between C_i and C_j. By construction it is $q_{C_{min}} \leq q_{C_1} \leq \ldots \leq q_{C_4} \leq q_{C_{max}}$. The three cases are addressed hereinafter.

Solution in case $\mathbf{C_1}$

This case presents itself in two different situations, as shown in Fig. 2.4a. In the first situation it is $q_C \in [q_{C_{min}},\, q_{C_1}]$, i.e., $\tau_1, \tau_2 \in \widehat{s_1\, s_2}$. The constraint on the pivots implies that $\mathcal{F}_F$ is oriented exactly as $\mathcal{F}_C$, i.e. $\beta = 0$, and ${}^F\boldsymbol{p}_C = \begin{pmatrix} q_C + \frac{d}{2} - L & 0 & \lambda \end{pmatrix}^T$.

In the second situation it is $q_C \in [q_{C_4},\, q_{C_{max}}]$, i.e., $\tau_1, \tau_2 \in \widehat{s_3\, s_4}$. The constraint on the pivots implies that $\mathcal{F}_F$ is rotated by 90° w.r.t. $\mathcal{F}_C$, i.e. $\beta = \frac{\pi}{2}$, and ${}^F\boldsymbol{p}_C = \begin{pmatrix} \frac{d}{2} - L + q_C - \lambda\frac{\pi}{2} & 0 & \lambda \end{pmatrix}^T$.



(a) Case C_1. (b) Case C_2.

Figure 2.4

Solution in case $\mathbf{C_2}$

In this case it results $q_C \in [q_{C_2},\, q_{C_3}]$ and $\tau_1, \tau_2 \in \widehat{s_2\, s_3}$. The constraint on the pivots implies that the frame $\mathcal{F}_F$ moves on a circumference of radius λ, centered in O_C and with $\vec{Z}_F$ always pointing towards the center. By visual inspection of Fig. 2.4b, it is clear that the vector ${}^F\boldsymbol{p}_C$ is fixed and equal to ${}^F\boldsymbol{p}_C = \begin{pmatrix} 0 & 0 & -\lambda\cos\theta \end{pmatrix}^T$. As for the relative orientation, it is $\beta = \theta + \phi$ with $\phi = \frac{(q_C - L)}{\lambda}$.

Solution in case $\mathbf{C_3}$

Once again, this case presents itself in two different situations, as shown in Fig. 2.5. Consider first the situation in which $q_C \in [q_{C_1},\, q_{C_2}]$. In this situation, the relative pose of $\mathcal{F}_C$ and $\mathcal{F}_F$ is subject to three constraints: i) $\tau_1 \in \widehat{s_1\, s_2}$, ii) $\tau_2 \in \widehat{s_2\, s_3}$ and iii) constraint (2.2) which implies that τ_1 and τ_2 lie on a circle centered in the projection of O_F on Σ and having radius $\frac{d}{2}$. The first constraint implies that ${}^C\boldsymbol{p}_1 = ({}^Cp_{1,x}\ 0\ {}^Cp_{1,z})^T = \begin{pmatrix} L - q_C & 0 & -\lambda \end{pmatrix}^T$. The second and third constraints together allow to determine the position of the second pivot ${}^C\boldsymbol{p}_2 = \begin{pmatrix} {}^Cp_{2,x} & 0 & {}^Cp_{2,z} \end{pmatrix}^T$ by

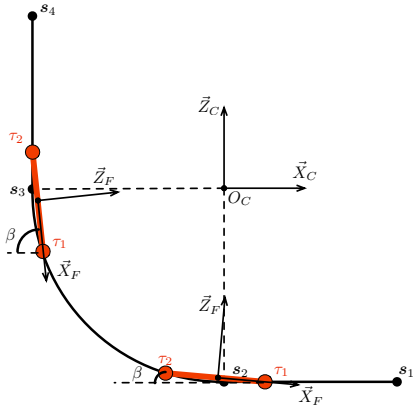


Figure 2.5: Case C_3.

solving the following system

$$\begin{cases} {}^{C}p_{2,x}^2 + {}^{C}p_{2,z}^2 = \lambda^2 \\ ({}^{C}p_{2,x} - {}^{C}p_{1,x})^2 + ({}^{C}p_{2,z} - {}^{C}p_{1,x})^2 = d^2 \end{cases} . \tag{2.7}$$

With the parameters of Table 2.2, system (2.7) always admits two solutions (intersections), and the correct one corresponds to ${}^{C}p_{2,x} < 0$.

Consider now the second situation in which $q_C \in [q_{C_3}, q_{C_4}]$. The constraints in this situation are i) $\tau_1 \in \widehat{s_2\, s_3}$, ii) $\tau_2 \in \widehat{s_3\, s_4}$ and iii) constraint (2.2) on the distance between τ_1 and τ_2. The first constraint implies that ${}^{C}\boldsymbol{p}_1 = ({}^{C}p_{1,x} \;\; 0 \;\; {}^{C}p_{1,z})^T = \begin{pmatrix} -\lambda \sin\phi & 0 & -\lambda\cos\phi \end{pmatrix}^T$. The second constraint gives a component of ${}^{C}\boldsymbol{p}_2$, i.e., ${}^{C}p_{2,x} = -\lambda$. Finally, similarly to the previous situation, the third constraint provides a second order equation, i.e.,

$$(-\lambda + \lambda \sin\phi)^2 + ({}^{C}p_{2,z} + \lambda\cos\phi)^2 = d^2. \tag{2.8}$$

whose positive solution is ${}^{C}p_{2,z}$.

To conclude this case, in both situations here discussed it is $\beta = \arctan \frac{({}^{C}p_{2,z} - {}^{C}p_{1,z})}{({}^{C}p_{1,x} - {}^{C}p_{2,x})}$.

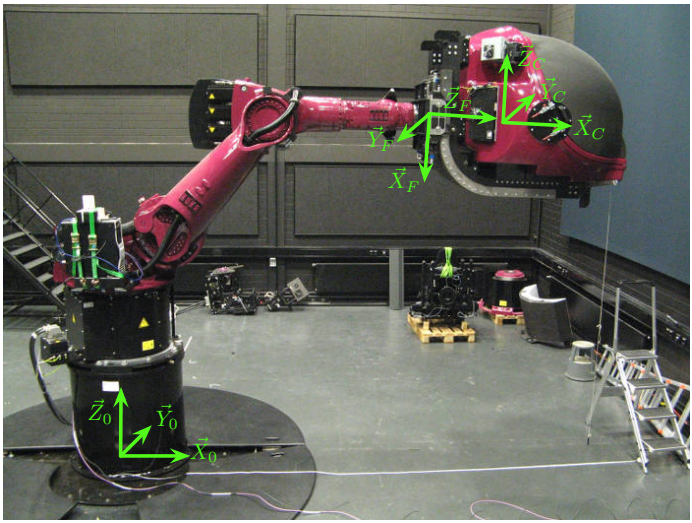


Figure 2.6: Side view of CyberMotion Simulator with the actuated cabin

Complete Forward Kinematics

So far, it was shown that the forward kinematics (2.3) to (2.5) of the 7th joint is described by a function that is split in five parts, according to the partition of the range $[q_{C,min}, q_{C,max}]$ that is introduced by (2.6). The complete forward kinematics of the 7 DoF system, follows straightforwardly by plugging the cabin kinematics (2.3) to (2.5) from $\mathcal{F}_F$ to $\mathcal{F}_C$ into the well known forward kinematics of the anthropomorphic manipulator from the chosen world frame $\mathcal{F}_0$ to $\mathcal{F}_F$ (see Fig. 2.6). Additionally, it can be included a fixed transformation from the cabin frame $\mathcal{F}_C$ to a frame $\mathcal{F}_P$ that is approximatively positioned where the head of the pilot, and therefore his/her vestibular sensors, is located. The overall transformation is obtained as

$$\begin{pmatrix} {}^0R_P(\boldsymbol{q}) & {}^0\boldsymbol{p}_P(\boldsymbol{q}) \\ \mathbf{0} & 1 \end{pmatrix} = \begin{pmatrix} {}^0R_F(\boldsymbol{q}_M) & {}^0\boldsymbol{p}_F(\boldsymbol{q}_M) \\ \mathbf{0} & 1 \end{pmatrix} \cdot \begin{pmatrix} {}^FR_C(q_C) & {}^F\boldsymbol{p}_C(q_C) \\ \mathbf{0} & 1 \end{pmatrix} \cdot \begin{pmatrix} I_{3\times 3} & {}^C\boldsymbol{p}_P \\ \mathbf{0} & 1 \end{pmatrix}$$

For the sake of readability, hereinafter the dependency from $\boldsymbol{q} \in \mathbb{R}^7$ will be omitted, unless strictly necessary.

So far, the orientation of frame $\mathcal{F}_P$ w.r.t. $\mathcal{F}_0$ has been described by the rotation matrix ${}^0R_P$, however for the design of the controller more efficient representations can be used. A common choice is to resort to Euler angles (see Robuffo Giordano et al. (2010b)), since the attitude of vehicles such as aircrafts or marine vehicles is generally given in this form. However, Euler angles introduce representation singularities, i.e., configurations in which the representation used for the orientation is not unique. For example, with a roll-pitch-yaw (RPY) Euler representation, a singularity occurs when the pitch angle is $\pm\frac{\pi}{2}$. This problem can be avoided by using unit quaternions to represent orientations. Hereinafter, the orientation of frame $\mathcal{F}_P$ w.r.t. $\mathcal{F}_0$ will be indicated by the unit quaternion $\boldsymbol{\eta} = (\mu\ \boldsymbol{\epsilon}^T)^T \in \mathbb{H}$, where $\mu \in \mathbb{R}$ is the scalar part of the quaternion and $\boldsymbol{\epsilon} \in \mathbb{R}^3$ is the vector part.

Analogously to the orientation, also the position of frame $\mathcal{F}_P$ w.r.t. $\mathcal{F}_0$ can be expressed using an alternative representation to the Cartesian coordinates ${}^0\boldsymbol{p}_C = (x\ y\ z)^T$. In particular, for the simulation of a vehicle motion the reference trajectory for the robot is produced by a motion cueing algorithm which takes as inputs the linear acceleration ${}^P\boldsymbol{a}$ and angular velocity ${}^P\boldsymbol{\omega}$ that act on the pilot according to the vehicle model. In this case and for the tests presented in Sec. 2.5, it is adopted the cylindrical MCA for anthropomorphic manipulators introduced by Robuffo Giordano et al. (2010c) and adapted to accommodate the use of Quaternions. The use of cylindrical coordinates in this MCA is motivated to better fit the workspace of an anthropomorphic manipulator. Therefore, hereinafter the position of the cabin in $\mathcal{F}_0$ is expressed in cylindrical coordinates $\boldsymbol{\xi} = (R\ \alpha\ z)^T \in \mathbb{R}^3$. The transformation from ${}^0\boldsymbol{p}_C$ to $\boldsymbol{\xi}$ is given by

$$\begin{cases} R = \sqrt{x^2 + y^2} \\ \alpha = \text{atan2}(y, x) \\ z = z \end{cases} \tag{2.9}$$

Summarizing, $\boldsymbol{r} = [\boldsymbol{\xi}^T\ \ \boldsymbol{\eta}^T] \in \mathbb{R}^7$ will be taken as *task variables* to be controlled.

2.3.2 Differential Kinematics

Similarly to the discussion of Sec. 2.3.1 for the forward kinematics, in order to compute the differential kinematics of the 7 DoF CMS it is necessary to

consider the contribution of the motion of the gondola w.r.t. the flange, i.e, how $\dot{q}_C$ determines the Cartesian/angular velocity of $\mathcal{F}_C$ w.r.t. $\mathcal{F}_F$. This contribution, expressed in world frame $\mathcal{F}_0$, is represented by the geometric Jacobian matrix (see Siciliano et al. (2009)) ${}^{F}J_C(\boldsymbol{q},\beta) \in \mathbb{R}^{6\times 1}$ defined as

$$J_C(\boldsymbol{q},\beta) = \begin{pmatrix} {}^{0}R_F(\boldsymbol{q}_M) & 0_{3\times 3} \\ 0_{3\times 3} & {}^{0}R_F(\boldsymbol{q}_M) \end{pmatrix} \begin{pmatrix} \frac{\partial^F \boldsymbol{p}_C}{\partial q_C} \\ \begin{pmatrix} 0 \\ -\frac{\partial \beta}{\partial q_C} \\ 0 \end{pmatrix} \end{pmatrix}, \tag{2.10}$$

where ${}^{0}R_F(\boldsymbol{q}_M) \in SO(3)$ is the rotation matrix from $\mathcal{F}_F$ to $\mathcal{F}_C$ and the quantities ${}^{F}\boldsymbol{p}_C$ and β were introduced in Sec. 2.3. Note that for this special joint the geometric Jacobian J_C is equivalent to the analytic Jacobian, i.e., the vector of roll-pitch-yaw rates of $\mathcal{F}_C$ w.r.t. $\mathcal{F}_F$, namely $(0 \;\; -\dot{\beta} \;\; 0)^T$, is equal to the angular velocity ${}^{C}\boldsymbol{\omega}$. This is because the rotation of $\mathcal{F}_C$ w.r.t. $\mathcal{F}_F$ is around a fixed axis in $\mathcal{F}_F$, i.e., $\vec{Y}_F$, (see Section 3.6 of Siciliano et al. (2009)). This behaviour can also be visualized by considering the transformation from Euler rates to angular velocity which is given by the following expression

$${}^{C}\boldsymbol{\omega} = \underbrace{\begin{pmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{pmatrix}}_{T_A(\beta)} \begin{pmatrix} 0 \\ -\dot{\beta} \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ -\frac{\partial \beta}{\partial q_C} \\ 0 \end{pmatrix} \dot{\boldsymbol{q}_C}.$$

The Jacobian matrix $J_M \in \mathbb{R}^{6\times 6}$ for the 6 DoF anthropomorphic manipulator is well known and it maps a joint velocity $\dot{\boldsymbol{q}}_M$ to a Cartesian/angular velocity of $\mathcal{F}_F$ w.r.t. $\mathcal{F}_0$. In order to map $\dot{\boldsymbol{q}}_M$ to velocities of the cabin it is necessary to take into account the displacement ${}^{F}\boldsymbol{p}_C$ from $\mathcal{F}_F$ to $\mathcal{F}_C$. This yields the following $\mathbb{R}^{6\times 6}$ Jacobian matrix

$$\bar{J}_M(\boldsymbol{q}) = \begin{pmatrix} I_{3\times 3} & [-{}^{0}R_F \, {}^{F}\boldsymbol{p}_C]_\wedge \\ 0_{3\times 3} & I_{3\times 3} \end{pmatrix} J_M(\boldsymbol{q}_M), \tag{2.11}$$

where $[\cdot]_\wedge \in \mathbb{R}^{3\times 3}$ is the skew-symmetric matrix obtained from a vector. With the same approach, the displacement from $\mathcal{F}_C$ to the pilot frame $\mathcal{F}_P$ is accounted by the following transformation

$$J_P(\boldsymbol{q}) = \begin{pmatrix} I_{3\times 3} & [-{}^{0}R_F \, {}^{C}\boldsymbol{p}_P]_\wedge \\ 0_{3\times 3} & I_{3\times 3} \end{pmatrix} \begin{pmatrix} \bar{J}_M(\boldsymbol{q}) & J_C(\boldsymbol{q}) \end{pmatrix}, \tag{2.12}$$

Finally, the complete task Jacobian mapping the joint velocity $\dot{q} \in \mathbb{R}^7$ of 7 DoF CMS to cyilindrical/angular velocity of the pilot in world frame is obtained as

$$J(\boldsymbol{q}) = \begin{pmatrix} T(\boldsymbol{\xi}(\boldsymbol{q})) & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} \end{pmatrix} J_P(\boldsymbol{q}) \tag{2.13}$$

where $T(\boldsymbol{\xi}(\boldsymbol{q}))$ is the matrix mapping from Cartesian to cylindrical coordinates, which has the form

$$T(\boldsymbol{\xi}(\boldsymbol{q})) = \begin{pmatrix} \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\dfrac{\sin\alpha}{R} & \dfrac{\cos\alpha}{R} \\ 0 & \end{pmatrix} & 0 \\ & I \end{pmatrix}$$

with α and R previously defined in (2.9). Note that the evaluation of $J(\boldsymbol{q})$ depends on the particular cases introduced in Sec. 2.3.1.

2.4 High-Level Control

As shown in Sec. 2.3.1 the forward kinematics of the robot has the form

$$\boldsymbol{r}(t) = \boldsymbol{f}(\boldsymbol{q}(t)), \tag{2.14}$$

with the corresponding differential relation derived in Sec. 2.3.2

$$\dot{\boldsymbol{r}}(t) = \frac{\partial\, \boldsymbol{f}(\boldsymbol{q}(t))}{\partial\, \boldsymbol{q}} \dot{\boldsymbol{q}}(t) = J(\boldsymbol{q}(t))\dot{\boldsymbol{q}}(t) \tag{2.15}$$

Now it is necessary to design a control law $\boldsymbol{u} = g(\boldsymbol{q}, \boldsymbol{r}_d, \boldsymbol{r})$ such that $\boldsymbol{r}(t)$ tracks $\boldsymbol{r}_d(t)$. In the classic kinematic control framework for manipulators (see Chiacchio et al. (1991); Siciliano (1990)) this problems is tackled in two stages: first, the desired trajectory is transformed via inverse kinematics into a desired joint trajectory $\boldsymbol{q}_d(t)$, i.e., by inverting the mapping (2.15); then the control problem is solved in the joint space. In the case of redundant manipulators, multiple joint trajectories can generate the same task trajectory and the typical approach to select a solution of the inversion is to the require a minimum-norm velocity, thus yielding a least-squares solution

$$\dot{\boldsymbol{q}}_d = J^{\dagger}\dot{\boldsymbol{r}}_d \tag{2.16}$$

where $J^{\dagger}$ indicates the Moore-Penrose pseudo inverse of J (see Meyer (2001)). However, in this case $\boldsymbol{r}_d$ might also be unfeasible because the

structure of the MCA does not allow to take explicitly into account the joint constraints (joint range, limited joint velocity and acceleration), that are summarized as

$$\begin{cases} \text{a)} & \forall i, \forall t \geq 0, & q_{i,min} \leq q_i(t) \leq q_{i,max} \\ \text{b)} & \forall i, \forall t \geq 0, & |\dot{q}_i(t)| \leq \dot{q}_{i,max} \\ \text{c)} & \forall i, \forall t \geq 0, & |\ddot{q}_i(t)| \leq \ddot{q}_{i,max} \end{cases} . \tag{2.17}$$

The numeric values for the terms $q_{i,min}$, $q_{i,max}$, $\dot{q}_{i,max}$ and $\ddot{q}_{i,max}$ are contained in Tab. 2.1. In addition to these limitations, the inversion scheme should avoid singularities or soften their effect by passing as 'smoothly' as possible through them.

The problem of tracking a trajectory with these issues has been widely addressed in literature. If $\boldsymbol{r}_d$ is known in advance, at least in its geometric path, then the problem can be solved by modifying the trajectory offline. Classic approaches for the offline optimization of the timing-law in presence of constraints are presented in Bobrow et al. (1985); Shiller (1994); Slotine and Yang (1989). However, for the problem here considered $\boldsymbol{r}_d$ is not known a priori because it depends on the unpredictable inputs that are provided by the human to the model of the vehicle. As discussed in Sec. 1.4, this is one of the main challenges of RTHL. Several works in literature have proposed online methods for joint limits and singularity avoidance based on artificial potentials (see Siciliano et al. (2009)) or on the local optimization of some index (see Chang and Dubey (1995); Nelson and Khosla (1995)). Nevertheless, such methods might start to degrade the tracking of $\boldsymbol{r}_d$ even when there is still margin before reaching a joint limit (2.17) or a singularity.

The problem of dealing online with the aforementioned constraints was tackled by implementing a control architecture for the 7 DoF CMS that builds upon the controller presented in Robuffo Giordano et al. (2010b) for a 6 DoF anthropomorphic manipulator. That approach is based on the combination of a Task Priority (TP) architecture (see Chiaverini (1997)) with a bang-bang control that is used to deterministically avoid joint limits given the maximum velocities/accelerations characteristics of Table 2.1.

The main insight of the bang-bang strategy is that each joint q_i has a bounded acceleration $\ddot{q}_{i,max}$ that can be used to determine exactly, given the current q_i, $\dot{q}_i$, the last moment the joint can be stopped before hitting its range limit. Without going into the well known details of bang-bang optimal control, the strategy is briefly summarized as follows: at every time and for every joint q_i the system has to check the intersection with

the switching curves

$$\begin{cases} \gamma^- : & q_i = q_{i,max} - \dfrac{\dot{q}_i^2}{2\ddot{q}_{i,max}} & \text{if } \dot{q}_i > 0 \\ \gamma^+ : & q_i = q_{i,min} + \dfrac{\dot{q}_i^2}{2\ddot{q}_{i,max}} & \text{if } \dot{q}_i < 0 \end{cases}$$

and then apply a maximum acceleration command $\pm\ddot{q}_{i,max}$ which is numerically integrated to generate the velocity control. Once the bang-bang control is active for a joint, then that joint will be *locked* until it fully stops and a command from the TP inversion scheme tries to move it away from the limit. When one or more joints are locked they cannot be used in the TP controller to track $\boldsymbol{r}_d$ and there can be a performance degradation. Therefore, the remaining joints must try to mitigate this effect. Indicating with the subscripts $\mathcal{L}$ and $\mathcal{U}$ the properties (joints, inputs, Jacobians) referred to the joints that are respectively locked/unlocked in the bang-bang strategy, the tracking problem can be reformulated to determining the input $\boldsymbol{u}_\mathcal{U}$ such that the trajectory

$$\dot{\boldsymbol{r}}_\mathcal{U} = \dot{\boldsymbol{r}} - J_\mathcal{L}\boldsymbol{u}_\mathcal{L} = J_\mathcal{U}\boldsymbol{u}_\mathcal{U} \tag{2.18}$$

tracks $\boldsymbol{r}_d$. The reader is referred to Robuffo Giordano et al. (2010b) for more details on this strategy.

Having summarized the bang-bang strategy to avoid joint range limits, the TP strategy to track $\boldsymbol{r}_d$ is now presented with the understanding that, if any joint is locked the strategy can be referred to $\boldsymbol{r}_\mathcal{U}$ instead of $\boldsymbol{r}$. The idea of TP architectures is that, for a redundant system, multiple tasks can be arranged with different priorities in order to try and simultaneously fullfill as many of them as possible. In this case, following the idea proposed in Robuffo Giordano et al. (2010b), the TP strategy is based on the consideration that correct orientation of gravity should be favoured in comparison to the other task variables because MCAs exploit reorientations of the gravitational acceleration to provide a correct feedback (tilt-coordination, see Grant and Reid (1997)). The advantage of this solution is that, close to singularities, even though the simulator cannot track the complete $\boldsymbol{r}_d$ it might be able to follow exactly the attitude. Furthermore, the 7 DoF CMS is redundant for the task and this redundancy can be exploited to reduce the risk of locking some joints. This is important because, despite the bang-bang strategy discussed before guarantees safe behaviours in all conditions, the performed motion can result heavily distorted when several joints become locked.

Three tasks have been considered for the TP controller, i.e.,

$$\begin{aligned} \boldsymbol{w}_1 &= \boldsymbol{f}_1(\boldsymbol{q}) \in \mathbb{R}^3 \\ \boldsymbol{w}_2 &= \boldsymbol{f}_2(\boldsymbol{q}) \in \mathbb{R}^3 \\ w_3 &= \boldsymbol{f}_3(\boldsymbol{q}) \in \mathbb{R} \end{aligned} \tag{2.19}$$

with priority decreasing from 1 to 3. Following the previous observations, the task trajectory $\boldsymbol{r}_d$ is split in two subtasks: highest priority task, $\boldsymbol{w}_1$, is the orientation task, and the secondary task $\boldsymbol{w}_2$ is the position task. Finally, the third task w_3 can be specified to optimize some aspects of the execution, by exploiting the redundancy of the system.

Starting from the first task, the form of $\boldsymbol{w}_1$ depends on the representation chosen for the orientation. For example, using a representation with actual and desired Euler angles $\boldsymbol{\varphi}$ and $\boldsymbol{\varphi}_d$ respectively, $\boldsymbol{w}_1$ could be taken as

$$\boldsymbol{w}_1 = \dot{\boldsymbol{\varphi}}_d + K_O \Delta\boldsymbol{\varphi}$$

where $\Delta\boldsymbol{\varphi} = \boldsymbol{\varphi}_d - \boldsymbol{\varphi}$ denotes the orientation error and $K_O \in \mathbb{R}^{3\times3}$ is a positive definite and diagonal matrix of gains. The orientation error is included to overcome the numerical drift due to the discrete-time integration of $\dot{\boldsymbol{r}}(t)$, according to the well known CLIK paradigm Chiacchio et al. (1991). In order to express $\boldsymbol{w}_1$ for the unit Quaternion representation, it is necessary to use a suitable orientation error. For this purpose, the error between the desired orientation $\boldsymbol{\eta}_d = (\mu_d \ \boldsymbol{\epsilon}_d^T)^T$ and the actual orientation $\boldsymbol{\eta} = (\mu \ \boldsymbol{\epsilon}^T)^T$ is computed, as shown in Siciliano (1998), as

$$\Delta\boldsymbol{\epsilon} = \mu\boldsymbol{\epsilon}_d - \mu_d\boldsymbol{\epsilon} - [\boldsymbol{\epsilon}_d]_\wedge\boldsymbol{\epsilon}. \tag{2.20}$$

The error $\Delta\boldsymbol{\epsilon}$ defined in (2.20) is the vector part of the unit quaternion that describes the relative orientation between the frames identified by $\boldsymbol{\eta}_d$ and $\boldsymbol{\eta}$. The orientation task is finally chosen as

$$\boldsymbol{w}_1 = \boldsymbol{\omega}_d + K_O\Delta\boldsymbol{\epsilon}, \tag{2.21}$$

according to the unit Quaternion CLIK algorithm (see Caccavale and Siciliano (2001))

Similarly to the primary task, the position task $\boldsymbol{w}_2$ is chosen according to the CLIK algorithm as

$$\boldsymbol{w}_2 = \dot{\boldsymbol{\xi}}_d + K_P\Delta\boldsymbol{\xi} \tag{2.22}$$

where $\Delta\boldsymbol{\xi} = \boldsymbol{\xi}_d - \boldsymbol{\xi}$ and $K_P \in \mathbb{R}^{3\times3}$ is a positive definite diagonal gain matrix.

Finally, the lowest priority task w_3 can be designed to locally optimize some metric of the robot, e.g., dexterity. A common choice in the kinematic control of a redundant manipulator is to maximize the manipulability ellipsoid, which was introduced by Yoshikawa (1985) to represent the directional capability of the robot to execute a motion from a certain configuration, and that is measured by the manipulator Jacobian as $\det(J(\boldsymbol{q})J^T(\boldsymbol{q}))$. Different extensions of Yoshikawa's manipulability ellipsoid have been proposed, as discussed in Doty et al. (1995). In the case of the CMS, it is not only important to maximize dexterity but also to keep the joints away from their limits in order to prevent distortion of the motion. For this purpose, it was used a measure that was introduced in Nelson and Khosla (1995) and which combines Yoshikawa's manipulability measure with a penalty term that goes to zero close to joint limits. The expression of this function is

$$H(\boldsymbol{q}) = \left(1 - e^{-k\prod_{i=1}^{n} \frac{(q_i - q_{i,min})(q_{i,max} - q_i)}{(q_{i,max} - q_{i,min})^2}} \right) \sqrt{\det(J(\boldsymbol{q})J^T(\boldsymbol{q}))} \tag{2.23}$$

where k is a design parameter and n is the number of joints. By maximizing $H(\boldsymbol{q})$ the robot will stay away from the singularities and joint limits. The threshold between these two actions is determined by k. The lowest priority task is then implemented as a single step of a gradient ascent as

$$w_3 = k_C \frac{\partial H(\boldsymbol{q})}{\partial \boldsymbol{q}}^T , \tag{2.24}$$

where $k_C > 0$ is a user defined scalar that weights the task.

There are different variants of TP strategies, as shown in Antonelli (2009), and in this case it was used an augmented inverse projection scheme based on the classic recursive implementation from Siciliano and Slotine (1991), i.e.,

$$\dot{\boldsymbol{q}}_i = \dot{\boldsymbol{q}}_{i-1} + (J_i N_{A,i-1})^{\#}(\boldsymbol{w}_i - J_i \dot{\boldsymbol{q}}_{i-1}) \tag{2.25}$$

where $i = 1, 2, ...$ is the priority of the tasks, the superscript $^{\#}$ indicates the generalised inverse of a matrix, J_i is the corresponding Jacobian and $N_{A,i-1} = (I - J_{A,i}^{\#} J_{A,i})$ is the null-space projection matrix for the augmented Jacobian $J_{A,i} = [J_1^T \; J_2^T \; \ldots \; J_{i-1}^T \; J_i^T]^T$. By applying (2.25), the control

law becomes

$$\begin{aligned}\boldsymbol{u} =& J_1^{\#}\boldsymbol{w}_1 + (J_2 N_{A,1})^{\#}(\boldsymbol{w}_2 - J_2 J_1^{\#}\boldsymbol{w}_1) + \\ & (N_{A,2})^{\#}(\boldsymbol{w}_3 - J_1^{\#}\boldsymbol{w}_1 - (J_2 N_{A,1})^{\#}(\boldsymbol{w}_2 - J_2 J_1^{\#}\boldsymbol{w}_1)).\end{aligned} \tag{2.26}$$

Moreover, in order to avoid ill-conditioning in the implementation of (2.26), the generalised inverse was computed with a singularity-robust pseudoinversion based on the numerical filtering by Maciejewski and Klein (1985). In particular, indicating with $J = \sum_{i=1}^{s} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$ the singular value decomposition of matrix J, then $J^{\#}$ is implemented as

$$J^{\#} = \sum_{i=1}^{s} \frac{\sigma_i}{\sigma_i^2 + \lambda_i^2} \boldsymbol{v}_i \boldsymbol{u}_i^T \tag{2.27}$$

where the parameter λ_i is the damping factor and it is chosen according to Chiaverini et al. (1991), as

$$\lambda_i^2 = \begin{cases} 0 & \text{if } \sigma_i \geq \epsilon \\ \left(1 - \left(\frac{\sigma_i}{\epsilon}\right)^2\right) \lambda_{max}^2 & \text{if } \sigma_i < \epsilon \end{cases} \tag{2.28}$$

where $\epsilon > 0$ is a parameter that defines the size of the singular region and $\lambda_{max} > 0$ is the maximum damping value. Note that this implementation introduces a damping only along those directions that are unfeasible while keeping the remaining directions unchanged.

The control scheme described so far has not considered the limits on joint velocities and accelerations (2.17). In order to ensure feasibility of the trajectory given the aforementioned constraints it has been used the saturation strategy that was proposed in Robuffo Giordano et al. (2010b). This saturation strategy relies on the discretization of the control loop to unify (2.17)a and (2.17)b. In particular, the constraint on joint velocities defines a region of admissible velocities

$$B_1 = \left\{ \dot{\boldsymbol{q}} \in \mathbb{R}^7 : |\dot{q}_i| \leq \dot{q}_{i,max} \right\}.$$

Similarly, using the control loop interval T_s to define the maximum joint velocity increment $\Delta\dot{q}_{i,max} = T_s\, \ddot{q}_{i,max}$, constraint (2.17)b determines a region of feasible velocity increments

$$B_2 = \left\{ \dot{\boldsymbol{q}} \in \mathbb{R}^7 : |\dot{q}_i - \dot{q}_{k,i}| \leq \Delta\dot{q}_{i,max} \right\}.$$

The idea of the saturation strategy is then to apply a uniform saturation of the velocity command if possible, i.e., if there is a $0 \leq k \leq 1$ such that $k\boldsymbol{u} \in B_1 \cap B_2$. In this way, the direction of motion of the end-effector remains unaltered. When that is not possible, it is used a non-uniform saturation that does not change the direction of the acceleration $\ddot{\boldsymbol{q}}$.

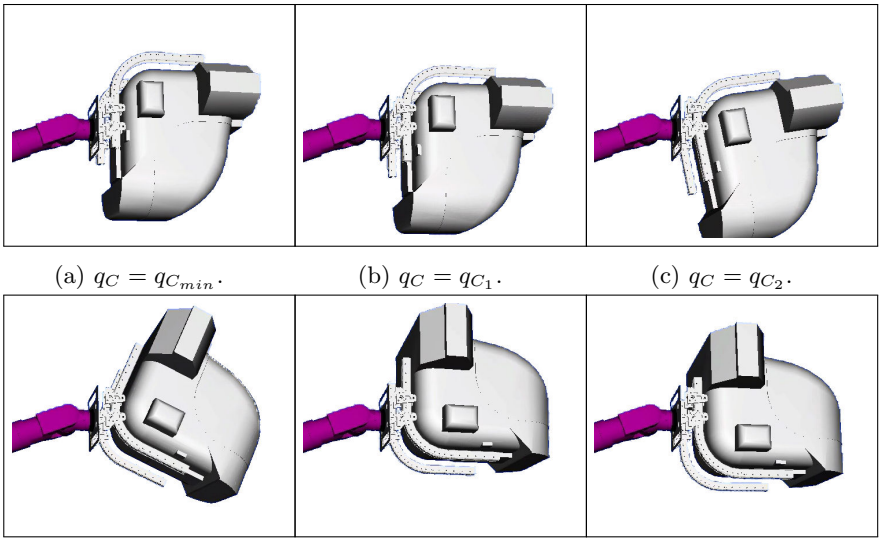
2.5 Results

The 7 DoF CMS here presented was evaluated in simulation and with experiments.

Simulation 1 First, the kinematic model of the cabin was tested in a Matlab/Simulink simulation. For this test, the cabin is mounted on the 6 DoF robot and commanded a joint velocity $\dot{q}_C = 0.2$, from a starting configuration $q_C = q_{C_{min}}$ until $q_C = q_{C_{max}}$, while the arm is kept fixed, i.e., $\dot{\boldsymbol{q}}_M = \mathbf{0}_6$. The 3D models of the cabin and of the robot used for the simulation are faithful representations of the real hardware based on measurements of the various parts (see also the parameters in Tab. 2.2). The snapshots from the simulation in Fig. 2.7 show the motion of the cabin and in particular the configurations corresponding to the switching conditions described in Sec. 2.3.1.

The plots in Fig. 2.8 illustrate the trajectory of the frame $\mathcal{F}_\mathcal{C}$ in $\mathcal{F}_0$, in particular showing the Cartesian position ${}^0\boldsymbol{p}_C$ (Fig. 2.8a), RPY orientation $\boldsymbol{\varphi}_C$ (Fig. 2.8b), linear velocity ${}^0\dot{\boldsymbol{p}}_C$ (Fig. 2.8c) and angular velocity $\boldsymbol{\omega}_C$ (Fig. 2.8d). In all the plots, vertical dashed lines indicate the time when a switching condition is met. Observe that, in accordance to the discussion from Sec. 2.3.1, the motion is split in several phases: i) until $\boldsymbol{q}_C = \boldsymbol{q}_{C_1}$ the joint has a linear vertical motion (in $\mathcal{F}_0$) with constant speed; ii) from $\boldsymbol{q}_C = \boldsymbol{q}_{C_1}$ until $\boldsymbol{q}_C = \boldsymbol{q}_{C_2}$ the joint also starts rotating; iii) from $\boldsymbol{q}_C = \boldsymbol{q}_{C_2}$ until $\boldsymbol{q}_C = \boldsymbol{q}_{C_3}$ the joint only rotates with constant angular velocity; iv) from $\boldsymbol{q}_C = \boldsymbol{q}_{C_3}$ until $\boldsymbol{q}_C = \boldsymbol{q}_{C_4}$ there is again a mixed translation and rotation; v) finally, from $\boldsymbol{q}_C = \boldsymbol{q}_{C_4}$ until $\boldsymbol{q}_C = \boldsymbol{q}_{C_{max}}$ the joint has a linear vertical motion (in $\mathcal{F}_0$) with constant speed.

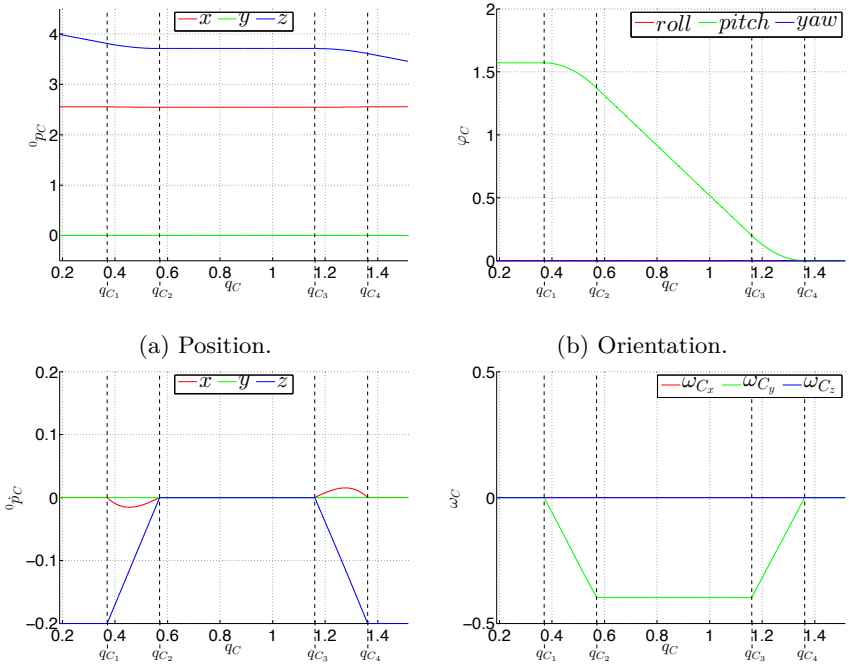
Simulation 2 The overall framework was then tested in a Matlab/Simulink simulation and compared to the original 6 DoF system presented in Robuffo Giordano et al. (2010b,c). The goal of this simulation is to realize a constant linear deceleration ${}^C\boldsymbol{a}_d = [-7\ 0\ 0]^T$ [m/s^2] in $\mathcal{F}_C$ that acts on the pilot



(a) $q_C = q_{C_{min}}$. (b) $q_C = q_{C_1}$. (c) $q_C = q_{C_2}$.

(d) $q_C = q_{C_3}$. (e) $q_C = q_{C_4}$. (f) $q_C = q_{C_{max}}$.

Figure 2.7: Simulation 1 - snapshots.



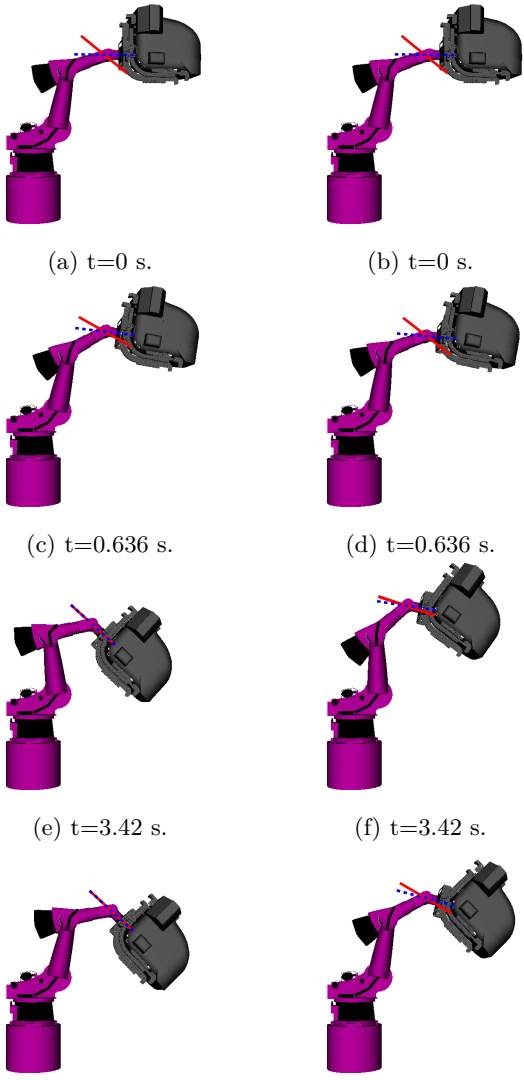
(a) Position.

(b) Orientation.

(c) Linear velocity.

(d) Angular velocity.

Figure 2.8: Simulation 1 - plots



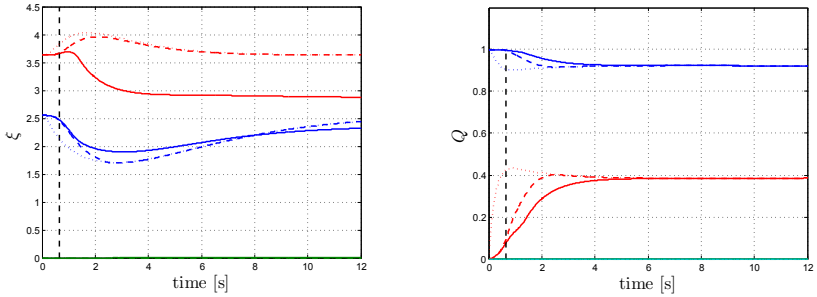
(a) t=0 s. (b) t=0 s.

(c) t=0.636 s. (d) t=0.636 s.

(e) t=3.42 s. (f) t=3.42 s.

(g) t=12 s. (h) t=12 s.

Figure 2.9: Snapshots of the robot during the execution of the task, without using the 7-th joint (left column: a,c,e,g) and using it (right column: b,d,f,h). The red line indicates the current orientation of the 5-th joint and the dashed blue line indicates its limit.



(a) (b)

Figure 2.10: Evolution of the task variables. Solid lines refer to the fixed cabin case, dashed lines refer to the actuated cabin case and dotted lines represent the reference task. In Fig. 2.10(a), the color red is used for z, blue for R and green for α. In Fig. 2.10(b), the color blue is associated to μ, while green, red and cyan are used to depict the components of $\boldsymbol{\epsilon}$, in this exact order.

under the assumption that ${}^{C}\boldsymbol{p}_P \simeq \mathbf{0}_3$. This deceleration is translated by an MCA into a desired task trajectory r_d that has the following characteristics: the cabin should move backwards (in $\mathcal{F}_C$) to reproduce the onset cue, and tilt downwards to reproduce the persistent cue by exploiting gravity. The manipulator starts from the configuration $\boldsymbol{q}_M(t_0) = [0 \ -80 \ 60 \ 0 \ 20 \ 0]^T$ [deg] and the cabin from the configuration $q_C(t_0) = 1.34$ [m], both well inside the joint limits. The corresponding initial task configuration is $\boldsymbol{r}(t_0) = [2.55 \ 0 \ 3.63 \ 1 \ 0 \ 0 \ 0]^T$, with the cabin being perfectly vertical (i.e., $\vec{Z}_C = \vec{Z}_0$).

This task was chosen because the sustained brake is a relevant cue for a car simulator. In Robuffo Giordano et al. (2010b,c) the CyberMotion Simulator was used with a fixed seat to simulate a Formula 1 car. However, in that case the maximum simulated forward deceleration without significant false cues was up to -4 [m/s^2], while the simulated car was capable of braking well beyond this value. This simulation will then show that the new CyberMotion Simulator/cabin allows to increase the level of sustained deceleration reproducible without significant artefacts.

In the rest of this section, the smallest singular value of the primary task Jacobian J_1 is indicated with σ_P and the smallest singular value of the

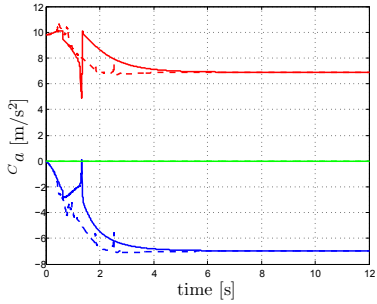


Figure 2.11: Simulated acceleration ${}^C\boldsymbol{a}$ in $\mathcal{F}_C$. Solid lines refer to the fixed cabin case, dashed lines refer to the actuated cabin case. Blue, green and red lines refer to accelerations along $\vec{X}_C$, $\vec{Y}_C$ and $\vec{Z}_C$, respectively

secondary task 'coupling' matrix $J_2 N_{A,1}$ is denoted with σ_S. Furthermore, in the following plots the quantities relative to the fixed cabin case (6 DoF) are represented with solid lines, those relative to the actuated cabin with dashed lines, and any reference quantity with dotted lines.

The behaviour of the robot during the execution of the task can be understood with the help of Fig. 2.9, that shows snapshots of the simulation for the non-actuated and actuated cabin case respectively. In the snapshots it is also depicted the current direction of the 5th link of the robot (blue dashed line) and its limit (solid red line). Consider first the fixed cabin case. From the starting position in Fig. 2.9(a), the cabin must move backwards along $\vec{X}_C$ and at the same time must tilt downwards to correctly reorient the gravity vector. In order to reorient the seat without deviating from $\boldsymbol{\xi}_d$, joints 3 and 5 are mainly exploited. In particular the latter is driven towards its limit and at the time $T = 0.636$ [s] the bang-bang controller activates and locks it (Fig. 2.9(c)). Hereafter, this event is indicated in the plots with a vertical dashed line. At this point, the robot is not able to execute simultaneously $\boldsymbol{\eta}_d$ and $\boldsymbol{\xi}_d$ anymore, joint 5 reaches its limit and the cabin visibly moves downwards (Fig. 2.9(e)). Finally, the onset cues expires and the desired orientation of the cabin is reached (Fig. 2.9(g)). The execution of the task trajectory is also shown in Fig. 2.10. In particular, starting at time T the robot is not able to track $\boldsymbol{r}_d$ anymore. While the primary task $\boldsymbol{\eta}$, despite a little deviation from the reference, follows $\boldsymbol{\eta}_d$ (Fig. 2.10(b)), the execution of the secondary task $\boldsymbol{\xi}$ is visibly different from $\boldsymbol{\xi}_d$ (Fig. 2.10(a)).

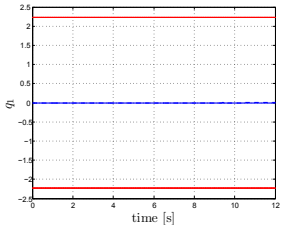
Since $\boldsymbol{\xi}_d$ is generated by the motion cueing algorithm to reproduce onset accelerations, this mismatch on $\boldsymbol{\xi}$ produces a very strong false cue during the transient. Note also that the desired forward deceleration -7 [m/s^2] is reached at about $t = 6$ [s].

Consider now the actuated cabin case. Starting from the same initial configuration (Fig. 2.9(b)), the reorientation is more efficiently executed by exploiting the 7-th axis. At the time T (Fig. 2.9(d) joint 5 is not locked and the robot can continue to track $\boldsymbol{\xi}_d$ to reproduce the onset cue (Fig. 2.9(f)). Eventually, the movement stops and the cabin remains in the desired orientation (Fig. 2.9(h)). The task trajectory is tracked accurately (see Fig. 2.10) and thus the desired deceleration is simulated without significant false cues (Fig. 2.11). The reference acceleration -7 [m/s^2] is also reached earlier, at about $t = 2.9$ [s].

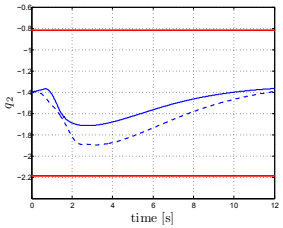
The different performance in the two cases can be analyzed by looking at the evolution of the joint variables during the task (Fig. 2.12). In the fixed cabin case, after being locked at the time T, joint 5 remains at its upper limit until the end (Fig. 2.12(e)). This also results in a larger evolution for joint 3 (Fig. 2.12(c)), that compensates for the loss of mobility. Clearly, at time T the modified manipulability function (2.23) rapidly goes to zero (Fig. 2.12(h)). Moreover, when joint 5 becomes locked, σ_S goes to zero as the secondary task 'coupling' matrix $J_2 N_{A,1}$ corresponding to the unlocked joints is singular (Fig. 2.13(b)). Anyway, thanks to the TP control, this singularity does not affect the primary task and σ_P is always greater than its starting value (Fig. 2.13(a)). With the addition of the actuated cabin, the burden on joints 3 and 5 is reduced and the vector q remains well inside the limits. Also, the manipulability function (2.23) never decreases below the starting value (Fig. 2.12(h)), and neither σ_P nor σ_S encounter singularities (Fig. 2.13).

Experiment Finally, the complete system with the 7 DoF CMS was tested in an experiment with a human-in-the-loop, who is asked to drive the virtual racing car modelled in Robuffo Giordano et al. (2010c). Figure 2.14 shows a snapshot taken during the experiment from a camera mounted inside the cabin, behind the pilot. In the figure it is clearly visible the visual feedback projected on the interior of the gondola as well as the input device used to command the virtual car.

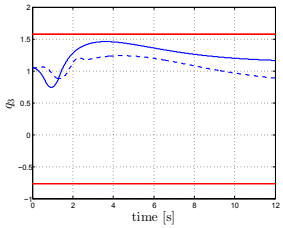
Figures 2.15a and 2.15b illustrate the desired forward and lateral acceleration expressed $\mathcal{F}_P$ that should be acting on the pilot, according to



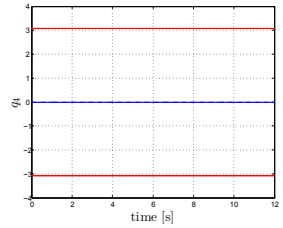
(a) q_1



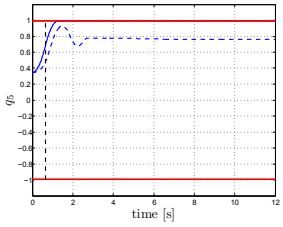
(b) q_2



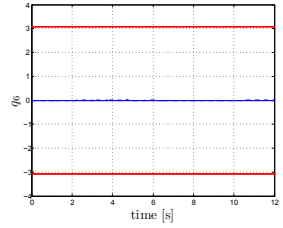
(c) q_3



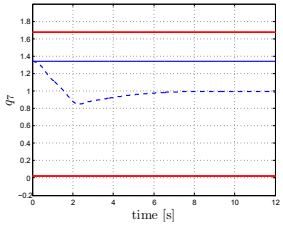
(d) q_4



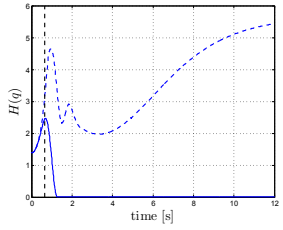
(e) q_5



(f) q_6

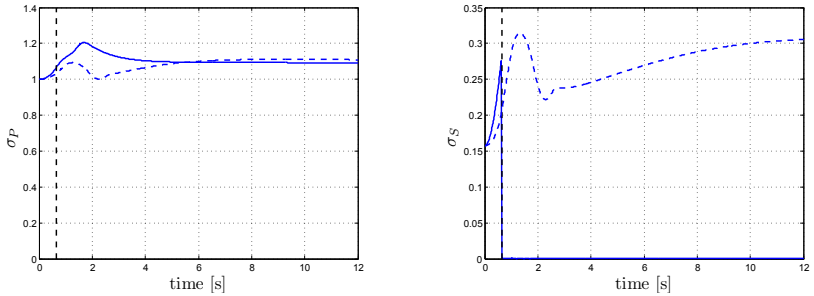


(g) q_7



(h) $H(\boldsymbol{q})$

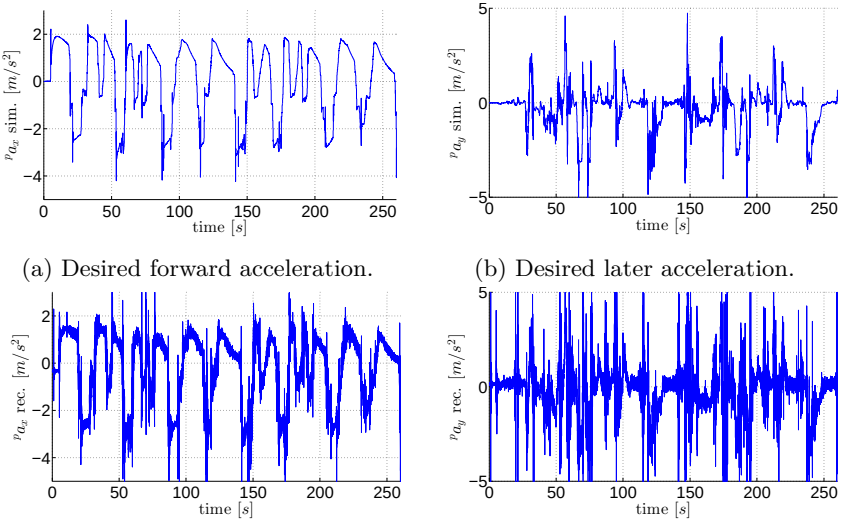
Figure 2.12: Figures 2.12(a-g) show the evolution of the joint variables during the execution. The red lines indicate the limits for each joint. (h) shows the evolution of the modified manipulability function (2.23). In all the figures solid lines refer to the fixed cabin case, dashed lines refer to the actuated cabin case.



(a) σ_P (b) σ_S

Figure 2.13: Evolution of σ_P and σ_S during the execution. Solid lines refer to the fixed cabin case, dashed lines refer to the actuated cabin case.

Figure 2.14: Experiment - snapshot from a camera inside the cabin.



(a) Desired forward acceleration.

(b) Desired later acceleration.

(c) Measured forward acceleration.

(d) Measured lateral acceleration.

Figure 2.15: Experiment - plots.

the vehicle model and to the motion cueing. Note that these accelerations have been scaled down with respect to the original accelerations produced by the car model in order to avoid excessive strain on the untrained pilot. The accelerations acting on the pilot within the cabin were also measured by an IMU mounted within the gondola. The measured forward and lateral accelerations are plotted in Figs. 2.15a and 2.15b. By comparing the plots of the desired and measured accelerations it is visible a substantial agreement of the two, however the measured acceleration clearly exhibits strong vibrations. Apart from the unavoidable noise, this is mainly due to the actual vibrations of the motion platform caused by the large mass added at the end of the manipulator and not accounted for by the low-level controller. Indeed these vibrations severely degrade the vestibular feedback, therefore suggesting for the future developments to i) reduce the weight of the cabin, and ii) account, if possible, for the additional mass in the low-level controller. Furthermore, it should be noted that this experiment considers an extreme scenario (racing car), as a proof of concept, but most applications (e.g., driving a normal car) present smaller and smoother accelerations.

2.6 Summary and Possible Extensions

Summarizing, the following results have been presented:

1. It was presented a novel design of actuated cabin that extends the motion envelope of anthropomorphic robotic arms to be used as motion simulator. The kinematic model of the cabin was formally given, showing a mixed translational/rotational behaviour.

2. It was described the control architecture of the 7 DoF system which exploits the redundancy of the system to optimise the motion of the platform and is capable to cope with the unpredictable trajectory generated online by the human operator.

3. The framework was validated both in simulation and with a real experiment, using the CMS simulator to simulate the sensation of driving a race car.

Because of the prioritization in (2.26), an extension to the proposed high-level controller could be the adoption of a saturation algorithm that takes into account the different priorities. A similar idea was developed

in Arrichiello et al. (2009) by considering the case of the sole actuator velocity saturations, and by applying different uniform scalings to the tasks starting from the highest priority one until the lowest priority one. Another possible extension to this research, could be the adoption of a model predictive control for the high-level controller, in case it is provided a model of the pilot or if the commanded trajectory has some given geometric structure (for example exploiting the knowledge of the street or track where the simulated vehicle moves). Moreover, additional tests and experiments could be executed to evaluate the benefit of the chosen lower priority task of the TP controller in comparison to other choices was well as the general benefit of using the actuated cabin in term of the motion perceived by test subjects.

Chapter 3

Shared control of a UAV bearing-formation

The discussion presented in this chapter is based upon the work that I have done under the supervision of Dr. Franchi and published in Franchi et al. (2011a, 2012a).

3.1 Introduction

As mentioned in Sec. 1.1, field applications represent one of the most important domains of RTHL, with tasks ranging from exploration to coverage and surveillance, see e.g. Franchi et al. (2009); Howard et al. (2006); Renzaglia et al. (2012); Schwager et al. (2011). Such applications are often carried out using multi-robot systems due to their resilience to single-point failures and adaptability to different scenarios. Among the others, UAVs are probably the most suitable robotic platform for remote exploration of large and/or unstructured areas since they possess large adaptability and potential pervasiveness in many different scenarios. The problem, as already discussed in Sec. 1.4.2, is that the robots seldom operate in full autonomy, on the contrary the *human-to-robot ratio* is typically larger than 1, i.e. it is usually needed more than one human for operating a single robot. In order to make the whole multi-robot system easily manageable by a single person, in this chapters it is explored the idea of i) increasing the autonomy of the robots by letting them autonomously achieve and keep a desired formation, and ii) improving the human-robot interaction with suitable cues (such as haptic) that let the human co-operator 'feel' the remote robots and site.

The design of a formation controller for the UAVs is however dependent on the measurements that are available. In multi-robot research, it

is generally considered a requirement to devise motion controllers based only on *relative measurements* (see Franchi et al. (2010)). This way, the controller is independent of the knowledge of the robot absolute positions in space and, thus, does not require global localization systems such as GPS or SLAM algorithms (see, e.g., Durham et al. (2012)). In the same spirit, substantial research efforts have been devoted to the development of solutions based on standard, light-weight, low-energy, and cheap sensors (like *onboard cameras*), rather than active and energetically-expensive sensing devices (like, e.g., laser/structured-light range-sensors), as shown in Schwager et al. (2011). When mapping/surveillance tasks are based on relative measurements obtained by cameras, a parallel objective for the group of robots is the maintenance of a 3D formation that optimizes some suitable 'visually-motivated' performance criterion. For example, in the context of environmental coverage (see Schwager et al. (2011)), one can minimize the overlap of the camera field-of-views (FoVs) or maximize the focus on some areas of interest. Similarly, in visual-based cooperative map building, the robot 3D formations can be carefully designed so as to facilitate the acquisition of those relative measurements strictly needed for solving the mutual localization problem (see Cognetti et al. (2012)) — a necessary prerequisite for accurate merging of the individual maps. Since, regardless of the particular application, cameras eventually provide as a direct measurement only a *bearing* (angular) information, in this chapter it is considered the problem of realizing and keeping a desired UAV 3D formation defined in terms of *sole relative (inter-robot) bearings* (i.e., a *bearing-formation*).

Summarizing, the main characteristics of this RTHL scenario are depicted in Fig. 3.1:

Robot: The robots (UAVs) implement a formation controller based only on relative bearing measurements that autonomously achieves and maintains the desired formation.

Human: While fulfilling the relative bearing constraints, the *remaining* DoF of the formation are exploited to let the human co-operator steer the whole group of UAVs.

Feedback: Together from the visual feedback from the onboard cameras, a suitable bilateral controller provides force cues informative of how well the UAVs, as a group, are executing the human's intended motion.

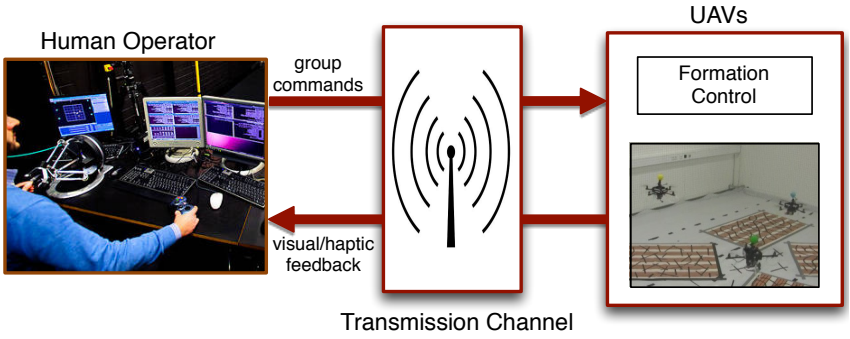


Figure 3.1: Shared formation control scheme.

3.1.1 Related Works

The previous works related to the topics of this paper can be roughly grouped in the following two categories:

Bearing/Vision-based Formation control

In literature, the use of bearing measurements has been mainly explored for groups of non-holonomic ground (2D) robots with a special attention to leader-follower configurations. In Das et al. (2002) a leader-follower control based on input-output feedback linearization is proposed. Every robot can control bearing and distance from another robot, or a combination of them from two other robots, and obstacle points. The quantities needed for the controller are either retrieved by an omnidirectional camera or estimated via EKF. Shakernia and Sastry (2003) presents a visual-servoing approach exploiting motion segmentation from panoramic images to estimate position and velocity of the leaders. In Johnson et al. (2004) distance estimation from the leader using bearing plus acceleration measurements is achieved using an EKF and a neural network. In Orqueda and Fierro (2006) a leader-follower approach based on feedback linearization is proposed. The relative pose (bearing, distance and orientation) is estimated directly from the image using fiducial markers and a high gain non-linear observer. In Moshtagh et al. (2009) parallel and circular flocking formations (i.e., with constant non-zero speed) are obtained using the bearing angle, optical flow and time to collision.

Even though all the aforementioned approaches do not assume availability of distance measurements, they also aim at concurrently regulating all the robot *inter-distances*. This is typically achieved by fusing bearing-only measurements with additional metric information such as acceleration, velocity, or known size of objects. Therefore the fulfillment of suitable observability conditions must be taken into account as, e.g., shown in Mariottini et al. (2009), as well as the maintenance of some sort of *persistency of excitation* condition during the motion. This can be obtained by either requiring the presence of a leader in charge of constantly "pulling" the formation, or by perturbing the motion of the robots with sinusoidal-like inputs.

Such a persistent-excitation behavior is not needed by the formation controller presented in this chapter, which neither uses nor regulates the inter-distances between the robots (while albeit ensuring their boundedness). The lack of a metric scale could be overcome, in some limited situations, by fusing visual information with onboard accelerometer readings as done in Kelly and Sukhatme (2011); Kneip et al. (2011). However, the solution presented here delegates to the human co-operator the role of regulating the expansion/contraction rate of the UAV group.

High-level steering of multiple mobile robots

Any approach involving a group of robots tracking a collective reference trajectory can be loosely considered as non-bilateral (*unilateral*, without force feedback) steering of multiple mobile robots. For instance, in Belta and Kumar (2004) a group of robots is made able to track given trajectories in the reduced space of some global quantities (e.g., the centroid of the formation), and conceptually similar problems are addressed in Antonelli et al. (2009); Kitts and Mas (2009) and references therein. Compared to the scenario considered here, all these approaches: i) propose different solutions to multi-robot formation control, ii) do not focus on the specific case of bearing-only measurements, and iii) only consider the unilateral steering case.

While many papers in the past literature did consider bilateral (with force feedback) teleoperation of a *single mobile robot*, only a few addressed the multiple mobile robot case. In Lee and Spong (2005) a passivity-based approach to bilaterally teleoperate a group of holonomic/non-holonomic ground robots is presented, and in Rodríguez-Seda et al. (2010) bilateral teleoperation of a group of UAVs is realized by directly coupling the position of the haptic device to the position of the formation centroid. This solution

does not take into account the *kinematic dissimilarity* between the haptic device and the slave mobile robots (bounded vs. unbounded workspace), which, on the contrary, is explicitly considered in Lee et al. (2011). Here the authors present a UAV bilateral teleoperation scheme where the *velocity* of the group is controlled by the *position* of the haptic device, while still guaranteeing passivity (I/O stability) of the teleoperation system. Along similar lines, in Franchi et al. (2011b, 2012c,d); Robuffo Giordano et al. (2011a,b); Secchi et al. (2012) a different bilateral teleoperation control strategy is proposed with the emphasis on the possibility to allow autonomous split and rejoin decisions within the group in a passive/stable way. Nevertheless, all the aforementioned approaches are not bearing-only based but explicitly require knowledge of metric information.

3.2 Preliminaries

This section introduces the models of the UAV and of a virtual kinematic system, called agent, that generates the reference trajectory for the UAV. The symbols used to define this models follow the general guidelines of the manuscript. In particular, vectors will be denoted with boldfaced symbols, scalars with normal weighted characters, and matrices with normal wighted capital characters.

3.2.1 UAV Model

The multi-robot system is composed of N UAVs, with $N \geq 3$, that are modeled as rigid bodies in space. The pose (position and attitude) of a rigid object in the environment is expressed with respect to an inertial (world) frame denoted with $\mathcal{F}_{\mathcal{W}} : \{O_{\mathcal{W}}; \vec{X}_{\mathcal{W}}; \vec{Y}_{\mathcal{W}}; \vec{Z}_{\mathcal{W}}\}$, where $O_{\mathcal{W}}$ indicates the origin of the frame and $\vec{X}_{\mathcal{W}}, \vec{Y}_{\mathcal{W}}, \vec{Z}_{\mathcal{W}}$ indicate its orthogonal axes (unit vectors). In order to describe the pose of the UAVs, each robot is endowed with a body fixed frame that is attached to the center of mass of the UAV itself. The body frame for the i-th UAV is denoted with $\mathcal{F}_{\mathcal{A}_i} : \{O_{\mathcal{A}_i}, \vec{X}_{\mathcal{A}_i}, \vec{Y}_{\mathcal{A}_i}, \vec{Z}_{\mathcal{A}_i}\}$, in accordance to the standard notation used for frames. An example of these frames is illustrated in Fig. 3.2. With this setting, the *configuration* of the i-th UAV is represented by the *position* ${}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_i} \in \mathbb{R}^3$ of $O_{\mathcal{A}_i}$ in $\mathcal{F}_{\mathcal{W}}$ and by the *rotation matrix* ${}^{\mathcal{W}}R_{\mathcal{A}_i} \in SO(3)$ that expresses the orientation of $\mathcal{F}_{\mathcal{W}}$ with respect to $\mathcal{F}_{\mathcal{A}_i}$. Clearly, a rotation matrix is only one of the possible representations for the orientation of the

robots in world frame. Another representation that is used hereinafter is in terms RPY Euler angles, with *roll* $\phi_{\mathcal{A}_i}$, *pitch* $\theta_{\mathcal{A}_i}$ and *yaw* $\psi_{\mathcal{A}_i}$ [1]. Each UAV is assumed to possess several properties, that are summarized in the following.

Measure of relative bearings Each UAV, e.g. the i-th UAV, is assumed capable of measuring the direction pointing towards the center of mass of another UAV, e.g. the j-th UAV, and expressed in body frame $\mathcal{F}_{\mathcal{A}_i}$ (*relative bearing*). Namely, this relative bearing is the direction from $O_{\mathcal{A}_i}$ to $O_{\mathcal{A}_j}$ in $\mathcal{F}_{\mathcal{A}_i}$, and it is formally defined as

$$\boldsymbol{\beta}_{\mathcal{A}_i\mathcal{A}_j} = {}^{\mathcal{A}_i}R_{\mathcal{W}} \frac{{}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_j} - {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_i}}{\|{}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_j} - {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_i}\|} \in \mathbb{S}^2, \tag{3.1}$$

where, as usual, a subscript/superscript shift denotes the inverse/transpose of a rotation matrix (i.e., ${}^{\mathcal{A}_i}R_{\mathcal{W}} = {}^{\mathcal{W}}R_{\mathcal{A}_i}{}^{-1}$). Note that the local frame of reference is not indicated in $\boldsymbol{\beta}_{\mathcal{A}_i\mathcal{A}_j}$, i.e. it is dropped the superscript from ${}^{\mathcal{A}_i}\boldsymbol{\beta}_{\mathcal{A}_i\mathcal{A}_j}$, because it is clear that relative bearings are expressed in the local frame. An example of relative bearings is shown in Fig. 3.2.

One important thing to point out is that, even though definition (3.1) uses the *absolute positions* ${}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_j}$ and ${}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_i}$, in practice it is not necessary to measure ${}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_j}$ and ${}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_i}$ to evaluate $\boldsymbol{\beta}_{\mathcal{A}_i\mathcal{A}_j}$. In fact, a direct measure of $\boldsymbol{\beta}_{\mathcal{A}_i\mathcal{A}_j}$ can be obtained by means of a calibrated monocular camera mounted on the i-th UAV. For this reason, in the following $\boldsymbol{\beta}_{\mathcal{A}_i\mathcal{A}_j}$ will be sometimes referred to as *measured* relative-bearing.

Measure of attitude Each UAV is assumed capable of locally measuring the roll and pitch angles ($\phi_{\mathcal{A}_i}$, $\theta_{\mathcal{A}_i}$). In practice, UAVs are always equipped with onboard inertial measurement units (IMUs consisting of accelerometers and gyros) that allow to retrieve these measures, for instance by implementing standard estimation algorithms such as complementary filters (see e.g., Mahony et al. (2008)). It will be shown in the following that these measures are used by the bearing-formation controller, in particular to evaluate (3.5).

[1] In literature these representation is also known as ZYX angles to indicate the order according to which the elementary rotations around the current coordinate axes are composed.

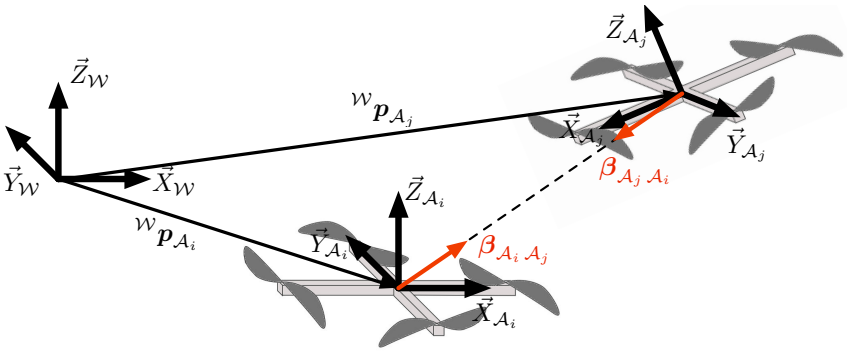


Figure 3.2: Model of two UAVs (in this case two quadrotors), each of them with a body-fixed frame, and the corresponding relative bearings.

The assumption above does not include the (absolute) yaw angle $\psi_{\mathcal{A}_i}$, because this information cannot be recovered from standard IMUs. Clearly, one could use additional specialized sensors, such as a compass, in which case the measure of $\psi_{\mathcal{A}_i}$ could be used by the bearing-formation controller and by the low-level trajectory controller. However, there are two reasons that deter from this approach:

- these additional sensors typically have a limited reliability (e.g., compasses do not work well indoor or close to strong magnetic fields);
- adding more sensors to the UAVs increases the overall weight and power draw, which is undesirable especially for very small vehicles.

In view of these considerations, the framework discussed in this chapter has been developed without requiring the measure of the absolute yaw angle.

Trajectory tracking The i-th UAV is assumed capable of tracking any smooth reference trajectory $(\boldsymbol{p}_i(t), \psi_i(t)) \in \mathbb{R}^3 \times \mathbb{S}^1$. This requirement is less demanding and more feasible than tracking any arbitrary trajectory $({}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_i}(t), {}^{\mathcal{W}}R_{\mathcal{A}_i}(t)) \in SE(3)$. A sufficient condition satisfying this assumption is that position and yaw angle $\left({}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_i}, \psi_{\mathcal{A}_i}\right)$ of the UAV are flat outputs (see Fliess et al. (1995)) of the system, i.e, together with their derivatives they algebraically define the state and control inputs of the UAV. Differential flatness is essentially equivalent to exact dynamic feedback

linearizability with $\left({}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_i},\, \psi_{\mathcal{A}_i}\right)$ taken as linearizing outputs (see Isidori (1995)). Helicopters and quadrotors are examples of differentially flat systems (see Mistler et al. (2001); Nieuwstadt and Murray (1998)).

From an implementation point of view, the trajectory tracker of each UAV is assumed to operate at the kinematic level, i.e., it is assumed capable of following a reference velocity trajectory $(\dot{\boldsymbol{p}}_i(t),\, \dot{\psi}_i(t))$ with good performance and keeping the tracking error small enough. This assumption has been used in previous related works that also included experimental validations: for example, in Schwager et al. (2009) the authors consider the UAV a simple kinematic integrator, in Fink et al. (2010) the UAVs are abstracted again as kinematic integrators with some additional details on low-level PID controls.

3.2.2 Agent Model

A virtual kinematic system, henceforth called *agent*, is introduced in order to generate a reference velocity trajectory $(\dot{\boldsymbol{p}}_i(t),\, \dot{\psi}_i(t))$ that is given as input to the trajectory tracker of the i-th UAV. The i-th agent can be imagined as a moving frame $\mathcal{F}_i : \{O_i;\, \vec{X}_i;\, \vec{Y}_i\, \vec{Z}_i\}$ that is free to translate in 3D, but whose orientation is constrained to have roll and pitch angles $\phi_i = \theta_i = 0$. With this setting, the pose of the i-th agent in the world frame is described by its position ${}^{\mathcal{W}}\boldsymbol{p}_i \in \mathbb{R}^3$ and by rotation matrix ${}^{\mathcal{W}}R_i = R_{\vec{Z}_i}(\psi_i)$ which indicates the canonical rotation matrix around the axis $\vec{Z}_i$. In order to make the notation less cumbersome, in the following the superscript/subscript $\mathcal{W}$ that denotes the world frame will be dropped from the pose of the agent. Namely, this means that the notation ${}^{\mathcal{W}}\boldsymbol{p}_i$, ${}^{\mathcal{W}}R_i$ will will be replaced by $\boldsymbol{p}_i$, R_i.

The kinematic model of the i-th agent is

$$\begin{pmatrix} \dot{\boldsymbol{p}}_i \\ \dot{\psi}_i \end{pmatrix} = \begin{pmatrix} R_i & \mathbf{0}_3 \\ \mathbf{0}_3^T & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_i \\ w_i \end{pmatrix}, \tag{3.2}$$

where $\mathbf{0}_3 = (0\,0\,0)^T$, and the *body-frame* linear velocity $\boldsymbol{u}_i \in \mathbb{R}^3$ and yaw-rate $w_i \in \mathbb{R}$ are the inputs. Vector $\boldsymbol{q}_i = (\boldsymbol{p}_i,\, \psi_i) \in \mathbb{R}^3 \times \mathbb{S}^1$ is defined as the i-th agent *configuration*, and vector $\boldsymbol{q} = (\boldsymbol{q}_1, \ldots, \boldsymbol{q}_N) \in (\mathbb{R}^3 \times \mathbb{S}^1)^N$ denotes the collection of configurations of all the agents. Lastly, for the well known properties of rotation matrices, the rotation between the body frames of agents i and j is described by ${}^iR_j = {}^iR R_j$, where ${}^iR = R_i^T$ as usual.

Analogously to UAVs (cf. (3.1)) a notion of relative bearing is formulated also for agents, as detailed hereinafter.

Agent relative bearing The direction between the i-th and j-th agent (seen from the body frame of the i-th agent) is defined as *agent relative bearing*. Formally, it is

$$\boldsymbol{\beta}_{ij}(\boldsymbol{q}) = \boldsymbol{\beta}_{ij}(\boldsymbol{q}_i, \boldsymbol{p}_j) = \frac{{}^{i}R\,\boldsymbol{p}_{ij}}{\delta_{ij}} \in \mathbb{S}^2, \tag{3.3}$$

where $\boldsymbol{p}_{ij} = \boldsymbol{p}_j - \boldsymbol{p}_i$, and

$$\delta_{ij}(\boldsymbol{q}) = \delta_{ij}(\boldsymbol{p}_i, \boldsymbol{p}_j) = \|\boldsymbol{p}_j - \boldsymbol{p}_i\| \tag{3.4}$$

is the *inter-distance* between agents i and j.

There is a subtle difference between the agent relative bearing defined in (3.3) and the measured UAV relative bearing introduced in (3.1). Besides possible mismatches as the UAV tracks the trajectory (3.2), the two bearings differ because the simplified agent model does not account for roll and pitch rotations of the body frame. Nevertheless, an evaluation of ${\boldsymbol{\beta}_{ij}}^2$ can be obtained from the roll $\phi_{\mathcal{A}_i}$, pitch $\theta_{\mathcal{A}_i}$, and relative bearing $\boldsymbol{\beta}_{\mathcal{A}_i\,\mathcal{A}_j}$ measured from the UAV, as

$$\boldsymbol{\beta}_{ij} \simeq R_{\vec{Y}_{\mathcal{A}_i}}(\theta_{\mathcal{A}_i}) R_{\vec{X}_{\mathcal{A}_i}}(\phi_{\mathcal{A}_i}) \boldsymbol{\beta}_{\mathcal{A}_i\,\mathcal{A}_j}, \tag{3.5}$$

where $R_{\vec{X}_{\mathcal{A}_i}}(\cdot), R_{\vec{Y}_{\mathcal{A}_i}}(\cdot)$ are the canonical rotation matrixes around the axes $\vec{X}_{\mathcal{A}_i}$ and $\vec{Y}_{\mathcal{A}_i}$, respectively. Approximation (3.5) holds as long as the discrepancy between the position of the j-th UAV in the frame of the i-th UAV, i.e., ${}^{\mathcal{A}_i}R_{\mathcal{W}}({}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_j} - {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{A}_i})$, and that of the associated virtual agents, i.e., ${}^{i}R(\boldsymbol{p}_j - \boldsymbol{p}_i)$, stays small enough. This has been confirmed by the extensive set of simulations and experimental results presented in Sec. 3.8. Furthermore, practical applications of UAV formations, such as map reconstruction or monitoring, do not feature aggressive motions but they rather require that the vehicles keep the angles $(\phi_{\mathcal{A}_i}, \theta_{\mathcal{A}_i})$ very small, for example for collecting data with down-facing sensors.

Among all the possible configurations of agents, there is one special situation that deserves attention and that is characterized by the following definition.

[2]The dependence from $\boldsymbol{q}$ is omitted here and in the following, whenever not needed or clear from the context.

Definition 3.1 (Degenerate configuration)**.** A configuration $\boldsymbol{q}$ is called *degenerate* if the associated agent positions $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_N$ are all aligned, i.e., $\boldsymbol{\beta}_{12}(\boldsymbol{q}) = \pm\boldsymbol{\beta}_{13}(\boldsymbol{q}) = \ldots = \pm\boldsymbol{\beta}_{1N}(\boldsymbol{q})$.

In the next section it will be shown that degenerate configurations correspond to a loss of several properties of a bearing-formation.

3.3 Relative bearings

Before proceeding with the design of the sought shared controller, it is of paramount importance

1) to understand what are the fundamental properties of the agent relative bearings;

2) to formally define a bearing-formation and to study its parameterization.

All these aspects are addressed in this section.

For the upcoming discussion, it is necessary to introduce few more concepts that will simplify the formulation. Firstly, recall from the notation guidelines that $[\boldsymbol{v}]_\wedge \in so(3)$ denotes the skew-symmetric matrix associated to a vector $\boldsymbol{v} = (v_1\, v_2\, v_3)^T \in \mathbb{R}^3$ (cf. (0.1)). The second concept regards the inter-distances among the agents:

Definition 3.2 (Inter-distance ratio)**.** For any 3 agents l, m, and n, the scalar ratios among their relative inter-distances are denoted as

$$\gamma_{lmn} = \frac{\delta_{ln}}{\delta_{lm}} \in \mathbb{R}^+, \quad \gamma_{mln} = \frac{\delta_{mn}}{\delta_{ml}} \in \mathbb{R}^+, \tag{3.6}$$

assuming that $\boldsymbol{p}_l$ and $\boldsymbol{p}_m$ are not coincident (see Fig. 3.3a). In the singular case $l = n$ and $m = n$, definition (3.6) is extended as

$$\gamma_{lml} = 0, \qquad \gamma_{lmm} = 1, \tag{3.7}$$

(see Figs. 3.3b and 3.3c).

3.3.1 Properties of Relative Bearings

It is now possible to introduce the three fundamental properties of *Orientation*, *Triangulation* and *Composition* for relative bearings.

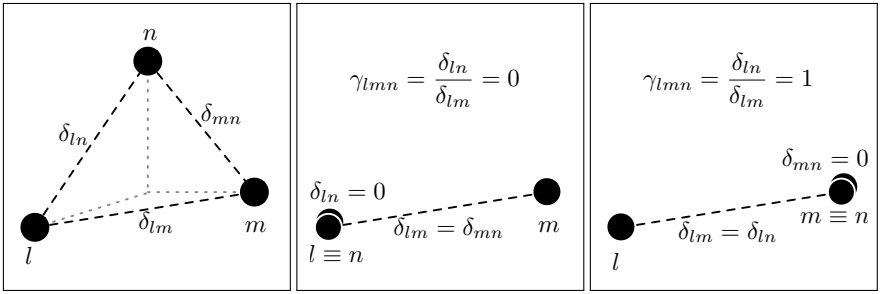


Figure 3.3: Inter-distances. a) For three non-coincident agents, formula (3.6) is valid. b) When $l \equiv n$, then $\gamma_{lmn} = 0$. c) When $m \equiv n$, then $\gamma_{lmn} = 1$.

Orientation

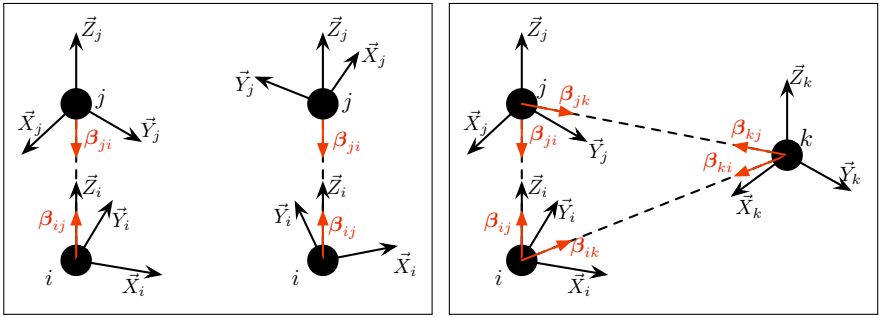
Property 3.1. *(Orientation)* If $\boldsymbol{\beta}_{ij} \neq \pm(0\ 0\ 1)^T$ then

$$ {}^iR_j = R\left(\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}\right) = \exp\left(\arccos(c_{ij})\,[\boldsymbol{v}_{ij}]_\wedge\right) \tag{3.8} $$

where $\boldsymbol{v}_{ij} = -\boldsymbol{\beta}_{ij} \times \boldsymbol{\beta}_{ji}$ and $c_{ij} = \boldsymbol{\beta}_{ij} \cdot \boldsymbol{\beta}_{ji}$. Moreover if $\boldsymbol{\beta}_{ij} = \pm(0\ 0\ 1)^T$ but $\boldsymbol{\beta}_{ik} \neq \pm(0\ 0\ 1)^T$ then

$$ {}^iR_j = {}^iR_k\, {}^kR_j = R\left(\boldsymbol{\beta}_{ik}, \boldsymbol{\beta}_{ki}\right) R\left(\boldsymbol{\beta}_{kj}, \boldsymbol{\beta}_{jk}\right). \tag{3.9} $$

Proof of Property 3.1. Multiplying both sides of (3.3) with R_i yields $R_i\boldsymbol{\beta}_{ij} = \boldsymbol{p}_{ij}/\delta_{ij}$, and symmetrically $R_j\boldsymbol{\beta}_{ji} = \boldsymbol{p}_{ji}/\delta_{ji}$. Noting that $\boldsymbol{p}_{ij} = -\boldsymbol{p}_{ji}$ and $\delta_{ij} = \delta_{ji}$, the left sides of the previous equations can be equated thus obtaining $R_i\boldsymbol{\beta}_{ij} = -R_j\boldsymbol{\beta}_{ji}$. This can be rewritten as ${}^jR_i\boldsymbol{\beta}_{ij} = -\boldsymbol{\beta}_{ji}$ showing that ${}^jR_i$ is the rotation matrix among the directions $\boldsymbol{\beta}_{ij}$ and $-\boldsymbol{\beta}_{ji}$. This consists of a rotation about the axis $\boldsymbol{v}_{ij}$ of an angle $\arccos(c_{ij})$, and takes the exponential form (3.8). To conclude the proof, note that ${}^iR_j$ is always a rotation matrix of the form $R_{\overline{Z}_j}(\cdot)$ by construction. Therefore, for any $a = b = \pm(0\,0\,1)^T$, the equation ${}^iR_j\, a = b$ does not admit a unique solution (any rotation matrix $R_{\overline{Z}_j}(\cdot)$ would be a solution), implying that ${}^iR_j$ cannot be evaluated whenever $\boldsymbol{\beta}_{ij} = \pm(0\ 0\ 1)^T$. Finally, the last statement of the proposition is proven by applying the first part of Property 3.1 twice. □



(a) (b)

Figure 3.4: Property 3.1: a) With just two vertically aligned agents i and j, it is not possible to uniquely define their relative orientation, because the body fixed frames $\mathcal{F}_i$ and $\mathcal{F}_j$ can freely rotate around $\vec{Z}_i \equiv \vec{Z}_j$. b) Adding a third agent that is not vertically aligned allows to disambiguate the rotation around the vertical axis.

Property 3.1 states that the relative orientation ${}^iR_j$ between agents i and j can be always retrieved from $\boldsymbol{\beta}_{ij}$ and $\boldsymbol{\beta}_{ji}$, as long as the two agents are not vertically aligned. If the two agents are vertically aligned, then the bearings $\boldsymbol{\beta}_{ij}$ and $\boldsymbol{\beta}_{ji}$ constrain the frames $\mathcal{F}_i$ and $\mathcal{F}_j$ to have their Z axes aligned, i.e., $\vec{Z}_i \equiv \vec{Z}_j$, however the rotation around $\vec{Z}_i$ (or $\vec{Z}_j$) is undetermined (see Fig. 3.4a). In this case it is necessary a third agent not vertically aligned with the first two in order to disambiguate the rotation of $\mathcal{F}_i$ and $\mathcal{F}_j$ (see Fig. 3.4b).

Triangulation

Property 3.2. *(Triangulation)* Consider 3 agents l, m, and n, s.t. $\boldsymbol{p}_l$, $\boldsymbol{p}_m$, and $\boldsymbol{p}_n$ are not aligned, i.e., $\boldsymbol{\beta}_{lm} \neq \pm\boldsymbol{\beta}_{ln}$. Then the following identities hold:

$$\gamma_{lmn} = \Gamma(\boldsymbol{\beta}_{mn}, \boldsymbol{\beta}_{ml}, \boldsymbol{\beta}_{ln}) \tag{3.10}$$

$$\gamma_{mln} = \Gamma(\boldsymbol{\beta}_{ln}, \boldsymbol{\beta}_{lm}, \boldsymbol{\beta}_{mn}), \tag{3.11}$$

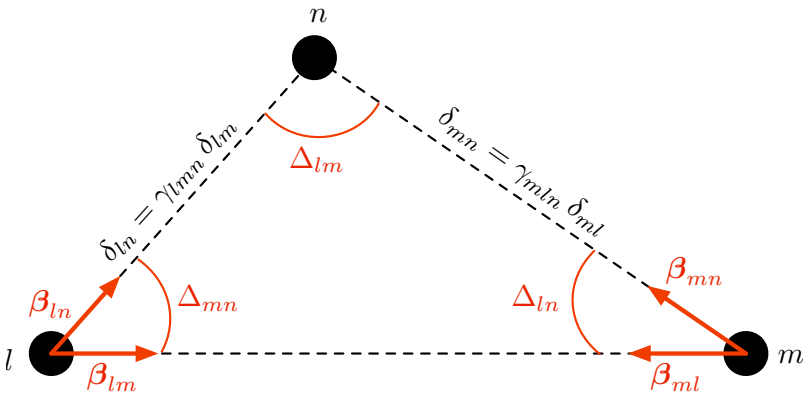


Figure 3.5: The plane spanned by the three agents l, n, m and Property 3.2.

where the function $\Gamma : (\mathbb{S}^2)^3 \to \mathbb{R}^+$ is defined as

$$\Gamma(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \boldsymbol{\beta}_3) = \frac{\|\boldsymbol{\beta}_1 \times \boldsymbol{\beta}_2\|}{\|\boldsymbol{\beta}_3 \times R(\boldsymbol{\beta}_3, \boldsymbol{\beta}_2)\,\boldsymbol{\beta}_1\|}. \tag{3.12}$$

If $\boldsymbol{\beta}_{lm} = \pm\boldsymbol{\beta}_{ln}$ but $\boldsymbol{\beta}_{lm} \neq \pm\boldsymbol{\beta}_{lo}$, γ_{lmn} is still obtainable as:

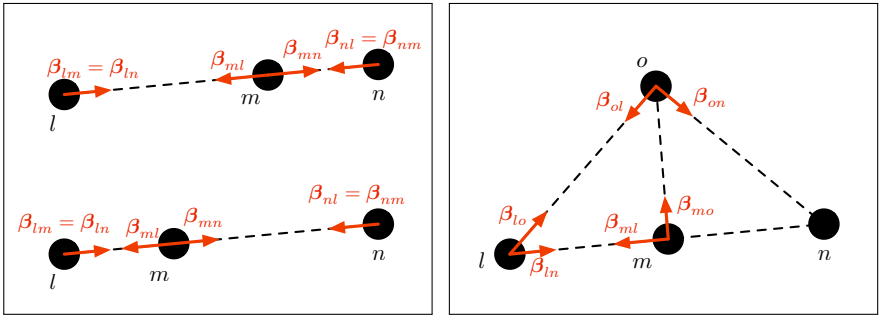
$$\gamma_{lmn} = \frac{\delta_{ln}}{\delta_{lo}}\frac{\delta_{lo}}{\delta_{lm}} = \Gamma(\boldsymbol{\beta}_{on}, \boldsymbol{\beta}_{ol}, \boldsymbol{\beta}_{ln})\Gamma(\boldsymbol{\beta}_{mo}, \boldsymbol{\beta}_{ml}, \boldsymbol{\beta}_{lo}). \tag{3.13}$$

Proof of Property 3.2. Figure 3.5 represents the plane spanned by the points $\boldsymbol{p}_l, \boldsymbol{p}_m, \boldsymbol{p}_n$. From the law of sines, it follows $\frac{\delta_{ln}}{\delta_{lm}} = \frac{\sin(\Delta_{ln})}{\sin(\Delta_{lm})}$, where $\sin(\Delta_{ln}) = \|\boldsymbol{\beta}_{mn} \times \boldsymbol{\beta}_{ml}\|$ and

$$\begin{aligned}\sin(\Delta_{lm}) &= \|\boldsymbol{\beta}_{nl} \times \boldsymbol{\beta}_{nm}\| = \|{}^l R_n \boldsymbol{\beta}_{nl} \times {}^l R_n \boldsymbol{\beta}_{nm}\| = \\ &= \| - \boldsymbol{\beta}_{ln} \times {}^l R_m {}^m R_n \boldsymbol{\beta}_{nm}\|,\end{aligned}$$

thus proving (3.10). Finally, (3.11) follows from (3.6) via suitable relabeling. □

Property 3.2 states that the inter-distance ratios (3.6) for three agents that are not aligned can always be computed from the relative bearings



(a) (b)

Figure 3.6: Property 3.2: a) When the three agents l, m, n are aligned, their relative-bearings do not allow to determine the inter-distances ratios. b). Adding a fourth agent o not aligned to l, m, n allows to disambiguate the inter-distances ratios. The example shows the relative-bearings that allow to compute γ_{lmn}.

among the three agents. However, if the three agents are aligned, then the information provided by their reciprocal relative bearings is not sufficient to retrieve the ratios between the inter-distances (see Fig. 3.6a). In this case it is necessary to triangulate with a fourth agent that is not aligned with the first three, in order to disambiguate the ratios (see Fig. 3.6b).

Composition

Property 3.3. *(Composition)* Consider 3 agents l, m, n then, if $\boldsymbol{\beta}_{lm} \neq \pm\boldsymbol{\beta}_{ln}$, the following identity holds:

$$\boldsymbol{\beta}_{mn} = {}^{m}R_{l}\frac{\boldsymbol{\beta}_{ln}\gamma_{lmn} - \boldsymbol{\beta}_{lm}}{\|\boldsymbol{\beta}_{ln}\gamma_{lmn} - \boldsymbol{\beta}_{lm}\|}. \tag{3.14}$$

Proof of Property 3.3. Identity (3.14) follows from

$$\begin{aligned}\boldsymbol{\beta}_{mn} &= \frac{{}^m R(\boldsymbol{p}_n - \boldsymbol{p}_m)}{\|\boldsymbol{p}_n - \boldsymbol{p}_m\|} \\ &= \frac{{}^m R(\boldsymbol{p}_l + R_l \boldsymbol{\beta}_{ln} \gamma_{lmn} \delta_{lm} - \boldsymbol{p}_l - R_l \boldsymbol{\beta}_{lm} \delta_{lm})}{\|\boldsymbol{p}_n - \boldsymbol{p}_m\|} \\ &= \frac{{}^m R R_l (\boldsymbol{\beta}_{ln} \gamma_{lmn} - \boldsymbol{\beta}_{lm})}{(\|\boldsymbol{p}_n - \boldsymbol{p}_m\| \delta_{lm}^{-1})},\end{aligned}$$

where the denominator $\|\boldsymbol{p}_n - \boldsymbol{p}_m\| \delta_{lm}^{-1} = \|\boldsymbol{\beta}_{ln} \gamma_{lmn} - \boldsymbol{\beta}_{lm}\|$ since $\boldsymbol{\beta}_{mn}$ is a unit vector. □

Property 3.3 provides a tool that allows to compose the relative bearings $\boldsymbol{\beta}_{ln}$, from agent l to agent n, and $\boldsymbol{\beta}_{lm}$, from agent l to agent m, to determine the bearing $\boldsymbol{\beta}_{mn}$, from agent m to agent n.

3.3.2 Bearing-Formations

The concept of formation is expressed in a clear and compact form by borrowing the formalism of graph-theory. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ indicate a generic graph with nodes $\mathcal{V} = \{1, \dots N\}$ and edges $\mathcal{E} \subseteq \mathcal{T} = \{(i, j) \in \mathcal{V} \times \mathcal{V} \,|\, i \neq j\}$. A formation is then represented as the 'graph-plus-configuration' structure obtained by associating to each node of the graph the configuration of an agent or robot. This kind of representation is also known in literature as *framework* or *point-formation*, see, e.g. Eren et al. (2003, 2006). The formalism just introduced is now applied to define two different typologies of formations.

Agent-formation A formation of agents, or just *agent-formation*, is a pair $(\mathcal{G}, \boldsymbol{q})$ consisting of a generic graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a configuration vector $\boldsymbol{q} = (\boldsymbol{q}_1, \dots, \boldsymbol{q}_N)$ that assigns to every node $i \in \mathcal{V}$ the configuration $\boldsymbol{q}_i = (\boldsymbol{p}_i, \psi_i) \in \boldsymbol{q}$ (introduced in Sec. 3.2.2 to model agents).

There are several attributes that can be associated to an agent-formation $(\mathcal{G}, \boldsymbol{q})$:

- An agent-formation $(\mathcal{G}, \boldsymbol{q})$ is called degenerate if $\boldsymbol{q}$ is degenerate in the sense of Definition 3.1.

- Two agent-formations $(\mathcal{G}, \boldsymbol{q})$ and $(\mathcal{G}, \boldsymbol{q}')$ with the same graph $\mathcal{G}$ but with different configurations $\boldsymbol{q}$ and $\boldsymbol{q}'$ are said *edge-equivalent* if

$$\boldsymbol{\beta}_{ij}(\boldsymbol{q}) = \boldsymbol{\beta}_{ij}(\boldsymbol{q}') \quad \forall (i,j) \in \mathcal{E}, \tag{3.15}$$

  and *similar* if

$$\boldsymbol{\beta}_{ij}(\boldsymbol{q}) = \boldsymbol{\beta}_{ij}(\boldsymbol{q}') \quad \forall (i,j) \in \mathcal{T}. \tag{3.16}$$

- An agent-formation $(\mathcal{G}, \boldsymbol{q})$ is called *rigid* if all its edge-equivalent agent-formations are also similar.

Bearing-formation A formation of bearings, or simply a *bearing-formation*, is a pair $(\mathcal{G}, \boldsymbol{\alpha})$ made of a generic graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a collection of $|\mathcal{E}|$ unit vectors $\boldsymbol{\alpha} = (\ldots, \boldsymbol{\alpha}_{ij}, \ldots) \in \mathbb{S}^{2|\mathcal{E}|}$ that assigns to every edge $(i,j) \in \mathcal{E}$ the unit vector[3] $\boldsymbol{\alpha}_{ij} \in \boldsymbol{\alpha}$.Analogously to agent-formations, bearing-formations can have various attributes:

- A bearing-formation $(\mathcal{G}, \boldsymbol{\alpha})$ is said *feasible* if all the unit vectors contained in $\boldsymbol{\alpha}$ can simultaneously exist as *actual relative bearings* $\boldsymbol{\beta}_{ij}(\boldsymbol{q}^\alpha)$ for some configuration $\boldsymbol{q}^\alpha$, i.e., such that $\boldsymbol{\beta}_{ij}(\boldsymbol{q}^\alpha) = \boldsymbol{\alpha}_{ij}\ \forall\, (i,j) \in \mathcal{E}$. In this case, $(\mathcal{G}, \boldsymbol{q}^\alpha)$ is called a *realization* of $(\mathcal{G}, \boldsymbol{\alpha})$. Figure 3.7 provides an example of feasible and unfeasible bearing-formations.

- A feasible bearing-formation $(\mathcal{G}, \boldsymbol{\alpha})$ is called *rigid* if it only has rigid realizations. This also implicitly defines (feasible) *non-rigid* bearing-formations.

- A feasible bearing-formation $(\mathcal{G}, \boldsymbol{\alpha})$ is called *degenerate* if it has at least a degenerate realization. This also implicitly defines (feasible) *non-degenerate* bearing-formations.

- Two rigid bearing-formations $(\mathcal{G}, \boldsymbol{\alpha})$ and $(\mathcal{G}', \boldsymbol{\alpha}')$ are said *equivalent* if (3.16) holds for *any pair* of realizations $(\mathcal{G}, \boldsymbol{q}^\alpha)$ and $(\mathcal{G}', \boldsymbol{q}^{\alpha\prime})$ of $(\mathcal{G}, \boldsymbol{\alpha})$ and $(\mathcal{G}', \boldsymbol{\alpha}')$, respectively. Therefore, two equivalent rigid bearing-formations share the same set of realizations.

Before proceeding further, it is important to stress out, despite the 'duality' between the definitions of $(\mathcal{G}, \boldsymbol{q})$ and $(\mathcal{G}, \boldsymbol{\alpha})$, the substantial difference

[3]Notice that here the term 'bearing' is used as a synonym of 'unit vector'.

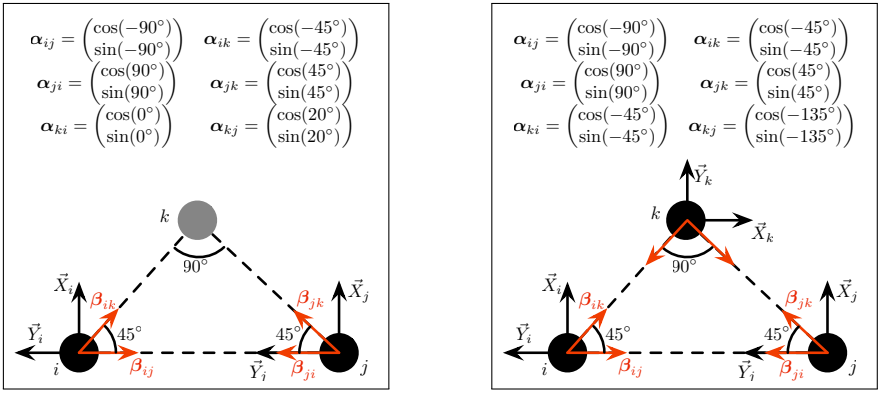


Figure 3.7: Example of planar bearing-formations. a) Unfeasible bearing-formation. b) Feasible bearing-formation.

between agent-formations and bearing-formations. A bearing-formation can be considered as a set of constraints that is expressed in the form of relative-bearings $\boldsymbol{\alpha}$. On the other hand, an agent-formation $(\mathcal{G}, \boldsymbol{q}^\alpha)$ whose relative bearings satisfy the constraints imposed by $(\mathcal{G}, \boldsymbol{\alpha})$ is only one specific realization. In fact, there can be other agent-formations with configurations that are different from $\boldsymbol{q}^\alpha$ but that still satisfy the constraints of $(\mathcal{G}, \boldsymbol{\alpha})$, or there can be no agent-formation at all that can satisfy the constraints. Therefore, the bearings $\boldsymbol{\alpha}$ of a bearing-formation $(\mathcal{G}, \boldsymbol{\alpha})$ are decoupled from the notions of agent configuration and relative bearings of an agent-formation.

To summarize and clarify some of previous definitions, a bearing-formation is rigid if it is feasible and implicitly defines all the relative bearings of its realizations, i.e., if it provides the maximum information obtainable using only relative bearings. This is a powerful concept and it will be further extended and exploited to design the formation controller. Furthermore, if a rigid bearing-formation is *degenerate* then no constraint exists among the inter-distances of its realizations, as shown by the example of Fig. 3.6a. On the other hand, in case of non-degeneracy, the inter-distance ratios are strictly constrained, as shown in the example of Fig. 3.6b.

Minimality for Rigid Bearing-Formations

By adopting the previous definitions, the goal of the formation control can be naively identified with the requirement of regulating *all* the relative bearings $\boldsymbol{\beta}_{ij}(\boldsymbol{q})$, $(i,j) \in \mathcal{T}$, where $\mathcal{T}$ is the set of all possible links among the agents, to some desired values $\boldsymbol{\beta}^d_{ij}$, i.e.

$$\boldsymbol{\beta}(\boldsymbol{q}) = (\ldots, \boldsymbol{\beta}_{ij}(\boldsymbol{q}), \ldots)_{(i,j)\in\mathcal{T}} \to \boldsymbol{\beta}^d = (\ldots, \boldsymbol{\beta}^d_{ij}, \ldots)_{(i,j)\in\mathcal{T}}.$$

This naive objective can be revised by using the concept of rigidity. Assume that there exists a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ such that the (desired) bearing-formation $(\mathcal{G}, \boldsymbol{\beta}^d_{\mathcal{E}})$, where $\boldsymbol{\beta}^d_{\mathcal{E}} = (\ldots, \boldsymbol{\beta}^d_{ij} \ldots)_{(i,j)\in\mathcal{E}}$ is the restriction of desired bearings $\boldsymbol{\beta}^d$ to $\mathcal{E}$, is *rigid*. From the property of rigidity it follows directly that $\boldsymbol{\beta}_{ij}(\boldsymbol{q}) \to \boldsymbol{\beta}^d_{ij}$, $\forall\,(i,j) \in \mathcal{E}$, implies $\boldsymbol{\beta}_{ij}(\boldsymbol{q}) \to \boldsymbol{\beta}^d_{ij}$, $\forall\,(i,j) \in \mathcal{T}\backslash\mathcal{E}$. Moreover, the desired bearing-formation $(\mathcal{G}, \boldsymbol{\beta}^d_{\mathcal{E}})$ can be replaced by any *equivalent* bearing-formation $(\mathcal{G}', \boldsymbol{\beta}^d_{\mathcal{E}'})$ for realizing the *same* formation control goal. Therefore, in order to reduce the number of measured quantities and the overall computational load of the formation controller it is desirable to use a minimal number of controlled and measured relative bearings. This point raises the issue of *minimality* for rigid bearing-formations, i.e., of the minimal cardinality of $|\mathcal{E}|$ needed to define a rigid bearing-formation.

Lemma 3.1. *(Parameterization of a bearing-formation)* The equivalence class of all the realizations of a non-degenerate and rigid bearing-formation $(\mathcal{G}, \boldsymbol{\alpha})$ is the set $\{(\mathcal{G}, \boldsymbol{q}^\alpha)\,|\boldsymbol{q}^\alpha \in \mathcal{Q}\}$ where $\mathcal{Q}$ is a 5-dimensional manifold embedded in $(\mathbb{R}^3 \times \mathbb{S}^1)^N$, and globally parameterized by the configuration $\boldsymbol{q}^\alpha_i$ of one agent (4 parameters) and the distance $\delta_{ij}(\boldsymbol{q}^\alpha)$ between two agents (1 parameter).

Proof of Lemma 3.1. Consider any realization $(\mathcal{G}, \boldsymbol{q}^\alpha)$. In order to prove the Lemma, the proof will show that, without loss of generality, if $\boldsymbol{q}^\alpha_1$, $\delta_{12}(\boldsymbol{q}^\alpha)$ and all the relative bearings $\boldsymbol{\beta}(\boldsymbol{q}^\alpha)$ are known, then all the rotation matrices $R_j(\boldsymbol{q}^\alpha)$ and positions $\boldsymbol{p}^\alpha_j$, $j \neq 1$, can be explicitly obtained. In the following the dependency from $\boldsymbol{q}^\alpha$ is omitted for conciseness.

First of all, since $R_j = R_1\,{}^1R_j$ and $\boldsymbol{p}^\alpha_j = \boldsymbol{p}^\alpha_1 + R_1\boldsymbol{\beta}_{1j}\delta_{1j}$, the problem can be reduced to the computation of ${}^1R_j$ $\forall j > 1$ and δ_{1j} $\forall j > 2$. If $\boldsymbol{\beta}_{1j} \neq \pm(0\,0\,1)^T$ then ${}^1R_j$ follows from (3.8) using the bearings $\boldsymbol{\beta}_{1j}$ and $\boldsymbol{\beta}_{j1}$.
If $\boldsymbol{\beta}_{1j} = \pm(0\,0\,1)^T$ (i.e., agents 1 and j are on on top of each other), let h be

any agent such that $\boldsymbol{\beta}_{1h} \neq \pm(0\,0\,1)^T$ (non-degeneracy guarantees existence of at least one such agent). Then, one can compute ${}^1R_h$ and ${}^hR_j$ from $\boldsymbol{\beta}_{1h}, \boldsymbol{\beta}_{h1}, \boldsymbol{\beta}_{hj}, \boldsymbol{\beta}_{jh}$ using (3.8) twice, to finally obtain ${}^1R_j = {}^1R_h\,{}^hR_j$.

As for the inter-distances, since $\boldsymbol{q}^\alpha$ is non-degenerate, there exists at least an agent h such that $\boldsymbol{\beta}_{12} \neq \pm\boldsymbol{\beta}_{1h}$. Then, applying (3.6) with $(l, m, n) = (1, 2, h)$, one can obtain $\delta_{1h} = \gamma_{12h}\delta_{12}$ and $\delta_{2h} = \gamma_{21h}\delta_{12}$ as a function of δ_{12} and the needed relative bearings. Finally, consider the case $j \neq 1, 2, h$: if $\boldsymbol{\beta}_{1j} \neq \pm\boldsymbol{\beta}_{12}$ then $\delta_{1j} = \gamma_{12j}\delta_{12}$, otherwise $\delta_{1j} = \gamma_{1hj}\delta_{1h}$, thus concluding the proof. □

Although $\mathcal{E}$ can contain up to $|\mathcal{T}| = N(N-1) = O(N^2)$ pairs, Lemma 3.1 shows that any rigid non-degenerate bearing-formation defines a 5-dimensional manifold in a $4N$-dimensional space. Therefore, only $4N - 5$ independent constraints are actually needed to determine a bearing-formation. This fact motivates the following definition:

Definition 3.3 (Minimally-linear rigid bearing-formation)**.** A non degenerate and rigid bearing-formation $(\mathcal{G}, \boldsymbol{\alpha})$ is called *minimally-linear rigid* if any $(\mathcal{G}', \boldsymbol{\alpha}')$ with $\mathcal{E}' \subsetneq \mathcal{E}$, $\boldsymbol{\alpha}' \subsetneq \boldsymbol{\alpha}$ is not rigid, and $|\mathcal{E}| = O(N)$.

The following Lemma shows how to explicitly construct an important class of minimally-linear rigid bearing-formations which will then be used by the formation controller.

Lemma 3.2 (Construction of a minimally-linear rigid bearing formation)**.** Given a non-degenerate rigid bearing-formation $(\mathcal{G}, \boldsymbol{\alpha})$, consider any corresponding realization $(\mathcal{G}, \boldsymbol{q}^\alpha)$. Then there exists at least one agent relabeling such that $\boldsymbol{\beta}_{12}(\boldsymbol{q}^\alpha) \neq \pm\boldsymbol{\beta}_{13}(\boldsymbol{q}^\alpha) \neq \pm(0\,0\,1)^T$. Consider any one of these relabelings and define the following sets $\mathcal{I}_2, \mathcal{I}_3 \subset \mathcal{V}$ and $\hat{\mathcal{E}} \subsetneq \mathcal{T}$:

$$\mathcal{I}_2 = \{3\} \cup \{j \in \{4, \dots, N\} \mid \boldsymbol{\beta}_{1j}(\boldsymbol{q}^\alpha) \neq \pm\boldsymbol{\beta}_{12}(\boldsymbol{q}^\alpha)\} \tag{3.17}$$

$$\mathcal{I}_3 = \{4, \dots, N\} \backslash \mathcal{I}_2. \tag{3.18}$$

$$\hat{\mathcal{E}} = \{(1, j), (j, 1)\}_{j=2}^{N} \cup \{(j, 2)\}_{j \in \mathcal{I}_2} \cup \{(j, 3)\}_{j \in \mathcal{I}_3}. \tag{3.19}$$

Define also $\hat{\mathcal{G}} = (\mathcal{V}, \hat{\mathcal{E}})$ and $\boldsymbol{\beta}^{\alpha}_{\hat{\mathcal{E}}} = (\ldots, \boldsymbol{\beta}_{ij}(\boldsymbol{q}^{\alpha}), \ldots)_{(i,j)\in\hat{\mathcal{E}}}$. Then the bearing-formation $(\hat{\mathcal{G}}, \boldsymbol{\beta}^{\alpha}_{\hat{\mathcal{E}}})$ is non-degenerate, minimally-linear rigid, and equivalent to $(\mathcal{G}, \boldsymbol{\alpha})$.

Proof of Lemma 3.2. It is easy to check that non-degeneracy implies the existence of at least one relabeling meeting the requirement $\boldsymbol{\beta}_{12}(\boldsymbol{q}^{\alpha}) \neq \pm\boldsymbol{\beta}_{13}(\boldsymbol{q}^{\alpha}) \neq \pm(0\,0\,1)^T$. Furthermore, the equivalence follows from the fact that $(\hat{\mathcal{G}}, \boldsymbol{\beta}^{\alpha}_{\hat{\mathcal{E}}})$ has been generated from a realization of $(\mathcal{G}, \boldsymbol{\alpha})$. The linearity can be checked computing the cardinality of $\hat{\mathcal{E}}$ that is $|\hat{\mathcal{E}}| = 2(N-1) + |\mathcal{I}_2| + |\mathcal{I}_3| = 3N - 4$, which is linear in the number of agents N.

In order to demonstrate rigidity, the proof shows that all the bearings $\boldsymbol{\beta}_{ij}(\boldsymbol{q}^{\alpha})$ with $(i,j) \notin \hat{\mathcal{E}}$ can be uniquely computed from the bearings in $\boldsymbol{\beta}^{\alpha}_{\hat{\mathcal{E}}}$. Hereinafter the dependency from $\boldsymbol{q}^{\alpha}$ is omitted for brevity.

All the bearings $\boldsymbol{\beta}_{2j}$, $\forall j \in \mathcal{I}_2$, can be computed by evaluating γ_{1j2} from (3.6) with $(l,m,n) = (1,j,2)$, and then applying (3.14) with $(l,m,n) = (1,2,j)$ and noting that $\gamma_{12j} = \gamma_{1j2}^{-1}$. With the same approach it is also possible to determine $\boldsymbol{\beta}_{3j}$ $\forall j \in \mathcal{I}_3$.

The rotation matrixes of the form ${}^{j}R_1$, $\forall j > 1$, can be computed as follows: if $\boldsymbol{\beta}_{1j} \neq \pm(0\,0\,1)$ then ${}^{j}R_1$ is determined using (3.8) with $(i,j) = (1,j)$. Otherwise, if $j \in \mathcal{I}_2$, then ${}^{j}R_1$ can be evaluated as ${}^{j}R_2 {}^{2}R_1$ using (3.8) twice, first with $(i,j) = (j,2)$ and then with $(i,j) = (2,1)$.

Finally, if $j \in \mathcal{I}_3$, it is possible can determine ${}^{j}R_1 = {}^{j}R_3 {}^{3}R_1$ in a similar fashion. This further allows to obtain any ${}^{j}R_i$, $\forall j \neq i$, since ${}^{j}R_i = {}^{j}R_1 {}^{i}R_1^T$. Consequently, for any known relative bearing $\boldsymbol{\beta}_{ij}$, one also gets $\boldsymbol{\beta}_{ji} = -{}^{j}R_i \boldsymbol{\beta}_{ij}$.

To conclude the proof of rigidity, it is necessary to show how to compute $\boldsymbol{\beta}_{ij}$ for any $i > 3, j > i$. To this end it is sufficient to apply (3.14) using $(l,m,n) = (1,i,j)$, where γ_{1ji} can be obtained as $\gamma_{12i}/\gamma_{12j}$ if $j \in \mathcal{I}_2$, or as $\gamma_{13i}/\gamma_{13j}$ if $j \in \mathcal{I}_3$.

Finally, in order to demonstrate minimality, the proof shows that rigidity is lost if any relative bearing is removed from $\hat{\mathcal{E}}$. First, if any bearing of the form $\boldsymbol{\beta}_{1i}$ or $\boldsymbol{\beta}_{i1}$, $i > 1$, is removed, then ${}^{i}R_1$ can be any rotation matrix of the form $R_z(\cdot)$, thus contradicting Lemma 3.1. Second, if any bearing of the form $\boldsymbol{\beta}_{ji}$, with $i = 2, 3$, $j \in \mathcal{I}_i$, is removed, then the ratio γ_{1ij} can take any value, i.e., δ_{1i} and δ_{1j} can be chosen freely, thus leading again to a contradiction with Lemma 3.1. Therefore no relative bearing can be removed from $\hat{\mathcal{E}}$ without losing rigidity, hence proving the last part of the statement. □

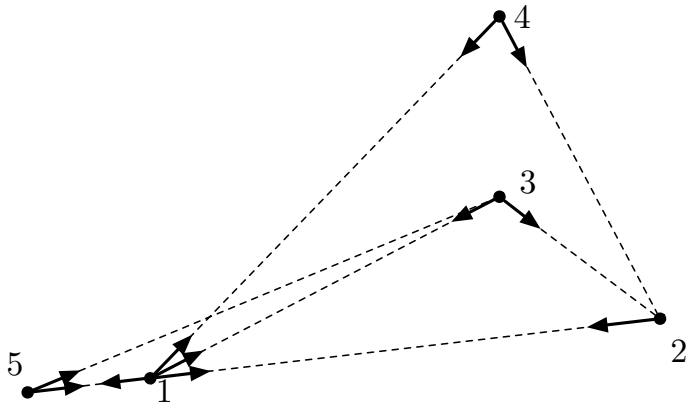


Figure 3.8: Construction of the minimally-linear rigid bearing-formation described in Lemma 3.2 for the case of 5 agents. Agents 1, 2, and 5 are aligned, therefore $\mathcal{I}_2 = \{3, 4\}$, $\mathcal{I}_3 = \{5\}$, and, as a consequence, $\hat{\mathcal{E}} = \{(1,2),(1,3),(1,4),(1,5),(2,1),(3,1),(4,1),(5,1)\} \cup \{(3,2),(4,2)\} \cup \{(5,3)\}$. Note that $|\hat{\mathcal{E}}| = 3 \cdot 5 - 4 = 11$ instead of $|\mathcal{T}| = (5-1) \cdot 5 = 20$ (the number of all possible relative bearings).

As an example, Fig. 3.8 illustrates the construction of a minimally-linear rigid set in the case of 5 agents where agents 1, 2 and 5 are aligned.

3.4 Overview of the Framework

Following the notation introduced in Sec. 3.3.2, let $(\mathcal{G}, \boldsymbol{\beta}^d_{\mathcal{E}})$ denote the *desired* rigid and non-degenerate bearing-formation that specifies the aim of the formation-control. Let also $(\mathcal{G}, \boldsymbol{q})$ indicate the agent-formation consisting of the same graph $\mathcal{G}$ and of the *current* agent configuration $\boldsymbol{q}$. Finally, indicate with $\boldsymbol{q}(t_0)$ the generic initial configuration of the agents. The goal of the shared control of the bearing-formation is:

1. to automatically bring and maintain $(\mathcal{G}, \boldsymbol{q}(t))$ within the class of realizations of $(\mathcal{G}, \boldsymbol{\beta}^d_{\mathcal{E}})$ (the control objective);

2. to allow the human to steer $(\mathcal{G}, \boldsymbol{q}(t))$ *inside* the class of realizations of $(\mathcal{G}, \boldsymbol{\beta}^d_{\mathcal{E}})$;

3. to generate a force feedback that is informative of the mismatch between the command from the human and the motion of the robots.

In order to achieve points 1 and 2, the control input of the i-th agent (3.2) is split into two terms

$$(\boldsymbol{u}_i, w_i) = (\boldsymbol{u}_i^h, w_i^h) + (\boldsymbol{u}_i^f, w_i^f), \tag{3.20}$$

where the term $(\boldsymbol{u}_i^h, w_i^h)$ represents the action of the human[4] in charge of steering the collective motion of the UAV group, and the term $(\boldsymbol{u}_i^f, w_i^f)$ enforces convergence to the desired bearing formation $(\mathcal{G}, \boldsymbol{\beta}_{\mathcal{E}}^d)$.

The framework, depicted in Fig. 3.9, is organized according to the following structure:

Human Steering: it lets the human operator steer the whole formation by reading the configuration of actuated multi-DoF input devices and by accordingly generating the signals $(\boldsymbol{u}_i^h, w_i^h)$ for the i-th agent, with $i = 1, \ldots, N$. The control inputs $(\boldsymbol{u}_i^h, w_i^h)$ are computed in the 'orthogonal' set of *virtual inputs* able to steer $(\mathcal{G}, \boldsymbol{q}(t))$ *inside* the class of realizations of $(\mathcal{G}, \boldsymbol{\beta}_{\mathcal{E}}^d)$.

Formation Control: it generates the control signals $(\boldsymbol{u}_i^f, w_i^f)$ by using *only* a minimally linear number of relative-bearing measurements.

Haptic Feedback Algorithm: it closes the interaction-loop between the human operator and the autonomous formation controller by rendering on the actuated input devices an appropriate torque $\boldsymbol{\tau}$ that is helpful to increase the user's SA.

3.5 Human Steering

As previously stated in Sec. 3.4, the Human Steering has to be designed so that the signals $(\boldsymbol{u}_i^h, w_i^h)$ steer $(\mathcal{G}, \boldsymbol{q}(t))$ *inside* the class of realizations of $(\mathcal{G}, \boldsymbol{\beta}_{\mathcal{E}}^d)$. To this aim, consider instead of $(\mathcal{G}, \boldsymbol{\beta}(\boldsymbol{q}))$ the bearing-formation $(\mathcal{K}, \boldsymbol{\beta}(\boldsymbol{q}))$, where $\mathcal{K} = (\mathcal{V}, \mathcal{T})$ is the complete graph. This choice is motivated by two considerations:

[4] The paradigm can be generalised by substituting the human with an 'external high-level planner'.

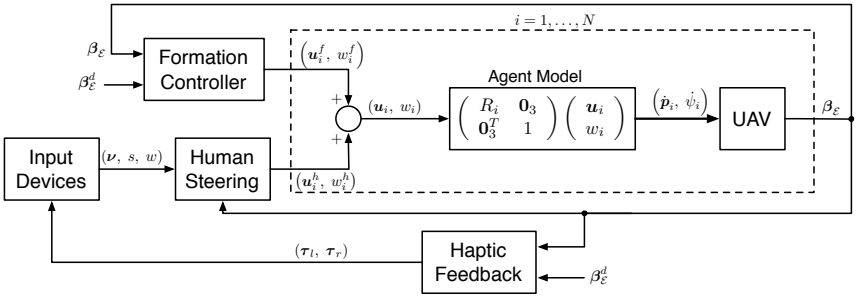


Figure 3.9: Overview of the framework.

i) using $\mathcal{K}$ instead of a generic $\mathcal{G}$ automatically guarantees rigidity of $(\mathcal{K}, \boldsymbol{\beta}(\boldsymbol{q}))$ for any configuration $\boldsymbol{q}$;

ii) the class of realizations of $(\mathcal{K}, \boldsymbol{\beta}(\boldsymbol{q}))$, denoted with $\mathcal{R}_{\mathcal{K}}(\boldsymbol{q})$, was proven in Lemma 3.1 to be a 5-dimensional manifold. The class $\mathcal{R}_{\mathcal{K}}(\boldsymbol{q})$ is the same class of realizations obtained by any other *rigid* bearing formation $(\mathcal{G}, \boldsymbol{\beta}_{\mathcal{E}}(\boldsymbol{q}))$, with $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and $\boldsymbol{\beta}_{\mathcal{E}}(\boldsymbol{q}) = (\ldots, \boldsymbol{\beta}_{ij}(\boldsymbol{q}), \ldots)_{(i,j)\in\mathcal{E}}$.

With this setting, denote as $\dot{\boldsymbol{q}} = (\dot{\boldsymbol{p}}_1^T\ \dot{\psi}_1\ \ldots\ \dot{\boldsymbol{p}}_N^T\ \dot{\psi}_N)^T \in \mathbb{R}^{4N}$ the vector of *generalized velocities* of the agents. By plugging (3.20) into (3.2) it is clear that $\dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}^f + \dot{\boldsymbol{q}}^h$, where $\dot{\boldsymbol{q}}^f$ and $\dot{\boldsymbol{q}}^h$ depend only on $(\boldsymbol{u}^f_i, w^f_i)$ and $(\boldsymbol{u}^h_i, w^h_i)$, respectively, with $i = 1 \ldots N$. Therefore the goal of the Human Steering is reformulated as follows:

Objective 3.1. Design $(\boldsymbol{u}^h_i, w^h_i)$ such that $\dot{\boldsymbol{q}}^h$ belongs to $T_{\boldsymbol{q}}\mathcal{R}_{\mathcal{K}}(\boldsymbol{q})$, the tangent space at $\boldsymbol{q}$ of $\mathcal{R}_{\mathcal{K}}(\boldsymbol{q})$.

The motivation of Objective 3.1 is quite intuitive. If the human operator were free to command any motion to the UAVs, than his/her commands could interfere with the formation control objective. Rather than letting the Formation Controller cope with any unpredictable human command, the solution sought here is to rather design the human commands so that they preserve the current relative bearings $\boldsymbol{\beta}(\boldsymbol{q})$ and thus will not alter the bearing formation.

To achieve Objective 3.1 it is necessary to have an analytical expression for $T_{\boldsymbol{q}}\mathcal{R}_{\mathcal{K}}(\boldsymbol{q})$, for which it will be used the following notation: i) $\hat{\boldsymbol{p}}_{ij} = \frac{\boldsymbol{p}_{ij}}{\delta_{ij}} \in \mathbb{S}^2$, $\forall i \neq j$ indicates the relative position of two agents, with the convention

that $\hat{\boldsymbol{p}}_{ii} = \mathbf{0}_3$, $\forall i = 1, \dots, N$; ii) $\mathfrak{R}(A)$ denotes the range (or column) space of a matrix A; iii) $S = [e_3]_\wedge$ is the skew-symmetric matrix associated to $e_3 = (0\,0\,1)^T$; iv) I_3 is the 3×3 identity matrix and $\mathbf{0}_3 = (0\,0\,0)^T$. As a first step towards the formalisation of $T_{\boldsymbol{q}}\mathcal{R}_\mathcal{K}(\boldsymbol{q})$, the following lemma provides a description of the tangent space at $\boldsymbol{q}$ of $\mathcal{R}_\mathcal{K}(\boldsymbol{q})$ in the case of three agents ($N = 3$).

Lemma 3.3 (Bearing-invariant motions for 3 agents)**.** If $N = 3$ and $\boldsymbol{q}$ is non-degenerate, then $T_{\boldsymbol{q}}\mathcal{R}_\mathcal{K}(\boldsymbol{q}) = \mathfrak{R}(T_3)$, being

$$T_3 = \begin{bmatrix} I_3 & \mathbf{0}_3 & I_3 & \mathbf{0}_3 & I_3 & \mathbf{0}_3 \\ \mathbf{0}_3^T & 0 & \hat{\boldsymbol{p}}_{12}^T & 0 & \gamma_{123}\hat{\boldsymbol{p}}_{13}^T & 0 \\ \mathbf{0}_3^T & 1 & -(S\boldsymbol{p}_{12})^T & 1 & -(S\boldsymbol{p}_{13})^T & 1 \end{bmatrix}^T \in \mathbb{R}^{12\times 5}. \quad (3.21)$$

Proof of Lemma 3.3. The time derivative of a bearing $\dot{\boldsymbol{\beta}}_{ij}$ is obtained by differentiating (3.3) as

$$\dot{\boldsymbol{\beta}}_{ij} = \frac{1}{\delta_{ij}}{}^i R \left[\dot{\psi}_i S\boldsymbol{p}_{ij} + P(\hat{\boldsymbol{p}}_{ij})\dot{\boldsymbol{p}}_{ij}\right], \quad (3.22)$$

where $P(\hat{\boldsymbol{p}}_{ij}) = (I - \hat{\boldsymbol{p}}_{ij}\hat{\boldsymbol{p}}_{ij}^T) \in \mathbb{R}^{3\times 3}$ is the projection matrix onto the plane perpendicular to $\hat{\boldsymbol{p}}_{ij}$. Imposing $\dot{\boldsymbol{\beta}}_{ij} = 0$ in (3.22) results in the following condition

$$0 = \dot{\psi}_i S\boldsymbol{p}_{ij} + P(\hat{\boldsymbol{p}}_{ij})\dot{\boldsymbol{p}}_{ij}, \quad (3.23)$$

which, combined with the symmetric condition $\dot{\boldsymbol{\beta}}_{ji} = 0$, yields

$$\left(\dot{\psi}_j - \dot{\psi}_i\right) S\boldsymbol{p}_{ij} = \dot{\psi}_{ij} S\boldsymbol{p}_{ij} = 0, \quad (3.24)$$

where it was exploited the fact that $\boldsymbol{p}_{ij} = -\boldsymbol{p}_{ji}$, $\dot{\boldsymbol{p}}_{ij} = -\dot{\boldsymbol{p}}_{ji}$, $\hat{\boldsymbol{p}}_{ij} = -\hat{\boldsymbol{p}}_{ji}$, and $P(\hat{\boldsymbol{p}}_{ji}) = P(\hat{\boldsymbol{p}}_{ij})$.
In the particular case of $N = 3$ agents considered here, the constraints (3.24) for all possible agent pairs are simply

$$\dot{\psi}_{12} S\boldsymbol{p}_{12} = 0, \quad \dot{\psi}_{13} S\boldsymbol{p}_{13} = 0, \quad \dot{\psi}_{23} S\boldsymbol{p}_{23} = 0. \quad (3.25)$$

Being $\ker S = \operatorname{span}\{(0\,0\,1)^T\}$, these can be satisfied when either $\boldsymbol{p}_{ij} \propto (0\,0\,1)^T$ or $\dot{\psi}_{ij} = 0$. However, since the positions $\boldsymbol{p}_1, \boldsymbol{p}_2$, and $\boldsymbol{p}_3$ are not aligned, there exist at least two vectors among $(\boldsymbol{p}_{12}, \boldsymbol{p}_{13}, \boldsymbol{p}_{23})$ not in $\ker S$. Therefore, it is possible to conclude that at least two entries among

($\dot{\psi}_{13}$, $\dot{\psi}_{12}$, $\dot{\psi}_{23}$) must be zero. Being $\dot{\psi}_{13} = \dot{\psi}_{12} + \dot{\psi}_{23}$ by construction, it necessarily follows that $\dot{\psi}_{13} = \dot{\psi}_{12} = \dot{\psi}_{23} = 0$.

Consider now the (globally invertible) change of coordinates

$$\Pi \boldsymbol{q} = \boldsymbol{q}' = (\boldsymbol{p}_1\, \boldsymbol{p}_{12}\, \boldsymbol{p}_{13}\, \psi_1\, \psi_{12}\, \psi_{13})^T$$

and the associated new generalized velocities

$$\Pi \dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}' = (\dot{\boldsymbol{p}}_1\, \dot{\boldsymbol{p}}_{12}\, \dot{\boldsymbol{p}}_{13}\, \dot{\psi}_1\, \dot{\psi}_{12}\, \dot{\psi}_{13})^T,$$

with $\Pi \in \mathbb{R}^{12\times 12}$, the determinant $\det \Pi \neq 0$, being the nonsingular transformation matrix associated to this change of coordinates. Constraints (3.23) can be rewritten in a matrix form in terms of $\boldsymbol{q}'$ and $\dot{\boldsymbol{q}}'$ as:

$$\underbrace{\begin{bmatrix} \mathbf{0}_{3\times 3} & P(\hat{\boldsymbol{p}}_{12}) & \mathbf{0}_{3\times 3} & S\boldsymbol{p}_{12} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & P(\hat{\boldsymbol{p}}_{13}) & S\boldsymbol{p}_{13} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_{3\times 3} & -P(\hat{\boldsymbol{p}}_{23}) & P(\hat{\boldsymbol{p}}_{23}) & S\boldsymbol{p}_{23} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3^T & \mathbf{0}_3^T & \mathbf{0}_3^T & \mathbf{0}_3^T & 1 & 0 \\ \mathbf{0}_3^T & \mathbf{0}_3^T & \mathbf{0}_3^T & \mathbf{0}_3^T & 0 & 1 \end{bmatrix}}_{A(\boldsymbol{q}')\in\mathbb{R}^{11\times 12}} \dot{\boldsymbol{q}}' = A(\boldsymbol{q}')\dot{\boldsymbol{q}}' = 0,$$

where $\mathbf{0}_{3\times 3}$ is the 3×3 null matrix, and the facts that $\dot{\boldsymbol{p}}_{23} = \dot{\boldsymbol{p}}_{13} - \dot{\boldsymbol{p}}_{12}$ and $\dot{\psi}_{12} = \dot{\psi}_2 - \dot{\psi}_1 = 0$ were exploited.

Then, one can identify $T_{\boldsymbol{q}'}\mathcal{R}_{\mathcal{K}}(\boldsymbol{q})$ with $\ker A(\boldsymbol{q}')$. Note that, although $\max(\operatorname{rank}(A(\boldsymbol{q}'))) = 11$, implying that $\min(\dim \ker A(\boldsymbol{q}')) = 1$, it is already known that $\dim \ker A(\boldsymbol{q}') = 5$ (resp. $\operatorname{rank}(A(\boldsymbol{q}')) = 7$) by construction. In fact, being $\mathcal{R}_{\mathcal{K}}(\boldsymbol{q})$ a regular manifold of dimension 5 (Lemma 3.1), the dimension of its tangent space at any point, i.e., of $\ker(A(\boldsymbol{q}'))$, must be necessarily 5. An explicit expression for $T_{\boldsymbol{q}'}\mathcal{R}_{\mathcal{K}}(\boldsymbol{q})$ can then be found by inspection as $T_{\boldsymbol{q}'}\mathcal{R}_{\mathcal{K}}(\boldsymbol{q}) = \ker A(\boldsymbol{q}') = \mathfrak{R}(T_3')$ where

$$T_3' = \begin{bmatrix} I_3 & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3^T & \hat{\boldsymbol{p}}_{12}^T & \gamma_{123}\hat{\boldsymbol{p}}_{13}^T & 0 & 0 & 0 \\ \mathbf{0}_3^T & -(S\boldsymbol{p}_{12})^T & -(S\boldsymbol{p}_{13})^T & 1 & 0 & 0 \end{bmatrix}^T \in \mathbb{R}^{12\times 5}.$$

In fact, letting $\boldsymbol{t}_i' \in \mathbb{R}^{12}$ be the i-th column of T_3', one can easily check that $(\boldsymbol{t}_1'^T, \boldsymbol{t}_2'^T, \boldsymbol{t}_3'^T)^T$ are in $\ker A(\boldsymbol{q}')$. Furthermore, $\boldsymbol{t}_4' \in \ker A(\boldsymbol{q}')$ since $P(\hat{\boldsymbol{p}}_{12})\hat{\boldsymbol{p}}_{12} = \mathbf{0}_3$ and $P(\hat{\boldsymbol{p}}_{13})\hat{\boldsymbol{p}}_{13} = \mathbf{0}_3$ by construction, and $-P(\hat{\boldsymbol{p}}_{23})\hat{\boldsymbol{p}}_{12} +$

$\gamma_{123}P(\hat{\boldsymbol{p}}_{23})\hat{\boldsymbol{p}}_{13}$ can be rewritten as

$$\begin{aligned}
&\frac{1}{\delta_{12}}(-P(\hat{\boldsymbol{p}}_{23})\boldsymbol{p}_{12} + P(\hat{\boldsymbol{p}}_{23})\boldsymbol{p}_{13}) = \\
&\frac{1}{\delta_{12}}(-P(\hat{\boldsymbol{p}}_{23})(\boldsymbol{p}_{13} - \boldsymbol{p}_{23}) + P(\hat{\boldsymbol{p}}_{23})\boldsymbol{p}_{13}) = \\
&\frac{1}{\delta_{12}}(-P(\hat{\boldsymbol{p}}_{23})\boldsymbol{p}_{13} + P(\hat{\boldsymbol{p}}_{23})\boldsymbol{p}_{13}) = \mathbf{0}_3.
\end{aligned}$$

Finally, $\boldsymbol{t}'_5 \in \ker A(\boldsymbol{q}')$ since $-P(\hat{\boldsymbol{p}}_{12})S\boldsymbol{p}_{12} + S\boldsymbol{p}_{12} = -S\boldsymbol{p}_{12} + S\boldsymbol{p}_{12} = \mathbf{0}_3$ and, similarly, $-P(\hat{\boldsymbol{p}}_{13})S\boldsymbol{p}_{13} + S\boldsymbol{p}_{13} = -S\boldsymbol{p}_{13} + S\boldsymbol{p}_{13} = \mathbf{0}_3$, implying that $P(\hat{\boldsymbol{p}}_{23})S\boldsymbol{p}_{12} - P(\hat{\boldsymbol{p}}_{23})S\boldsymbol{p}_{13} + S\boldsymbol{p}_{23} = P(\hat{\boldsymbol{p}}_{23})S\boldsymbol{p}_{13} - P(\hat{\boldsymbol{p}}_{23})S\boldsymbol{p}_{23} - P(\hat{\boldsymbol{p}}_{23})S\boldsymbol{p}_{13} + S\boldsymbol{p}_{23} = -S\boldsymbol{p}_{23} + S\boldsymbol{p}_{23} = \mathbf{0}_3$.

Note that $\operatorname{rank}(T'_3) = 5$ as expected. This can be easily verified by noting the 'upper triangular' form of T'_3. As a last step, it is possible to recover the sought T_3 in (3.21) by going back to the original coordinates $\boldsymbol{q}$, i.e., by taking $T_3 = \Pi^{-1}T'_3$. The resulting matrix is explicitly shown in (3.21). □

Before generalizing the result of Lemma 3.3 formations with $N \geq 3$ agents, it is worth analyzing with the help of Fig. 3.10 the geometrical meaning of the columns of T_3 and get an intuitive understanding of the actual agent motion they represent. To this end, let $\boldsymbol{t}_i \in \mathbb{R}^{12}$ be the i-th column of T_3.
A motion along $(\boldsymbol{t}_1\ \boldsymbol{t}_2\ \boldsymbol{t}_3)$, i.e.,

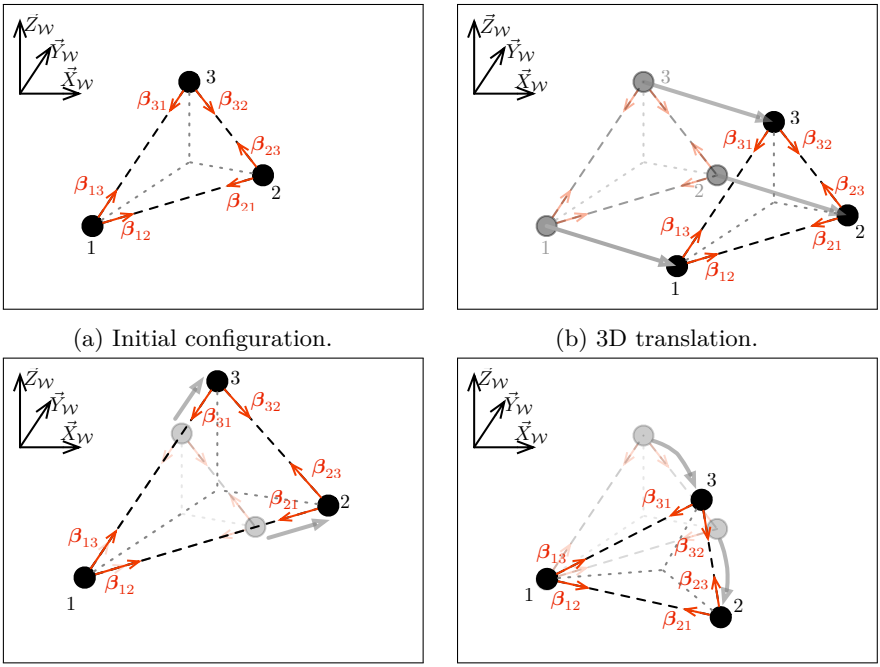
$$\dot{\boldsymbol{q}} = [\boldsymbol{t}_1\ \boldsymbol{t}_2\ \boldsymbol{t}_3]\boldsymbol{\nu} = [I_3 \quad \mathbf{0}_3 \quad I_3 \quad \mathbf{0}_3 \quad I_3 \quad \mathbf{0}_3]^T\boldsymbol{\nu}$$

for some $\boldsymbol{\nu} \in \mathbb{R}^3$ represents a *synchronized translation* of the formation with velocity $\boldsymbol{\nu}$ (see Fig. 3.10b).

A motion along $\boldsymbol{t}_4$, i.e.,

$$\dot{\boldsymbol{q}} = \boldsymbol{t}_4 s = [\mathbf{0}_3^T \quad 0 \quad \hat{\boldsymbol{p}}_{12}^T \quad 0 \quad \gamma_{123}\hat{\boldsymbol{p}}_{13}^T \quad 0]^T s$$

for some $s \in \mathbb{R}$ represents a *synchronized expansion* of the formation with expansion rate s. In particular, agent 1 does not move, agent 2 moves along the connecting line with 1 at speed s, and agent 3 moves along the connecting line with 1 at speed γ_{123}s, that is, exactly the speed needed to keep the relative bearings unchanged (see Fig. 3.10c).



(a) Initial configuration.

(b) 3D translation.

(c) Expansion.

(d) Synchronized rotation.

Figure 3.10: Bearing invariant motions with a formation of three agents.

Finally, a motion along $\boldsymbol{t}_5$, i.e.,

$$\dot{\boldsymbol{q}} = \boldsymbol{t}_5 w = [\mathbf{0}_3^T \quad 1 \quad -(S\boldsymbol{p}_{12})^T \quad 1 \quad -(S\boldsymbol{p}_{13})^T \quad 1]^T w$$

for some $w \in \mathbb{R}$ represents a *synchronized rotation* around agent 1 with rate w. Agent 1 rotates in place with an angular speed $\dot{\psi}_1 = w$, agent 2 rotates with the same angular speed $\dot{\psi}_2 = w$ but is also translating around agent 1 with linear velocity $-(S\boldsymbol{p}_{12})w$, and likewise for agent 3 (see Fig. 3.10d). Note also that synchronized translations and expansions can be performed with the sole knowledge of relative bearings, while execution of a synchronized rotation necessarily requires an additional metric information (the magnitudes of $\boldsymbol{p}_{12}$ and $\boldsymbol{p}_{13}$).

The result of Lemma 3.3 can now be generalized to a group of N agents.

Propositon 3.1 (Bearing-invariant motions for N agents)**.** Given a non-degenerate configuration $\boldsymbol{q}$, consider the sets $\mathcal{I}_2(\boldsymbol{q})$ and $\mathcal{I}_3(\boldsymbol{q})$ obtained by applying Lemma 3.2 to $(\mathcal{K}, \boldsymbol{\beta}(\boldsymbol{q}))$. Then, the set of motions keeping constant the relative bearings at $\boldsymbol{q}$ can always be expressed, with an agent relabeling if necessary, as $T_{\boldsymbol{q}}\mathcal{R}_{\mathcal{K}}(\boldsymbol{q}) = \mathfrak{R}(T_N)$, where $T_N = \begin{bmatrix} T_3^T & T_4^T & T_5^T \end{bmatrix}^T \in \mathbb{R}^{4N\times 5}$, T_3 is defined in (3.21),

$$T_4 = \begin{bmatrix} \cdots & I_3 & \mathbf{0}_3 & \cdots \\ \cdots & \gamma_{12i}\hat{\boldsymbol{p}}_{1i}^T & 0 & \cdots \\ \cdots & -(S\boldsymbol{p}_{1i})^T & 1 & \cdots \end{bmatrix}^T_{i\in\mathcal{I}_2(\boldsymbol{q})\setminus\{3\}} \in \mathbb{R}^{4(|\mathcal{I}_2(\boldsymbol{q})|-1)\times 5} \tag{3.26}$$

$$T_5 = \begin{bmatrix} \cdots & I_3 & \mathbf{0}_3 & \cdots \\ \cdots & \gamma_{13i}\hat{\boldsymbol{p}}_{1i}^T & 0 & \cdots \\ \cdots & -(S\boldsymbol{p}_{1i})^T & 1 & \cdots \end{bmatrix}^T_{i\in\mathcal{I}_3(\boldsymbol{q})} \in \mathbb{R}^{4|\mathcal{I}_3(\boldsymbol{q})|\times 5}. \tag{3.27}$$

Proof of Proposition 3.1. Applying Lemma 3.2 to $(\mathcal{K}, \boldsymbol{\beta}(\boldsymbol{q}))$ yields the equivalent rigid and non-degenerate bearing-formation $(\hat{\mathcal{G}}, \boldsymbol{\beta}_{\hat{\mathcal{E}}})$, where $\boldsymbol{\beta}_{\hat{\mathcal{E}}} = (\ldots,\ \boldsymbol{\beta}_{ij}(\boldsymbol{q}),\ \ldots)_{(i,j)\in\hat{\mathcal{E}}}$. Then the sought tangent space can be defined as the set of those velocities $\dot{\boldsymbol{q}}$ keeping all the bearings in $\boldsymbol{\beta}_{\hat{\mathcal{E}}}$ constant. Consider the sets composed by the relative bearings $\hat{\boldsymbol{\beta}}_i(\boldsymbol{q}) = \{\boldsymbol{\beta}_{1j}, \boldsymbol{\beta}_{j1}, \boldsymbol{\beta}_{i1}, \boldsymbol{\beta}_{1i}, \boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}\}$ where $j = 2$, if $i \in \mathcal{I}_2(\boldsymbol{q})$ and $j = 3$, if $i \in \mathcal{I}_3(\boldsymbol{q})$. It is clear that $\boldsymbol{\beta}_{\hat{\mathcal{E}}} \subset \cup_{i=3}^N \hat{\boldsymbol{\beta}}_i(\boldsymbol{q})$. Apply now Lemma 3.3 replacing agents

2 with j and 3 with a generic agent $i \in \{3 \dots N\}$, and consider the corresponding matrix T_i (3.21) defined as

$$T_i = \begin{bmatrix} I_3 & \mathbf{0}_3 & I_3 & \mathbf{0}_3 & I_3 & \mathbf{0}_3 \\ \mathbf{0}_3^T & 0 & \hat{\boldsymbol{p}}_{1j}^T & 0 & \gamma_{1ji}\hat{\boldsymbol{p}}_{1i}^T & 0 \\ \mathbf{0}_3^T & 1 & -(S\boldsymbol{p}_{1j})^T & 1 & -(S\boldsymbol{p}_{1i})^T & 1 \end{bmatrix}^T \in \mathbb{R}^{12\times 5}. \quad (3.28)$$

This matrix defines the motions of agents $1, j$, and i which keep all the relative bearings in $\hat{\boldsymbol{\beta}}_i(\boldsymbol{q})$ constant. Note that the motions of agents 1 and j are independent from the state of agent i. Therefore, in order to keep the bearings of all the subsets $\hat{\boldsymbol{\beta}}_i(\boldsymbol{q})$ constant, one can: i) choose the motion of agent 1 and 2, then ii) select the motion of every agent $i \in \mathcal{I}_2(\boldsymbol{q})$ as per the last four rows of (3.28), with $j = 2$, and finally iii) select the motion of every other agent $i \in \mathcal{I}_3(\boldsymbol{q})$ as per the last four rows of (3.28), with $j = 3$. This procedure directly yields matrix T_N and thus proves the Proposition. □

Similarly to the previous case of 3 agents, the bearing-invariant motions represented by T_N are of three kinds: a synchronized translation with velocity $\boldsymbol{\nu} \in \mathbb{R}^3$, a synchronized expansion with rate $s \in \mathbb{R}$, and a synchronized rotation with speed $w \in \mathbb{R}$.

Having provided an analytical expression of $T_{\boldsymbol{q}}\mathcal{R}_{\mathcal{K}}(\boldsymbol{q})$, it is now possible to design the control terms $(\boldsymbol{u}_i^h, w_i^h)$ so that they impose the aforementioned bearing-invariant motions to the group of UAVs. Without loss of generality, It is assumed that the human operator controls the group from the body-frame of agent 1. This is a conventional choice, since many works on multi-robot systems in literature assume the presence of a leader or preferred robot, see e.g. Das et al. (2002). From Proposition 3.1, this is achieved by letting, $\forall i = 1 \dots N$,

$$\boldsymbol{u}_i^h = {}^iR_1\boldsymbol{\nu} - s\gamma_{12i}\boldsymbol{\beta}_{i1} + w\delta_{12}\gamma_{12i}S\boldsymbol{\beta}_{i1} \quad (3.29)$$

$$w_i^h = w. \quad (3.30)$$

where γ_{12i} is computed as $\gamma_{123}\gamma_{13i}$ for those $i \in \mathcal{I}_3(\boldsymbol{q})$. The following statement characterizes the choice of the human control terms $(\boldsymbol{u}_i^h, w_i^h)$.

Lemma 3.4. Commands (3.29–3.30) result in $\dot{\boldsymbol{p}}_1 = R_1\boldsymbol{\nu}$, $\dot{\psi}_1 = w$ for agent 1, and in exactly those coordinated motions preserving all the relative bearings, for all the remaining agents $i = 2 \dots N$. Furthermore,

it also results in $\dot{\delta}_{12} = s$.

Proof. To prove the first part of the statement, consider (3.29) and (3.30) with $i = 1$. Since it is ${}^1R_1 = I_3$ and, from Definition 3.2, $\gamma_{121} = 0$, then it follows that $\boldsymbol{u}_1^h = R_1\boldsymbol{\nu}$. By plugging $(\boldsymbol{u}_1^h,\ w_1^h)$ into (3.2) one gets immediately $\dot{\boldsymbol{p}}_1 = R_1\boldsymbol{\nu}$, $\dot{\psi}_1 = w$. Consider now the i-th UAV, assuming without loss of generality that $i \in \mathcal{I}_s(\boldsymbol{q})$. By injecting (3.29) and (3.30) into (3.2), and exploiting the expressions of the bearings (3.3) and of the inter-distance ratios (3.6), it follows

$$\begin{aligned}\dot{\boldsymbol{p}}_i &= R_1\boldsymbol{\nu} - s\gamma_{12i}\frac{\boldsymbol{p}_1 - \boldsymbol{p}_i}{\delta_{1i}} + w\delta_{12}\gamma_{12i}S\frac{\boldsymbol{p}_1 - \boldsymbol{p}_i}{\delta_{1i}} \\ &= R_1\boldsymbol{\nu} - \frac{s}{\delta_{12}}(\boldsymbol{p}_1 - \boldsymbol{p}_i) + wS(\boldsymbol{p}_1 - \boldsymbol{p}_i)\end{aligned} \tag{3.31}$$

Therefore for two generic agents i and j (w.l.o.g. $j \in \mathcal{I}_s(\boldsymbol{q})$) it is

$$\dot{\boldsymbol{p}}_{ij} = \dot{\boldsymbol{p}}_j - \dot{\boldsymbol{p}}_i = \frac{s}{\delta_{12}}\boldsymbol{p}_{ij} - wS\boldsymbol{p}_{ij}.$$

Substituting this expression of $\dot{\boldsymbol{p}}_{ij}$ in the time derivative of $\boldsymbol{\beta}_{ij}$ (3.22), it finally follows

$$\begin{aligned}\dot{\boldsymbol{\beta}}_{ij} &= \frac{1}{\delta_{ij}}{}^iR\left[wS\boldsymbol{p}_{ij} + P(\hat{\boldsymbol{p}}_{ij})\dot{\boldsymbol{p}}_{ij}\right] \\ &= \frac{1}{\delta_{ij}}{}^iR\left[wS\boldsymbol{p}_{ij} + \underbrace{P(\hat{\boldsymbol{p}}_{ij})\frac{s}{\delta_{12}}\boldsymbol{p}_{ij}}_{=\mathbf{0}_3} - \underbrace{P(\hat{\boldsymbol{p}}_{ij})wS\boldsymbol{p}_{ij}}_{wS\boldsymbol{p}_{ij}}\right] = \mathbf{0}_3,\end{aligned}$$

where it was used the fact that $P(\hat{\boldsymbol{p}}_{ij})\boldsymbol{p}_{ij} = \mathbf{0}_3$ and $P(\hat{\boldsymbol{p}}_{ij})S\boldsymbol{p}_{ij} = S\boldsymbol{p}_{ij}$. This proves that the bearings are preserved. Finally, by differentiating the δ_{12} and using (3.29), it follows

$$\begin{aligned}\dot{\delta}_{12} &= \hat{\boldsymbol{p}}_{12}^T\left(R_1\boldsymbol{\nu} - sR_2\boldsymbol{\beta}_{21} + w\delta_{12}SR_2\boldsymbol{\beta}_{21} - R_1\boldsymbol{\nu}\right) \\ &= s\hat{\boldsymbol{p}}_{12}^T\hat{\boldsymbol{p}}_{12} + w\delta_{12}\hat{\boldsymbol{p}}_{12}^TS\hat{\boldsymbol{p}}_{12} = s\end{aligned}$$

where it was used the property that $\hat{\boldsymbol{p}}_{12}^TS\hat{\boldsymbol{p}}_{12} = \mathbf{0}_3$. □

Note that control (3.29–3.30) is a function of only bearing measurements (using Properties 3.1 and 3.2) with the exception of the unique metric

quantity δ_{12}. This is in fact needed in (3.29) to correctly implement a synchronized group rotation. If δ_{12} is available through direct measurement or online estimation, then (3.29) can be exactly implemented. If the distance δ_{12} is not available then it can be replaced with an arbitrary initial guess $\hat{\delta}_{12} > 0$, e.g., chosen by the human operator. This choice will result in a non-perfect execution of the synchronized rotation command, which will pull the bearing-formation away from the desired one. On the other hand, the feedback action of the term $(\boldsymbol{u}_i^f, w_i^f)$ in (3.20) – see (3.35–3.37) – will keep these disturbances bounded by trying to achieve the desired formation, eventually keeping a (bounded) non-zero bearing error at steady-state. These intuitive considerations have been empirically proven by the simulations and experiments of Sec. 3.8, while a formal characterization is destined to future works.

Input Devices The interaction with the human is realized by means of two actuated input devices: a 3 DoF device for controlling the group linear velocity $\boldsymbol{\nu}$, and a 2 DoF device for commanding the group expansion/rotation rates $(s,\, w)$. These haptic devices are modeled as generic mechanical systems

$$M_t(\boldsymbol{x}_t)\ddot{\boldsymbol{x}}_t + C_t(\boldsymbol{x}_t, \dot{\boldsymbol{x}}_t)\dot{\boldsymbol{x}}_t = \boldsymbol{\tau}_t + \boldsymbol{f}_t \tag{3.32}$$

$$M_r(\boldsymbol{x}_r)\ddot{\boldsymbol{x}}_r + C_r(\boldsymbol{x}_r, \dot{\boldsymbol{x}}_r)\dot{\boldsymbol{x}}_r = \boldsymbol{\tau}_r + \boldsymbol{f}_r \tag{3.33}$$

where $\boldsymbol{x}_t \in \mathbb{R}^3$ and $\boldsymbol{x}_r = \begin{pmatrix} x_s & x_w \end{pmatrix}^T \in \mathbb{R}^2$ are the device position vectors, $M_t(\boldsymbol{x}_t) \in \mathbb{R}^{3\times3}$ and $M_r(\boldsymbol{x}_r) \in \mathbb{R}^{2\times2}$ their positive-definite and symmetric inertia matrices, $C_t(\boldsymbol{x}_t, \dot{\boldsymbol{x}}_t) \in \mathbb{R}^{3\times3}$ and $C_r(\boldsymbol{x}_r, \dot{\boldsymbol{x}}_r) \in \mathbb{R}^{2\times2}$ represent Coriolis and centrifugal terms, and the pairs $(\boldsymbol{f}_t,\, \boldsymbol{\tau}_t) \in \mathbb{R}^3 \times \mathbb{R}^3$, $(\boldsymbol{f}_r,\, \boldsymbol{\tau}_r) \in \mathbb{R}^2 \times \mathbb{R}^2$ are the human/control forces acting on each device, respectively. The control forces $\boldsymbol{\tau}_t$ and $\boldsymbol{\tau}_r$ will be discussed in Sec. 3.7. As usually done, gravity effects are assumed to be locally compensated.

The control actions (3.29) and (3.30) are then implemented by setting

$$\boldsymbol{\nu} = \lambda_t \boldsymbol{x}_t, \quad \begin{pmatrix} s \\ w \end{pmatrix} = \begin{pmatrix} \lambda_s & 0 \\ 0 & \lambda_w \end{pmatrix} \boldsymbol{x}_r, \tag{3.34}$$

where $\lambda_t > 0$, $\lambda_s > 0$, and $\lambda_w > 0$ are suitable scaling factors from the device positions $(\boldsymbol{x}_t,\, \boldsymbol{x}_r)$ to the generalized velocity commands. The proposed architecture implements a position-velocity coupling between the haptic device and the UAV group. This is the most natural choice in order

to handle the *kinematic dissimilarity*, i.e., the fact that the haptic device has a bounded workspace but the UAVs are characterized by an unbounded workspace, e.g., see also Franchi et al. (2011b).

3.6 Formation Controller

By applying Lemma 3.2 with $\boldsymbol{\beta}^d_{\mathcal{E}}$ playing the role of $\boldsymbol{\alpha}$, the agents are able to compute a minimally-linear rigid bearing-formation $(\hat{\mathcal{G}}, \boldsymbol{\beta}^d_{\hat{\mathcal{E}}})$ equivalent to $(\mathcal{G}, \boldsymbol{\beta}^d_{\mathcal{E}})$, thus sharing the same class of realizations. Without loss of generality, agents 1, 2, and 3 as per the definition of $\hat{\mathcal{E}}$ are hereinafter called *beacon agents*. However, their role can be taken by any triplet whose positions in a realization of $(\mathcal{G}, \boldsymbol{\beta}^d_{\mathcal{E}})$ (and then also of $(\hat{\mathcal{G}}, \boldsymbol{\beta}^d_{\hat{\mathcal{E}}})$) are not collinear. Note also that by exploiting the desired bearings in $\boldsymbol{\beta}^d_{\hat{\mathcal{E}}}$, the agents are also able to compute all the desired relative rotations ${}^1R^d_i$ and desired distance ratios γ^d_{12i} for any $i = 2 \dots N$ as shown by Properties 3.1 and 3.2, respectively. The objective of the formation controller is stated as follows:

Objective 3.2. Design $(\boldsymbol{u}^f_i, w^f_i)$ such that $\boldsymbol{\beta}_{\hat{\mathcal{E}}} \to \boldsymbol{\beta}^d_{\hat{\mathcal{E}}}$.

To achieve this objective, the term $(\boldsymbol{u}^f_i, w^f_i)$ in (3.20) is designed as follows:

$$\begin{pmatrix} \boldsymbol{u}^f_1 \\ w^f_1 \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ 0 \end{pmatrix} \tag{3.35}$$

$$\begin{pmatrix} \boldsymbol{u}^f_2 \\ w^f_2 \end{pmatrix} = \begin{pmatrix} K_p \, {}^2R_1 [\boldsymbol{\beta}_{12} \times (\boldsymbol{\beta}^d_{12} \times \boldsymbol{\beta}_{12})] \\ K_\omega \left[{}^1R^d_2 \, {}^2R_1 - {}^1R_2 \, {}^2R^d_1 \right]_{\vee,3} \end{pmatrix} \tag{3.36}$$

$$\begin{pmatrix} \boldsymbol{u}^f_i \\ w^f_i \end{pmatrix} = \begin{pmatrix} K_p \, {}^iR_1 \left(\gamma^d_{12i} \boldsymbol{\beta}^d_{1i} - \gamma_{12i} \boldsymbol{\beta}_{1i} \right) \\ K_\omega \left[{}^1R^d_i \, {}^iR_1 - {}^1R_i \, {}^iR^d_1 \right]_{\vee,3} \end{pmatrix} \tag{3.37}$$

with $i = 3, \dots, N$, $K_p, K_\omega > 0$ being positive gains, and $[A]_{\vee,3}$ representing the third component of the vector associated to a skew-symmetric matrix A. The following statement provides a formal validation of the effectiveness of the chosen control term $(\boldsymbol{u}^f_i, w^f_i)$ in controlling the bearing-formation.

Propositon 3.2 (Formation Control)**.** Given a desired non-degenerate rigid bearing-formation $(\mathcal{G}, \boldsymbol{\beta}^d_{\mathcal{E}})$, consider the relabeling and equivalent minimally-linear rigid formation $(\hat{\mathcal{G}}, \boldsymbol{\beta}^d_{\hat{\mathcal{E}}})$ obtained by applying Lemma 3.2. Control (3.20), together with (3.29–3.30) and (3.35–3.37), asymptotically and almost globally steers $(\hat{\mathcal{G}}, \boldsymbol{q}(t))$ towards the particular realization of $(\hat{\mathcal{G}}, \boldsymbol{\beta}^d_{\hat{\mathcal{E}}})$ such that

$$\boldsymbol{q}_1(t) = \boldsymbol{q}_1(t_0) + \int_{t_0}^{t} \begin{pmatrix} R(\psi_1)\boldsymbol{\nu} \\ w \end{pmatrix} dt \tag{3.38}$$

$$\delta_{12}(t) = \delta_{12}(t_0) + \int_{t_0}^{t} s \; dt, \tag{3.39}$$

provided that $\int_{t_0}^{t} s\, dt > -\delta_{12}(t_0)$ and $K_p > s$ for any $t \geq 0$. Furthermore $\delta_{1i} \to \gamma^d_{12i}\delta_{12}$, for any $i = 3 \dots N$.

Proof. The first condition (3.38) is a trivial consequence of (3.35) and (3.29–3.30) evaluated for $i = 1$. In order to show (3.39), differentiate the dynamics of δ_{12} using (3.4) together with (3.29),(3.35–3.36) and $i = 1$, $j = 2$. This yields

$$\begin{aligned} \dot{\delta}_{12} &= \boldsymbol{\beta}_{12}^T({}^1R_2\boldsymbol{u}_2^f) + s = \\ &= K_p\boldsymbol{\beta}_{12}^T {}^1R_2 {}^2R_1[\boldsymbol{\beta}_{12} \times (\boldsymbol{\beta}^d_{12} \times \boldsymbol{\beta}_{12})] + s = s. \end{aligned}$$

Since the control action (3.29-3.30) is designed to not change any bearing, the dynamics of $\boldsymbol{\beta}_{ij}$, obtained differentiating (3.3), is

$$\dot{\boldsymbol{\beta}}_{ij} = -\begin{pmatrix} 0 \\ 0 \\ w_i^f \end{pmatrix} \times \boldsymbol{\beta}_{ij} + \frac{I - \boldsymbol{\beta}_{ij}\boldsymbol{\beta}_{ij}^T}{\delta_{ij}} \left({}^iR_j\boldsymbol{u}_j^f - \boldsymbol{u}_i^f \right). \tag{3.40}$$

In order to prove that $(\hat{\mathcal{G}}, \boldsymbol{q}(t))$ converges to the set of realizations of $(\hat{\mathcal{G}}, \boldsymbol{\beta}^d_{\hat{\mathcal{E}}})$, it is sufficient to show that $\boldsymbol{\beta}_{ij}(t) \to \boldsymbol{\beta}^d_{ij}$, $\forall (i,j) \in \hat{\mathcal{E}}$. For a better readability, the proof is now divided in several sub-cases:

Convergence of $\boldsymbol{\beta}_{1i}$ to $\boldsymbol{\beta}^d_{1i}$ the first step is to prove that $\boldsymbol{\beta}_{12} \to \boldsymbol{\beta}^d_{12}$ by plugging the control inputs (3.35–3.36) into (3.40) (with $i = 1$, $j = 2$) and

obtaining

$$\begin{aligned}\dot{\boldsymbol{\beta}}_{12} &= (K_p/\delta_{12}(t))(\boldsymbol{\beta}_{12} \times (\boldsymbol{\beta}_{12}^d \times \boldsymbol{\beta}_{12})) = \\ &= (K_p/\delta_{12}(t))[\boldsymbol{\beta}_{12}^d(\boldsymbol{\beta}_{12}^T\boldsymbol{\beta}_{12}) - \boldsymbol{\beta}_{12}(\boldsymbol{\beta}_{12}^T\boldsymbol{\beta}_{12}^d)].\end{aligned}$$

Consider the dynamics of the error term $e_{12} = \boldsymbol{\beta}_{12}^T\boldsymbol{\beta}_{12}^d - 1$, that is,

$$\begin{aligned}\dot{e}_{12} &= \dot{\boldsymbol{\beta}}_{12}^T\boldsymbol{\beta}_{12}^d = -(K_p/\delta_{12}(t))((\boldsymbol{\beta}_{12}^T\boldsymbol{\beta}_{12}^d)^2 - 1) = \\ &= -(K_p/\delta_{12}(t))(\boldsymbol{\beta}_{12}^T\boldsymbol{\beta}_{12}^d - 1)(\boldsymbol{\beta}_{12}^T\boldsymbol{\beta}_{12}^d + 1) = \\ &= -(K_p/\delta_{12}(t))(\boldsymbol{\beta}_{12}^T\boldsymbol{\beta}_{12}^d + 1)e_{12}.\end{aligned}$$

Except from the zero-measure case of an initial condition $\boldsymbol{\beta}_{12}(t_0) = -\boldsymbol{\beta}_{12}^d$, it is $\boldsymbol{\beta}_{12}^T\boldsymbol{\beta}_{12}^d + 1 > 0$. Since also $\delta_{12}(t) > 0$ by assumption, it follows that $e_{12} \to 0$ almost everywhere and, therefore, that $\boldsymbol{\beta}_{12} \to \boldsymbol{\beta}_{12}^d$.
To prove that $\boldsymbol{\beta}_{1i} \to \boldsymbol{\beta}_{1i}^d$, with $i = 3 \dots N$, one can show that $\boldsymbol{\beta}_{1i}\delta_{1i} \to \boldsymbol{\beta}_{1i}^d\gamma_{12i}^d\delta_{12}$, which also implies $\delta_{1i} \to \gamma_{12i}^d\delta_{12}$, i.e., $\gamma_{12i} \to \gamma_{12i}^d$. Consider now the error $e_{1i} = \boldsymbol{\beta}_{1i}\delta_{1i} - \boldsymbol{\beta}_{1i}^d\gamma_{12i}^d\delta_{12}$. Since $\boldsymbol{\beta}_{1i}\delta_{1i} = {}^1R\boldsymbol{p}_{1i}$, after some straightforward algebra the error dynamics results in:

$$\begin{aligned}\dot{e}_{1i} &= {}^1R R_i \boldsymbol{u}_i^f - (\boldsymbol{\beta}_{1i}^d\gamma_{12i}^d - \boldsymbol{\beta}_{1i}\gamma_{12i})s = \\ &= (K_p - s)\,{}^1R_i\,{}^iR_1 \left(\gamma_{12i}^d\boldsymbol{\beta}_{1i}^d - \gamma_{12i}\boldsymbol{\beta}_{1i}\right) = -\frac{K_p - s}{\delta_{12}(t)}e_{1i},\end{aligned}$$

thus proving global exponential convergence of e_{1i} to zero since $\delta_{12}(t) > 0$ and $K_p - s > 0$ by assumption.

Convergence of $\boldsymbol{\beta}_{i1}$ to $\boldsymbol{\beta}_{i1}^d$ First, one can start showing that ${}^iR_1$ converges to ${}^iR_1^d$. Let $\psi_{1i} = \psi_1 - \psi_i$ represent the relative yaw among agents 1 and i, and note that, because of their definitions, ${}^iR_1 = R_z(\psi_{1i})$ and ${}^iR_1^d = R_z(\psi_{1i}^d)$. Therefore $\psi_{1i} \to \psi_{1i}^d$ implies ${}^iR_1 \to {}^iR_1^d$. Define $e_{\psi_{1i}} = \psi_{1i}^d - \psi_{1i}$ as error term and note that $\dot{e}_{\psi_{1i}} = -w_i^f$ since $w_1^h = w_i^h = w$ from (3.30). By applying control (3.37) one obtains

$$\dot{e}_{\psi_{1i}} = -K_\omega \left[{}^1R_i^d\,{}^iR_1 - {}^1R_i\,{}^iR_1^d\right]_{\vee,3} = -2K_\omega \sin(e_{\psi_{1i}}),$$

showing that $e_{\psi_{1i}} \to 0$ for all initial conditions apart from the zero-measure case $e_{\psi_{1i}}(t_0) = \pm\pi$.
Since $\boldsymbol{\beta}_{i1} = -{}^iR_1\boldsymbol{\beta}_{1i}$, and having proven that $\boldsymbol{\beta}_{1i} \to \boldsymbol{\beta}_{1i}^d$ (previous point) and ${}^iR_1 \to {}^iR_1^d$, it follows straightforwardly that $\boldsymbol{\beta}_{i1} \to \boldsymbol{\beta}_{i1}^d$.

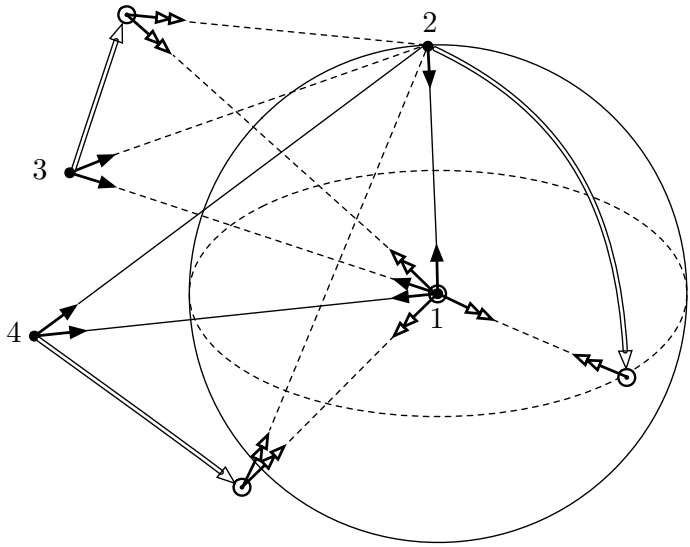


Figure 3.11: Visual representation of the action of the formation controller (3.35–3.37) in the case of 4 agents. Current agent positions are marked with solid dots, while black arrows and double-white arrows denote measured and corresponding desired bearings, respectively. The target positions for every agent are marked with white circles (note that these are not explicitly computed by the controller). Long and thick white arrows indicate the paths followed by the agents under the control action: agent 1 is stationary; agent 2 moves on the sphere centered around 1 following a geodesic path; agent 3 and 4 move along a straight line. All the motions are executed without resorting to any distance measurement.

Convergence of $\boldsymbol{\beta}_{jk}$ to $\boldsymbol{\beta}^d_{jk}$, for any remaining $(j,k) \in \hat{\mathcal{E}}$ from the proof of Property 3.1 and Property 3.3 with $l = 1$, $m = k$, $l = j$, it follows that

$$\boldsymbol{\beta}_{jk} = -{}^{j}R_k \boldsymbol{\beta}_{kj} = -{}^{j}R_1 \frac{\boldsymbol{\beta}_{1j}\gamma_{1kj} - \boldsymbol{\beta}_{1k}}{\|\boldsymbol{\beta}_{1j}\gamma_{1kj} - \boldsymbol{\beta}_{1k}\|}.$$

Note that from (3.6) it follows $\gamma_{1kj} = \gamma_{12j}/\gamma_{12k}$. Since from the previous points, it has been shown that $(\boldsymbol{\beta}_{1j}, \boldsymbol{\beta}_{1k}, {}^{j}R_1, \gamma_{12j}, \gamma_{12k})$ converge to their desired values, it obviously follows that $\boldsymbol{\beta}_{jk} \to \boldsymbol{\beta}^d_{jk}$. □

In order to better understand the action of control (3.35–3.37) it is useful

to highlight, with the help of Fig. 3.11, some properties depending on the human's steering inputs $\boldsymbol{\nu}$, s, and w:

1. the translational dynamics of agent 1 is affected only by the high-level command $\boldsymbol{\nu}$, therefore it will stay fixed in space when $\boldsymbol{\nu} = \mathbf{0}$;

2. if $s = 0$ then agent 2 will rotate and travel along the geodesic path on the sphere centered on agent 1 by keeping the 'radius' δ_{12} constant. Indeed, *i)* the control action $\boldsymbol{u}_2^f$ is always orthogonal to $\boldsymbol{\beta}_{12}(\boldsymbol{q})$ which represents the direction of the line connecting agents 1 and 2; *ii)* the external input $\boldsymbol{\nu}$ moves the agents 1 and 2 cohesively, and *iii)* the external input w generates a motion that is also orthogonal to $\boldsymbol{\beta}_{12}(\boldsymbol{q})$;

3. if $\boldsymbol{\nu} = \mathbf{0}$ and $s = w = 0$ then agents $3 \dots N$ will rotate and move along the straight lines connecting their initial position with the final position satisfying the constraints $\boldsymbol{\beta}_{ij}(\boldsymbol{q}) = \boldsymbol{\beta}_{ij}^d$, $\forall (i,j) \in \hat{\mathcal{E}}$, $\boldsymbol{q}_1 = \boldsymbol{q}_1(t_0)$, and $\delta_{12} = \delta_{12}(t_0)$.

There are several observations to be made regarding the proposed formation controller.

Remark 1. The measured quantities needed to implement controller (3.35–3.37) are $\boldsymbol{\beta}_{21}$, $\boldsymbol{\beta}_{1i}$, ${}^iR_1$, and γ_{12i} $\forall i = 2 \dots N$. Since ${}^iR_1$ and γ_{12i} can always be obtained as a function of relative bearings (see Properties 3.1 and 3.2), the control law (3.35–3.37) uses only relative bearings as expected.

Remark 2. Condition $K_p > s$ does not constitute a limit of the proposed controller, since it can be always guaranteed by choosing a K_p large enough, given the range of variation of s in the specific application. Furthermore, in practical applications s cannot grow unbounded, mainly because of the actuation limits of a real UAV.

Remark 3. For a generic desired bearing-formation (i.e., whose realizations have no special alignments), any triplet of agents can be chosen as beacon agents in Lemma 3.2. A possible direction for future developments could be to study the problem of optimizing this choice of beacon agents with respect to any suitable criterion, such as robustness against measurement noise.

Remark 4. Control (3.35–3.37) becomes singular only when *all* the agents happen to be aligned during a transient phase (this is structurally avoided at steady-state since $(\mathcal{G}, \boldsymbol{\beta}_{\mathcal{E}}^d)$ is assumed to be non-degenerate). Such situation

represents a zero-measure case, and, in practice, is very unlikely to occur. For instance, it was never encountered during simulations or experiments. Nevertheless, since it is in principle possible to fall in its neighborhood, a practical workaround is to apply a suitable constant control action for a short phase in order to quickly exit from this singular configuration.

Remark 5. The Formation Controller does not have a built-in inter-agent collision avoidance, because this would require the UAVs to be able to measure their inter-distances while here it is assumed that the from the onboard camera only measure of relative-bearings are retrieved. Clearly, an inter-agent collision avoidance could be integrated if distance measures would be available. Nevertheless, if the UAVs are already close to a realization of the desired bearing-formation, then the operator can also assume the responsibility of avoiding inter-UAVs collisions, for example by expanding the formation if necessary.

3.6.1 Computational and Communication Complexity

One of the issues in multi-agent problems is the computational and communication complexity of the controller. These aspects are analyzed in the following propositions.

Propositon 3.3 (Computational Complexity)**.** The computational complexity of control (3.35–3.37) is $O(N)$. In fact, it can be implemented by only using the $3N-4$ relative-bearing measurements defined by the pairs included in $\hat{\mathcal{E}}$ obtained by applying Lemma 3.2 with $\boldsymbol{\beta}^d_{\mathcal{E}}$ playing the role of $\boldsymbol{\alpha}$.

Proof. Matrix ${}^2R_1$ can be computed from $\boldsymbol{\beta}_{12}$ and $\boldsymbol{\beta}_{21}$, and ${}^iR_1$ and γ_{12i}, $i = 3 \dots N$, can be obtained as follows: i) if $i \in \mathcal{I}_2$, ${}^iR_1 = {}^iR_2 {}^2R_1$ and ${}^iR_2$ can be evaluated by first applying Property 3.3 and then Property 3.1. As for γ_{12i}, it can be obtained by applying Property 3.2; ii) if $i \in \mathcal{I}_3$, ${}^iR_1 = {}^iR_3 {}^3R_1$ and ${}^iR_3$ follows by first applying Property 3.3 and then Property 3.1. Furthermore, from its definition, $\gamma_{12i} = \gamma_{13i}/\gamma_{123}$, and γ_{13i}, γ_{123} follow from Property 3.2. □

The applicability of Properties 3.1–3.3 is always verified at the desired bearing-formation and, thus, when the controller (3.35–3.37) has reached a steady-state regime. However, during initial transients, some of the assumptions needed by the Properties could temporarily not be met, e.g.,

when, for $i \in \mathcal{I}_2$, $\boldsymbol{\beta}_{12} = \pm\boldsymbol{\beta}_{1i}$. These transient situations can always be disambiguated by (temporarily) exploiting additional measurements in order to recover the loss of information, see for instance the second parts of Properties 3.1–3.2.

As for the issue of communication complexity of controller (3.35–3.37), let N_C be the number of exchanged messages per unit of time. Since not all the needed measurements are locally available, the agents need to share some information among themselves. For example, in order to implement (3.37), agent 2 needs to receive $\boldsymbol{\beta}_{12}$ from agent 1. When considering inter-agent communication issues, in order to prevent network congestions it is important to render N_C bounded with respect to the number of agents N. Typically, $N_C = O(N)$ is considered as a good tradeoff, see for instance the case of consensus/agreement algorithms with a bounded number of neighbors per agent (Lynch (1997)).

Propositon 3.4 (Communicational Complexity)**.** The communication complexity of controller (3.35–3.37) is $O(N)$.

Proof. Agent 1 needs no external information, agent 2 needs $\boldsymbol{\beta}_{12}$, agents $i \in \mathcal{I}_2$ need $\boldsymbol{\beta}_{1i}$ and $\boldsymbol{\beta}_{2i}$, and agents $i \in \mathcal{I}_3$ need $\boldsymbol{\beta}_{1i}$ and $\boldsymbol{\beta}_{3i}$, for a total of $N_C = 2(N-1)+1$ exchanged messages over the network per unit of time. □

3.6.2 Time-varying desired bearings

Formation controller (3.35) to (3.37) can be extended to the case where the desired relative bearings are not constant but (known) time-varying quantities defined as $\tilde{\boldsymbol{\beta}}^d_{ij}(t) = \tilde{R}_i(t)\boldsymbol{\beta}^d_{ij}$, with $\tilde{R}_i(t)$ given. This extension will be exploited in the experiments reported in Sec. 3.8.2, and is important to both overcome limited FoV of the relative-bearing sensor, and to allow for a possible scanning of the environment during a 3D coverage task, e.g., exploration, mapping, or surveillance.

The rotation matrices $\tilde{R}_i(t)$ considered here are only of the kind $R_{\vec{Z}_i}(\cdot)$ since these are the ones affected by the agent inputs w_i. Denote with $\tilde{\boldsymbol{\omega}}_i(t) = (0\ 0\ \tilde{\omega}_{i,3})^T \in \mathbb{R}^3$ the angular velocity associated to $\tilde{R}_i(t)$, i.e., such that $\tilde{\boldsymbol{\omega}}_i(t) = [\tilde{R}_i^T(t)\dot{\tilde{R}}_i(t)]_\vee$, where the $[\cdot]_\vee$ represents the vector associated to a skew-symmetric matrix. Presence of a time-varying bearing can be easily handled by using $\tilde{\boldsymbol{\beta}}_{ij}$ instead of $\boldsymbol{\beta}^d_{ij}$ and adding a suitable feedforward

term to the yaw-rate part of the controller (3.35–3.37), i.e., using:

$$w_1^f = -\tilde{\omega}_{1,3} \tag{3.41}$$

$$w_2^f = -\tilde{\omega}_{2,3} + K_\omega \left[{}^1\tilde{R}_2 \, {}^2R_1 - {}^1R_2 \, {}^2\tilde{R}_1 \right]_{\vee,3} \tag{3.42}$$

$$w_i^f = -\tilde{\omega}_{i,3} + K_\omega \left[{}^1\tilde{R}_i \, {}^iR_1 - {}^1R_i \, {}^i\tilde{R}_1 \right]_{\vee,3}. \tag{3.43}$$

Note that presence of a time-varying component in the desired bearings does not affect the overall formation shape, but only causes the agents to suitably rotate in space. In this way, the agents can track any exogenous rotation signal while still keeping the same shape implicitly defined by the static component of the desired bearings.

3.7 Haptic Feedback Algorithm

The loop with the human is closed by generating the control torques $\boldsymbol{\tau}_t$ and $\boldsymbol{\tau}_r$ that are rendered on the haptic devices (3.32) and (3.33) to increase the performances in human-robot cooperation. The underlying idea is to design the haptic cues so that they are informative of how well the *real* UAVs, as a group, are executing the desired human commands (3.34). Recalling the definition from Sec. 3.2.1, let $\dot{\boldsymbol{p}}_{\mathcal{A}_i} \in \mathbb{R}^3$ be the *body-frame* velocity vector of the i-th UAV, and $\dot{\psi}_{\mathcal{A}_i} \in \mathbb{R}$ its yaw rate. It is important to stress, again, that these represent real (measured) UAV quantities and not the reference (virtual) velocities of the agent model (3.2) tracked by the UAVs.

After the initial transient needed for reaching the desired bearing formation, i.e., when controller (3.35–3.37) has reached its steady-state, the components $(\boldsymbol{u}_i^f, w_i^f)$ in (3.20) become negligible and the only motion input for the UAV group is due to the high-level commands $(\boldsymbol{u}_i^h, w_i^h)$ (3.29–3.30). As first haptic cue, it is then considered the mismatch between the commanded translational velocity $\boldsymbol{\nu}$ and its actual execution by the UAVs. From (3.29) and (3.34) it follows that, for each i-th UAV,

$$\begin{aligned} \boldsymbol{x}_t = \frac{1}{\lambda_t}\boldsymbol{\nu} &= \frac{1}{\lambda_t} {}^1R_i(\boldsymbol{u}_i^h + s\gamma_{12i}\boldsymbol{\beta}_{i1} - w\delta_{12}\gamma_{12i}S\boldsymbol{\beta}_{i1}) \\ &= \frac{1}{\lambda_t} {}^1R_i \left(\boldsymbol{u}_i^h + \gamma_{12i}(\lambda_s x_s \boldsymbol{\beta}_{i1} - \lambda_w x_w \delta_{12} S\boldsymbol{\beta}_{i1})\right) \\ &\simeq \frac{1}{\lambda_t} {}^1R_i \left(\dot{\boldsymbol{p}}_{\mathcal{A}_i} + \gamma_{12i}(\lambda_s x_s \boldsymbol{\beta}_{i1} - \lambda_w x_w \delta_{12} S\boldsymbol{\beta}_{i1})\right) =: \boldsymbol{z}_{ti}. \end{aligned} \tag{3.44}$$

The approximation in (3.44) has two sources: (*i*) temporary presence of nonzero formation control inputs $(\boldsymbol{u}_i^f,\, w_i^f)$ in (3.20) because of transient disturbances in the maintenance of the bearing-formation, and (*ii*) a generic non-perfect tracking of the commanded velocities $\boldsymbol{u}_i$ from UAVs, that is, $\dot{\boldsymbol{p}}_{\mathcal{A}_i} \neq \boldsymbol{u}_i^h$ (e.g., presence of wind, actuator saturations, UAV inertia, etc.). Whatever the reason, the mismatch $\boldsymbol{e}_{ti} = \boldsymbol{x}_t - \boldsymbol{z}_{ti}$ represents a good measurement of how well the i-th UAV is executing the human *translational motion command* $\boldsymbol{\nu}$. The average translational mismatch is then obtained as

$$\boldsymbol{e}_t = \boldsymbol{x}_t - \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{z}_{ti} = \boldsymbol{x}_t - \boldsymbol{z}_t. \tag{3.45}$$

The mismatch between commanded expansion/rotation rates $(s,\, w)$ and their actual execution by the UAVs are derived analogously. From (3.29) and (3.34) it can be seen that, for each i-th UAV,

$$\begin{aligned} x_s = \frac{s}{\lambda_s} &= \frac{1}{\lambda_s \gamma_{12i}} ({}^iR_1\boldsymbol{\nu} - \boldsymbol{u}_i^h + w\delta_{12} S\boldsymbol{\beta}_{i1}) \cdot \boldsymbol{\beta}_{i1} = \\ &= \frac{1}{\lambda_s \gamma_{12i}} ({}^iR_1\boldsymbol{\nu} - \boldsymbol{u}_i^h) \cdot \boldsymbol{\beta}_{i1} \simeq \frac{1}{\lambda_s \gamma_{12i}} ({}^iR_1\boldsymbol{\nu} - \dot{\boldsymbol{p}}_{\mathcal{A}_i}) \cdot \boldsymbol{\beta}_{i1} \\ &=: z_{si} \end{aligned} \tag{3.46}$$

where it was exploited the fact that $S\boldsymbol{\beta}_{i1} \perp \boldsymbol{\beta}_{i1}$. Averaging over all UAVs yields the overall expansion error

$$e_s = x_s - \frac{1}{N}\sum_{i=1}^{N} z_{si} = x_s - z_s. \tag{3.47}$$

As for the rotation rate, eq. (3.30) simply yields

$$x_w = \frac{w}{\lambda_w} \simeq \frac{\dot{\psi}_{\mathcal{A}_i}}{\lambda_w} = z_{wi} \tag{3.48}$$

with the corresponding average error

$$e_w = x_w - \frac{1}{N}\sum_{i=1}^{N} z_{wi} = x_w - z_w. \tag{3.49}$$

Finally, the mismatch between the commanded and actual expansion/rotation rates is expressed as the vector

$$\boldsymbol{e}_r = \begin{pmatrix} e_s \\ e_w \end{pmatrix} = \boldsymbol{x}_r - \begin{pmatrix} z_s \\ z_w \end{pmatrix} = \boldsymbol{x}_r - \boldsymbol{z}_r. \tag{3.50}$$

Note that an evaluation of $\boldsymbol{z}_t$ and $\boldsymbol{z}_r$ requires each UAV to send to the haptic device its body-frame velocity and yaw rate $(\dot{\boldsymbol{p}}_{\mathcal{A}_i}, \dot{\psi}_{\mathcal{A}_i})$, its relative bearing $\boldsymbol{\beta}_{i1}$ w.r.t. agent 1, and the scaling factor γ_{12i}.

The control torques in (3.32–3.33), aimed at providing a useful force-feedback to the human operator, are then computed as

$$\boldsymbol{\tau}_t = -B_t\dot{\boldsymbol{x}}_t - K_t\boldsymbol{x}_t - K_t^e\boldsymbol{e}_t \tag{3.51}$$

$$\boldsymbol{\tau}_r = -B_r\dot{\boldsymbol{x}}_r - K_r\boldsymbol{x}_r - K_r^e\boldsymbol{e}_r. \tag{3.52}$$

Here, $B_t \in \mathbb{R}^{3\times3}$, $B_r \in \mathbb{R}^{2\times2}$ are positive definite damping matrices whose role is to stabilize the haptic devices, $K_t \in \mathbb{R}^{3\times3}$, $K_r \in \mathbb{R}^{2\times2}$ are semi-definite diagonal matrices (possibly null) for giving the user a perception of the distance to the zero-commanded velocity, and $K_t^e \in \mathbb{R}^{3\times3}$, $K_r^e \in \mathbb{R}^{2\times2}$ are positive definite diagonal matrices meant as scaling factors for $\boldsymbol{e}_t$ and $\boldsymbol{e}_r$.

Control (3.51–3.52) is, however, not robust against the destabilizing effects of the typical non-idealities in haptic-device/UAVs communication channels, that is, possible presence of discrete sampling, delays and packet losses. In order to guarantee teleoperation stability despite these effects, let $(\boldsymbol{z}_t[k], \boldsymbol{z}_r[k])$ be the discrete (received) versions of $(\boldsymbol{z}_t(t), \boldsymbol{z}_r(t))$. Then, the actual implementation of the force controller takes the form

$$\boldsymbol{\tau}_t = -B_t\dot{\boldsymbol{x}}_t - K_t\boldsymbol{x}_t - K_t^e(\boldsymbol{x}_t - \bar{\boldsymbol{z}}_t[k]) \tag{3.53}$$

$$\boldsymbol{\tau}_r = -B_r\dot{\boldsymbol{x}}_r - K_r\boldsymbol{x}_r - K_r^e(\boldsymbol{x}_r - \bar{\boldsymbol{z}}_r[k]). \tag{3.54}$$

where the quantities $(\bar{\boldsymbol{z}}_t[k], \bar{\boldsymbol{z}}_r[k])$ are the *passive set-position modulation* (PSPM) versions of the received $(\boldsymbol{z}_t[k], \boldsymbol{z}_r[k])$. The PSPM framework by Lee and Huang (2010) is a general tool for guaranteeing haptic-device passivity and, therefore, stability of the closed-loop system when dealing with possible delays/ discretization/ packet losses of the signals exchanged over the haptic-device/UAVs communication channel. The action of the PSPM is to *modulate* a received signal $\boldsymbol{z}[k]$ into a (possibly) attenuated version $\bar{\boldsymbol{z}}[k]$ so that implementation of (3.53–3.54) will meet the haptic-device (energetic) passivity constraint over its external power port. Being

the slave considered in this work a kinematic (first-order) system (agent model (3.2)), this is indeed sufficient to ensure stability of the overall closed-loop teleoperation scheme by further assuming, as usually done (see, e.g., Hogan (1989)), passivity of the human side. The interested reader is referred to Lee et al. (2011); Lee and Huang (2010) and references therein for a complete treatment and formal proofs of these statements. Note that, besides the passifying action against non-idealities of the communication channel, the PSPM framework also allows to enforce closed-loop stability despite the nonstandard position-velocity coupling between haptic-device and the UAVs adopted in this work, see again Lee et al. (2011) for a more thorough treatment.

The resulting haptic cues will not only give a general feeling of the execution accuracy of the human commands, but are specifically designed to represent the execution mismatches along the 5 motion directions in a *decoupled* way.

3.8 Simulations and Experiments

3.8.1 Experimental Testbed

The two haptic devices used both for the simulations and real experiments are shown in Fig. 3.12a: the device on the right is the 3 DoF device[5] modeled by (3.32) and responsible for the command $\boldsymbol{\nu}$ in (3.34) and the force feedback term $\boldsymbol{\tau}_t$ in (3.53). The device on the left is also a 3 DoF device but constrained via software to only move on the 2D horizontal plane in order to behave as (3.33). This device is responsible for the commands $(s,\, w)$ in (3.34) and the force feedback term $\boldsymbol{\tau}_r$ in (3.54). Both devices are connected to a GNU-Linux machine where a local control loop implements the forces $\boldsymbol{\tau}_t$ and $\boldsymbol{\tau}_r$ at a frequency of 2.5 kHz. The control program also sends over a wireless channel the inputs $(\boldsymbol{\nu},\, s,\, w)$ to the UAV controllers at 120 Hz, and receives the UAV measurements needed to implement $\boldsymbol{\tau}_t$ and $\boldsymbol{\tau}_r$.

The UAVs used in these simulations/experiments are quadrotors, see Figs. 3.12 and 3.14. Indeed, the use of quadrotors makes it possible to empirically validate the preliminary assumption of Sec. 3.2, i.e., to exactly track the virtual agent dynamics (3.2) thanks to the flatness of the quadrotor outputs $(\boldsymbol{p}_{\mathcal{A}_i},\, \psi_{\mathcal{A}_i})$. As summarized by Fig. 3.13, the controller of each

[5] `http://www.forcedimension.com`

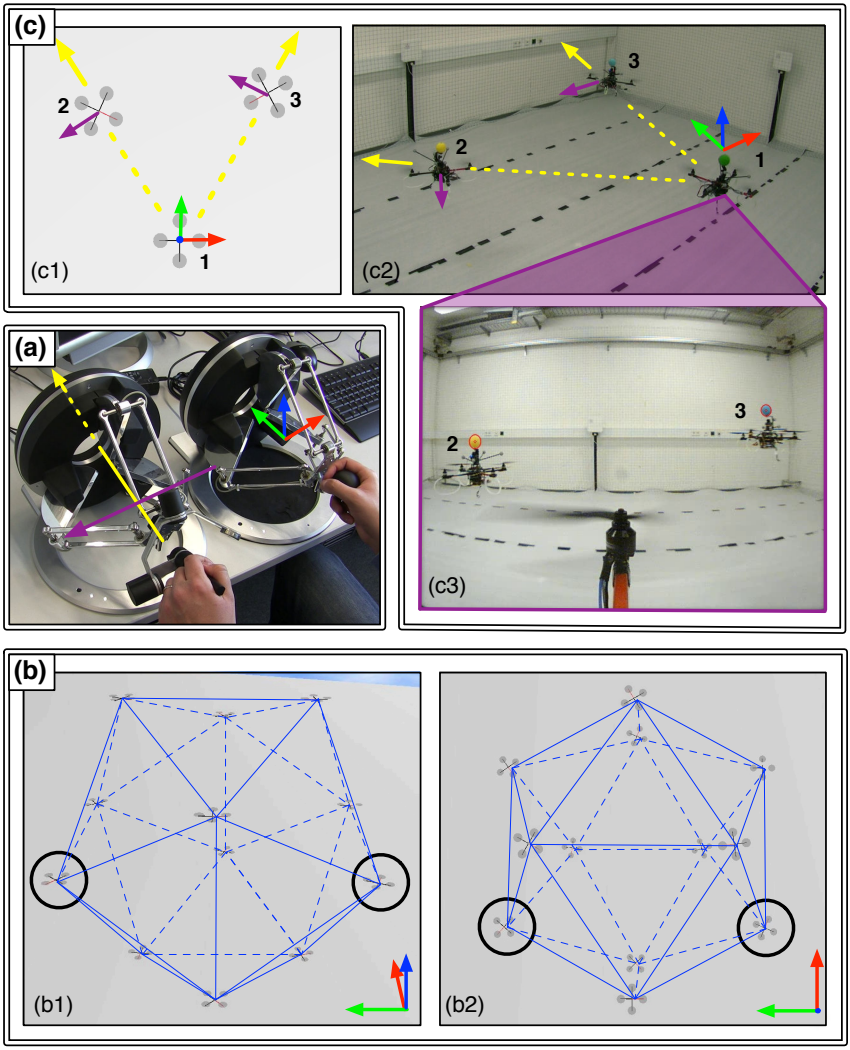


Figure 3.12: Experimental setup: (a) 3 DoF haptic-feedback devices used to perform the bilateral high-level steering. (b) Simulation environment used to physically simulate the quadrotors: side (b1) and top (b2) views of 12 quadrotors in a icosahedron formation with agents 1 and 2 being highlighted. (c) Real UAV setup with 3 quadrotors: (c1) top-view of the formation in the 3D visualizer; (c2) triangular formation used during the experiments; (c3) onboard camera view of agent 1 with agent 2 and 3 detected by the image processing algorithm.

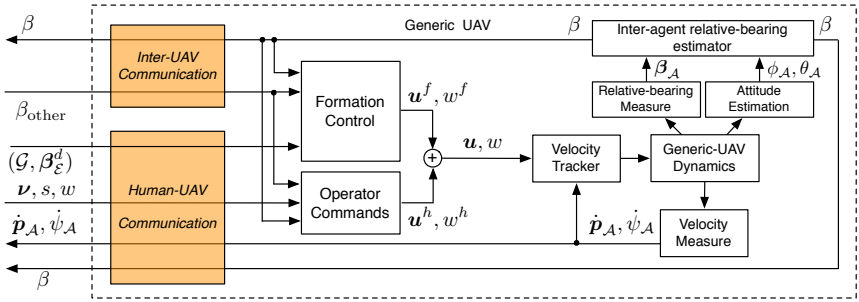


Figure 3.13: Overall system architecture as seen from the point-of-view of a generic UAV.

quadrotor (both simulated/real) runs three processes: the *agent-process*, the *velocity-tracker* and the *inter-agent bearing estimator*.

The agent process locally implements the Human Steering and Formation Control algorithms. At the beginning of the task, the i-th UAV receives from the human the desired bearing-formation. Then, during the task the UAV receives the commands ($\boldsymbol{\nu}$, s, w) in (3.34) from the haptic devices to implement (3.29) and (3.30). Analogously, the UAV communicates via a wireless link with the agent processes of the other UAVs to receive the bearing measures that are necessary to compute the formation control terms (3.35) to (3.37).

The velocity-tracker enables the quadrotor to track smooth body-frame *reference velocities* ($\boldsymbol{u}_i$, w_i) by implementing a standard cascaded controller similar to the one used in Michael et al. (2010) but without position error terms. Namely, the velocity tracker regulates the real (measured) UAV velocities ($\dot{\boldsymbol{p}}_{\mathcal{A}}$, $\dot{\psi}_{\mathcal{A}}$) to the desired ones. These measured velocities are also sent back to the Haptic Feedback Algorithm to compute the haptic cues (3.53–3.54)

Finally, the inter-agent bearing estimator measures the UAV relative bearings $\boldsymbol{\beta}_{\mathcal{A}_i\mathcal{A}_j}$ and roll/pitch angles ($\phi_{\mathcal{A}_i}$, $\theta_{\mathcal{A}_i}$) to obtain the corresponding inter-agent relative bearings (3.5) that are used by the UAV controller and communicated to the Haptic Feedback Algorithm and to the other UAVs in the group.

The simulation environment is made of a custom software Lächele et al.

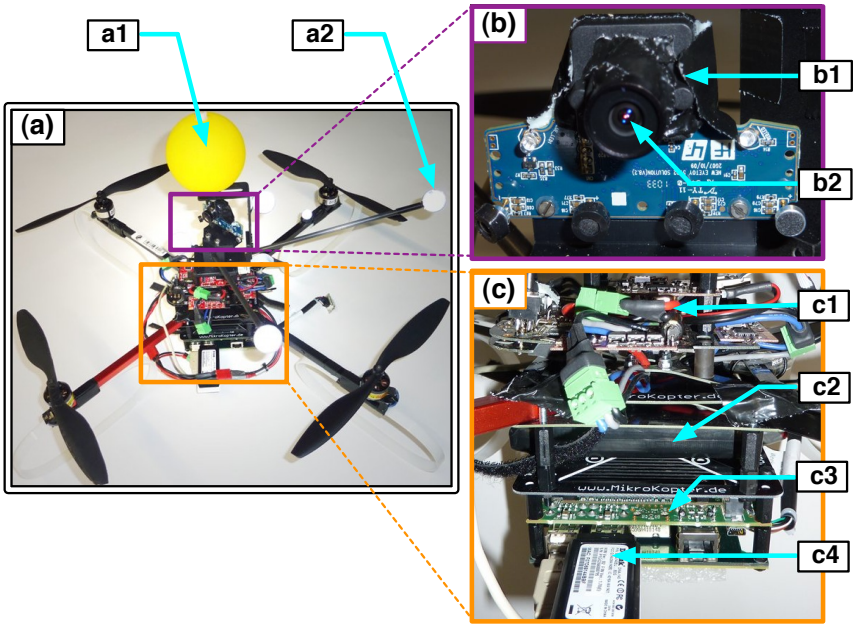


Figure 3.14: Quadrotor setup. (a) Quadrotor in its flight configuration: a1) colored sphere used for the visual tracking, a2) reflective marker used for the ground-truth tracking system (b) Camera setup: b1) Consumer-market camera, b2) 140° lens. (c) Computational setup: c1) Microcontroller and IMU, c2) Battery, c3) GNU-Linux PC Board, c4) Wireless adapter.

(2012) based on third party 3D graphics and physics engines[6] (see Fig. 3.12b for two screenshots). This environment simulates the quadrotor rigid-body dynamics and generates the measurements needed by the controllers.

As for the experiments, 3 real quadrotors[7] have been used with the setup shown in Fig. 3.14a. The velocity-tracker and agent-process implementation is split between the onboard microcontroller (Fig. 3.14c1), a desktop PC and a small GNU-Linux PC-board[8] mounted underneath the quadrotor (Fig. 3.14c3). Measurements of the quadrotor current roll and pitch angles

[6] `http://www.ogre3d.org`, `http://www.nvidia.com/object/physx_new.html`
[7] `http://www.mikrokopter.de`
[8] `http://www.seco.it/en/`, `http://www.qseven-standard.org/`

(a) $t = 0$s. (b) $t = 4$s. (c) $t = 17$s.

Figure 3.15: Screenshots of the simulation. The Formation Controller drives the robot to a realization of the desired bearing-formation.

$(\phi_{\mathcal{A}_i}, \theta_{\mathcal{A}_i})$ are obtained by fusing the onboard IMU readings by means of a standard complementary filter. The absolute yaw $\psi_{\mathcal{A}_i}$, on the other hand, is not measured nor estimated since it is not needed by the framework. An external optical tracking system[9] is used to *only* retrieve the quadrotor body-frame velocity since, again, no additional global position measurements were needed. Note however that this velocity information could also be obtained exploiting vision/optical flow Grabe et al. (2012) or range finders.

The relative bearings needed by the controller were obtained from an onboard monocular camera (Fig. 3.14b) with an horizontal/vertical FoV of about 88/60 deg. In particular, by equipping every quadrotor with a colored sphere on its top (Fig. 3.14a1), measurements of relative bearings are estimated by segmenting these spheres from the onboard camera images. This algorithm could run with a rate of about 7 Hz and with an average latency of 500 ms. This relatively poor performance was mainly due to the absence of a dedicated driver for the camera and to the lack of a GPU in the used small PC-board. Nevertheless, the approach was robust enough to deal with these and additional non-idealities representative of real-world conditions. A visualization of the ball-tracker output is shown in Fig. 3.12c3, where the detection of the balls is highlighted with red circles.

3.8.2 Results

Simulation 1

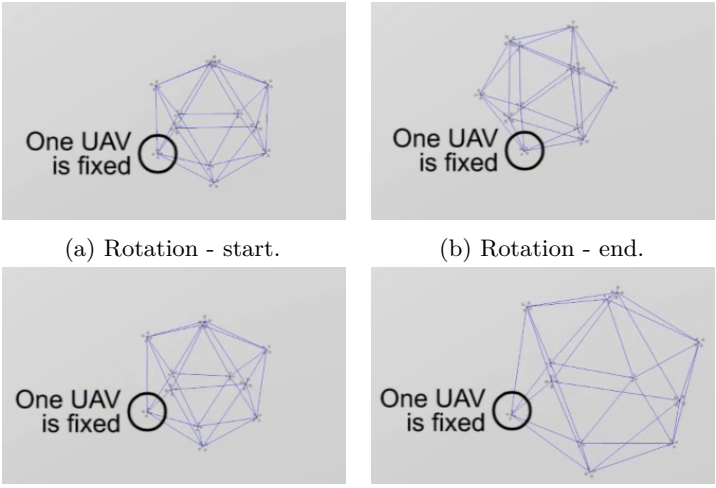
The first results here presented are from a Human/Hardware in-the-loop simulation involving 12 UAVs that start far from the desired bearing-formation. The desired bearing-formation has been chosen so that realizations are

[9] `http://www.vicon.com/`

regular icosahedra having the agents as vertices. In this test the UAVs have been simulated with an unlimited FoV capability so that relative bearings could always be retrieved. The simulation is articulated in two phases: at the beginning, only the formation control is active while the human operator holds the two haptic devices fixed at their neutral position. This first phase is illustrated by the screenshots in Fig. 3.15. After the time $t = 17\,\mathrm{s}$ (vertical dashed black line in Figs. 3.17a to 3.17f), the desired formation is eventually reached, and the human operator starts commanding the overall group. Screenshots from this second phase are shown in Fig. 3.16. This second phase is itself split into two sub-phases: at first (from $t = 17\,\mathrm{s}$ to $t = 110\,\mathrm{s}$), the human operator intentionally commands the 5 available motion directions once at a time by following this particular order $\boldsymbol{\nu}_y \to \boldsymbol{\nu}_x \to \boldsymbol{\nu}_z \to r \to w$ (see Fig. 3.17b). This is done to isolate the effects of each command. Then, during the last sub-phase (from $t = 110\,\mathrm{s}$ to $t = 150\,\mathrm{s}$), the human operator gives a generic command which shuffles the 5 motion directions all together.

Figures 3.17a to 3.17c show the average tracking errors between the actual UAV velocities $(\dot{\boldsymbol{p}}_{\mathcal{A}_i}, \dot{\psi}_{\mathcal{A}_i})$ and the reference velocities $(\boldsymbol{u}_i, w_i)$ associated to the agent model (3.2). As expected, due to the flatness of the quadrotor, the velocity tracker is able to keep the tracking error sufficiently small, thus confirming the assumptions of Sec. 3.2. Figure 3.17e, shows the evolution of the average quadratic error of the bearing-formation $\boldsymbol{\beta}_{ij}^d - \boldsymbol{\beta}_{ij}$, $\forall \boldsymbol{\beta}_{ij}^d \in \boldsymbol{\beta}_{\mathcal{E}}^d$: this goes exponentially to zero until $t = 17\,\mathrm{s}$, and then does not increase when the human starts commanding the group during the second phase. The only exception is a bounded error occurring whenever the human is commanding a rotation w: this is due to the mismatch between the actual distance δ_{12} and the constant guess $\hat{\delta}_{12}$ used to implement (3.29), see the remarks in Sec. 3.5. This is in perfect agreement with the theoretical analysis and expectations.

Figure 3.17d shows the behavior of the error vectors $(\boldsymbol{e}_t, \boldsymbol{e}_r)$ in (3.45)–(3.50). The peaks occur when the commanded acceleration is at its maximum as a consequence of the non-negligible inertia of the physically simulated quadrotors. Note also how the components of $(\boldsymbol{e}_t, \boldsymbol{e}_r)$ show a good decoupling during the first phase of the human operation, roughly from $t = 17\,\mathrm{s}$ to $t = 110\,\mathrm{s}$: activation of one particular command among $(\boldsymbol{\nu}, r, w)$ *only* affects its associated component of the error vectors $(\boldsymbol{e}_t, \boldsymbol{e}_r)$, with, again, the only exception of the command w because of the wrong estimate $\hat{\delta}_{12}$. Finally, Fig. 3.17f shows the control torques $(\boldsymbol{\tau}_t, \boldsymbol{\tau}_r)$ computed from (3.53–3.54) during the motion. These are basically proportional to



(a) Rotation - start.

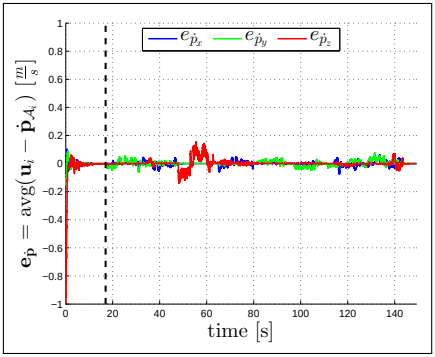
(b) Rotation - end.

(c) Expansion - start.

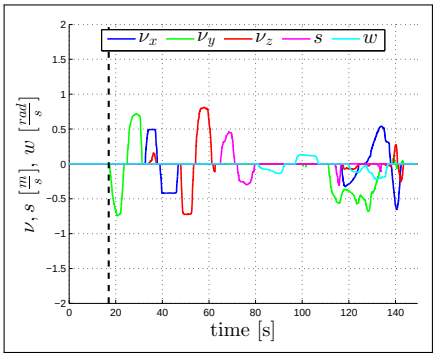
(d) Expansion - end.

Figure 3.16: Screenshots of the simulation - Rotation and change of scale commanded by the Human Steering.

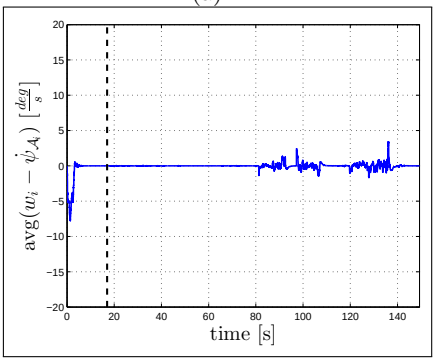
the error vectors ($\boldsymbol{e}_t$, $\boldsymbol{e}_r$) as expected, since matrices K_t^e and K_r^e have been chosen to be dominant with respect to B_t, K_t, and B_r, K_r, respectively, in (3.51-3.52).



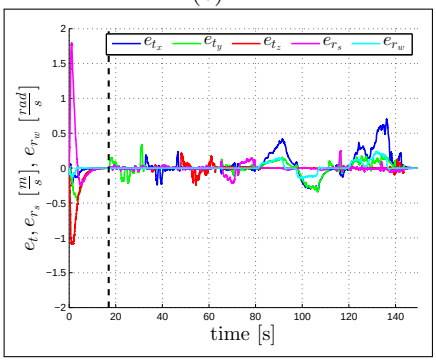
(a)



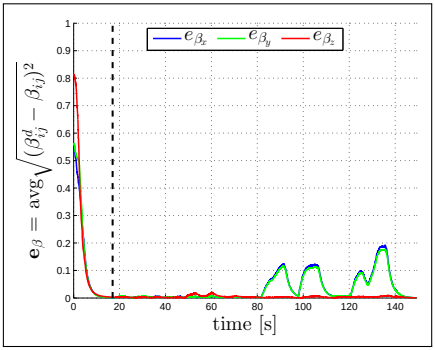
(b)



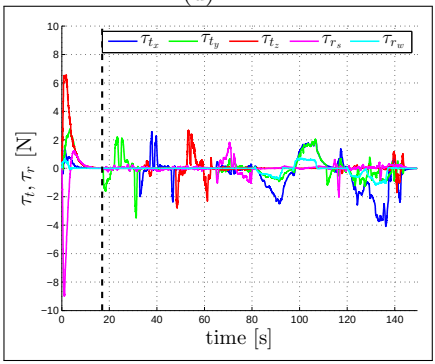
(c)



(d)



(e)



(f)

Figure 3.17: Results with a group of 12 simulated quadrotors. a,c: Average velocity and yaw rate tracking errors; b: Operator commands: translation velocity, expansion rate and rotation speed; d: Mismatch between commands and executions; f: Force feedback; e: Mean square error w.r.t. desired bearing-formation.

Experiment 1

Next results are from an experiment executed with the 3 real quadrotors whose setup is shown in Figs. 3.12 and 3.14. In this case, the quadrotors are already at the beginning of the experiment close to the desired bearing-formation and the human operator can immediately control their motion. Screenshots of the experiments at three different times ($t_1 = 36\,\mathrm{s}$, $t_2 = 62\,\mathrm{s}$ and $t_3 = 79\,\mathrm{s}$) are shown in Fig. 3.18. There are two main reasons why the quadrotors start close to the desired formation: i) the limited workspace available for the experiment, and ii) the limited FoV of the onboard cameras, which required to choose the desired bearing-formation as well as the starting one such that every UAV is in the visibility range of the others.

In Figs. 3.19a and 3.19c it can be observed again that the average tracking errors of the quadrotor actual velocities ($\dot{\boldsymbol{p}}_{\mathcal{A}_i}$, $\dot{\psi}_{\mathcal{A}_i}$) w.r.t. the reference velocities ($\boldsymbol{u}_i$, w_i) stays approximately zero despite the noisier measurements and all the unmodeled dynamics not taken into account in the analysis and simulation results — see Figs. 3.17a and 3.17c for a comparison. Similarly, Fig. 3.19e shows that the average quadratic bearing error also remains approximately zero during the motion. A bounded error only appears when the human operator commands a rotation of the formation for the reasons explained before, thus confirming again the theoretical analysis and the simulation results. The behavior of the user commands and error vectors ($\boldsymbol{e}_t$, $\boldsymbol{e}_r$) is reported in Figs. 3.19b and 3.19d, respectively. Once again, note the bounded translational error triggered by the rotation command to the whole formation. Finally, Fig. 3.19f shows the control torques ($\boldsymbol{\tau}_t$, $\boldsymbol{\tau}_r$) presented as force cues to the human operator.

Onboard cameras provide the position of the tracked object in their image plane, i.e., an information equivalent to a pair of horizontal/vertical angles (azimuth and elevation). Relative bearings $\boldsymbol{\beta}_{ij}$ can then be computed in terms of relative azimuth ζ_{ij} and elevation η_{ij} as

$$\boldsymbol{\beta}_{ij} = \begin{pmatrix} \cos\eta_{ij}\cos\zeta_{ij} & \cos\eta_{ij}\sin\zeta_{ij} & \sin\eta_{ij} \end{pmatrix}^T . \tag{3.55}$$

Figure 3.20a shows the mean square error between the azimuth/elevation measurements obtained from the onboard cameras and those obtained from the ground-truth. One can verify that the mismatch always stays below 3 deg. During the experiment it was observed a constant temporal lag of 0.5, w.r.t. the ground-truth, as visible in Fig. 3.20b. Nevertheless, the control framework proved to be robust enough to this unmodeled delay.

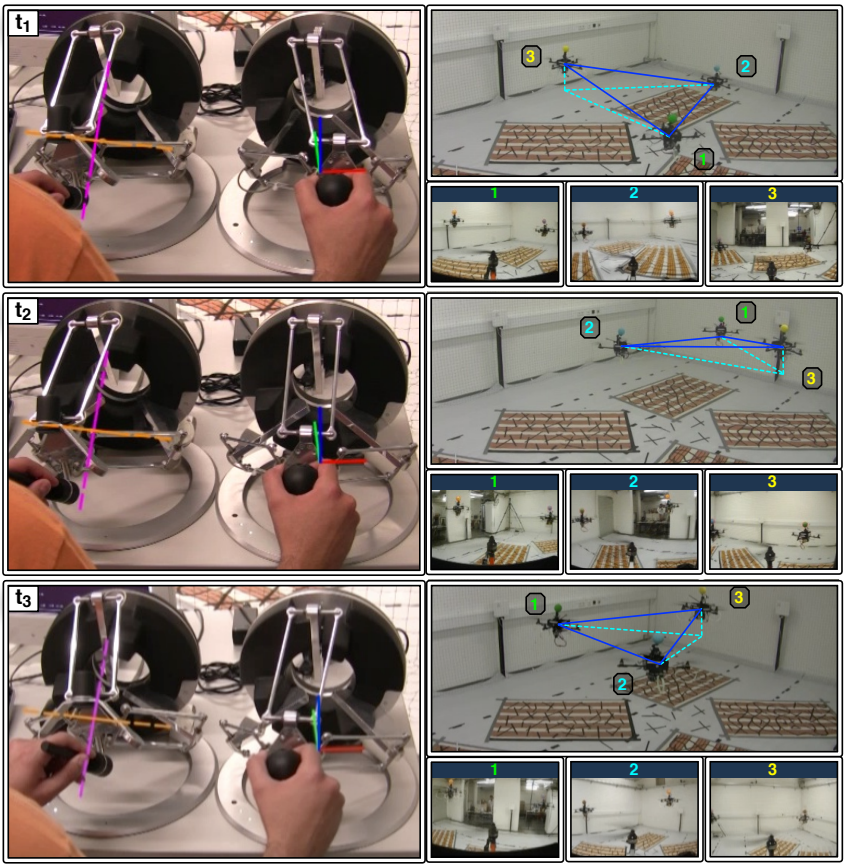
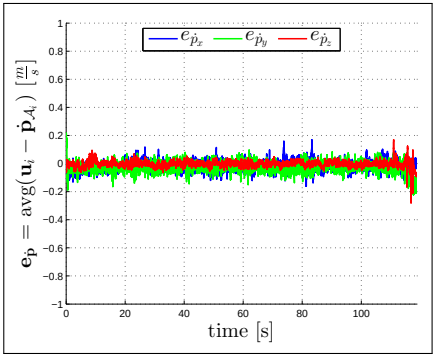
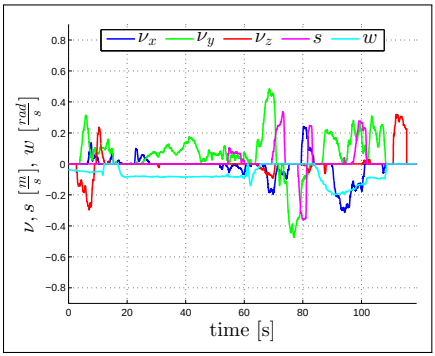


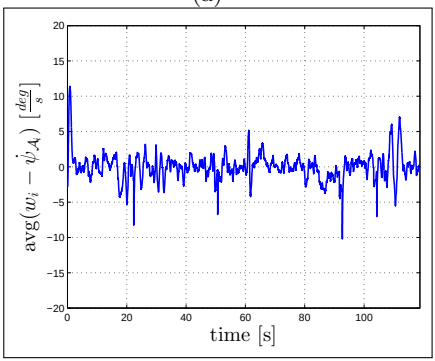
Figure 3.18: Three snapshots of the experiment with 3 quadrotors. Pictures on the left show the haptic interfaces used to command the group motion and receive suitable haptic cues. Pictures on the right report the corresponding external views of the formation and, superimposed, the local views from the onboard cameras of each UAV.



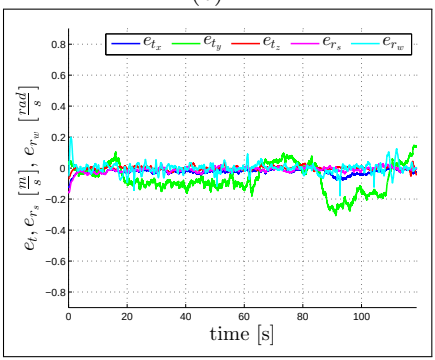
(a)



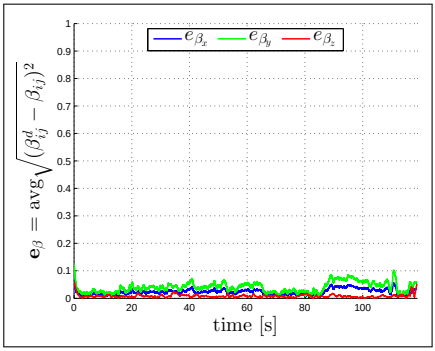
(b)



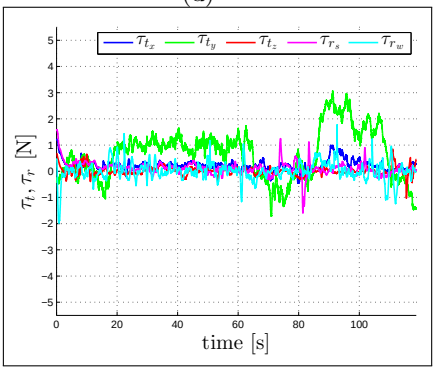
(c)



(d)

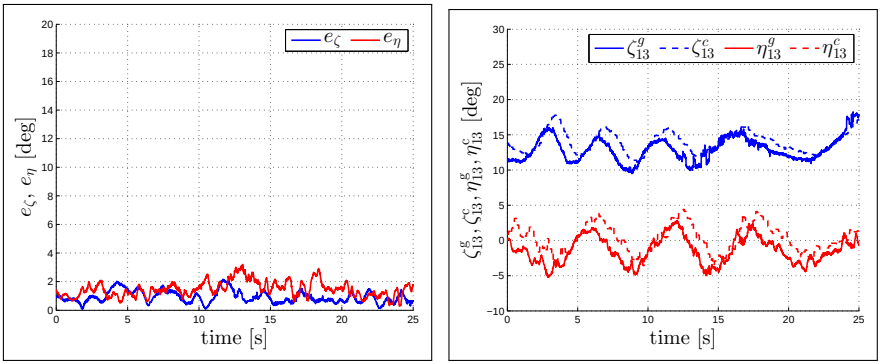


(e)



(f)

Figure 3.19: Results with a group of 3 real quadrotors. a),c): Average velocity and yaw rate tracking errors; b): Operator commands: translation velocity, expansion rate and rotation speed; d): Mismatch between commands and executions; f): Force feedback; e): Mean square error w.r.t. desired bearing-formation.



(a) (b)

Figure 3.20: Experiments: mean square measurement error a) and measurement lag b) when comparing the measures obtained from the on-board cameras with the one computed from the ground-truth.

Environmental Scanning in case of Limited Field-of-View

In the discussion of the previous experiment it was noted that the desired bearing-formations achievable by the UAVs were limited by the finite FoV of the onboard cameras. To overcome the restrictions due to the limited FoV in the horizontal plane, every UAV was forced to rotate with some desired rotation speed $\tilde{\omega}_{i,3}(t)$ in order to 'scan' the environment and periodically detect all the other UAVs. This is recast into the problem of letting the UAVs tracking a *time-varying* desired bearing trajectory without affecting the geometrical shape of the formation, as explained in Sec. 3.6.2.

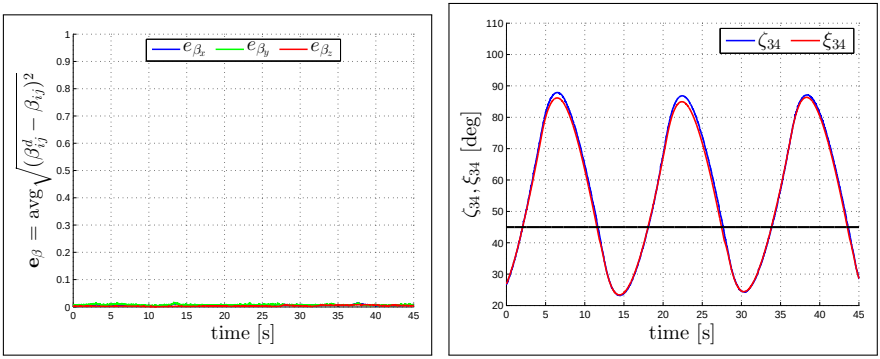
Every agent $i \in (1, \dots, N)$ maintains a local estimation of the azimuth ζ_{ij} relative to another agent j by means of the following dynamical system

$$\dot{\xi}_{ij} = K_\xi(\zeta_{ij} - \xi_{ij}) - \dot{\psi}_{\mathcal{A}_i} \tag{3.56}$$

where $\dot{\psi}_{\mathcal{A}_i}$ is the measured yaw-rate of the UAV, the estimation gain $K_\xi > 0$ if a measurements ζ_{ij} is available, and $K_\xi = 0$ otherwise. This simple estimation scheme will converge if the bearing-formation is kept close enough to the desired one, so that no additional dynamics influence the evolution of ζ_{ij}. For instance, this estimation will not exactly track the real ζ_{ij} whenever a rotation command w is applied with a wrong guess for $\hat{\delta}_{12}$.

Then one can freely design the scan rotation speed $\tilde{\omega}_{i,3}(t)$ in (3.41–3.43) as long as it can guarantee that the relative bearing measurements are acquired often enough to refresh the estimation (3.56). For the tests that will be presented in the following it was chosen the following refreshing strategy: at the beginning of the task, every UAV performs a complete 360° scan to initialize the estimations of all the needed relative azimuths. During normal motion, on the other hand, the i-th UAV will rotate towards the UAV that was not seen for the longest time, say j. The sign of $\tilde{\omega}_{i,3}(t)$ is chosen in order to travel the smallest angle. When j is in the FoV of i, the measurement ζ_{ij} is plugged into (3.56), and the procedure is repeated for another agent k. Depending on the configuration of the formation, this strategy will make some agent to periodically invert the rotation direction, and others to persistently rotate in the same direction. Finally, if a UAV can measure the bearings relative to all the other UAVs, it simply rotates to keep them in the most centered way.

This strategy was first tested in simulation assuming a horizontal FoV of $[-44°, +44°]$. Figure 3.21 shows the results of 7 simulated quadrotors which,

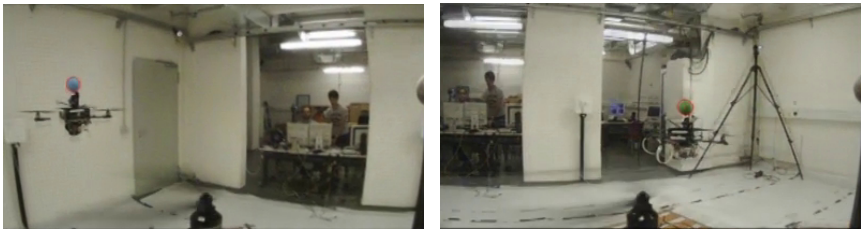


(a) (b)

Figure 3.21: Simulation with 7 quadrotors waving to overcome the limited FoV. a): bearing formation error. b): Estimated azimuth vs. ground-truth measure

starting from the desired bearing formation, are commanded with translations and expansion rates while implementing the estimation law (3.56) with the proposed scanning strategy. Figure 3.21a reports the average quadratic error of the bearing formation computed from the ground-truth position measurements available in simulation. Note how this error is almost zero during the whole simulation, showing that the desired bearing-formation was indeed correctly realized. Figure 3.21b depicts the time behavior of an estimated azimuth (red line) and of the corresponding real value (blue line). One can appreciate how the estimation is able to track the true azimuth value also when the other UAV is out of the FoV (above the horizontal black line in the plot). In this case, the maximum estimation error was around 2 deg and occurred when the other UAV was not in visibility.

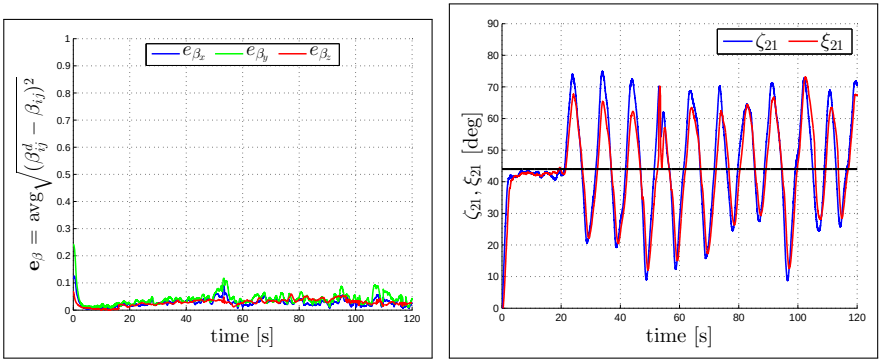
The same scenario was then repeated in an experiment with 3 real quadrotors as before. All the bearing measurements were obtained from the cameras shown in Fig. 3.14b, which have an horizontal FoV of about $[-44°, +44°]$. Figure 3.23a shows again the mean square error of the bearing-formation from the ground-truth data. The error remains very small although the quadrotors are often using the estimated bearings from (3.56). Similarly to before, Fig. 3.23b reports the evolution of an estimated azimuth and the corresponding ground-truth value obtained from the external tracking system. In this case, the estimation degrades when the



(a) (b)

Figure 3.22: Experiment with scanning: screenshots from onboard camera. a) Blue UAV acquired. b) Green UAV acquired.

tracked UAV is outside of the camera FoV (above the horizontal black line in the plot), with a maximum error of about 10 deg. This poorer performance w.r.t. the simulation case is due to all the unmodeled effects and real-world conditions and also to the wrong estimate $\hat{\delta}_{12}$ when commanding a rotation. However, the controller was still able to keep the desired formation despite of these degrading effects.



(a) (b)

Figure 3.23: Experiment with scanning. a): bearing formation error. b): Estimated azimuth vs. ground-truth measure.

3.9 Summary and Possible Extensions

Summarizing, the following results have been presented:

1. It was introduced and rigorously analyzed the concept of *3D bearing-formations*, describing the minimal sets of bearings needed to unequivocally define a bearing-formation with linear cardinality in the number of robots.

2. It was proposed a 3D bearing-formation controller for UAVs based on only relative bearings. The controller is proven to be almost globally convergent, to stabilize the UAV inter-distances to a finite value despite the lack of metric measurements, and to not require any persistency of excitation condition to accomplish this task.

3. The set of generalized velocities of the remaining DoF of a UAV group constrained to maintain a desired bearing-formation was formally characterized and exploited to implement a bilateral teleoperation system which allows the human operator to steer the formation.

4. The framework was validated by means of extensive human/hardware-in-the-loop simulations and real experiments employing a group of quadrotors with onboard cameras and two force-feedback devices.

The formation control could be extended by including robots with different sensors. For example, there could be a small number of robots equipped with expensive sensors and in charge of a specialised aspect of the task. Another possible extension of the framework could be to include an estimation of the actual inter-agent distances by exploiting the steady-state error present during synchronized rotations and due to wrong initial guesses in the quantity δ_{12}. Estimating inter-distances, or in general metric quantities, could also be used for obstacle avoidance purposes. Finally, already developed visual-based techniques using onboard optical flow extraction and decomposition could be included to become fully independent from external tracking devices.

Chapter 4

Shared Planning with Integral Haptic Feedback

The discussion presented in this chapter is based upon the work previously published and presented Franchi et al. (2012b); Masone et al. (2012a,b), accepted Masone et al. (2014a) and submitted Masone et al. (2014b).

4.1 Introduction

Many of the applications considered in RTHL involve mobile robots. In particular, their mobility is often exploited for tasks that require traveling along some predefined routes, for example to perform environmental monitoring (see Smith et al. (2011)), border patrolling (see Girard et al. (2004); Matveev et al. (2010)) or urban structure coverage (see Cheng et al. (2008)). The drawback, discussed already in Sec. 1.3, is that real-world applications are often too challenging to be fully tackled by completely autonomous (mobile) robots. In fact, despite the algorithmic and sensorial advancements, robots are still limited in their world awareness and cognitive capabilities, thus being incapable of making complex decisions in real-world unstructured scenarios. For this reason, safety regulations generally require or suggest the presence of a human co-operator to control of the robot.

These factors have motivated the application of teleoperation frameworks to mobile robots with the integration of some kind of feedback, in order to provide the human with suitable cues that are informative of the state of the robot and of the remote site. For instance, it has been often shown that haptic shared control brings an increase of performance with respect to manual control while still preserving the humans' control skills (see Mulder et al. (2012)). However, letting the human operator directly pilot the robot (robots) during the whole execution of the aforementioned tasks requires

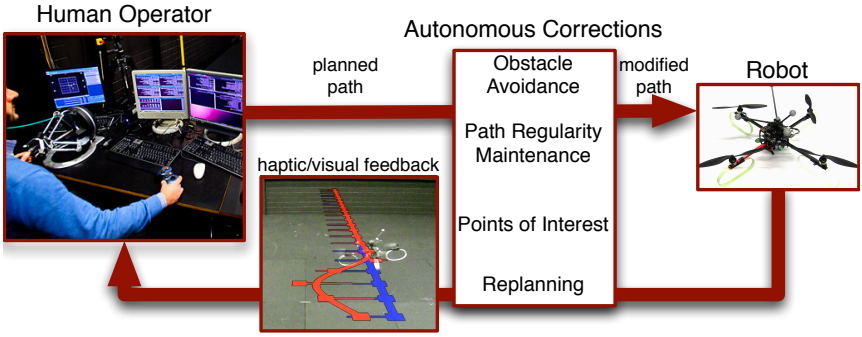


Figure 4.1: Shared planning scheme.

him/her a profound commitment and can result in intense workload and stress level. In view of these considerations, in this chapter it is presented an extension of the shared control framework for mobile robots with multimodal feedback, in which the action of the human is moved at the planning level by letting him/her control online the planned route. The conceptual scheme of this framework is depicted in Fig. 4.1:

Human: The operator modifies online the planned trajectory to be tracked by the mobile robot for a given future *time horizon*. The modification is achieved by changing the planned path.

Robot: The robot autonomously corrects the path specified by the operator in order to avoid collisions, to proven irregularities of the path or to reach important locations;

Feedback: A force-feedback is computed on the the base of the mismatch between the path planned by the human and the one corrected by the robot. A visual feedback using a virtual representation of the environment or an augmented video stream can be used to show the path to the operator.

4.1.1 Related Works

The previous works related to the topics of this paper can be roughly grouped in the following two categories:

Path planning/reshaping

This study relates to the topic of corrective path planning, i.e., adapting in real-time a given path to a new, slightly modified situation. A common solution to this problem is pure replanning, i.e., calculating from scratch a new path. This is generally achieved by using graphs (roadmaps) whose nodes are free-space configurations, and which are built via sampling-based algorithms (see LaValle (2006) for an overview). The two main paradigms for the construction of such graphs, RRT (see e.g., LaValle (1998); LaValle and Kuffner (2001)) and PRM (see e.g., Kavraki et al. (1996)) were originally conceived to be used off-line or for static environments, but with the increase in computational power they have been adapted to on-line usage. The main idea is to add a preprocessing stage that computes an initial roadmap which is then updated on-line when environmental changes occur (see e.g., Leven and Hutchinson (2002), Sa and Corke (2006)). Despite the efficiency improvements, path replanning alone is still not a viable solution for real-time corrections in complex scenarios. A big downfall is that the robot is forced to stop whenever the new path is not calculated in time, and this can severely limit the performances of fast and agile robots.

Reshaping methods are better suited for real-time applications because they only refine locally the path without starting anew. The "elastic bands" or "elastic strips" approaches by Quinlan and Khatib (1993) and Brock and Khatib (2002) fall within this categorization: therein, the sequence of configurations defining the path to be tracked is corrected by obstacle artificial potentials in order to minimize some 'energy' criterium. A descent optimization method is also used in Lamiraux et al. (2004) to determine perturbations of the input function of the system along the admissible directions of motion. Other methods rely on group transformations to modify (segments of) the path, such as Lie group symmetries (in Seiler et al. (2011)) and affine groups transformation (in Pham (2011); Pham and Nakamura (2012)).

Even though reshaping approaches have proven to be effective for online path corrections, they suffer from the *local* nature typical of local methods. The strengths of reshaping and replanning can also be combined by using both methods, as shown in Yoshida and Kanehiro (2006). Nevertheless, none of the aforementioned approaches integrates a human operator in the planning scheme and therefore are limited in terms of cognitive and decisional capabilities.

Bilateral teleoperation of mobile robots

This study is also related to the bilateral shared control paradigm for mobile robots (see Niemeyer et al. (2008)), i.e., shared control with a force feedback returned to the human operator. In the classical bilateral teleoperation framework, the human operator commands with a haptic device the *current*[1] desired state of the robotic system (e.g., the current desired position, velocity, and acceleration). The robot executes the command while retaining some autonomy in order, e.g., to avoid obstacles or other dangers. Finally, the loop is closed by rendering on the haptic feedback a force that is proportional to the mismatch between commanded and executed motion in order to increase the operator's situational awareness. In the recent literature, this paradigm has been successfully applied to mobile robots, as already shown by several examples mentioned in Secs. 1.2 and 3.1. To recall a few examples, in Lee and Spong (2005) a passivity-based approach to bilaterally teleoperate a group of holonomic/non-holonomic ground robots is presented, and in Rodríguez-Seda et al. (2010) bilateral teleoperation of a group of UAVs is realized by directly coupling the position of the haptic device to the position of the formation centroid. This solution does not take into account the *kinematic dissimilarity* between the haptic device and the slave mobile robots (bounded vs. unbounded workspace), which, on the contrary, is explicitly considered in Lee et al. (2011). Other similar approaches have been seen in Lam et al. (2006a,b) and in Franchi et al. (2012c,d).

In comparison to these prior works, the novel framework presented in Chap. 3 constitutes a step towards a higher-level of human steering, because the operator is allowed to command various motions of the whole group of robots and not just translations. The feedback is also presented on the same input channels, i.e., translational, rotational and scaling. Nevertheless, like in all the aforementioned approaches, in this case the robots are still directly 'controlled' by the operator's commands and their autonomy is only exploited to keep some desired geometric formation. Without commands from the operator the formation of robots stays still, therefore the human is required to constantly guide the robots throughout the environment in order to execute a task. In comparison to these prior works, the idea presented in this chapter is to let the robot follow a planned trajectory even without commands from the user. The operator can modify the planned motion of the robot but he/she does not have to directly steer

[1] i.e., the state at the current time or in the very next future.

the robot.

4.2 Preliminaries

This section introduces the problem setting, with models of the robot, of the path and of the environment where the task takes place. The notation adopted follows the guidelines described in the Notation Section.

Robot

The scenario considered for the design of a shared planning framework features a single mobile robot that is tasked to travel along a desired path. The robot is modelled as a rigid body in space and its position in the environment is expressed with respect to an inertial frame denoted with $\mathcal{F}_\mathcal{W} : \{O_\mathcal{W}; \vec{X}_\mathcal{W}; \vec{Y}_\mathcal{W}; \vec{Z}_\mathcal{W}\}$, where $O_\mathcal{W}$ indicates the origin of the frame and $\vec{X}_\mathcal{W}$, $\vec{Y}_\mathcal{W}$, $\vec{Z}_\mathcal{W}$ are its coordinate axes.

In order to keep the formulation general, the robot is simply assumed to possess a characteristic point (output), whose position in $\mathcal{F}_\mathcal{W}$ is denoted as $\boldsymbol{p}_r$, that is capable of traveling with a non-zero speed along any sufficiently smooth regular path. This assumption is easily achievable for flat systems if the trajectory is sufficiently smooth and the initial condition is consistent[2], see e.g Faiz et al. (2001). Exact path following of sufficiently smooth path for flat systems has been proven with very mild conditions (see Faulwasser et al. (2011)). These results make flatness a very useful tool for path planning, because the problem to go from one location in space to another one is reduced to finding a suitable interpolation between two points. The second advantage of this results is that this property is possessed by the large majority of mobile robots, such as aircrafts (see Hauser and Hindman (1997); Khan and Agrawal (2007); Mistler et al. (2001)), wheeled vehicles (see Agrawal and Jin (2003); De Luca et al. (1997)), underwater vehicles (see Fraga et al. (2003)) and others (see Murray et al. (1995)). In the scenario here considered, these results are applicable when the robotic system is differentially flat and the characteristic point $\boldsymbol{p}_r$ is part of its flat output (see Fliess et al. (1995)), or, equivalently, when the robot is feedback linearizable with the characteristic point $\boldsymbol{p}_r$ taken as linearizing output (see Isidori (1995)).

[2]For example, a non-holonomic car-like vehicle must start aligned with the direction of motion De Luca et al. (1997).

Path

In the shared planning framework paths are represented as B-splines (see Biagiotti and Melchiorri (2008)), i.e., as a linear combination of a certain number of suitable basis functions. The use of B-splines is motivated by their generality and versatility, since they can describe or approximate arbitrary functions. Moreover, the relation between the parameters of a B-spline and the shape of the corresponding path is easy to manage and can be exploited to create an intuitive interface for a human operator. Even though the framework here proposed generalizes to $\mathbb{R}^3$, the formulation is henceforth restricted to planar paths in order to simplify the exposition. Nevertheless, the planar case is relevant *per se* in many real-world scenarios, e.g., in the case of a ground robot or of an aircraft monitoring the earth surface while flying at a constant altitude.

The family of planar B-spline curves here considered is described by the function

$$\boldsymbol{\gamma} : \mathbb{R}^{2n} \times S \to \mathbb{R}^2, \tag{4.1}$$

where $S \subset \mathbb{R}$ is a compact set and $\mathbb{R}^{2n}$ is the parameterization domain. A B-spline curve of this family is a function

$$\boldsymbol{\gamma}(\boldsymbol{x}, \cdot) : S \to \mathbb{R}^2, \quad s \mapsto \boldsymbol{\gamma}(\boldsymbol{x}, s) \tag{4.2}$$

that is parameterized by the vector of coplanar control points $\boldsymbol{x} = \begin{pmatrix} \boldsymbol{x}_1^T & \cdots & \boldsymbol{x}_n^T \end{pmatrix}^T \in \mathbb{R}^{2n}$. According to this notation, $\boldsymbol{\gamma}(\boldsymbol{x}, s) \in \mathbb{R}^2$ is a single point of the B-spline curve, i.e., the point obtained by evaluating the function $\boldsymbol{\gamma}(\boldsymbol{x}, \cdot)$ in $s \in S$. Finally, the path corresponding to the B-spline curve $\boldsymbol{\gamma}(\boldsymbol{x}, \cdot)$ is

$$\boldsymbol{\gamma}_S(\boldsymbol{x}) = \{\boldsymbol{\gamma}(\boldsymbol{x}, s) \in \mathbb{R}^2 \,|\, s \in S\}, \tag{4.3}$$

i.e., the set of points obtained by varying the coordinate s within S. Therefore, according to (4.3) the control points $\boldsymbol{x}$ parameterizing the B-spline define the shape of the path $\boldsymbol{\gamma}_S(\boldsymbol{x})$. Figure 4.2 shows an example of path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ given by six control points ($n = 6$).

In order to make the notation less cumbersome, several additional constant parameters have been omitted from definitions (4.1) to (4.3): the degree $\lambda \in \mathbb{N}_{>0}$, which relates to the smoothness of the B-spline curve with respect to the variable s, and the knots $s_1, s_2, \ldots, s_l \in \mathbb{R}$ with $s_1 \leq s_2 \leq \ldots \leq s_l$, that determine the set $S = [s_1, s_l]$ and affect the basis functions of the spline. For compactness, the vector of all the knots

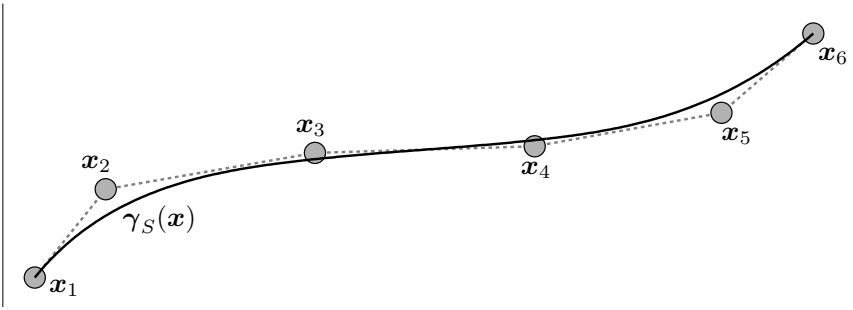


Figure 4.2: Example of B-spline path.

is denoted as $\boldsymbol{s}$. Further details on the parametrization and structure of the B-spline curves (4.2) are given in Sec. 4.3.

Recalling the hypothesis of exact path-following made in the model of the robot, one important requirement to be verified is that $\boldsymbol{\gamma}(\boldsymbol{x}, \cdot)$ is at least a $\mathcal{C}^k$ function ($k > 0$) with respect to both the parameters $\boldsymbol{x}$ and the coordinate s. This requirement can be easily satisfied by choosing a sufficiently high degree for the B-spline, as it will be made clear in Sec. 4.3. However, there is another issue to be accounted which relates to the well known concepts of *singularity* and *regularity* of parametric curves (see Manocha and Canny (1992); Stone and DeRose (1989)). These concepts, in reference to the function $\boldsymbol{\gamma}(\boldsymbol{x}, \cdot)$, are defined as follows:

Definition 4.1. A point $\boldsymbol{\gamma}(\boldsymbol{x}, s)$ with $\boldsymbol{x} \in \mathbb{R}^{2n}$, $s \in S$ and such that $\left.\frac{\partial \boldsymbol{\gamma}}{\partial s}\right|_{(\boldsymbol{x},s)} = (0\ 0)^T \in \mathbb{R}^2$ is called a *singularity* of $\boldsymbol{\gamma}_S(\boldsymbol{x})$. A path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ without singularities is called *regular*.

Namely, a singularity is a point in which the tangent space of the path, i.e., the directions where the motion is allowed, vanishes. Geometrically, this situation could correspond to a cusp or a backtracking in the path, e.g., see Fig. 4.3. Therefore, besides the requirement of sufficient smoothness, it is also necessary to guarantee that $\boldsymbol{\gamma}(\boldsymbol{x}, \cdot)$ is regular in order to avoid that the direction of motion vanishes. This ulterior requirement can be satisfied by appropriately choosing the vector of control points $\boldsymbol{x}$. To clarify this statement, it is first necessary to introduce some notations:

- Let $b_i^\lambda(\boldsymbol{s}, \cdot) : S \to \mathbb{R}$ indicate the basis function associated to the i-th control point $\boldsymbol{x}_i$ (the expression of $b_i^\lambda(\boldsymbol{s}, s)$ will be detailed in Sec. 4.3).
- Define the knots subrange $S_i = \{s \in [s_j,\, s_i), \text{with } j = \max(\lambda - i, 1) : \frac{d\, b_i^\lambda(\boldsymbol{s}, s)}{d\, s} \neq 0\}$.
- Finally, define the following function.

$$\boldsymbol{x}_i^\star(\boldsymbol{x}, \cdot) :\ S_i \to \mathbb{R}^2$$
$$s \mapsto \boldsymbol{x}_i^\star(\boldsymbol{x}, s) :\ \left.\frac{\partial \boldsymbol{\gamma}}{\partial s}\right|_{\left((\boldsymbol{x}_1^T \,\dots\, \boldsymbol{x}_{i-1}^T\ \boldsymbol{x}_i^\star(\boldsymbol{x},s)^T\ \boldsymbol{x}_{i+1}^T \,\dots\, \boldsymbol{x}_n^T)^T, s\right)} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{4.4}$$

Details on the computation of $\boldsymbol{x}_i^\star(\boldsymbol{x}, s)$ are given in Sec. 4.3, where it is also shown that for each $s \in S_i$ the point $\boldsymbol{x}_i^\star(\boldsymbol{x}, s)$ is unique and independent from $\boldsymbol{x}_i$. The restriction of $\boldsymbol{x}_i^\star(\boldsymbol{x}, \cdot)$ to $S_i \subset S$ is due to the fact that outside this range the control point $\boldsymbol{x}_i$ does not affect the tangent $\frac{\partial \boldsymbol{\gamma}}{\partial s}$, as it will be clear in Sec. 4.3.

With these tools it is possible to introduce the concept of *singular curves*, which provides the relation between the control points in $\boldsymbol{x}$ and the presence of singularities in $\boldsymbol{\gamma}_S(\boldsymbol{x})$.

Definition 4.2. Consider a regular path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ of degree $\lambda > 1$, with $S \subset \mathbb{R}$ and $\boldsymbol{x} = (\boldsymbol{x}_1^T\, \boldsymbol{x}_2^T\, \dots\, \boldsymbol{x}_n^T)^T \in \mathbb{R}^{2n}$. The 'singular curve' of the control point $\boldsymbol{x}_i \in \mathbb{R}^2$ is the collection of points $\Omega_i(\boldsymbol{x}) = \{\boldsymbol{x}_i^\star(\boldsymbol{x}, s) \,|\, s \in S_i\}$.

The interpretation of singular curves is twofold. Firstly, a path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ is regular if none of its control points lies on the corresponding singular curve. Secondly, if $\boldsymbol{\gamma}_S(\boldsymbol{x})$ is regular, the singular curve Ω_i describes how the control point $\boldsymbol{x}_i$ can be modified without creating singularities. These concepts are illustrated by the example in Fig. 4.3.

Environment

The environment where the task takes place is populated by static *obstacles* to be avoided and *points of interest* to be reached. Obstacles are modeled as a finite set of $n_{\mathcal{O}} \geq 0$ balls with fixed radius. The position of the center of an obstacle ball in $\mathcal{F}_{\mathcal{W}}$ is denoted as $\boldsymbol{o} \in \mathbb{R}^2$, and the vector with all

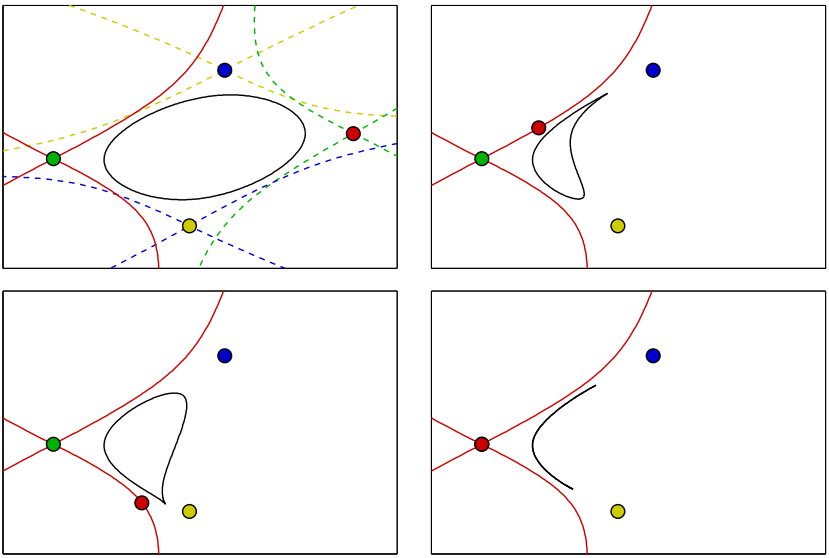


Figure 4.3: Example of a B-spline (black line) of degree $\lambda = 3$, with 4 control points (colored points). By moving one control point, the B-spline is made non-regular. Top-Left: initial regular B-spline and singular curves (colored lines) of the control points (with the same color pattern). The dashed lines are the singular curves of the fixed control points. Other boxes: the B-spline becomes non-regular when one control point (red one) is moved onto its singular curve.

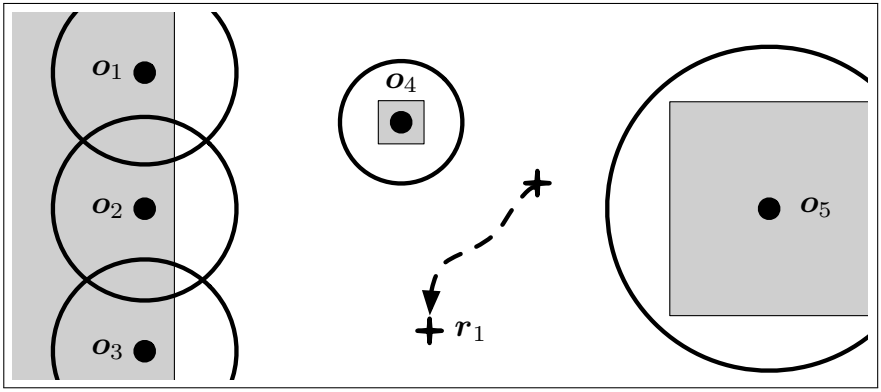


Figure 4.4: Example of environment. The obstacles (grey) are approximated by several obstacle balls of different radii. A single point of interest, $\boldsymbol{r}_1$, is moving in the environment.

the centers is indicated with $\mathcal{O} \in \mathbb{R}^{2\times n_\mathcal{O}}$. The path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ is considered as collision free if it lies outside the obstacle balls. This formulation is quite generic, because any obstacle can be approximated by several balls of various radii (see example of Fig. 4.4). Without loss of generality and to limit the amount of symbols used, hereinafter all obstacle balls $\boldsymbol{o} \in \mathcal{O}$[3] are assumed to have the same radius $R_\mathcal{O}$.

The $n_\mathcal{R} \geq 0$ points of interest (PoIs) represent important locations for the task. For example, they could be fixed stations that allow to exchange data within a limited range, victims and critical locations in search and rescue applications or moving objects to be monitored. The position of a generic PoI in $\mathcal{F}_\mathcal{W}$ is indicated as $\boldsymbol{r} \in \mathbb{R}^2$ and the vector of all PoIs is denoted as $\mathcal{R} \in \mathbb{R}^{2\times n_\mathcal{R}}$. Contrarily to the obstacles, PoIs are not necessarily static (see Fig. 4.4): they could be added or removed by the human operator during the task execution or they can be dynamically generated by an external algorithm such as a dynamic routing strategy Bullo et al. (2011). In this regard, in Sec. 4.8 it is presented an algorithm that dynamically updates the PoIs to facilitate the execution of a coverage task.

[3]Note that $\mathcal{O}$ is equally considered as a *set* of center positions.

4.3 B-Splines

This section briefly recalls some well known properties of B-splines that are instrumental to prove the fundamental results of the proposed framework, and it describes exactly the structure of the specific function $\boldsymbol{\gamma}(\boldsymbol{x}, s)$. The interested reader can refer to Biagiotti and Melchiorri (2008) for a more detailed introduction on the subject.

B-splines are linear combinations of independent polynomial basis functions that are completely parameterized by:

1. A sequence of scalars $s_1, \ldots, s_l$ (*knots* sequence), with $s_j \leq s_{j+1}$ for $j = 1, \ldots, l-1$. The knots sequence determines the pool of basis functions that define the spline.

2. A parameter $\lambda \in \mathbb{N}_{>0}$ (*spline degree*). It is related to the differentiability with respect to s at the knots because the B-spline is $\lambda - k$ times continuously differentiable at a knot with multiplicity k. In the interior of the knot intervals, the B-spline is differentiable infinite times.

3. A vector of planar[4] points $\boldsymbol{x} = [\boldsymbol{x}_1 \ \ldots \ \boldsymbol{x}_n]^T$ (*control points*), with $\boldsymbol{x}_j \in \mathbb{R}^2$ and $j = 1, \ldots, n$, which are the coefficients of the linear combination of basis functions.

It is important to underline that a B-spline path does not necessarily pass through its control points. The control points are the vertices of the so called control polygon (dashed line in Fig. 4.2), that is an approximation of the B-spline. As a rule of thumb, the lower the degree λ, the closer the B-spline path is to the control polygon.

The first important observation that can be derived from the elements introduced so far regards the requirement of smoothness made in Sec. 4.2. To satisfy this requirement, it is sufficient to choose the degree λ such that the function $\boldsymbol{\gamma}(\boldsymbol{x}, s)$ is sufficiently differentiable at the knots. Note also that, as mentioned in Sec. 4.2, here the knots are considered fixed and so it is also their multiplicity.

Given the vector of control points $\boldsymbol{x}$, the evaluation of the B-spline requires to compute the basis functions. For reasons that will be soon explained, this computation has been split in two cases: *open* B-splines and *cyclic* B-splines.

[4]Because the formulation is developed for planar paths.

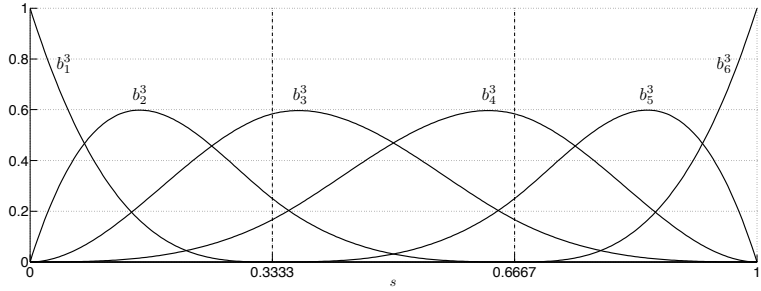


Figure 4.5: Basis functions for an open B-spline with $\boldsymbol{s} = [0\ 0\ 0\ 0\ 0.3333\ 0.6667\ 1\ 1\ 1\ 1]$.

open B-splines The relation between the number l of knots, the number n of control points and the degree λ of the B-spline is

$$l = n - \lambda + 1 \tag{4.5}$$

For the computation of the basis functions, it is introduced the knots vector $\boldsymbol{s}$

$$\boldsymbol{s} = [\underbrace{s_1, \ldots, s_1}_{\lambda+1}, s_2, \ldots, s_{l-1}, \underbrace{s_l, \ldots, s_l}_{\lambda+1}] \tag{4.6}$$

The repetition of the first and last knots is imposed in order to force the B-spline to start exactly at the first control point, $\boldsymbol{x}_1$, and end at the last control point, $\boldsymbol{x}_n$, as explained in Biagiotti and Melchiorri (2008). With this setting, the basis function of degree λ in the knot interval $[s_j,\ s_j + 1]$, with $j = 1, \ldots, n-1$, is denoted as $b_j^\lambda(\boldsymbol{s}, s)$ and it is computed recursively according to the algorithm from De Boor (1972)

$$\begin{aligned} b_j^0(\boldsymbol{s}, s) &= \begin{cases} 1, & \text{if } s_j \leq s < s_{j+1} \\ 0, & \text{otherwise} \end{cases} \\ b_j^\lambda(\boldsymbol{s}, s) &= \frac{s - s_j}{s_{j+\lambda} - s_j} b_j^{\lambda-1}(\boldsymbol{s}, s) + \frac{s_{j+\lambda+1} - s}{s_{j+\lambda+1} - s_{j+1}} b_{j+1}^{\lambda-1}(\boldsymbol{s}, s) \end{aligned} \tag{4.7}$$

An example of basis functions for an open B-spline is illustrated in Fig. 4.5.

cyclic B-splines Traditionally, closed B-splines are generated using (4.5) to (4.7) with the additional constraint $\boldsymbol{x}_1 = \boldsymbol{x}_n$ (the first and last control

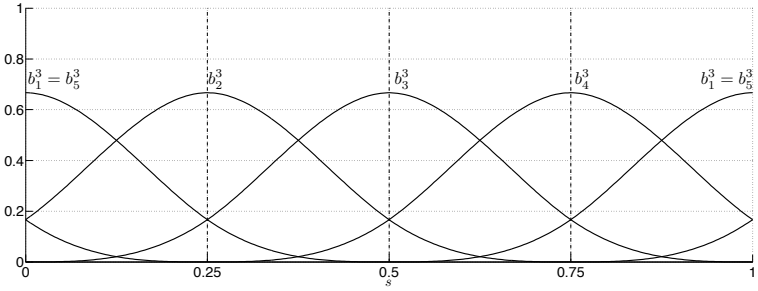


Figure 4.6: Basis functions for a cyclic B-spline with $\boldsymbol{s} = [0\ 0.25\ 0.5\ 0.75\ 1]$.

point are coincident). However, this approach is not suitable for the sought planning strategy for two reasons. Firstly, it does not impose continuity of the derivatives of $\boldsymbol{\gamma}(\boldsymbol{x}, s)$ at the initial and final knot, so additional constraints should be considered. Secondly, this approach is not suitable for online path modifications because a change at the beginning of the $\boldsymbol{\gamma}_S(\boldsymbol{x})$ would not be propagated also at the end, and the other way around. Therefore, the computation of closed (cyclic) B-splines has ben modified as follows.

The relation between the number l of knots, the number n of control points and the degree λ of the B-spline is

$$l = n \geq \lambda \tag{4.8}$$

The knots vector $\boldsymbol{s}$ is

$$\boldsymbol{s} = [s_1, \ldots, s_l] \tag{4.9}$$

Finally, the recursive algorithm (4.7) is modified by replacing all knot subtractions and knot index additions with modulus operations on $[s_1,\ s_l]$ and $[1,\ l]$. An example of basis functions for a cyclic B-spline is illustrated in Fig. 4.5.

A comparison of a closed B-spline generated by an open B-spline with $\boldsymbol{x}_1 = \boldsymbol{x}_n$ and by a cyclic B-spline is showed in Fig. 4.7. Moreover, note that the basis functions (both for open and cyclic B-splines) have the following property (cf. Biagiotti and Melchiorri (2008)):

Remark 6. The basis functions are always nonnegative. Furthermore, for any $s \in [s_1, s_l]$ it is $\sum_{j=1}^{j=n} b_j^{\lambda}(\boldsymbol{s}, s) = 1$.

For both open and cyclic B-splines, the expression of the point $\boldsymbol{\gamma}(\boldsymbol{x}, s)$ of the B-spline curve $\boldsymbol{\gamma}(\boldsymbol{x}, \cdot)$ is given by

$$\boldsymbol{\gamma}(\boldsymbol{x}, s) = \sum_{j=1}^{n} \boldsymbol{x}_j b_j^{\lambda}(\boldsymbol{s}, s) = B_{\boldsymbol{s}}(s)\boldsymbol{x} \tag{4.10}$$

where $B_{\boldsymbol{s}} \in \mathbb{R}^{2\times n}$ is

$$B_{\boldsymbol{s}}(s) = \begin{bmatrix} b_1^{\lambda}(\boldsymbol{s}, s) & 0 & \cdots & b_n^{\lambda}(\boldsymbol{s}, s) & 0 \\ 0 & b_1^{\lambda}(\boldsymbol{s}, s) & \cdots & 0 & b_n^{\lambda}(\boldsymbol{s}, s) \end{bmatrix} \tag{4.11}$$

Consequently, the k-th derivative of the B-spline curve (4.10) with respect to s is

$$\frac{d^k\, \boldsymbol{\gamma}(\boldsymbol{x}, s)}{d\, s^k} = \frac{d^k\, B_{\boldsymbol{s}}(s)}{d\, s^k}\boldsymbol{x} = \sum_{j=1}^{n} \boldsymbol{x}_j \frac{d^k\, b_j^{\lambda}(\boldsymbol{s}, s)}{d\, s^k}, \tag{4.12}$$

where $b_j^{\lambda}(\boldsymbol{s}, s)$ is computed efficiently with the following recursive algorithm

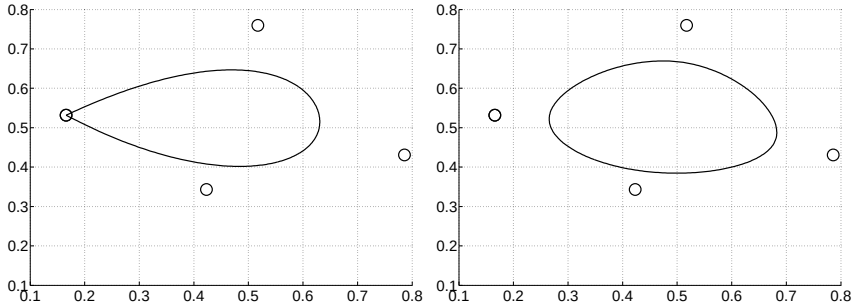
$$\frac{d^k\, b_j^{\lambda}(\boldsymbol{s}, s)}{d\, s^k} = \frac{\lambda!}{(\lambda - k)!} \sum_{i=0}^{k} a_{k,i} b_{j+i}^{\lambda-k} \tag{4.13}$$

with

$$\begin{aligned} a_{0,0} &= 1 \\ a_{k,0} &= \frac{a_{k-1,0}}{s_{j+\lambda-k+1} - s_j} \\ a_{k,i} &= \frac{a_{k-1,i} - a_{k-1,i-1}}{s_{j+\lambda+i-k+1} - s_{j+i}}, \qquad i = 1, \ldots, k-1 \\ a_{k,k} &= \frac{-a_{k-1,k-1}}{s_{j+\lambda+1} - s_{j+k}} \end{aligned}$$

From the implementation point of view, since the basis functions (and their derivatives) depend only on the constant parameters λ and $s_1, \ldots, s_l$, they can be rewritten as polynomials in s whose coefficients can precomputed offline. Therefore, the computational load to evaluate B-spline functions is very limited.

A significant property of B-spline functions (see Biagiotti and Melchiorri (2008)) that will be useful in the design of the framework is the following:



(a) Open B-spline. (b) Cyclic B-spline

Figure 4.7: Comparison of a closed B-spline with $\lambda = 3$, using both open and cyclic formulations. In both cases it is $\boldsymbol{x}_1 = [0.1660\ 0.5138]^T$, $\boldsymbol{x}_2 = [0.5173\ \ 0.7599]^T$, $\boldsymbol{x}_3 = [0.7860\ \ 0.4309]^T$, $\boldsymbol{x}_4 = [0.4232\ \ 0.3432]^T$, $\boldsymbol{x}_5 = \boldsymbol{x}_1$.

Remark 7. On any knot span $[s_i,\ s_{i+1})$, with $i = 1, \ldots, n-1$, at most $\lambda+1$ basis functions are non zero, i.e., $b^\lambda_{i-\lambda}, \ldots, b^\lambda_i$. As a consequence of (4.13) at any knot span $[s_i,\ s_{i+1})$ at most $\lambda + 1$ basis functions derivatives are non zero, i.e., $\frac{d^k\, b^\lambda_{i-\lambda}}{d\, s^k}, \ldots, \frac{d^k\, b^\lambda_i}{d\, s^k}$, with $k = 1, \ldots, \lambda$.

Singular curves With the B-spline representation introduced so far, the computation of $\boldsymbol{x}^\star_i(\boldsymbol{x}, s)$, for $s \in S_i$ (cfr. Definition 4.2), follows from (4.12) with $k = 1$, i.e.,

$$\frac{d\,\boldsymbol{\gamma}(\boldsymbol{x}, s)}{d\, s} = \sum_{j=1,\, j\neq i}^{n} \boldsymbol{x}_j \frac{d\, b^\lambda_j(\boldsymbol{s}, s)}{d\, s} + \boldsymbol{x}_i \frac{d\, b^\lambda_i(\boldsymbol{s}, s)}{d\, s}.$$

Then, by imposing that the left side is $(0\ 0)^T$ and rearranging, it follows

$$\boldsymbol{x}^\star_i(\boldsymbol{x}, s) = -\frac{\sum_{j=1, j\neq i}^{n} \boldsymbol{x}_j \frac{d\, b^\lambda_j(\boldsymbol{s}, s)}{d\, s}}{\frac{d\, b^\lambda_i(\boldsymbol{s}, s)}{d\, s}} \tag{4.14}$$

Observe that i) $\frac{d\, b^\lambda_i(\boldsymbol{s}, s)}{d\, s} \neq 0$ for $s \in S_i$, therefore on S_i $\boldsymbol{x}^\star_i(\boldsymbol{x}, s)$ is always defined and unique, and ii) $\boldsymbol{x}^\star_i(\boldsymbol{x}, s)$ does not depend on the value of $\boldsymbol{x}_i$.

4.4 Overview of the Proposed Framework

Given the problem setting described in Sec. 4.2 and the B-spline structure and properties introduced in Sec. 4.3, now it will be discussed the framework that has been developed for the shared planning.

The simple, but effective, idea behind the framework is to let a human operator influence in real time the shape of the path with the assistance of an autonomous algorithm that corrects, if necessary, the operator's directives so that the path remains regular and satisfies the requirements regarding obstacles and PoIs. Path modifications are realized by introducing a time dependency in $\boldsymbol{x}$, so that $\boldsymbol{\gamma}(\boldsymbol{x}(t), s)$ becomes a time-varying point and $\boldsymbol{\gamma}_S(\boldsymbol{x}(t))$ a time-varying path. Note that, by introducing also a signal $s(t)$, then $\boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))$ would provide the reference trajectory for the robot according to the well-known decoupled design in *path* and *timing law* shown in Kant and Zucker (1986), Peng and Akella (2005). However, the design of a timing-law $s(t)$ is not the focus of this project and in the simulations and experiments presented in Sec. 4.9.2 it was simply used a signal $s(t)$ that keeps the traveling speed small[5] and modulates it with the curvature of the path. The interested reader can however find in literature other algorithms for the generation of $s(t)$, see e.g., Faulwasser et al. (2011), Smith et al. (2012).

The signal $\boldsymbol{x}(t)$ is generated online according to the following dynamical system

$$\dot{\boldsymbol{x}} = N\left(\boldsymbol{u}_h + \boldsymbol{u}_a\right), \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0, \qquad N \in \mathbb{R}^{2n\times 2n} \tag{4.15}$$

where $\boldsymbol{u}_h \in \mathbb{R}^{2n}$ is the control term influenced by the human operator (described in Sec. 4.5), and $\boldsymbol{u}_a \in \mathbb{R}^{2n}$ and $N \in \mathbb{R}^{2n\times 2n}$ are two control terms generated by the autonomous algorithm (described in Sec. 4.6). The initial condition $\boldsymbol{x}_0$ is assumed to define a regular and collision free path and it can be specified by the human operator or by a preliminary planning algorithm tailored for the task at hand[6]. In general the trajectory initialization can be repeated over time, for instance when the robot has almost completed the current trajectory. For example, in Sec. 4.6.2 it is shown how, in presence of obstacles, $\boldsymbol{x}$ can be automatically reinitialized by

[5] For the applications considered, such as monitoring or coverage, the speed of the robot is generally small.

[6] For example, it can be an exploration algorithm planning the next move based on the current partial map, or a coverage method that selects one among predefined curve patterns.

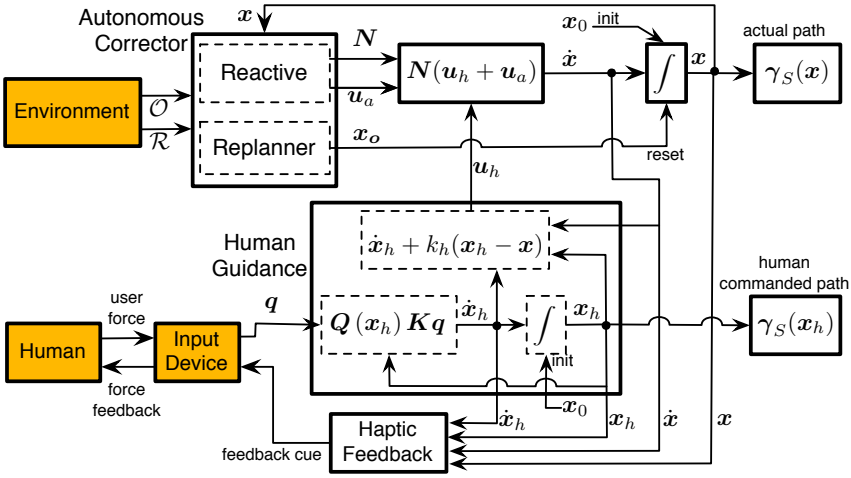


Figure 4.8: Overview of the framework. The signals $\boldsymbol{x}_h$ and $\dot{\boldsymbol{x}}_h$ indicate the desired corrections given by the human (see Sec. 4.5).

a replanning algorithm in order to overcome the limitations of the purely reactive corrections implemented by $\boldsymbol{u}_a$.

The framework, depicted in Fig. 4.8, is organized according to the following circular structure:

Human guidance: It provides the signal $\boldsymbol{u}_h$ in (4.15) that steers the actual path $\boldsymbol{\gamma}_S(\boldsymbol{x}(t))$ towards the desired path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h(t))$, which is modified by the human via an actuated multi-DoF input device.

Autonomous corrector: It consists of two parts: 1. a reactive algorithm that corrects, if necessary, the human commands such that the actual path $\boldsymbol{\gamma}_S(\boldsymbol{x}(t))$ complies with the requirements of staying regular, collision free and attracted by nearby points of interest; 2. a replanner that reinitializes the path in presence of obstacles.

Haptic feedback: It closes the interaction-loop between the human operator and the autonomous correction algorithm in order to increase his/her situational awareness. This is obtained by appropriately controlling the force exerted by the actuated input device, thus producing a haptic feedback that physically informs the operator about the

changes brought by the autonomous correction to his/her suggested modifications to the current path.

The three parts of the framework are explained in detail in Sections 4.5, 4.6 and 4.7 respectively.

4.5 Human Guidance

The human guidance is obtained by making use of an input device with $1 \leq m \leq 2n$ fully-actuated DoF[7]. An example of actuated input device is illustrated in Fig. 4.9. The device is modeled as a generic (gravity pre-compensated) mechanical system

$$M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \boldsymbol{\tau} + \boldsymbol{\tau}_h \tag{4.16}$$

where $\boldsymbol{q} \in \mathbb{R}^m$ is the configuration vector of the device, $M(\boldsymbol{q}) \in \mathbb{R}^{m \times m}$ is the positive-definite and symmetric inertia matrix, $C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in \mathbb{R}^m$ are the Coriolis and centrifugal terms, and $\boldsymbol{\tau}$, $\boldsymbol{\tau}_h \in \mathbb{R}^m$ are the control and human forces, respectively. The computation of $\boldsymbol{\tau}$ is done automatically by the haptic feedback algorithm and it is described in Sec. 4.7. Multiple input devices can also be used at once and in this case $\boldsymbol{q}$, $\boldsymbol{\tau}$ and $\boldsymbol{\tau}_h$ are obtained by stacking in columns the corresponding vectors of each device, while M and C become block diagonal matrices.

The configuration vector $\boldsymbol{q}$ is used by the operator to generate $\boldsymbol{u}_h$ in (4.15) and thus modify the reference path. The connection between $\boldsymbol{q}$ and $\boldsymbol{u}_h$ is provided by an auxiliary vector of control points $\boldsymbol{x}_h \in \mathbb{R}^{2n}$ that evolves according to the following dynamical system

$$\dot{\boldsymbol{x}}_h = Q\left(\boldsymbol{x}_h\right) K \boldsymbol{q}\,, \qquad \boldsymbol{x}_h(0) = \boldsymbol{x}_0, \tag{4.17}$$

where $K \in \mathbb{R}^{m \times m}$ is a diagonal matrix of positive gains and $Q : \mathbb{R}^{2n} \to \mathbb{R}^{2n \times m}$ is a nonlinear mapping. The vector $\boldsymbol{x}_h$ defines a 'virtual' path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ controlled by the operator alone, without any autonomous correction, i.e., $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ is the desired path planned according to the human operator. The term $\boldsymbol{u}_h$ in (4.15) is then designed to steer the actual $\boldsymbol{x}$ towards the desired $\boldsymbol{x}_h$ by implementing a feedforward/proportional-like action, i.e.,

$$\boldsymbol{u}_h = \dot{\boldsymbol{x}}_h + k_h(\boldsymbol{x}_h - \boldsymbol{x}), \tag{4.18}$$

[7] In practice, input devices have at most seven fully-actuated DoF (m) while the number n of control points that specifies a path easily reaches the hundreds even in simple cases.

with $k_h > 0$.

The matrix $Q(\boldsymbol{x}_h)$ in (4.17) determines how the human operator is allowed to interact with the path. Clearly any mapping Q could be used, but for the sake of making this approach usable with proficiency by pilots without extensive training it is fundamental that the effect of mapping (4.17), i.e., how $\boldsymbol{q}$ changes $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$, is intuitive for a human operator. Keeping this in mind, it is desirable to map each DoF (or group of DoF) of the input device to a 'canonical' transformation of the path that can be easily managed by the operator, such as translations, scalings and rotations. Following this idea, Q is designed as the juxtaposition of n_Q elementary matrices $Q_i(\boldsymbol{x}_h) \in \mathbb{R}^{2n \times \nu_i}$ with $i = 1 \dots n_Q$

$$Q(\boldsymbol{x}_h) = \begin{pmatrix} Q_1(\boldsymbol{x}_h) \mid & \dots & \mid Q_{n_Q}(\boldsymbol{x}_h) \end{pmatrix}, \tag{4.19}$$

where $1 \leq n_Q \leq m$, $1 \leq \nu_i \leq m$, and $\sum_{i=1}^{n_Q} \nu_i = m$. Partition (4.19) induces a corresponding partition of $\boldsymbol{q}$

$$\boldsymbol{q} = \begin{pmatrix} \boldsymbol{q}_1^T \mid & \dots & \mid \boldsymbol{q}_l^T \end{pmatrix}^T$$

with $\boldsymbol{q}_i \in \mathbb{R}^{\nu_i}$ for $i = 1 \dots n_Q$. Each $\boldsymbol{q}_i$ is thus mapped through the corresponding elementary matrix $Q_i(\boldsymbol{x}_h)$ to a different canonical transformation of the desired path.

Hereinafter are given three examples of elementary matrices Q_i that map $\boldsymbol{q}_i$ to selected transformations of a subset of control points of the path. The indices of the selected control point are denoted with $\mathcal{X} \subset \{1, \dots, n\}$, and $id_{\mathcal{X}} : \mathcal{X} \to \{0, 1\}$ is the indicator function[8] of $\mathcal{X}$ defined as

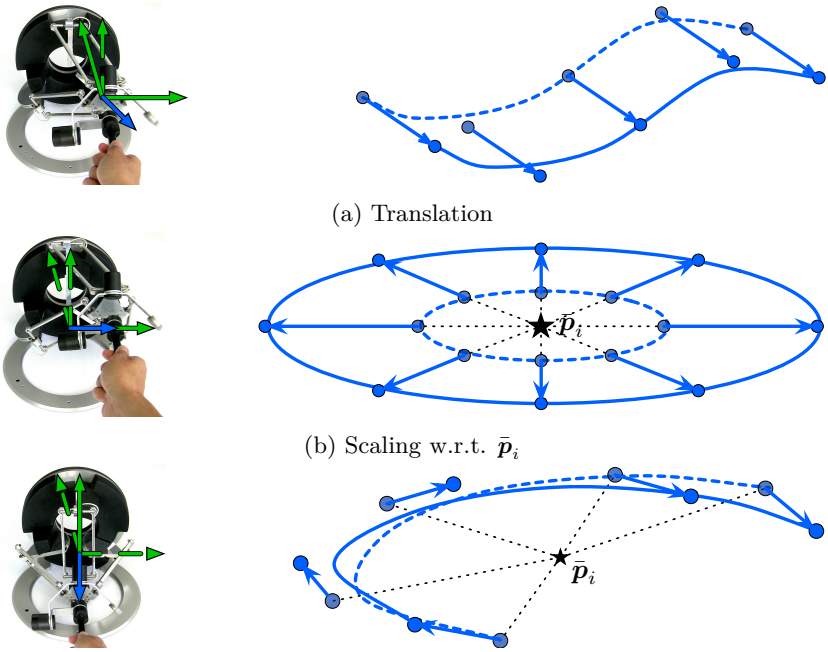
$$id_{\mathcal{X}}(j) := \begin{cases} 1 & \text{if } j \in \mathcal{X} \\ 0 & \text{if } j \notin \mathcal{X} \end{cases}$$

Translation

The configuration $\boldsymbol{q}_i \in \mathbb{R}^2$ is mapped to a translation of the selected subset of control points by the following elementary matrix

$$Q_i(\boldsymbol{x}_h) = \begin{pmatrix} id_{\mathcal{X}}(1) & \cdots & id_{\mathcal{X}}(n) \end{pmatrix}^T \otimes I_2 \tag{4.20}$$

[8] In literature, the indicator function is usually denoted as $\mathbf{1}_{\mathcal{X}}$. The notation here is different to avoid confusion with a vector of ones $\mathbf{1}$.



(a) Translation

(b) Scaling w.r.t. $\bar{p}_i$

(c) Rotation around $\bar{p}_i$

Figure 4.9: Examples of three canonical path transformations applied to different paths and commanded with the same actuated input device. Green arrows represent the DoF. Continuous green arrows indicate the DoF used by the specific transformation: 2 DoF for translation, 1 DoF for scaling, 1 DoF for rotation. Blue arrows represent the commands and corresponding motion of the control points.

where $I_2 \in \mathbb{R}^{2\times 2}$ is the identity matrix and $\otimes$ denotes the Kronecker product of two matrixes. Notice that Q_i does not depend on $\boldsymbol{x}_h$, since the velocity applied to each control point is simply a scaled version of vector $\boldsymbol{q}_i$, without further transformations. If $\mathcal{X} = \{1, \ldots, n\}$ then $\boldsymbol{q}_i$ commands a desired velocity to the whole path (see Fig. 4.9a). Moreover, multiple elementary matrices of the form (4.20) can be juxtaposed in (4.19) to command different translations to different subsets of control points.

Scaling

A single DoF $q_i \in \mathbb{R}$ is mapped to a scaling of the selected subset of control points, i.e., it moves these control points away or towards a given fixed point $\bar{\boldsymbol{p}}_i \in \mathbb{R}^2$. The sought Q_i is

$$Q_i(\boldsymbol{x}_h) = \text{diag}\left(id_{\mathcal{X}}(1)I_2, \cdots, id_{\mathcal{X}}(n)I_2\right)\left(\boldsymbol{x}_h - \mathbf{1}_n \otimes \bar{\boldsymbol{p}}_i\right) \tag{4.21}$$

where $\mathbf{1}_n$ is an n-dimensional column vector of ones and diag$(\cdot)$ denotes a diagonal (or block diagonal) matrix with its argument as the diagonal entries. If $\mathcal{X} = \{1, \ldots, n\}$ then q_i commands a dilation/contraction of the whole path (see Fig. 4.9b). Also in this case multiple elementary matrices of the form (4.21) can be juxtaposed in (4.19) to command, to different subsets of control points, changes of scale with respect to different fixed points.

Rotation

A single DoF $q_i \in \mathbb{R}$ can be used to command a rotation of the subset of the control points around a given fixed point $\bar{\boldsymbol{p}}_i \in \mathbb{R}^2$. The expression of Q_i in this case is

$$Q_i(\boldsymbol{x}_h) = \text{diag}\left(id_{\mathcal{X}}(1)R_{90}, \ldots, id_{\mathcal{X}}(n)R_{90}\right)\left(\boldsymbol{x}_h - \mathbf{1}_n \otimes \bar{\boldsymbol{p}}_i\right), \tag{4.22}$$

where $R_{90} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$. If $\mathcal{X} = \{1, \ldots, n\}$ then q_i commands a rotation of the whole path (see Fig. 4.9c). Also in this case multiple elementary matrices of the form (4.22) can be juxtaposed in (4.19) to command, for different subsets of control points, rotations with respect to different fixed points.

4.6 Autonomous Correction

Having described the interface with the human operator, it will now be discussed the autonomous component of the framework in its two parts.

4.6.1 Reactive Path Deformation

The reactive part of the autonomous corrector is responsible for generating the control terms N and $\boldsymbol{u}_a$ in (4.15). Hereinafter it will be used the notation $\bullet^{(k)}$ to indicate the k-th derivative of a function w.r.t. time, e.g., $\boldsymbol{x}^{(k)}(t) = \frac{d^k \boldsymbol{x}(t)}{d\, t^k}$.

The design of N and $\boldsymbol{u}_a$ must meet several objectives:

Objective 4.1. Suppose that an external algorithm provides a timing law $s(t) \in \mathcal{C}^k$ together with its first k derivatives, and denote with $\boldsymbol{p}(t) = \boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))$ the trajectory tracked by the robot. The trajectory time derivatives $\dot{\boldsymbol{p}}(t), \ddot{\boldsymbol{p}}(t), \ldots, \boldsymbol{p}^{(k)}(t)$ must not be affected by the time derivatives of the curve parameters $\boldsymbol{x}(t)$ at the current $s(t)$.

Objective 4.2. The distance between any obstacle point $\boldsymbol{o} \in \mathcal{O}$ and $\boldsymbol{\gamma}_S(\boldsymbol{x})$ is always greater than $R_{\mathcal{O}}$.

Objective 4.3. The path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ is regular.

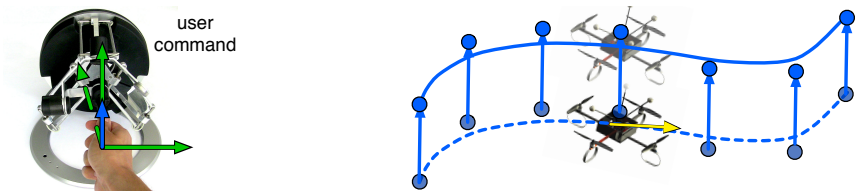
Objective 4.4. The path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ is attracted by every PoI that is closer than $R_{\mathcal{R}}$ to the path itself.[9]

In the following, it is first discussed Objective 4.1 and how this goal is satisfied by suitably choosing the control term N in (4.15). Afterwards, it is designed $\boldsymbol{u}_a$ in order to satisfy the remaining objectives.

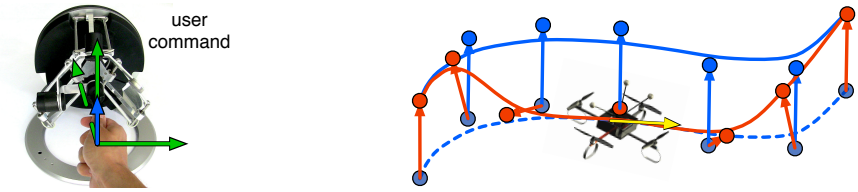
Realization of Objective 4.1

Objective 4.1 is important for preventing that path modifications caused by the exogenous human command $\boldsymbol{u}_h$ in (4.15) may result in an unfeasible reference trajectory for the robot *at its current location on the curve*. This could happen, for example, if the operator abruptly steers the path sideways with respect to the current velocity of the robot (see Fig. 4.10).

[9] Notice how this objective prescribes a qualitative behavior and therefore it intentionally represents a sort of soft constraint for the autonomous corrector.



(a) Without $\boldsymbol{N}$



(b) With $\boldsymbol{N}$

Figure 4.10: User commanding a desired translation (blue arrow) to the path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ (blue line), while the robot is traveling it with nonzero speed (yellow arrows). In this example it is assumed that the control term $\boldsymbol{u}_a$ is null. a) Without the projection matrix N, the actual path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ follows exactly the command, but the resulting motion is unfeasible for the robot. b) When using (4.23) and (4.24), the local geometric properties of $\boldsymbol{\gamma}_S(\boldsymbol{x})$ are preserved and the path translation does not affect the instantaneous motion of the robot.

Secondly, when exploiting the differential flatness of the system for the control design, the computation of the robot inputs requires knowledge of $\dot{\boldsymbol{p}}(t), \ddot{\boldsymbol{p}}(t), \ldots, \boldsymbol{p}^{(k)}(t)$ (see e.g. Mellinger and Kumar (2011) for the case of a quadrotor). However, since the derivatives of $\boldsymbol{u}_h$ are not assumed available, $\ddot{\boldsymbol{x}}(t), \ldots, \boldsymbol{x}^{(k)}(t)$ and, consequently, also $\ddot{\boldsymbol{p}}(t), \ldots, \boldsymbol{p}^{(k)}(t)$ can only be computed when meeting Objective 4.1.

Lastly, Objective 4.1 can simplify the design of external algorithms providing the timing law $s(t)$, since the trajectory derivatives only depend on the derivatives of $s(t)$ and not on the derivatives of $\boldsymbol{x}(t)$. For instance, Objective 4.1 allows to command the robot to remain still just by keeping $s(t)$ constant, regardless of any underlying path modification.

In order to achieve this goal, the control term N in (4.15) is designed as

$$N = I_{2n} - J^{\dagger} J \tag{4.23}$$

where $I_{2n} \in \mathbb{R}^{2n\times 2n}$ is the identity matrix and $J^\dagger$ indicates the Moore-Penrose pseudoinverse of $J \in \mathbb{R}^{2k\times 2n}$, with $k < n$ and J defined as

$$J(\boldsymbol{x}(t), s(t)) = \left(\frac{\partial \boldsymbol{\gamma}}{\partial \boldsymbol{x}}^T \; \frac{\partial}{\partial \boldsymbol{x}}\left(\frac{\partial \boldsymbol{\gamma}}{\partial s}\right)^T \; \cdots \; \frac{\partial}{\partial \boldsymbol{x}}\left(\frac{\partial^k \boldsymbol{\gamma}}{\partial s^k}\right)^T \right)\Bigg|^T_{(\boldsymbol{x}(t),s(t))} \tag{4.24}$$

The Jacobian J relates variations of $\boldsymbol{x}$ to changes of local geometric properties of the path in $s(t)$, such as the position of the point $\boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))$, the tangent vector $\frac{\partial}{\partial s}\boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))$, the curvature vector $\frac{\partial^2}{\partial s^2}\boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))$, and so on.

The matrix N in (4.23) is the well known orthogonal projection matrix in the null-space of J (see Chiaverini et al. (2008)), i.e., it is such that $JN = \mathbf{0}_{2k\times 2n}$. This property gives an intuitive interpretation to the choice of (4.23) and (4.24), and also of its effect on the path $\boldsymbol{\gamma}_S(\boldsymbol{x})$. Namely, this design imposes the invariance of the *local* geometric properties of the path at the current location of the robot regardless of the *global* changes brought by $\boldsymbol{u}_h$ and $\boldsymbol{u}_a$ in (4.15), as illustrated in the example of Fig. 4.10b. This local geometric invariance of the path is beneficial to ease the tracking of the reference trajectory $\boldsymbol{p}(t)$, as experimentally shown by the results in Sec. 4.9.2.

The following property is helpful to characterize the null-space projection matrix N.

Property 4.1. Let $\boldsymbol{\gamma}(\boldsymbol{x}, \cdot)$ be a B-spline of order λ and suppose that the current value of the path coordinate is $s(t) \in [s_i, s_{i+1})$ with $i \in \{1, \ldots, n-1\}$. The range space $\mathfrak{R}(J)$ of J has dimension $\dim(\mathfrak{R}(J)) \leq 2(\lambda+1)$ and the projection of $\boldsymbol{u}_h + \boldsymbol{u}_a$ through N is

$$N\left(\boldsymbol{u}_h + \boldsymbol{u}_a\right) = \begin{pmatrix} I_{2(i-\lambda-1)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & N_\lambda & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & I_{2(n-i)} \end{pmatrix} \left(\boldsymbol{u}_h + \boldsymbol{u}_a\right), \tag{4.25}$$

where $N_\lambda \in \mathbb{R}^{2(\lambda+1)\times 2(\lambda+1)}$ is a projection matrix and $\mathbf{0}$ indicate matrices of zeros with suitable dimensions.

Proof of Property 4.1. The proof Property 4.1 refers to the B-spline structure described in Sec. 4.3. From Remark 7 it follows that only the basis

function derivatives $\frac{d^j\, b_{i-\lambda}^{\lambda}}{d\, s^j}, \ldots, \frac{d^j\, b_i^{\lambda}}{d\, s^j}$ for $j = 0, \ldots, k$ can be not null. Therefore, the expression (4.24) of matrix J becomes

$$J = \left[\begin{array}{c|c|c} \mathbf{0}_{2k \times 2(i-\lambda-1)} & M & \mathbf{0}_{2k \times 2(n-i)} \end{array} \right] \tag{4.26}$$

where $M \in \mathbb{R}^{2k \times 2(\lambda+1)}$ is the following matrix

$$M = \begin{bmatrix} b_{i-\lambda}^{\lambda} & 0 & \cdots & b_i^{\lambda} & 0 \\ 0 & b_{i-\lambda}^{\lambda} & & 0 & b_i^{\lambda} \\ & & \vdots & & \\ \dfrac{d^k\, b_{i-\lambda}^{\lambda}}{d\, s^k} & 0 & & \dfrac{d^k\, b_i^{\lambda}}{d\, s^k} & 0 \\ 0 & \dfrac{d^k\, b_{i-\lambda}^{\lambda}}{d\, s^k} & \cdots & 0 & \dfrac{d^k\, b_i^{\lambda}}{d\, s^k} \end{bmatrix}. \tag{4.27}$$

Note that it was omitted the dependency from $(\boldsymbol{s}, s)$ to have a compact notation. Expression (4.27) proves that $\dim(\Re(J)) \leq 2(\lambda + 1)$.

Introduce now the matrix $N_\lambda = I_{2(\lambda+1)} - M^\dagger M$, with $N_\lambda \in \mathbb{R}^{2(\lambda+1)\times 2(\lambda+1)}$. By substituting (4.26) and (4.27) into (4.23), N has the following structure

$$N = \begin{bmatrix} I_{2(i-\lambda-1)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & N_\lambda & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & I_{2(n-i)} \end{bmatrix}, \tag{4.28}$$

where the terms $\mathbf{0}$ indicate matrices of zeros of appropriate size that complete the non-diagonal blocks of N, which concludes the proof. □

Property 4.1 gives a local characterization to the projection matrix N, because it only affects the velocity of the control points $\boldsymbol{x}_{i-\lambda}, \ldots, \boldsymbol{x}_i$ that (locally) parameterize the shape of the path around $\boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))$, while the rest of the path follows exactly the corrections specified by $\boldsymbol{u}_h$ and $\boldsymbol{u}_a$.

It is now possible to present the first important result which states that the proposed controller fullfills Objective 4.1.

Propositon 4.1. If $\boldsymbol{N}$ in (4.15) is chosen as in (4.23) and (4.24), then the trajectory derivatives $\dot{\boldsymbol{p}}(t), \ddot{\boldsymbol{p}}(t), \ldots, \boldsymbol{p}^{(k)}(t)$ are not functions of the time derivatives of $\boldsymbol{x}(t)$.

Proof. The proof proceeds by writing the trajectory derivatives $\dot{\boldsymbol{p}}, \ddot{\boldsymbol{p}}, \ldots, \boldsymbol{p}^{(k)}$ under the assumption that condition $J\dot{\boldsymbol{x}} = \mathbf{0}_{2n}$, with J defined in (4.24),

is verified. Starting from $\boldsymbol{p}(t) = \boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))$, the first derivative of the trajectory is

$$\begin{aligned}\frac{d\,\boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))}{d\,t} &= \frac{\partial\,\boldsymbol{\gamma}(\boldsymbol{x}, s)}{\partial\,\boldsymbol{x}}\dot{\boldsymbol{x}} + \frac{\partial\,\boldsymbol{\gamma}(\boldsymbol{x}, s)}{\partial\,s}\dot{s} = 0 + \frac{\partial\,\boldsymbol{\gamma}(\boldsymbol{x}, s)}{\partial\,s}\dot{s} \\ &= \dot{\boldsymbol{p}}(\boldsymbol{x}(t), s(t), \dot{s}(t))\end{aligned}$$

where $\frac{\partial\,\boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial\,\boldsymbol{x}}\dot{\boldsymbol{x}} = 0$ results from the initial assumption. Applying the chain rule, the second derivative is

$$\begin{aligned}\frac{d^2\,\boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))}{d\,t^2} &= \underbrace{\frac{\partial}{\partial\,\boldsymbol{x}}\left(\frac{\partial\,\boldsymbol{\gamma}(\boldsymbol{x}, s)}{\partial\,s}\dot{s}\right)\dot{\boldsymbol{x}}}_{=(0\;0)^T} + \sum_{j=1}^{2}\frac{\partial}{\partial s^{(j-1)}}\left(\frac{\partial\,\boldsymbol{\gamma}(\boldsymbol{x}, s)}{\partial\,s}\dot{s}\right)s^{(j)} \\ &= \ddot{\boldsymbol{p}}(\boldsymbol{x}(t), s(t), \dot{s}(t), \ddot{s}(t))\end{aligned}$$

where it was imposed $\frac{\partial}{\partial\,\boldsymbol{x}}\left(\frac{\partial\,\boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial\,s}\right)\dot{\boldsymbol{x}} = 0$ from the initial assumption. Iterating with the chain rule to compute higher order trajectory derivative, at each step the initial assumption $J\dot{\boldsymbol{x}} = \boldsymbol{0}_{2n}$ annihilates the terms containing the partial derivatives w.r.t. $\boldsymbol{x}$, and the i-th derivative, with $1 \leq i \leq k$, becomes

$$\begin{aligned}\frac{d^i\,\boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))}{d\,t^i} &= \underbrace{\frac{\partial}{\partial\boldsymbol{x}}\left(\boldsymbol{p}^{(i-1)}(\boldsymbol{x}, s, \dot{s}, \ldots, s^{(i-1)})\right)\dot{\boldsymbol{x}}}_{=(0\;0)^T} \\ &\quad + \sum_{j=1}^{i}\frac{\partial}{\partial s^{(j-1)}}\left(\boldsymbol{p}^{(i-1)}(\boldsymbol{x}, s, \dot{s}, \ldots, s^{(i-1)})\right)s^{(j)} \\ &= \boldsymbol{p}^{(i)}(\boldsymbol{x}(t), s(t), \dot{s}(t), \ldots, s^{(i)}(t))\end{aligned} \tag{4.29}$$

□

Realization of Objectives 4.2, 4.4 and 4.3

In order to satisfy Objectives 4.2, 4.3 and 4.4, the control term $\boldsymbol{u}_a$ is designed as the sum of three terms

$$\boldsymbol{u}_a = \boldsymbol{u}_{a,\mathcal{O}}(\boldsymbol{x}, \mathcal{O}) + \boldsymbol{u}_{a,\mathcal{I}}(\boldsymbol{x}) + \boldsymbol{u}_{a,\mathcal{R}}(\boldsymbol{x}, \mathcal{R}). \tag{4.30}$$

The three components of $\boldsymbol{u}_a$ are now discussed in detail.

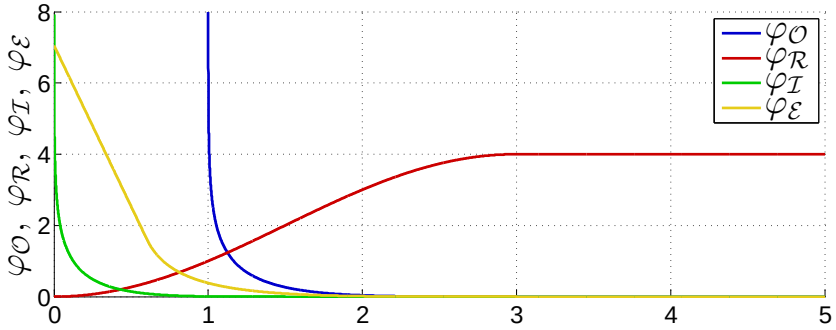


Figure 4.11: Example of the artificial potentials $\varphi_{\mathcal{O}}$, $\varphi_{\mathcal{R}}$ and $\varphi_{\mathcal{I}}$ used to compute $\boldsymbol{u}_a$, and of the potential $\varphi_{\mathcal{E}}$ that is used in Sec. 4.6.2.

Design of $\boldsymbol{u}_{a,\mathcal{O}}$ This term is designed to satisfy Objective 4.2. The expression of $\boldsymbol{u}_{a,\mathcal{O}}$ is

$$\boldsymbol{u}_{a,\mathcal{O}} = -\sum_{o\in\mathcal{O}} \int_S \left(\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial \boldsymbol{x}}^{\dagger} \underbrace{\frac{\partial \varphi_{\mathcal{O}}(\|\boldsymbol{\gamma}(\boldsymbol{x},s)-\boldsymbol{o}\|)}{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}^{T}}_{\dot{\boldsymbol{p}}_{\boldsymbol{o}}(s)} \right)\Bigg|_{\boldsymbol{x}(t)} ds \tag{4.31}$$

where $\varphi_{\mathcal{O}} : \mathbb{R}_{\geq R_{\mathcal{O}}} \to \mathbb{R}_{\geq 0}$ is a smooth distance-based artificial potential function chosen such that

$$\begin{array}{ll} \varphi_{\mathcal{O}} = 0 & \text{if } \|\boldsymbol{\gamma}(\boldsymbol{x},s)-\boldsymbol{o}\| \geq \bar{R}_{\mathcal{O}} \\ \varphi_{\mathcal{O}} \to \infty & \text{if } \|\boldsymbol{\gamma}(\boldsymbol{x},s)-\boldsymbol{o}\| \to R^{+}_{\mathcal{O}} \end{array}$$

with $\bar{R}_{\mathcal{O}} > R_{\mathcal{O}}$. Furthermore, $\varphi_{\mathcal{O}}$ is strictly monotonic in $[R_{\mathcal{O}}, \bar{R}_{\mathcal{O}}]$. An example of potential $\varphi_{\mathcal{O}}$ is depicted in Fig. 4.11.

The meaning of (4.31) is straightforward. For every obstacle $\boldsymbol{o} \in \mathcal{O}$, the artificial potential $\varphi_{\mathcal{O}}$ applies to every point $\boldsymbol{\gamma}(\boldsymbol{x}(t), s)$ of the path a repulsive velocity $-\dot{\boldsymbol{p}}_{\boldsymbol{o}}(s)$ that is directed away from the obstacle, and with intensity growing unbounded as $\boldsymbol{\gamma}(\boldsymbol{x}(t), s)$ approaches the boundary $R_{\mathcal{O}}$ of the obstacle ball. This repulsive velocity is mapped onto the $\mathbb{R}^{2n}$ space of control points by using the pseudo-inverse $\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial \boldsymbol{x}}^{\dagger}$ to invert the relation

$$-\dot{\boldsymbol{p}}_{\boldsymbol{o}}(s) = \frac{d\boldsymbol{\gamma}(\boldsymbol{x},s)}{dt} = \frac{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} + \underbrace{\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial s}}_{=(0\,0)^T}\dot{s}\,.$$

Here, $\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial s}\dot{s}$ is null since s in (4.31) is not a function of time. Finally, the line integral in (4.31) evaluates the effect of the artificial potential over all the points of the path. From a practical standpoint, the analytical expression of (4.31) can be hard to determine and a numerical evaluation of the integral may be needed.

Design of $\boldsymbol{u}_{a,\mathcal{I}}$ This term is designed to satisfy Objective 4.3. The expression of $\boldsymbol{u}_{a,\mathcal{I}}$ is

$$\boldsymbol{u}_{a,\mathcal{I}} = -\sum_{i=1}^{n} \int_{S_i} \frac{\partial \varphi_{\mathcal{I}}(\|\boldsymbol{x}_i - \boldsymbol{x}_i^\star(\boldsymbol{x},s)\|)}{\partial \boldsymbol{x}}^T \Bigg|_{\boldsymbol{x}(t)} ds \tag{4.32}$$

where S_i was introduced in Sec. 4.2 and $\varphi_{\mathcal{I}} : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a smooth distance-based artificial potential function such that

$$\begin{array}{ll} \varphi_{\mathcal{I}} = 0 & \text{if } \|\boldsymbol{x}_i - \boldsymbol{x}_i^\star\| \geq R_{\mathcal{I}} \\ \varphi_{\mathcal{I}} \to \infty & \text{if } \|\boldsymbol{x}_i - \boldsymbol{x}_i^\star\| \to 0^+ \end{array}$$

Furthermore, $\varphi_{\mathcal{I}}$ is strictly monotonic in $[0, R_{\mathcal{I}}]$. An example of $\varphi_{\mathcal{I}}$ is depicted in Fig. 4.11.

The effect of (4.32) is better understood by writing the control term as

$$\boldsymbol{u}_{a,\mathcal{I}} = -\sum_{i=1}^{n}\sum_{j=1}^{n} \begin{pmatrix} \mathbf{0}_{j-1} \\ 1 \\ \mathbf{0}_{n-j} \end{pmatrix} \otimes \int_{S_i} \frac{\partial \varphi_{\mathcal{I}}(\|\boldsymbol{x}_i - \boldsymbol{x}_i^\star(\boldsymbol{x},s)\|)}{\partial \boldsymbol{x}_j}^T \Bigg|_{\boldsymbol{x}(t)} ds. \tag{4.33}$$

From (4.33) and by recalling that $\boldsymbol{x}_i^\star(\boldsymbol{x},s)$ does not depend on $\boldsymbol{x}_i$ (see Sec. 4.3), it is clear that the action of potential $\varphi_{\mathcal{I}}(\|\boldsymbol{x}_i - \boldsymbol{x}_i^\star\|)$ on a the j-th control point $\boldsymbol{x}_j$ is twofold:

- If $j = i$, then the potential applies to $\boldsymbol{x}_i$ a velocity that steers it away from the singular point $\boldsymbol{x}_i^\star(\boldsymbol{x},s)$, $\forall s \in S_i$.
- If $j \neq i$, the potential applies to $\boldsymbol{x}_j$ a velocity such that the singular point $\boldsymbol{x}_i^\star(\boldsymbol{x},s)$ is moved away from $\boldsymbol{x}_i$.

As for the computation of (4.32), the same remarks apply to this case (a numerical evaluation of the integral may be needed in practice).

Design of $\boldsymbol{u}_{a,\mathcal{R}}$ This term is designed to satisfy the 'qualitative' Objective 4.4. The expression of $\boldsymbol{u}_{a,\mathcal{R}}$ is the following

$$\boldsymbol{u}_{a,\mathcal{R}} = -\sum_{\boldsymbol{r}\in\mathcal{R}} \left(\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial \boldsymbol{x}}^{\dagger} \underbrace{\frac{\partial \varphi_{\mathcal{R}}(\|\boldsymbol{\gamma}(\boldsymbol{x},s)-\boldsymbol{r}\|)}{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}^{T}}_{\dot{\boldsymbol{p}}_{\boldsymbol{r}}(\bar{s}_{\boldsymbol{r}})} \right) \Bigg|_{(\boldsymbol{x},\bar{s}_{\boldsymbol{r}})} \tag{4.34}$$

where $\bar{s}_{\boldsymbol{r}}$ indicates the point of $\boldsymbol{\gamma}_S(\boldsymbol{x})$ closest to $\boldsymbol{r}$, i.e., $\|\boldsymbol{\gamma}(\boldsymbol{x},\bar{s}_{\boldsymbol{r}})-\boldsymbol{r}\| = \min_{s\in S}\|\boldsymbol{\gamma}(\boldsymbol{x},s)-\boldsymbol{r}\|$, and $\varphi_{\mathcal{R}} : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a smooth distance-based artificial potential function that is designed such that

$$\begin{aligned} &\varphi_{\mathcal{R}} = 0 && \text{if } \|\boldsymbol{\gamma}(\boldsymbol{x},s)-\boldsymbol{r}\| = 0 \\ &\varphi_{\mathcal{R}} = U_{\mathcal{R}} > 0 && \text{if } \|\boldsymbol{\gamma}(\boldsymbol{x},s)-\boldsymbol{r}\| \geq R_{\mathcal{R}} \end{aligned}$$

Furthermore, $\varphi_{\mathcal{R}}$ is strictly monotonic in $[0, R_{\mathcal{R}}]$ and it has bounded slope that vanishes at 0 and $R_{\mathcal{R}}$. An example of $\varphi_{\mathcal{R}}$ is depicted in Fig. 4.11. Unlike the potential functions $\varphi_{\mathcal{O}}$ and $\varphi_{\mathcal{I}}$, potential $\varphi_{\mathcal{R}}$ is bounded with bounded derivative because reaching the PoIs is only a qualitative goal and it has a lower priority in comparison to maintaining collision avoidance and path regularity.

The structure of (4.34) is similar to that of $\boldsymbol{u}_{a,\mathcal{O}}$ (cf. (4.31)). The artificial potential $\varphi_{\mathcal{R}}$ applies to the point $\boldsymbol{\gamma}(\boldsymbol{x},\bar{s}_{\boldsymbol{r}})$ an attractive velocity $-\dot{\boldsymbol{p}}_{\boldsymbol{r}}(\bar{s}_{\boldsymbol{r}})$ that is directed towards $\boldsymbol{r}$ and has bounded intensity. This attractive velocity is projected on the $\mathbb{R}^{2n}$ space of the control points by the pseudo-inverse $\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial \boldsymbol{x}}^{\dagger}$. Note that the attractive action is not evaluated over the whole path since this is not required by Objective 4.4 and it could tend to collapse portions of $\boldsymbol{\gamma}_S(\boldsymbol{x})$ to a point.

From a practical point of view, $\bar{s}_{\boldsymbol{r}}$ in (4.34) can be computed numerically and the implementation done for the simulations and experiments presented in Sec. 4.9.2 was able to perform this computation (and the computation of all previous integrals) in real time on a standard CPU and for several points of interest.

It is now possible to present the second important result of the proposed controller, by showing that Objectives 4.2 to 4.4 are achieved. Regarding Objective 4.4, it has already been observed that it is only a qualitative goal which does not require the path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ to reach the PoIs. In this regard, $\boldsymbol{u}_{a,\mathcal{R}}$ is constructed to attract $\boldsymbol{\gamma}_S(\boldsymbol{x})$ to nearby PoIs but without guarantee of reaching them. Regarding the accomplishment of Objectives 4.2 and 4.3, the following Proposition is valid.

Propositon 4.2. Suppose that $\boldsymbol{u}_h$ is bounded and that $\|\boldsymbol{u}_h+\boldsymbol{u}_{a,\mathcal{R}}\| \leq \bar{u}$. Then, $\boldsymbol{\gamma}_S(\boldsymbol{x})$ remains collision free and regular.

Proof. The proof relies on the structure of B-splines that is described in Sec. 4.3. Consider the Lyapunov function

$$V(t) := \sum_{\boldsymbol{o}\in\mathcal{O}} \int_S \frac{1}{k(s)} \varphi_{\mathcal{O}}(\|\boldsymbol{\gamma}(\boldsymbol{x},s)-\boldsymbol{o}\|)ds + \sum_{i=1}^{n} \int_{s_{i-\lambda}}^{s_i} \varphi_{\mathcal{I}}(\|\boldsymbol{x}_i - \boldsymbol{x}_i^\star(\boldsymbol{x},s)\|)ds \tag{4.35}$$

where the dependence on time is in $\boldsymbol{x}(t)$ and $k(s) = \sum_{i=1}^{n} b_i^2(\boldsymbol{s},s) > 0$ owing to Remark 6. Therefore $V(t) > 0$ and, since i) $\varphi_{\mathcal{O}} \to \infty$ iff the path is approaching a collision, and ii) $\varphi_{\mathcal{I}} \to \infty$ iff the path is becoming non regular, then $V(t) < \infty$ implies that the path remains regular and collision free.

The time derivative of $V(t)$ is

$$\dot{V}(t) = \underbrace{\left(\sum_{\boldsymbol{o}\in\mathcal{O}} \int_S \frac{\partial \varphi_{\mathcal{O}}}{\partial \boldsymbol{\gamma}} \frac{B_{\boldsymbol{s}}}{k(s)} ds + \sum_{i=1}^{n} \int_{s_{i-\lambda}}^{s_i} \frac{\partial \varphi_{\mathcal{I}}}{\partial \boldsymbol{x}}^T ds \right)}_{\boldsymbol{w}^T = \frac{\partial V}{\partial \boldsymbol{x}}} \dot{\boldsymbol{x}} \tag{4.36}$$

where it was used the fact that $\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial \boldsymbol{x}} = B_{\boldsymbol{s}}(s)$, that $\dot{\boldsymbol{o}} = 0$ (i.e., static obstacles) and that $k(s)$ is not a function of time because the knots $\boldsymbol{s}$ are fixed (see Sec. 4.2).

By injecting (4.15) and (4.30) in (4.36) it follows

$$\dot{V}(t) = \boldsymbol{w}^T N\Big(\underbrace{\boldsymbol{u}_h + \boldsymbol{u}_{a,\mathcal{R}}}_{\boldsymbol{v}} - \underbrace{(-\boldsymbol{u}_{a,\mathcal{O}} - \boldsymbol{u}_{a,\mathcal{I}})}_{\tilde{\boldsymbol{w}}} \Big). \tag{4.37}$$

By comparing (4.36) to (4.31) and (4.32) it is clear that $\tilde{\boldsymbol{w}}$ differs from $\boldsymbol{w}$ only because in $\boldsymbol{u}_{a,\mathcal{O}}$ there is the pseudoinverse $B_{\boldsymbol{s}}(s)^\dagger$ rather than $\frac{B_{\boldsymbol{s}}(s)^T}{k(s)}$. However, from (4.11) and from the definition of $k(s)$ it results that

$$B_{\boldsymbol{s}}(s)^\dagger = (B_{\boldsymbol{s}}(s)^T B_{\boldsymbol{s}}(s))^{-1} B_{\boldsymbol{s}}(s)^T = \frac{B_{\boldsymbol{s}}(s)^T}{k(s)}, \tag{4.38}$$

hence $\tilde{\boldsymbol{w}} = \boldsymbol{w}$ and (4.37) becomes

$$\dot{V}(t) = -\boldsymbol{w}^T N \boldsymbol{w} + \boldsymbol{w}^T N \boldsymbol{v} \tag{4.39}$$

In (4.28) it was proven that N has a particular structure, and this can be exploited (with a possible row rearrangement) to rewrite (4.39) as

$$\dot{V}(t) = -(\boldsymbol{w}_1^T\ \boldsymbol{w}_2^T)\begin{pmatrix} I_{n-\lambda-1} & 0 \\ 0 & N_\lambda \end{pmatrix}\begin{pmatrix} \boldsymbol{w}_1 \\ \boldsymbol{w}_2 \end{pmatrix} + (\boldsymbol{w}_1^T\ \boldsymbol{w}_2^T)\begin{pmatrix} I_{n-\lambda-1} & 0 \\ 0 & N_\lambda \end{pmatrix}\begin{pmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \end{pmatrix} \tag{4.40}$$

where $\boldsymbol{w} = (\boldsymbol{w}_1^T\ \boldsymbol{w}_2^T)^T$ and $\boldsymbol{v} = (\boldsymbol{v}_1^T\ \boldsymbol{v}_2^T)^T$ are the partitions corresponding to the block diagonal structure of N. Matrix $N_\lambda \in \mathbb{R}^{2(\lambda+1)\times 2(\lambda+1)}$ is a projection matrix, thus it only has 0 and 1 as eigenvalues and it is diagonalizable as $N_\lambda = H\Lambda H^T$, where H and H^T are the right and left eigenvectors[10] of N_λ and Λ is the diagonal matrix of the eigenvalues. Without loss of generality, assume that the first elements of the diagonal of Λ are the 1 eigenvalues and indicate $\Lambda = \mathrm{diag}(\Lambda_1\, \Lambda_0)$. With the change of coordinates $H^T\boldsymbol{w}_2 = (\boldsymbol{w}_{\Lambda_1}^T\ \boldsymbol{w}_{\Lambda_0}^T)$ and $H^T\boldsymbol{v}_2 = (\boldsymbol{v}_{\Lambda_1}^T\ \boldsymbol{v}_{\Lambda_0}^T)$, equation (4.40) can be rewritten as

$$\begin{aligned}\dot{V}(t) = &-(\boldsymbol{w}_1^T\ \boldsymbol{w}_{\Lambda_1}^T\ \boldsymbol{w}_{\Lambda_0}^T)\begin{pmatrix} I_{n-\lambda-1} & 0 & 0 \\ 0 & \Lambda_1 & 0 \\ 0 & 0 & \Lambda_0 \end{pmatrix}\begin{pmatrix} \boldsymbol{w}_1 \\ \boldsymbol{w}_{\Lambda_1} \\ \boldsymbol{w}_{\Lambda_0} \end{pmatrix} + \\ &(\boldsymbol{w}_1^T\ \boldsymbol{w}_{\Lambda_1}^T\ \boldsymbol{w}_{\Lambda_0}^T)\begin{pmatrix} I_{n-\lambda-1} & 0 & 0 \\ 0 & \Lambda_1 & 0 \\ 0 & 0 & \Lambda_0 \end{pmatrix}\begin{pmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_{\Lambda_1} \\ \boldsymbol{v}_{\Lambda_0} \end{pmatrix} \\ = &-\boldsymbol{w}_{\Im}^T\boldsymbol{w}_{\Im} + \boldsymbol{w}_{\Im}^T\boldsymbol{v}_{\Im} \leq -\|\boldsymbol{w}_{\Im}\|^2 + \|\boldsymbol{w}_{\Im}\|\bar{u}\end{aligned} \tag{4.41}$$

where $\boldsymbol{w}_{\Im} = (\boldsymbol{w}_1^T\ \boldsymbol{w}_{\Lambda_1})$ and $\boldsymbol{v}_{\Im} = (\boldsymbol{v}_1^T\ \boldsymbol{v}_{\Lambda_1})$ are the components of $\boldsymbol{w}$ and $\boldsymbol{v}$ that are not annihilated by the null space of N and where it was used the assumption $\|\boldsymbol{v}_{\Im}\| \leq \|\boldsymbol{v}\| \leq \bar{u}$ from the statement of Proposition 4.2.

Since $\varphi_{\mathcal{O}}$ and $\varphi_{\mathcal{I}}$ are unbounded with unbounded gradients, one can always find a finite value $M > V(0)$ such that, when $V(t) \geq M$ and if $\|\boldsymbol{w}_{\Im}\| \neq 0$ then $\|\boldsymbol{w}_{\Im}\| \geq \bar{u}$. Hence, $\dot{V}(t) \leq 0$, i.e., $V(t)$ remains bounded and the path remains collision free and regular. □

4.6.2 Generation of Non-homotopic Alternative Paths

The reactive part of the Autonomous Corrector described in Sec. 4.6 ensures that the path is collision free, however the reactive obstacle avoidance also

[10]Since N_λ is symmetric H can be chosen as an orthonormal basis of $\mathbb{R}^{2(\lambda+1)}$.

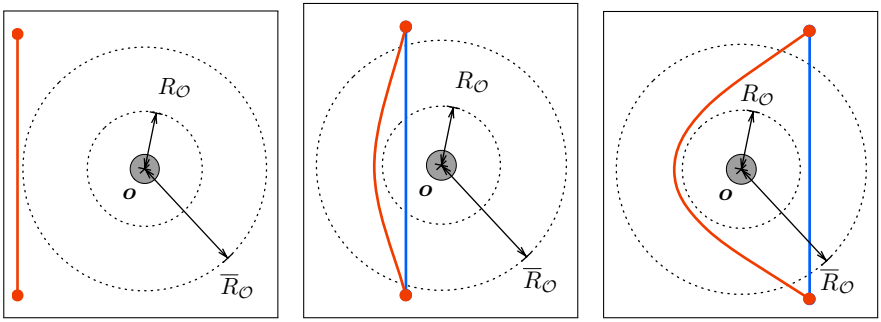


Figure 4.12: a) to c): Sequence showing the deformation of a path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ (red curve) with respect to the desired path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ (blue line) that is moved through an obstacle (from left to right). In the end, the deformed path becomes a suboptimal w.r.t., e.g., a straight line.

prevents $\boldsymbol{\gamma}_S(\boldsymbol{x})$ to 'pass over an obstacle' and this can lead to suboptimal paths (see Fig. 4.12 for an example). This is a well known limitation of reactive planners and it can severely degrade the capability of the human operator to steer the path, especially in a cluttered environment. In order to overcome this problem it is necessary a strategy for generating new *alternative* paths in presence of obstacles. For example, in the elastic strip framework by Brock and Khatib (2002) this is done by allowing the separation of the elastic strip so that it can cross over the obstacle, however the strip cannot be reconnected if the obstacle remains in between.

The replanning method that has been developed for this framework is still based on continuous deformations, but these deformations actively drive the path to the other side of an obstacle to create an alternative route. The underlying idea is that, given an obstacle $\boldsymbol{o}$ and a collision free path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ between two points[11], it is possible to find a new vector of control points $\boldsymbol{x}_{\boldsymbol{o}} \in \mathbb{R}^{2n}$ such that $\boldsymbol{\gamma}_S(\boldsymbol{x}_{\boldsymbol{o}})$ is collision-free, it has the same endpoints of $\boldsymbol{\gamma}_S(\boldsymbol{x}_{\boldsymbol{o}})$, and it is *non-homotopic* (LaValle (2006)) to $\boldsymbol{\gamma}_S(\boldsymbol{x})$ (i.e., it cannot be continuously morphed into $\boldsymbol{\gamma}_S(\boldsymbol{x})$ without intersecting $\boldsymbol{o}$ (see Fig. 4.12)). For each obstacle $\boldsymbol{o} \in \mathcal{O}$, the computation of $\boldsymbol{x}_{\boldsymbol{o}}$ is done in three steps, as detailed hereinafter.

[11]It can also be a portion of the path.

Crossing The step starts when the repulsion applied to $\boldsymbol{\gamma}_S(\boldsymbol{x})$ by $\boldsymbol{o}$ is greater than a predefined threshold $\overline{F} > 0$, i.e.,

$$(\textit{Cond. } \mathcal{C}1) \quad \begin{cases} \left\| \dfrac{\partial \varphi_{\mathcal{O}}(\|\boldsymbol{\gamma}(\boldsymbol{x},\bar{s}) - \boldsymbol{o}\|)}{\partial \boldsymbol{\gamma}(\boldsymbol{x},\bar{s})} \right\| \geq \overline{F} \\ \text{s.t. } \bar{s} = \text{argmin}_{s \in S} \|\boldsymbol{\gamma}(\boldsymbol{x},s) - \boldsymbol{o}\| \end{cases} \tag{4.42}$$

When condition $\mathcal{C}1$ becomes true, let this be at time t_1, vector $\boldsymbol{x_o}$ is initialized and evolved according to

$$\begin{aligned} \dot{\boldsymbol{x}}_{\boldsymbol{o}} &= \left. \frac{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial \boldsymbol{x}} \right|^{\dagger}_{(\boldsymbol{x_o},\hat{s})} G \frac{\boldsymbol{o} - \boldsymbol{\gamma}(\boldsymbol{x},\bar{s})}{\|\boldsymbol{o} - \boldsymbol{\gamma}(\boldsymbol{x},\bar{s})\|} \\ \boldsymbol{x_o}(t_1) &= \boldsymbol{x}(t_1) \end{aligned} \tag{4.43}$$

where $G > 0$ is a parameter and $\boldsymbol{\gamma}(\boldsymbol{x_o},\hat{s})$ is the intersection between $\boldsymbol{\gamma}_S(\boldsymbol{x_o})$ and the segment $\boldsymbol{o} - \boldsymbol{\gamma}(\boldsymbol{x},\bar{s})$. System (4.43) creates $\boldsymbol{\gamma}_S(\boldsymbol{x_o})$ as a copy of $\boldsymbol{\gamma}_S(\boldsymbol{x})$ and then 'pulls' the point $\boldsymbol{\gamma}(\boldsymbol{x_o},\hat{s}) \in \boldsymbol{\gamma}_S(\boldsymbol{x_o})$ along the direction $\frac{\boldsymbol{o}-\boldsymbol{\gamma}(\boldsymbol{x},\bar{s})}{\|\boldsymbol{o}-\boldsymbol{\gamma}(\boldsymbol{x},\bar{s})\|}$ with a force of intensity G (see Fig. 4.13a).

Expansion System (4.43) remains active until $\boldsymbol{\gamma}_S(\boldsymbol{x_o})$ becomes non-homotopic to $\boldsymbol{\gamma}_S(\boldsymbol{x})$ w.r.t. $\boldsymbol{o}$, i.e.,

$$(\textit{Cond. } \mathcal{C}2) \quad \frac{(\boldsymbol{o} - \boldsymbol{\gamma}(\boldsymbol{x},\bar{s}))^T (\boldsymbol{\gamma}(\boldsymbol{x_o},\hat{s}) - \boldsymbol{\gamma}(\boldsymbol{x},\bar{s}))}{\|\boldsymbol{o} - \boldsymbol{\gamma}(\boldsymbol{x},\bar{s})\|^2} \geq 1 + F_c \tag{4.44}$$

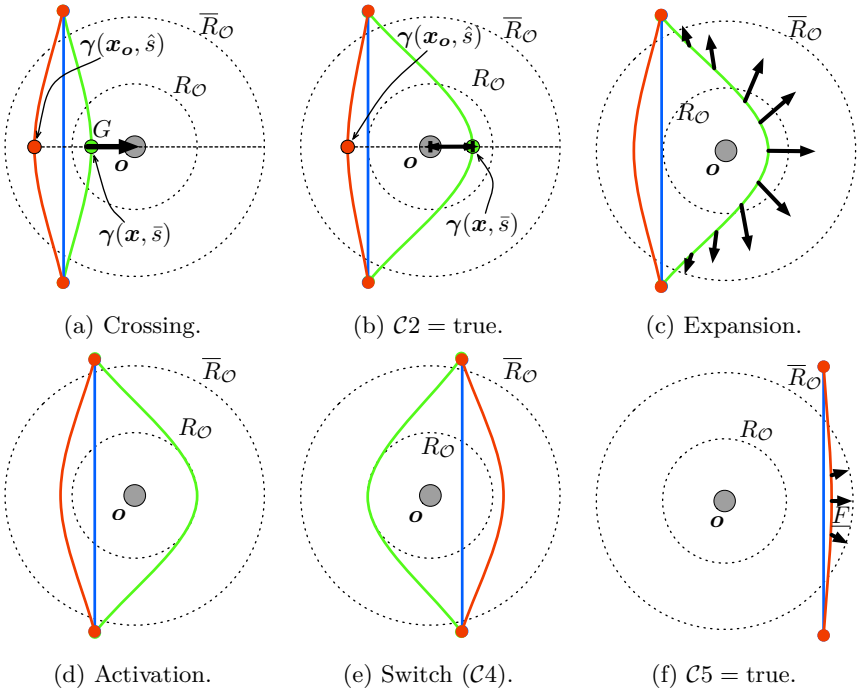
where $F_c > 0$ is a user defined threshold (see Fig. 4.13b). When (4.44) becomes true, the evolution of $\boldsymbol{x_o}$ switches to

$$\dot{\boldsymbol{x}}_{\boldsymbol{o}} = -\int_S \left(\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}{\partial \boldsymbol{x}}^{\dagger} \frac{\partial \varphi_{\mathcal{E}}(\|\boldsymbol{\gamma}(\boldsymbol{x},s) - \boldsymbol{o}\|)}{\partial \boldsymbol{\gamma}(\boldsymbol{x},s)}^{T} \right) \Bigg|_{\boldsymbol{x_o}(t)} ds \tag{4.45}$$

where $\varphi_{\mathcal{E}} : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a smooth distance-based artificial potential function, that is strictly monotonic in $[0, \bar{R}_{\mathcal{O}}]$, with finite slope and such that

$$\begin{aligned} \varphi_{\mathcal{E}} &= 0 \quad \text{if } \|\boldsymbol{\gamma}(\boldsymbol{x_o},s) - \boldsymbol{o}\| \geq \bar{R}_{\mathcal{O}} \\ \varphi_{\mathcal{E}} &\to U \quad \text{if } \|\boldsymbol{\gamma}(\boldsymbol{x_o},s) - \boldsymbol{o}\| \to 0^+ \end{aligned}$$

where $U > 0$ is a fixed parameter (see Fig. 4.11 for an example of $\varphi_{\mathcal{E}}$). System (4.45) 'pushes' $\boldsymbol{\gamma}_S(\boldsymbol{x_o})$ outside the obstacle ball centered in $\boldsymbol{o}$ (see Fig. 4.13c).



(a) Crossing. (b) $\mathcal{C}2 = \text{true}$. (c) Expansion.

(d) Activation. (e) Switch ($\mathcal{C}4$). (f) $\mathcal{C}5 = \text{true}$.

Figure 4.13: Generation of an alternative path: $\boldsymbol{\gamma}_S(\boldsymbol{x_o})$ (green line), $\boldsymbol{\gamma}_S(\boldsymbol{x})$ (red line), $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ (blue line), obstacle $\boldsymbol{o}$ (gray disc). From a) to f), $\boldsymbol{\gamma}_S(\boldsymbol{x})$ is moving from left to right, passing over the obstacle.

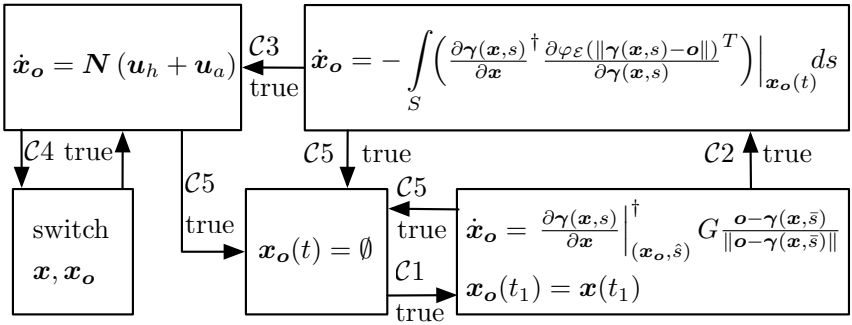


Figure 4.14: Block representation of the replanning algorithm. The algorithm starts from the central block and proceedes as conditions $\mathcal{C}1$ to $\mathcal{C}5$ are met.

Activation When $\boldsymbol{\gamma}_S(\boldsymbol{x}_{\boldsymbol{o}})$ is collision free, i.e.,

$$(\textit{Cond. } \mathcal{C}3) \quad \min_{s \in S} \|\boldsymbol{\gamma}(\boldsymbol{x}_{\boldsymbol{o}}, s) - \boldsymbol{o}\| > R_{\mathcal{O}} \tag{4.46}$$

the evolution of $\boldsymbol{x}_{\boldsymbol{o}}$ changes to (4.15)[12] and $\boldsymbol{\gamma}_S(\boldsymbol{x}_{\boldsymbol{o}})$ is ready to be selected (see Fig. 4.13d). The alternative path $\boldsymbol{\gamma}_S(\boldsymbol{x}_{\boldsymbol{o}})$ becomes active (i.e., $\boldsymbol{x}$ and $\boldsymbol{x}_{\boldsymbol{o}}$ are switched) only when

$$(\textit{Cond. } \mathcal{C}4) \quad \begin{cases} \|\boldsymbol{x}_{\boldsymbol{o}} - \boldsymbol{x}_h\| < \|\boldsymbol{x} - \boldsymbol{x}_h\| \\ \|\boldsymbol{\gamma}(\boldsymbol{x}, s(t)) - \boldsymbol{\gamma}(\boldsymbol{x}_{\boldsymbol{o}}, s(t))\| \simeq 0 \\ \vdots \\ \left\| \dfrac{d^k \boldsymbol{\gamma}(\boldsymbol{x}, s(t))}{d\, t^k} - \dfrac{d^k \boldsymbol{\gamma}(\boldsymbol{x}_{\boldsymbol{o}}, s(t))}{d\, t^k} \right\| \simeq 0 \end{cases} . \tag{4.47}$$

Condition $\mathcal{C}4$ requires that i) $\boldsymbol{x}_{\boldsymbol{o}}$ is closer to $\boldsymbol{x}_h$ than $\boldsymbol{x}$, and ii) the change from $\boldsymbol{x}$ to $\boldsymbol{x}_{\boldsymbol{o}}$ causes no discontinuities in the trajectory tracked by the robot. The switch to the new path is depicted in Fig. 4.13e.

Finally, $\boldsymbol{\gamma}_S(\boldsymbol{x}_{\boldsymbol{o}})$ is deleted from memory if the reaction applied to $\boldsymbol{\gamma}_S(\boldsymbol{x})$ by $\boldsymbol{o}$ becomes sufficiently small, i.e.,

$$(\textit{Cond. } \mathcal{C}5) \quad \max_{s \in S} \left\| \frac{\partial \varphi_{\mathcal{O}}(\|\boldsymbol{\gamma}(\boldsymbol{x}, s) - \boldsymbol{o}\|)}{\partial \boldsymbol{\gamma}(\boldsymbol{x}, s)} \right\| \leq \underline{F} \tag{4.48}$$

[12]Using $\boldsymbol{x}_{\boldsymbol{o}}$ instead of $\boldsymbol{x}$ for the computation of N, $\boldsymbol{u}_h$ and $\boldsymbol{u}_a$.

where $0 < \underline{F} < \overline{F}$ (see Fig. 4.13f) A block representation of the overall system is depicted in Fig. 4.14.

Although the algorithm has been discussed for a single obstacle, it generalizes to multiple obstacles. By taking into account all the $n_{\mathcal{O}}$ obstacles in $\mathcal{O}$, one would have up to $2^{n_{\mathcal{O}}} - 1$ new paths non-homotopic to $\boldsymbol{\gamma}_S(\boldsymbol{x})$ and to each other. In practice, the number of alternative paths considered was limited at once to one per obstacle. Although not complete, this solution resulted very effective due since the generation of an alternative path is very fast, as shown in the results presented in Sec. 4.9.2.

4.7 Haptic Feedback

The haptic feedback algorithm computes the force $\boldsymbol{\tau}$ rendered by the input device (cf. (4.16)) to inform the operator of the discrepancies between the path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ generated by the Human Guidance (see Sec. 4.5) and the actual path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ modified by the Autonomous Corrector (see Sec. 4.6). Recall that these discrepancies are due to the null space projection matrix N and to the reactive terms $\boldsymbol{u}_a$, as illustrated with examples in Fig. 4.15. The force $\boldsymbol{\tau}$ is designed as a function of two haptic cues, $\boldsymbol{e}_{\dot{\boldsymbol{x}}}$ and $\boldsymbol{e}_{\boldsymbol{x}}$, that are described hereinafter.

First haptic cue Analogously to what has been done in Sec. 3.7, the first haptic cue provides a feedback indicating how well the teleoperated system is following the instantaneous motion command given by the human. This is a common approach in the bilateral teleoperation of mobile robots, see e.g., Franchi et al. (2012a,c,d) and Lee et al. (2013). In this case, since the human operator commands the velocity $\dot{\boldsymbol{x}}_h$, the haptic cue $\boldsymbol{e}_{\dot{\boldsymbol{x}}}$ should represent the mismatch between $\dot{\boldsymbol{x}}_h$ itself and the actual velocity $\dot{\boldsymbol{x}}$. However, there are some observations to be made with the help of Fig. 4.16a:

1. The Human Guidance maps a scaled configuration $K\boldsymbol{q} \in \mathbb{R}^m$ of the haptic device to a velocity $\dot{\boldsymbol{x}}_h \in \mathbb{R}^{2n}$ of the control points through the matrix $Q(\boldsymbol{x}_h)$. Note that the map $Q(\cdot)$ depends on the application point and in general it is $Q(\boldsymbol{x}_h) \neq Q(\boldsymbol{x})$, therefore a command applied in $\boldsymbol{x}$ produces a different velocity than a command applied in $\boldsymbol{x}_h$.

2. The velocity $\dot{\boldsymbol{x}}$ computed by Autonomous Corrector is not achieved through the map $Q(\boldsymbol{x})$, i.e. $\dot{\boldsymbol{x}} \notin \mathfrak{R}(Q(\boldsymbol{x}))$, where $\mathfrak{R}(Q(\boldsymbol{x}))$ denotes

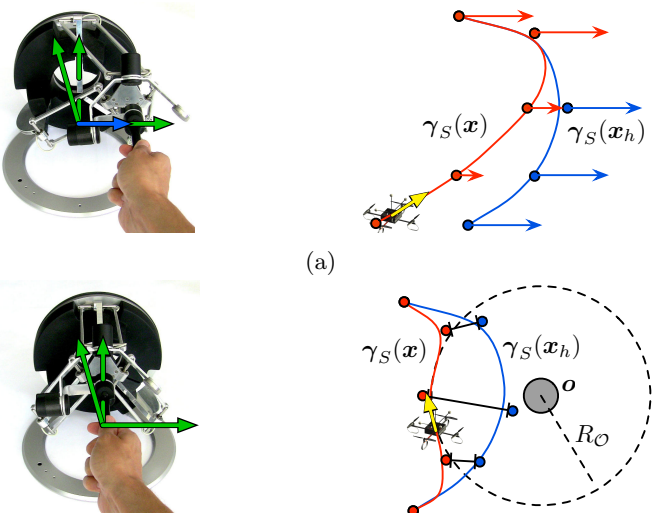


Figure 4.15: Examples of discrepancies between $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ (blue curve) and $\boldsymbol{\gamma}_S(\boldsymbol{x})$ (red curve). a) The projection term N causes the velocity $\dot{\boldsymbol{x}}$ (red arrows) to be different from the commanded $\dot{\boldsymbol{x}}_h$ (blue arrows). b) The reactive correction due to the obstacle $\boldsymbol{o}$ causes a mismatch between $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ and $\boldsymbol{\gamma}_S(\boldsymbol{x})$, even at the equilibrium when $\dot{\boldsymbol{x}}_h = \dot{\boldsymbol{x}} = 0$.

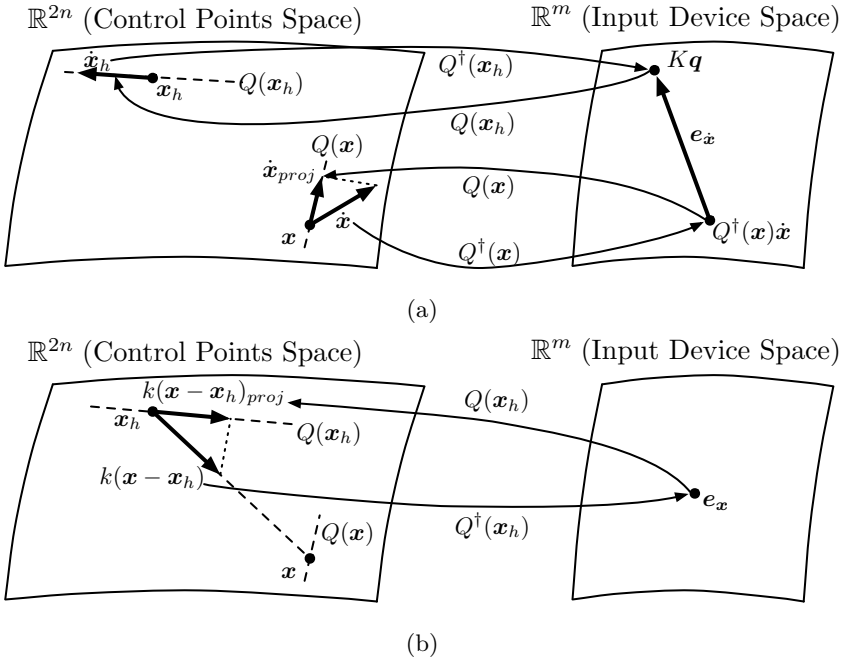


Figure 4.16: Computation of the haptic feedback. a) First haptic cue, $\boldsymbol{e}_{\dot{\boldsymbol{x}}}$. b) Second haptic cue, $\boldsymbol{e}_{\boldsymbol{x}}$.

the range space of $Q(\boldsymbol{x})$. Namely, $\dot{\boldsymbol{x}}$ in general does not correspond to a modification that could be commanded by the human through the input device.

These considerations suggest that the simple difference $\dot{\boldsymbol{x}}_h - \dot{\boldsymbol{x}}$ would not be a meaningful cue, for two reasons. Firstly, $\dot{\boldsymbol{x}}_h$ and $\dot{\boldsymbol{x}}$ are velocity vectors applied to different configurations of the control points and therefore their difference is not representative of how well a command from the human is executed. For example, imagine that $\boldsymbol{\gamma}_S(\boldsymbol{x})$ is exactly a scaled version of $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ and that the operator is commanding a rotation of the path around the center of mass of the control points (using the canonical map (4.22)). In this case, if both $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ and $\boldsymbol{\gamma}_S(\boldsymbol{x})$ rotate with the same angular rate, then the velocity vectors $\dot{\boldsymbol{x}}_h$ and $\dot{\boldsymbol{x}}$ would be different because the control points have to move on circles with different radius. Therefore, this example

shows that the difference $\dot{\boldsymbol{x}}_h - \dot{\boldsymbol{x}}$ does not provide a good information on how well a command from the user is executed. Secondly, in general $\dot{\boldsymbol{x}}_h - \dot{\boldsymbol{x}}$ is not a velocity achievable through the map $Q(\cdot)$, i.e., $(\dot{\boldsymbol{x}}_h - \dot{\boldsymbol{x}}) \notin \Re(Q(\boldsymbol{x}_h))$ and $(\dot{\boldsymbol{x}}_h - \dot{\boldsymbol{x}}) \notin \Re(Q(\boldsymbol{x}))$ (see Fig. 4.16a). This means that the difference $\dot{\boldsymbol{x}}_h - \dot{\boldsymbol{x}}$ cannot be transformed to a meaningful feedback along the directions (DoF) of the input device that are used by the human operator.

For these reasons, the approach adopted here is to first map $\dot{\boldsymbol{x}}_h$ and $\dot{\boldsymbol{x}}$ back onto the space of input device configurations and only afterwards consider their difference. Following this idea, the haptic cue $\boldsymbol{e}_{\dot{\boldsymbol{x}}}$ is

$$\begin{aligned} \boldsymbol{e}_{\dot{\boldsymbol{x}}} &= Q(\boldsymbol{x}_h)^\dagger \dot{\boldsymbol{x}}_h - Q(\boldsymbol{x})^\dagger \dot{\boldsymbol{x}} \\ &= K\boldsymbol{q} - Q(\boldsymbol{x})^\dagger \dot{\boldsymbol{x}} \end{aligned} \tag{4.49}$$

with $Q(\cdot)^\dagger = (Q(\cdot)^T\, Q(\cdot))^{-1} Q(\cdot)^T$. Observe that:

Property 4.2. $Q(\boldsymbol{x})^\dagger$ is the mapping onto the space of input device configurations of the orthogonal projection of $\dot{\boldsymbol{x}}$ on $\Re(Q(\boldsymbol{x}))$.

Proof. The statement is a simple result of linear algebra (Meyer (2001)). The well known orthogonal projection operator onto $\Re(Q(\boldsymbol{x}))$ is the matrix

$$Q(\boldsymbol{x})(Q(\boldsymbol{x})^T\, Q(\boldsymbol{x}))^{-1} Q(\boldsymbol{x})^T \equiv Q(\boldsymbol{x})Q(\boldsymbol{x})^\dagger,$$

therefore the orthogonal projection of $\dot{\boldsymbol{x}}$ on $\Re(Q(\boldsymbol{x}))$ is $\dot{\boldsymbol{x}}_{proj} = Q(\boldsymbol{x})Q(\boldsymbol{x})^\dagger \dot{\boldsymbol{x}}$. Thus, the mapping of $\dot{\boldsymbol{x}}_{proj}$ on the space of input device configurations is

$$Q(\boldsymbol{x})^\dagger \dot{\boldsymbol{x}}_{proj} = Q(\boldsymbol{x})^\dagger Q(\boldsymbol{x})Q(\boldsymbol{x})^\dagger \dot{\boldsymbol{x}} = Q(\boldsymbol{x})^\dagger \dot{\boldsymbol{x}}.$$

□

In other words, Property 4.2 means that, since $\dot{\boldsymbol{x}}$ cannot be brought back to any command of the user (or equivalently to a configuration of the input device), then it is taken the closest velocity $\dot{\boldsymbol{x}}_{proj}$ that represents a meaningful command.

Second haptic cue The first cue, $\boldsymbol{e}_{\dot{\boldsymbol{x}}}$, has been designed to represent how well $\dot{\boldsymbol{x}}$ follows the signal $\dot{\boldsymbol{x}}_h$ given by the human. However, this cue does not capture if and how much $\boldsymbol{\gamma}_S(\boldsymbol{x})$ differs from $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$. For example, imagine that the operator steers the desired path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ towards an obstacle, so

that the actual $\boldsymbol{\gamma}_S(\boldsymbol{x})$ results modified by the reactive corrections but not reinitialized by the replanner. In this situation, if the operator gives no commands anymore, when the actions of $\boldsymbol{u}_{a,\mathcal{O}}$ and $\boldsymbol{u}_h$ counteract each other the error $\boldsymbol{e}_{\dot{\boldsymbol{x}}}$ would vanish and therefore there would be no feedback even though $\boldsymbol{\gamma}_S(\boldsymbol{x}) \neq \boldsymbol{\gamma}_S(\boldsymbol{x})$ (see example in Fig. 4.16b). This phenomenon is due to the integration of the signals $\dot{\boldsymbol{x}}_h$ and $\dot{\boldsymbol{x}}$ that is done within the framework (cf. Fig. 4.8). In order to account for this integration the second haptic cue $\boldsymbol{e}_{\boldsymbol{x}}$ is chosen to represent the mismatch between $\boldsymbol{x}_h$ and $\boldsymbol{x}$.

The cue $\boldsymbol{e}_{\boldsymbol{x}}$ has been designed with the goal of providing a force feedback that 'guides' the operator so that his/her commands reduce the mismatch between $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ and $\boldsymbol{\gamma}_S(\boldsymbol{x})$. For instance, in the example of Fig. 4.16b a force feedback towards the left should be rendered on the input device so that the desired path is steered away from the obstacle and towards $\boldsymbol{\gamma}_S(\boldsymbol{x})$. To achieve this result, it is taken as cue the velocity vector $k(\boldsymbol{x} - \boldsymbol{x}_h)$, with $k > 0$, which drives $\boldsymbol{x}_h$ towards $\boldsymbol{x}$ (see Fig. 4.16b). Analogously to the previous discussion of $\boldsymbol{e}_{\dot{\boldsymbol{x}}}$, also in this case $k(\boldsymbol{x} - \boldsymbol{x}_h)$ in general does not correspond to a command achievable through the map $Q(\boldsymbol{x}_h)$, therefore it is mapped on the space of input device configurations by $Q(\boldsymbol{x}_h)^\dagger$. This yields

$$\boldsymbol{e}_{\boldsymbol{x}} = kQ(\boldsymbol{x}_h)^\dagger(\boldsymbol{x} - \boldsymbol{x}_h). \tag{4.50}$$

Force feedback The force $\boldsymbol{\tau}$ corresponding to the two haptic cues $\boldsymbol{e}_{\dot{\boldsymbol{x}}}$ and $\boldsymbol{e}_{\boldsymbol{x}}$ is

$$\boldsymbol{\tau} = -B\dot{\boldsymbol{q}} - K_M\boldsymbol{q} - K^*(\boldsymbol{e}_{\dot{\boldsymbol{x}}} - \boldsymbol{e}_{\boldsymbol{x}}) \tag{4.51}$$

where B is a positive definite damping matrix used to stabilize the device, K_M is a diagonal non-negative matrix used to provide a perception of the distance from the zero-commanded velocity[13], and K^* a diagonal positive definite matrix of gains. As in all bilateral teleoperation applications, the presence of the force feedback $\boldsymbol{\tau}$ may cause unstable behaviors of the haptic interface because of non-modeled dynamics, communication delays and packet losses. In order to guarantee stability despite all these shortcomings, like in Sec. 3.7, it is adopted the PSPM approach by Lee and Huang (2010) to guarantee stability (passivity) of the master side and of the closed-loop teleoperation system. Let $\bar{\boldsymbol{z}}[k]$ be the PSPM version of the following signal

$$\boldsymbol{z} = Q(\boldsymbol{x})^\dagger\dot{\boldsymbol{x}} + kQ(\boldsymbol{x}_h)^\dagger(\boldsymbol{x} - \boldsymbol{x}_h), \tag{4.52}$$

[13]If this effect is not desired, one can alway disable it by taking $K_M = 0$.

that is sampled and sent from the mobile robot to the haptic interface through the (possibly non-ideal) communication channel. Exploiting the PSPM action, the final *passive* implementation of τ in (4.51) then becomes

$$\tau = -B\dot{\boldsymbol{q}} - K_M\boldsymbol{q} - K^*\left(K\boldsymbol{q} - \bar{\boldsymbol{z}}[k]\right). \tag{4.53}$$

This is sufficient for guaranteeing stability (passivity) of the bilateral system assuming that the human operator behaves as a passive system (see Lee and Huang (2010)).

4.8 Coverage Task with Human-in-the-loop

Many applications discussed in Sec. 1.1, such as environmental monitoring, dusting, lawn mowing and patrolling, require the robot to persistently move in a compact environment $\mathcal{A} \subset \mathbb{R}^2$. The persistent motion in these *coverage* tasks is normally achieved by letting the robot circulate on a suitable[14] closed path (see e.g., Smith et al. (2012)), however the path is generally considered fixed. In this section it will be shown how, besides ensuring the Objectives while tracking the human commands, the Autonomous Corrector can be exploited to modify in real-time the coverage path, e.g., in response to environmental changes or to improve the performance with respect to the initial path. A naive idea to tackle this problem could be to steer the control points where the coverage is more required, however the path could transit far away from its control points (e.g., see Fig. 4.3). Instead, the approach adopted here is to design an algorithm that autonomously steers a set of PoIs that, in turn, modify the path.

Before designing the algorithm it is necessary to model the coverage task. It is assumed that the robot executes the task by means of a device[15] with a finite circular footprint $\mathcal{P}(\boldsymbol{p}_r(t), R) = \{\boldsymbol{a} \in \mathcal{A} : \|\boldsymbol{a} - \boldsymbol{p}_r(t)\| \leq R\}$, where $\boldsymbol{p}_r(t) \in \mathcal{A}$ is the position of the robot and $R > 0$ is the footprint radius. The coverage state of $\mathcal{A}$ is modelled as a time varying scalar field

$$C : \mathcal{A} \times \mathbb{R}_{\geq 0} \to [0, 1], \quad \text{s.t.} \quad (\boldsymbol{a}, t) \mapsto c$$

[14] In a monitoring task the path is chosen to cover locations where the data or material to be collected is expected to be more significant.

[15] It could be, e.g., a sensor, like a down-facing or omnidirectional camera, or a physical effector, like a tool to take samples of the environment or a rotating brush used to clean the floor.

where $C(\boldsymbol{a}, t) = 1$ indicates perfect coverage and $C(\boldsymbol{a}, t) = 0$ indicates no coverage in $\boldsymbol{a}$ at the time t. The field C evolves according to the following dynamical system

$$\dot{C}(\boldsymbol{a}, \boldsymbol{p}_r, t) = \begin{cases} -\alpha, & \text{if } \boldsymbol{a} \notin \mathcal{P}(\boldsymbol{p}_r, R) \wedge C(\boldsymbol{a}, \boldsymbol{p}_r, t) > 0 \\ \beta \frac{R - \|\boldsymbol{a} - \boldsymbol{p}_r\|}{R}, & \text{if } \boldsymbol{a} \in \mathcal{P}(\boldsymbol{p}_r, R) \wedge C(\boldsymbol{a}, \boldsymbol{p}_r, t) < 1 \\ 0 & \text{otherwise.} \end{cases} \tag{4.54}$$
$$C(\boldsymbol{a}, \boldsymbol{p}_r, 0) = 0$$

where $\alpha, \beta > 0$ are parameters assumed to be measured or estimated from the sensor/tool and from the environment. According to (4.54),

1. At each point $\boldsymbol{a} \notin \mathcal{P}$, the field decreases with a constant rate $-\alpha < 0$, for example because the information at $\boldsymbol{a}$ is becoming old or some physical quantity is consumed/accumulated.

2. At each point $\boldsymbol{a} \in \mathcal{P}$, the field increases with a non-uniform rate that represents a degradation effect at the edge of $\mathcal{P}(\boldsymbol{p}_r(t), R)$.

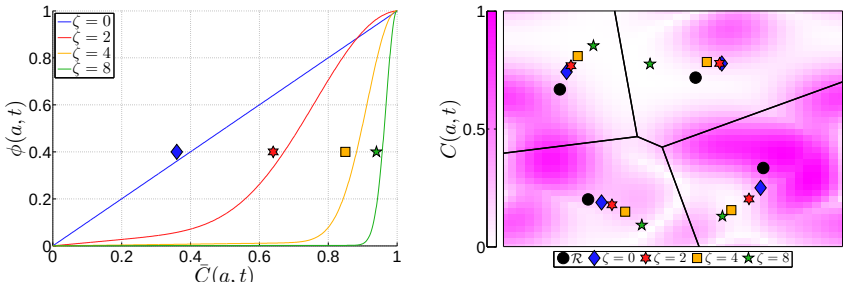
With this setting, the goal is now to design an algorithm that steers the PoIs where the field C is low. The sought algorithm has been developed according to the well known iterative method called Lloyd's algorithm or Voronoi iteration (see Cortés et al. (2004)), that is used to deploy robotic networks. The algorithm starts by initializing $\mathcal{R}$ with a uniform distribution over $\mathcal{A}$ and such that $\boldsymbol{r} \neq \boldsymbol{r}'$, $\forall \boldsymbol{r}, \boldsymbol{r}' \in \mathcal{R}$. Afterwards, each iteration consists of three steps:

1) Voronoi partitioning Given the current $\mathcal{R}$, compute the Voronoi partition (or tessellation) of $\mathcal{A}$. Each one of the $n_{\mathcal{R}}$ Voronoi cells contains a different PoI and in particular $V_{\boldsymbol{r}}(\mathcal{R})$ is the Voronoi cell associated to $\boldsymbol{r} \in \mathcal{R}$.

2) Centroids computation In the second step the algorithm computes the centroid of each cell. For this computation it is used the density function $\phi_\varsigma(\boldsymbol{a}, t) : \mathcal{A} \times \mathbb{R}_{\geq 0} \to [0, 1]$ defined as

$$\phi(\boldsymbol{a}, t) = e^{-\varsigma} \bar{C}(\boldsymbol{a}, t) + (1 - e^{-\varsigma}) \sin\left(\frac{\pi(\bar{C}(\boldsymbol{a}, t))}{2}\right)^{e^{\varsigma}} \tag{4.55}$$

where $\bar{C}(\boldsymbol{a}, t) = 1 - C(\boldsymbol{a}, t)$ and $\varsigma \geq 0$ is a free parameter. The use of $\bar{C}$ instead of C stems from the need of directing the points of interest



(a) Density function. (b) Centroids computation.

Figure 4.17: Effect of the parameter ς on the density function (4.55) and on the computation of the centroids (4.56).

towards low-coverage zones. If $\varsigma = 0$ then it is exactly $\phi(\boldsymbol{a}, t) = \bar{C}(\boldsymbol{a}, t)$. By increasing the parameter ς it is possible to amplify the density of the points where the coverage is lower and reduce the density of the points where the coverage is higher (see Fig. 4.17).

The centroid of $V_{\boldsymbol{r}}(\mathcal{R})$ is then given by

$$\bar{\boldsymbol{a}}_{\boldsymbol{r}}(t) = \frac{1}{\int_{V_{\boldsymbol{r}}(\mathcal{R})} \phi(\boldsymbol{a}, t)\, d\boldsymbol{a}} \int_{V_{\boldsymbol{r}}(\mathcal{R})} \boldsymbol{a}\, \phi(\boldsymbol{a}, t)\, d\boldsymbol{a}. \tag{4.56}$$

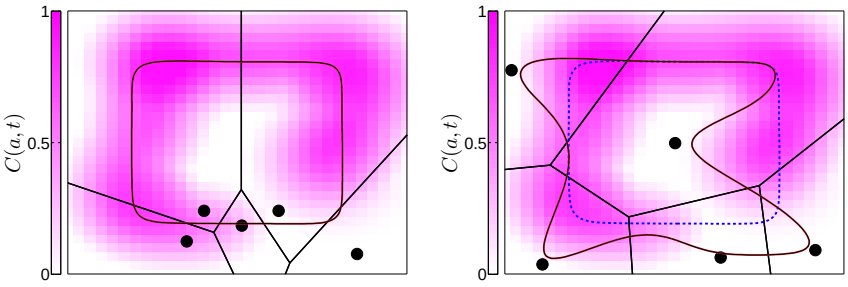
3) PoIs update Given the centroids of the Voronoi tessellation, each point of interest $\boldsymbol{r} \in \mathcal{R}$ is updated with a gradient ascent of the function $\|\bar{\boldsymbol{a}}_{\boldsymbol{r}} - \boldsymbol{r}\|$, i.e.,

$$\dot{\boldsymbol{r}} = k_c(\bar{\boldsymbol{a}}_{\boldsymbol{r}} - \boldsymbol{r}), \tag{4.57}$$

where $k_c > 0$.[16] As the PoIs are updated, the path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ reacts accordingly, thanks to the action of the autonomous corrector. An example of the evolution of a path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ with this algorithm is shown in Fig. 4.18.

The proposed iterative algorithm has several advantages over applying a pure gradient-descent approach to the point of interests without using the Voronoi tessellation. Firstly, it guarantees that the PoIs never overlap since they are constricted to the interior of different Voronoi cells. Secondly,

[16] In the practical (discrete) implementation of (4.57), the gain k_c is chosen using a simple line-search and ensuring that, at each step, every PoI remains inside its Voronoi cell.



(a) Initial configuration. (b) Updated configuration.

Figure 4.18: Evolution of the actual path (red line) with a fixed field C (magenta-scale), using the PoI steering algorithm. The desired (human-commanded) path is shown with a dashed blue curve. Black lines indicate the Voronoi cells. a) Initial configuration: actual path and desired path coincides and randomly selected PoIs (black points) are added in the scene. b) Starting from the initial configuration, the PoIs are updated with the algorithm and the path is modified accordingly, reaching the places where C is lower and following the desired curve if possible.

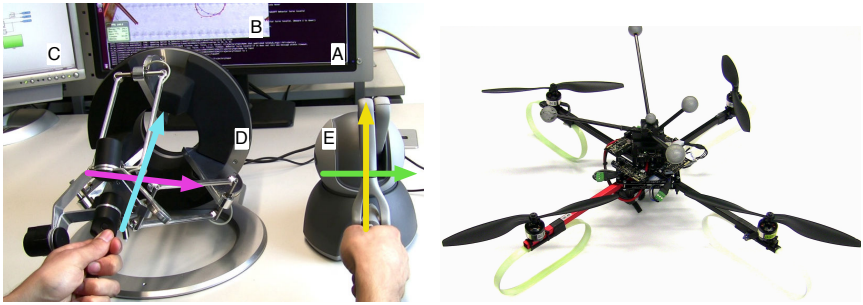
this method is more efficient because each update uses the information of an extended region, the Voronoi cell, thus making it more robust to local minima. The method has been tested in simulation and the results are presented in Sec. 4.9.2.

4.9 Simulations and Experiments

4.9.1 Experimental Testbed

The shared planning framework has been extensively tested both in simulation and in experiments with a real robot. In both cases, two haptic devices have been used and they are shown in Fig. 4.19a. The device on the left is an Omega.6[17] with 6 DoF (only 3 DoF are actuated), but only 2 DoF have been used to command changes of scale and rotations of the path with respect to the centre of mass of the control points. The

[17] `www.forcedimension.com`



(a) Input devices and software. (b) Quadrotor.

Figure 4.19: Experimental setup. a) Software components of the framework and input devices: A) TeleKyb; B) SwarmSimX; C) Matlab; D) Omega.6, used to command changes of scale (magenta) and rotations (cyan); E) Phantom Omni, used to command translations along $\vec{X}_{\mathcal{W}}$ (yellow) and along $\vec{Y}_{\mathcal{W}}$ (green). b) The quadrotor used for the experiments.

device on the right is a Phantom Omni[18] with 6 DoF (only 3 DoF are actuated), and only 2 DoF have been used to command translations in the plane spanned by $(\vec{X}_{\mathcal{W}}, \vec{Y}_{\mathcal{W}})$. The components of the force feedback $\boldsymbol{\tau}$ have been partitioned between the two devices according to the division of the commands. Both devices are connected to a Linux machine where a local control loop implements the force $\boldsymbol{\tau}$ at a frequency of 2.5kHz.

The software implementation of the framework consists of three parts that are all executed on the same Linux machine where the local controller of the input devices resides. The three parts, which are interfaced according to the diagram shown in Fig. 4.20, are the following:

A) The basis of the software implementation is TeleKyb (see Grabe et al. (2013)), a middleware open source software based on the Robot Operating System (ROS[19]). Telekyb was used to realize all the communication interfaces among the different components of the framework (input devices, simulation environment, robot, Matlab). In Fig. 4.19a Telekyb is shown running in the background together with the other processes.

B) The second component of the software is SwarmSimX (see Lächele et al.

[18] www.geomagic.com
[19] www.ros.org

(2012)), a real-time simulation environment that uses Ogre3D[20] as graphical engine and Nvidia PhysX to physically simulate robots. The role of SwarmSimX is twofold. In the experiments, only the graphical engine is used in order to provide a visual feedback to the operator by showing the evolution of the paths $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ and $\boldsymbol{\gamma}_S(\boldsymbol{x})$ in a virtual reconstruction of the real environment. In this case, the simulated robot moving in the virtual environment is only a graphical object whose state is updated with the measured state of the robot. On the other hand, in the simulations SwarmSimX is also used to physically simulate the robot and for this it receives the reference trajectory and implements internally the model of the robot.

C) The third component of the software is Matlab[21] and Simulink, that is used to compute the reference trajectory for the robot. In particular, in this part of the software implements the Human Guidance, the Autonomous Controller, the computation of the haptic cues and the PoIs update algorithm. All these algorithms were able run in real-time on a normal pc using single core.

The robot used both in the simulations and in the experiments is a quadrotor. The use of a quadrotor allows to empirically validate the preliminary assumption made in Sec. 4.2 thanks to the flatness of the point $\boldsymbol{p}_r$ (it is part of the flat output). For the experiments and simulations the trajectory commanded to the robot were not aggressive, since in typical applications such as monitoring or exploration the robot is required to travel at low speed to collect data and to allow the operator to observer the environment. Given this assumption, it was used a near-hovering trajectory tracker. The results presented hereinafter confirm that the robot could precisely follow the commanded trajectory.

4.9.2 Results

Test 1

The first test was conducted to assess the contribution of the control term N in (4.15) by letting the quadrotor travel along a straight reference path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ that is commanded lateral translations by the human operator. This test was initially conducted in simulation, with two conditions: once

[20] `http://www.ogre3d.org`
[21] `http://www.mathworks.com`

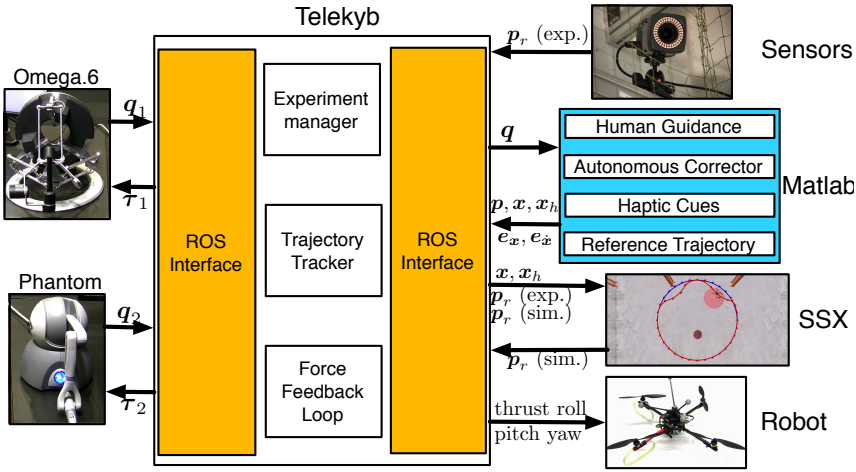


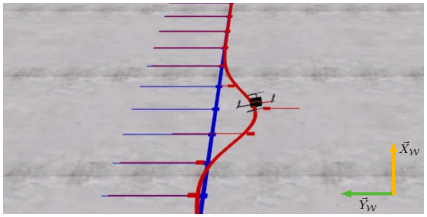
Figure 4.20: Experimental testbed.

using the control points evolution (4.15) (snapshot in Fig. 4.21a) and the second time removing the control term N from (4.15) (snapshot in Fig. 4.21b). In order to compare the two cases, the operator commands from the first trial are recorded and used as inputs in the second trial.

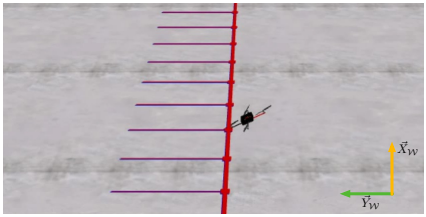
The reference trajectory $\boldsymbol{p}(t)$ to be tracked by the robot is shown in Fig. 4.21c for both cases. A simple visual inspection of Figs. 4.21a and 4.21b suggests that: i) without N, $\boldsymbol{\gamma}_S(\boldsymbol{x})$ is identical to $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ but the robot cannot follow it because of the human commands; ii) with N, $\boldsymbol{\gamma}_S(\boldsymbol{x})$ differs from $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ but the robot can follow it quite precisely. This analysis is confirmed by Fig. 4.22d, which displays the tracking error, in both conditions, of the real robot position with respect to $\boldsymbol{\gamma}(\boldsymbol{x}(t), s(t))$. Remember that the goal is not to track exactly $\boldsymbol{\gamma}(\boldsymbol{x}_h, s)$, because the human-commanded path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$ is not guaranteed to be collision-free and regular. Moreover, the unpredictable path variations commanded by the user could make tracking $\boldsymbol{\gamma}(\boldsymbol{x}_h, s)$ difficult and they could result in large and dangerous overshoots. Snapshots 4.21a and 4.21b also suggest that without the control term N the quadrotor has to tilt (roll) more in order to cope with the commands given by the human. Once again, this is confirmed by the plot of the roll angle in Fig. 4.22f.

The same test was then repeated in an experiment using the real quadro-

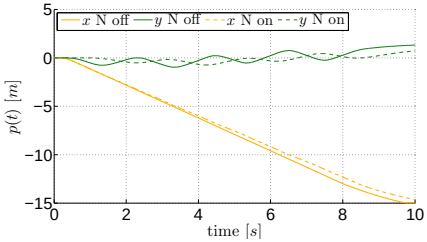
tor. The results plotted in Fig. 4.22 are consistent with the results from the simulation, thus confirming the positive effect of the projector N.



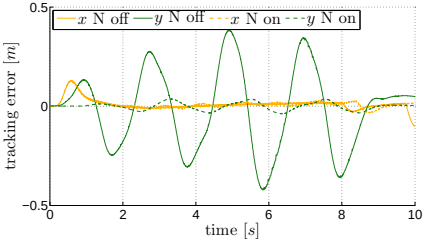
(a) Snapshot: N on.



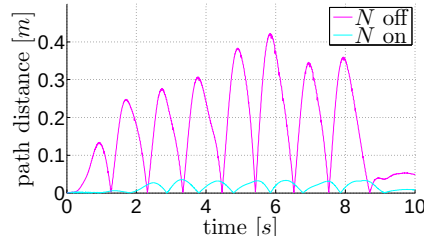
(b) Snapshot: N off.



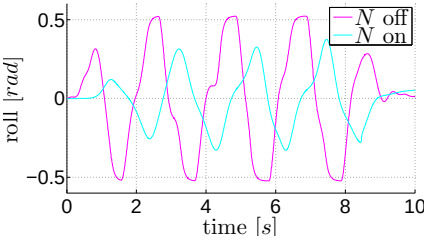
(c) Reference trajectory $\boldsymbol{p}(t)$.



(d) Tracking error.

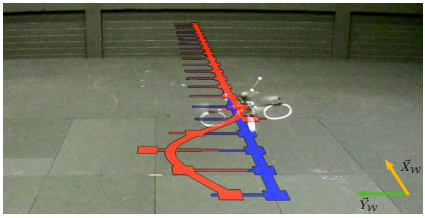


(e) Tilt angle

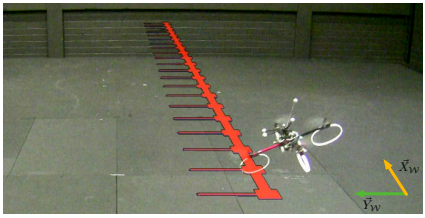


(f) Tilt angle

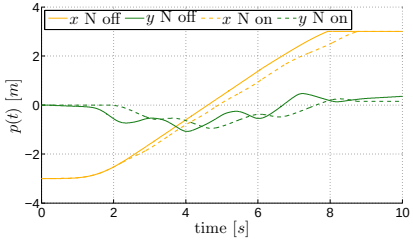
Figure 4.21: Simulation 1: Projection term N. In the snapshots a) and b): the desired path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$, the control points $\boldsymbol{x}_h$ and the velocities $\dot{\boldsymbol{x}}_h$ are depicted as a thick blue line, blue squares and thin blue lines respectively. The actual path $\boldsymbol{\gamma}_S(\boldsymbol{x})$, the control points $\boldsymbol{x}$ and the velocities $\dot{\boldsymbol{x}}$ are depicted as a thick red line, red squares and thin red lines respectively.



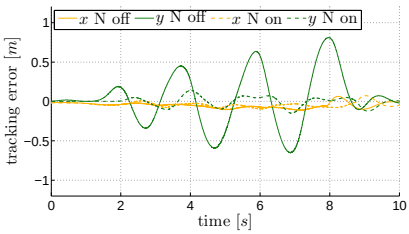
(a) Snapshot: N on.



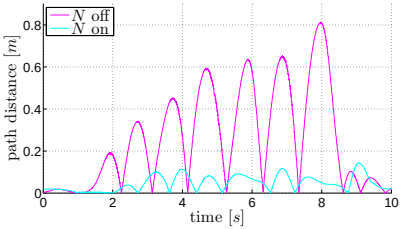
(b) Snapshot: N off.



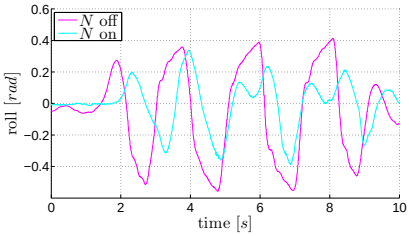
(c) Reference trajectory $\boldsymbol{p}(t)$.



(d) Tracking error.



(e) Tilt angle



(f) Tilt angle

Figure 4.22: Experiment 1: Projection term N. In the snapshots a) and b): the desired path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$, the control points $\boldsymbol{x}_h$ and the velocities $\dot{\boldsymbol{x}}_h$ are depicted as a thick blue line, blue squares and thin blue lines respectively. The actual path $\boldsymbol{\gamma}_S(\boldsymbol{x})$, the control points $\boldsymbol{x}$ and the velocities $\dot{\boldsymbol{x}}$ are depicted as a thick red line, red squares and thin red lines respectively.

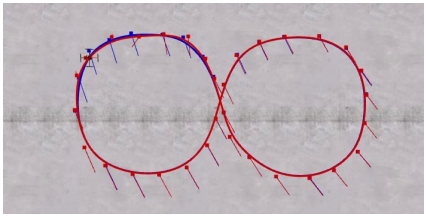
Test 2

The second test is designed to showcase the Human Guidance, by letting the operator command path modifications in a free environment (without obstacles an PoIs). For this test, both haptic devices in Fig. 4.19b have been used to command path translations, changes of scale and rotations as defined by (4.20) to (4.22). In particular, both rotation and scaling maps (4.21) and (4.22) have been implemented with the center of mass of the control points as the center of rotation/scaling. The test is repeated twice, using both an open and a closed self-intersecting path, in order to show the generality of the framework and its capability to deal even with even non-trivial paths.

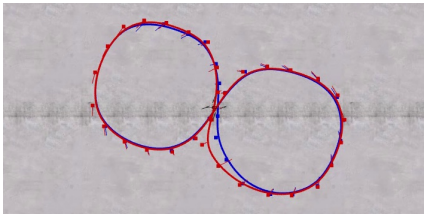
This test was first conducted in simulation, and Figs. 4.23a to 4.23c and Figs. 4.24a to 4.24c show snapshots of the execution, where the velocities of $\boldsymbol{x}_h$ (blue squares) and $\boldsymbol{x}$ (red squares) are depicted as blue and red thin lines respectively. Observe that during a translation (Figs. 4.23a and 4.24a) the velocity vectors are all the same, during a rotation (Figs. 4.23b and 4.24b) they are all moving around the center of rotation, and during a rescaling (Figs. 4.23c and 4.24c) they are all moving towards or from the center of scaling. Observe also that the control points of the vector $\boldsymbol{x}$ around the current robot location do not follow exactly the desired velocity $\dot{\boldsymbol{x}}_h$. This is due to the projection operator N, which modifies the human directives in order to preserve the properties specified by (4.24).

Once again, the tracking error is small (see Figs. 4.23d and 4.24d), thus confirming the hypothesis of precise tracking. Figures 4.23e and 4.24e show the commands given by the operator. Notice that the operator controls at the same time all the 4 DoF to command path modifications through (4.20) to (4.22). Finally, the difference between actual and desired path is captured by the force feedback rendered on the input devices, which is depicted in Figs. 4.23f and 4.24f.

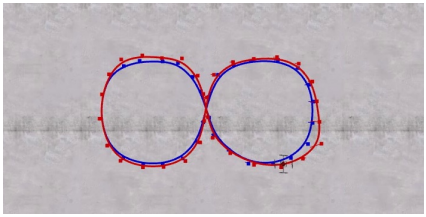
Analogous results were obtained in the experiments executed with the real quadrotor, as shown in Figs. 4.25 and 4.26



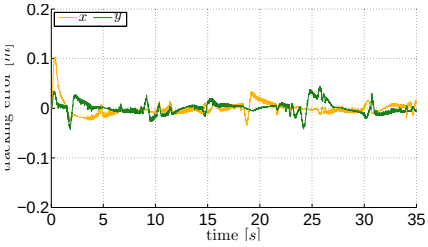
(a) Translation.



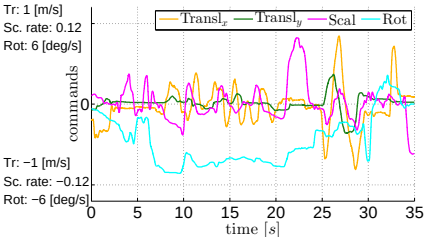
(b) Rotation.



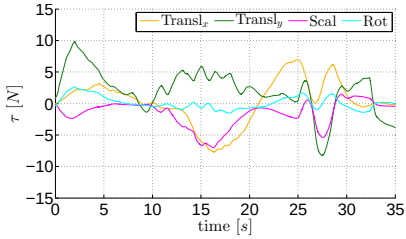
(c) Scaling.



(d) Tracking error.

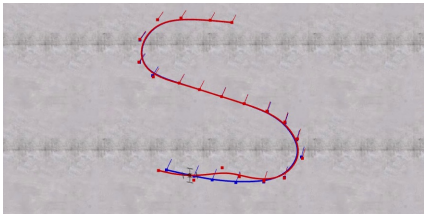


(e) Human commands

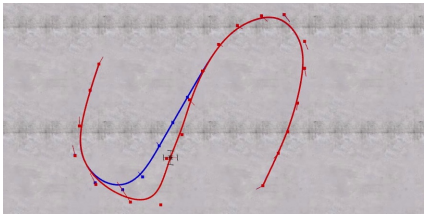


(f) Force feedback

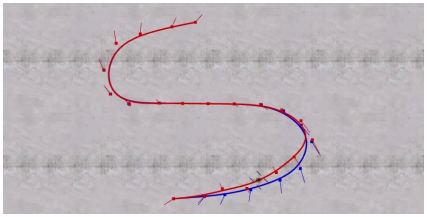
Figure 4.23: Simulation 2 - human guidance (closed path). In a) to c): the desired path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$, the control points $\boldsymbol{x}_h$ and the velocities $\dot{\boldsymbol{x}}_h$ are depicted as a thick blue line, blue squares and thin blue lines, respectively. The actual path $\boldsymbol{\gamma}_S(\boldsymbol{x})$, the control points $\boldsymbol{x}$ and the velocities $\dot{\boldsymbol{x}}$ are depicted as a thick red line, red squares and thin red lines, respectively.



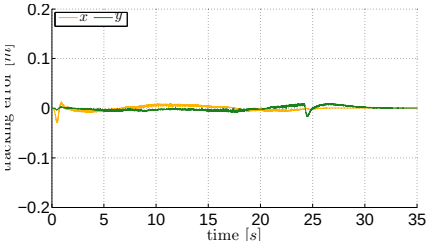
(a) Translation.



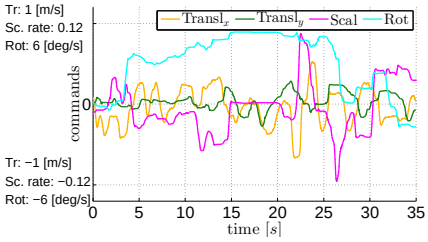
(b) Rotation.



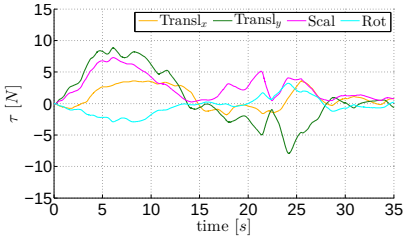
(c) Scaling.



(d) Tracking error.

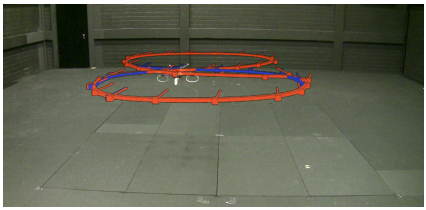


(e) Human commands

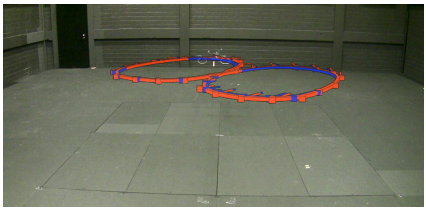


(f) Force feedback

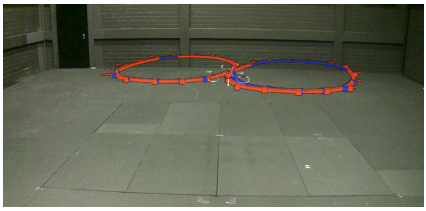
Figure 4.24: Simulation 2 - human guidance (open path). In a) to c): the desired path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$, the control points $\boldsymbol{x}_h$ and the velocities $\dot{\boldsymbol{x}}_h$ are depicted as a thick blue line, blue squares and thin blue lines, respectively. The actual path $\boldsymbol{\gamma}_S(\boldsymbol{x})$, the control points $\boldsymbol{x}$ and the velocities $\dot{\boldsymbol{x}}$ are depicted as a thick red line, red squares and thin red lines, respectively.



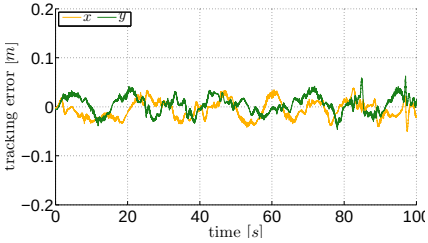
(a) Translation.



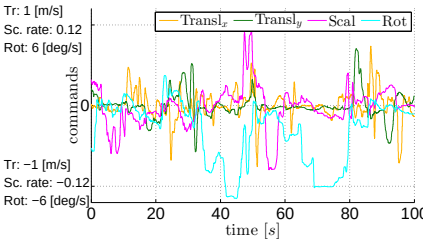
(b) Rotation.



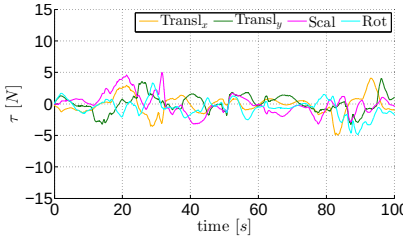
(c) Scaling.



(d) Tracking error.

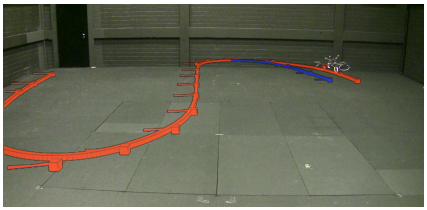


(e) Human commands

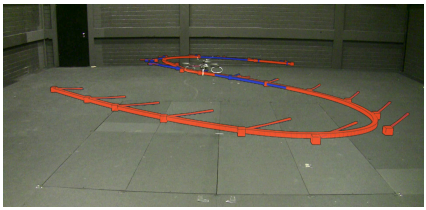


(f) Force feedback

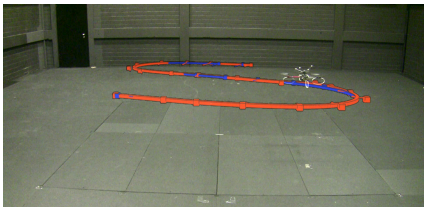
Figure 4.25: Experiment 2 - human guidance (closed path). In a) to c): the desired path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$, the control points $\boldsymbol{x}_h$ and the velocities $\dot{\boldsymbol{x}}_h$ are depicted as a thick blue line, blue squares and thin blue lines, respectively. The actual path $\boldsymbol{\gamma}_S(\boldsymbol{x})$, the control points $\boldsymbol{x}$ and the velocities $\dot{\boldsymbol{x}}$ are depicted as a thick red line, red squares and thin red lines, respectively.



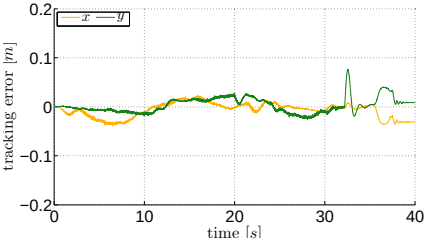
(a) Translation.



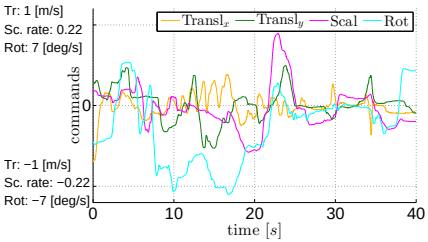
(b) Rotation.



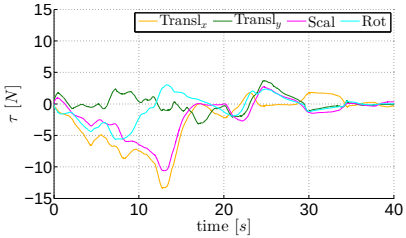
(c) Scaling.



(d) Tracking error.



(e) Human commands



(f) Force feedback

Figure 4.26: Experiment 2 - human guidance (open path). In a) to c): the desired path $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$, the control points $\boldsymbol{x}_h$ and the velocities $\dot{\boldsymbol{x}}_h$ are depicted as a thick blue line, blue squares and thin blue lines, respectively. The actual path $\boldsymbol{\gamma}_S(\boldsymbol{x})$, the control points $\boldsymbol{x}$ and the velocities $\dot{\boldsymbol{x}}$ are depicted as a thick red line, red squares and thin red lines, respectively.

Test 3

The third test is designed to asses the effect of the reactive control from the Autonomous Corrector in presence of obstacles and PoIs, while the human operator steers the desired path by commanding translations (2 DoF) and changes of scale (1 DoF).

The test is first executed in a simulation in which the PoIs are represented by barrels in the environment and the UAV should fly over them, for example to gather data from a down-facing camera (represented by a pink cone, as visible from Figs. 4.27a to 4.27c). Snapshots 4.27a to 4.27c illustrate some meaningful moments during the execution of the task:

a) at the time $t \approx 12s$, $\boldsymbol{\gamma}_S(\boldsymbol{x})$ is attracted by a nearby PoI (Fig. 4.27a);

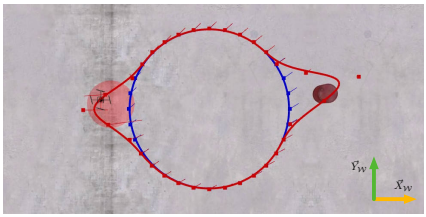
b) at the time $t \approx 33s$, the operator has discarded the previous PoI after it was visited and now steers the path towards two obstacles (Fig. 4.27b) while the Autonomous Corrector prevents collisions;

c) at the time $t \approx 47s$, the operator shrinks the path to make it pass between two obstacles (Fig. 4.27c).

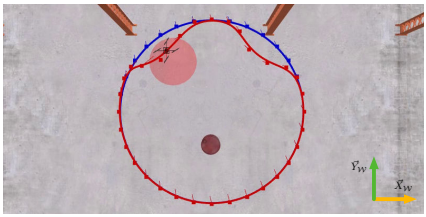
The commands given by the operator and the forces rendered on the input devices are depicted in Figs. 4.27d and 4.27e, respectively. Notice the effect of the PoIs when they are within the range of action of the artificial potential $\varphi_{\mathcal{R}}$, at around 4.6s and 61.3s (vertical dashed lines in the plots): i) the force feedback has peaks that inform the operator about the presence of the PoIs; ii) the human is guided by the force feedback and reacts by steering the path towards the PoIs. The distance of $\boldsymbol{\gamma}_S(\boldsymbol{x})$ from the PoIs is plotted in Fig. 4.27f. Note that the distance from the PoIs rapidly decreases when they are within the range $R_{\mathcal{R}}$ of action of the potential $\varphi_{\mathcal{R}}$. In Fig. 4.27f it is also shown how the operator discards the first PoI after it has been visited by the robot. The autonomous corrector also ensures that $\boldsymbol{\gamma}_S(\boldsymbol{x})$ is collision free and regular: Fig. 4.27g shows that the distance from the obstacles always stays above the threshold $R_{\mathcal{O}}$, and Fig. 4.27h shows that the distance $\min_{i=1,\dots,n} \|\Omega_i(\boldsymbol{x}) - \boldsymbol{x}_i\|$ between the control points and their singular curve is always greater than zero.

The test was then conducted in an experiment with the real quadrotor. In the experiment, the environment contains two obstacles which are two vertical bars and a single PoI shaped like a target that is placed on the ground. The environment is designed so that the operator must manipulate the path past the obstacles in order to reach the PoI. The quadrotor is

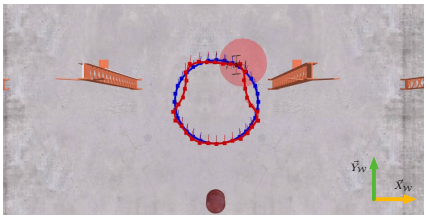
equipped with a down-facing camera which allows the operator to have a visual feedback of the target. Note that in comparison to the simulation the space available in the environment is quite smaller and the operator must steer the path in the limited corridor between the two obstacles in order to reach the target. For this reason the operator commands more frequent corrections with all the commands at his disposal (see Fig. 4.27d). Similarly to the simulation, also in this case the Autonomous Corrector keeps the traveled path regular and collision free, and it attracts it to the single PoI. Figure 4.28f shows a snapshot from the onboard camera when the UAV transits over the target, thanks to the attractive action implemented by $\boldsymbol{u}_{a,\mathcal{R}}$ on $\boldsymbol{\gamma}_S(\boldsymbol{x})$.



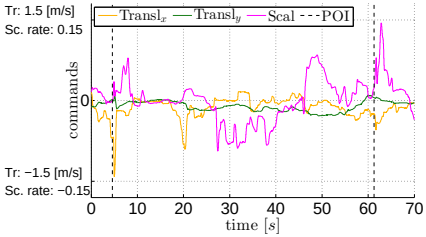
(a) Snapshot 1, $t \approx 12s$.



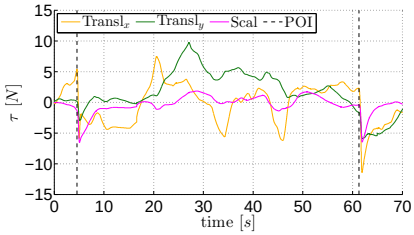
(b) Snapshot 2, $t \approx 33s$.



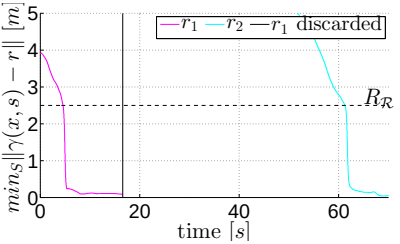
(c) Snapshot 3, $t \approx 47s$.



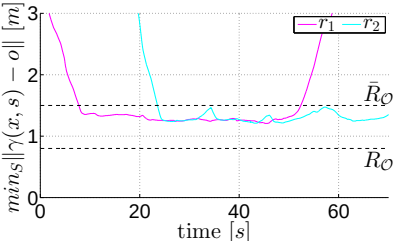
(d) Human commands.



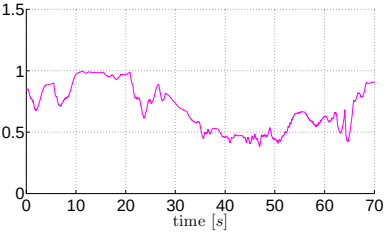
(e) Force feedback.



(f) Distance of the path from the PoIs.

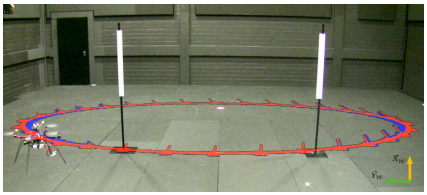


(g) Distance of the path from obstacles and singularities.

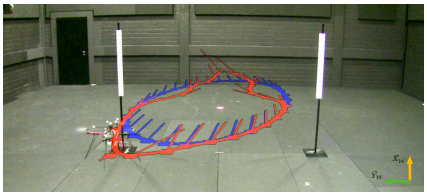


(h) Distance of the control points from singularity curves.

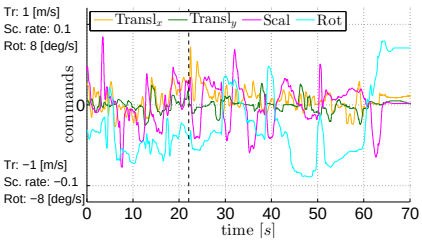
Figure 4.27: Simulation 3: obstacles and PoIs.



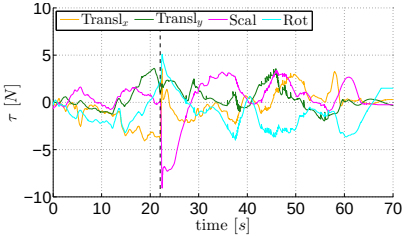
(a) Snapshot: $t \simeq 1$s.



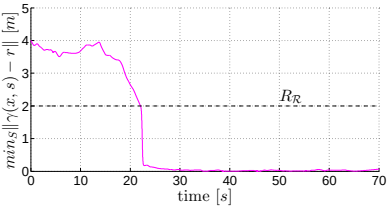
(b) Snapshot: $t \simeq 22$s.



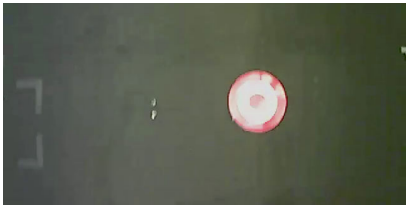
(c) Commands.



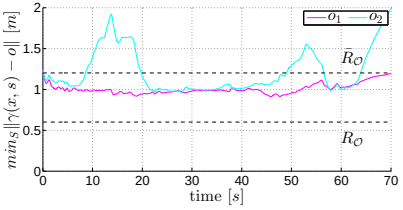
(d) Force feedback.



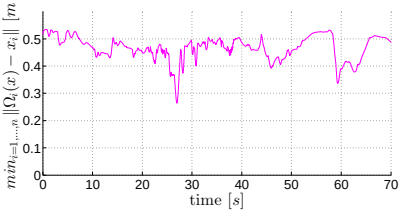
(e) Distance from PoI.



(f) PoI (onboard camera).



(g) Distance from obstacles.



(h) Distance from singularities.

Figure 4.28: Experiment 3: obstacles and PoIs.

Test 4

The fourth test showcases the mechanism for the generation of alternative paths in a cluttered environment. Figures 4.29a to 4.29d illustrate the different phases of the generation of an alternative path and the path switch, in simulation:

a) At time $t = 17.52s$ the algorithm is in the *crossing* step and the alternative path is 'pulled' to the other side of the obstacle. Note that the velocity vectors of the control points $\boldsymbol{x_o}$ (thin green lines) are pointing in the direction of the obstacle.

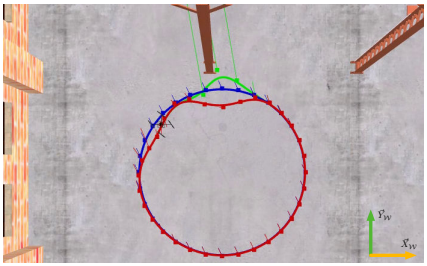
b) At time $t = 17.80s$ the algorithm is in the *expansion* step and $\boldsymbol{\gamma}_S(\boldsymbol{x_o})$, which is already non-homeotopic to $\boldsymbol{\gamma}_S(\boldsymbol{x})$, is 'pushed' out of the obstacle ball. Note that the velocity vectors of the control points $\boldsymbol{x_o}$ (thin green lines) are pointing away from the obstacle.

c) At time $t = 19.96s$ the algorithm is in the *activation* step and the alternative path is fully generated.

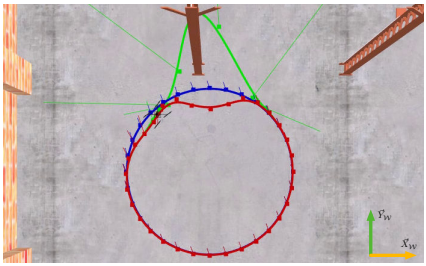
d) At time $t = 23.00s$, condition $\mathcal{C}4$ is verified and the current path switches to the alternative one.

Similarly to Simulation 3, the repulsive term (4.31) keeps the distance between $\boldsymbol{\gamma}_S(\boldsymbol{x})$ and the obstacles in the environment always greater than $R_{\mathcal{O}}$, even when multiple obstacles are present at the same time (see Fig. 4.29e). Furthermore, the alternative paths algorithm allows $\boldsymbol{\gamma}_S(\boldsymbol{x})$ to easily move past an obstacle: this prevents the algorithm to get stuck in a suboptimal path in terms of closeness to the desired $\boldsymbol{\gamma}_S(\boldsymbol{x}_h)$. The effect is clearly visible in Fig. 4.29f which shows how a reduction of the mismatch $\|\boldsymbol{x} - \boldsymbol{x}_h\|$ occurs whenever there is a path switch. Figures 4.29g and 4.29h show the evolution of the commands given by the human operator and of the force feedback. There is a clear jump in the forces, and consequently in the commands, when the path switches to a new alternative given by the replanner. This effect gives the natural sensation that the resistance produced by the obstacle is finally removed and therefore the path is free to move in that direction.

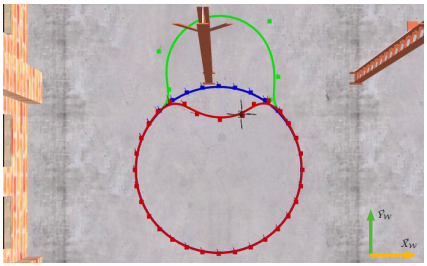
Similar result have also been obtained in experiments, as shown in Fig. 4.30. Note that the experimental environment contains three obstacles not too far apart, and the replacer is capable of generating at the same time one alternative path for each obstacle (see Fig. 4.30e).



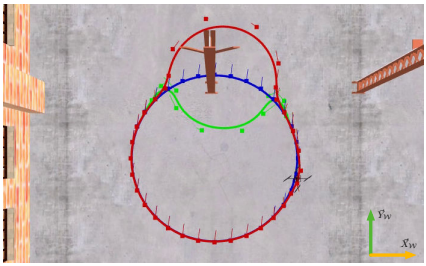
(a) Crossing, $t = 17.52s$.



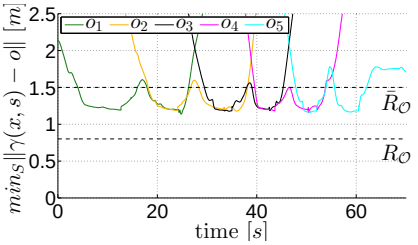
(b) Expansion, $t = 17.80s$.



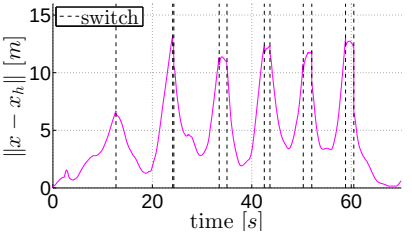
(c) Activation, $t = 19.96s$.



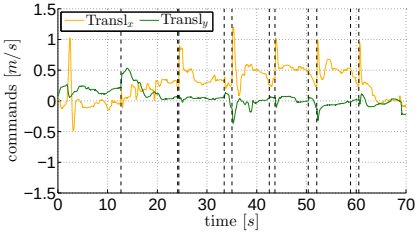
(d) Switch, $t = 23.00s$.



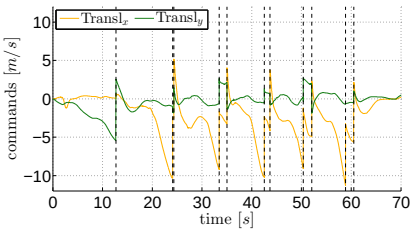
(e) Distance path - obstacles.



(f) Mismatch $\|\boldsymbol{x} - \boldsymbol{x}_h\|$.

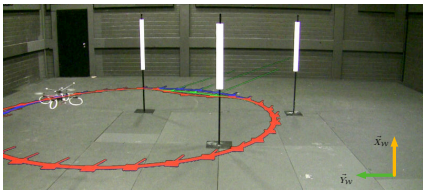


(g) Commands.

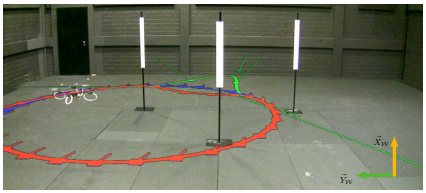


(h) Force feedback.

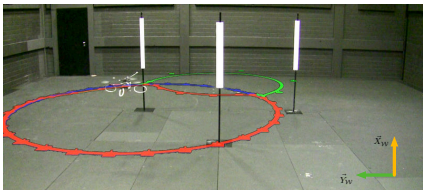
Figure 4.29: Simulation 4: generation of alternative paths.



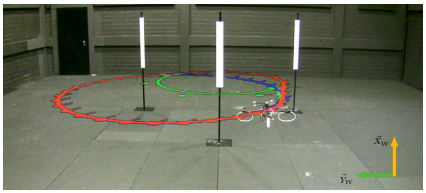
(a) Crossing, $t = 17.52s$.



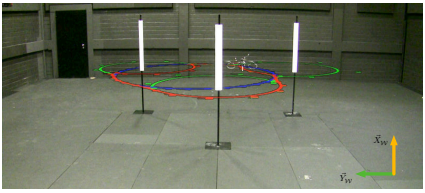
(b) Expansion, $t = 17.80s$.



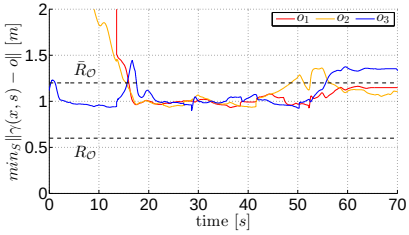
(c) Activation, $t = 19.96s$.



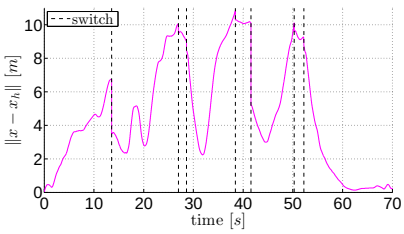
(d) Switch, $t = 23.00s$.



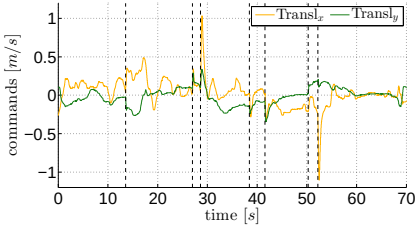
(e) Alternative paths with mutliple obstacles.



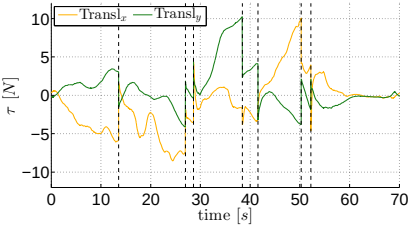
(f) Distance path - obstacles.



(g) Mismatch $\|\boldsymbol{x} - \boldsymbol{x}_h\|$.



(h) Commands.



(i) Force feedback.

Figure 4.30: Simulation 4: generation of alternative paths. Snapshots of the phases of the proposed algorithm.

Test 5

In the fifth simulation the framework is applied to a coverage task. The task is carried out in two phases: first, the human operator steers $\boldsymbol{\gamma}_S(\boldsymbol{x})$ in the environment and selects the area to be monitored; then, the monitoring of the selected area is executed autonomously by using the PoIs update procedure proposed in Sec. 4.8. Fig. 4.31a shows, first, the commands given by the user prior to select the initial desired path for the coverage. In the second part, Fig. 4.31a shows the normalized integral of the coverage function on the area monitored, i.e., $C_{\mathcal{A}} = \int_{\mathcal{A}} C(\boldsymbol{a}, t) d\boldsymbol{a} / \int_{\mathcal{A}} d\boldsymbol{a}$. The integral $C_{\mathcal{A}}$ rapidly increases and it stabilizes at around 0.7 because of the consumption in the coverage field model (4.54). Snapshots 4.31b to 4.31d clearly illustrate how the PoIs move during the task and how the path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ reacts. Observe in particular that at the beginning of the task (Fig. 4.31b), since the coverage field $\mathcal{C}$ is zero almost everywhere, the PoIs are close to the geometrical centre of the Voronoi cells. As the task is executed the PoIs reach all the locations that have not be covered, including the centre of the reference path (Fig. 4.31c) and the corners of $\mathcal{A}$ (Fig. 4.31c), and the path $\boldsymbol{\gamma}_S(\boldsymbol{x})$ readily adapts to the PoIs motion.

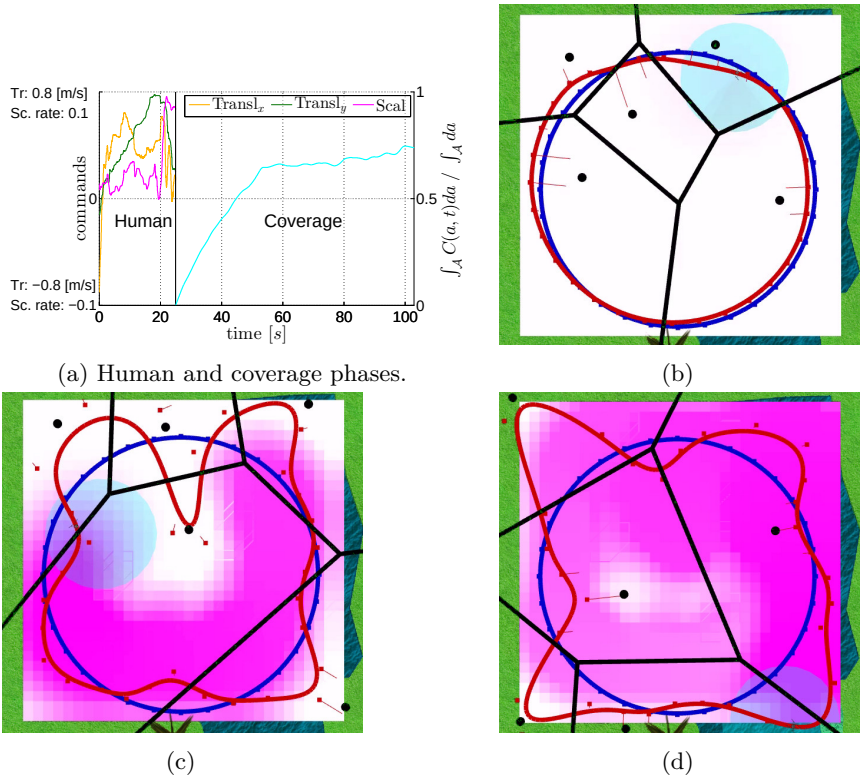
4.10 Summary and Possible Extensions

Summarizing, it has been presented an extension of the shared control approach for mobile robots in which:

1. The operator acts at the planning level and modifies online the desired path planned for the robot, thus reducing his/her commitment and increasing the autonomy of the system. The framework provides an intuitive human interface which allows the user to command a broad variety of path modifications and to use multiple input devices.

2. A reactive algorithm autonomously corrects the path planned by the operator to guarantee obstacle avoidance and path regularity, and to reach interesting locations in the environment. The guarantee of a collision free and regular path was formally proven. The algorithm is integrated with a separate module that allows online planning of alternative paths.

3. The force feedback is proportional to the mismatch between the planned path and the one corrected by the robot (i.e., the error is



(a) Human and coverage phases.

(b)

(c)

(d)

Figure 4.31: Simulation 5: coverage task. a) in a first phase the operator commands the initial desired path shape for the coverage; in the second phase the coverage is executed. b) to d) snapshots of the simulation that depict: $\boldsymbol{\gamma}_S(\boldsymbol{x})$ (red line), $\boldsymbol{\gamma}_S(\boldsymbol{x_o})$ (blue line), the footprint $\mathcal{F}(\boldsymbol{p}(t), R)$ covered by the robot (light blue disc), the PoIs (black dots) and the corresponding Voronoi tessellation of $\mathcal{A}$ (black lines), and the coverage function on $\mathcal{A}$ ($C = 0$ is white and $C = 1$ is magenta)

computed in an *integral* sense on the future trajectory). This force feedback is also paired with a visual feedback.

4. The framework was validated by means of extensive human/hardware-in-the-loop simulations and real experiments employing a quadrotor and two force-feedback devices.

Several extensions of this framework are possible. Firstly, it could be interesting to make an evaluation study with subjects to assess the benefits of the integral haptic feedback and also the importance of the virtual/augmented visual feedback. Another extension is to include an additional task, such as grasping an object in a certain point, which would constrain the path modifications. One observation is that in this project have not been considered possible kinodynamics constraints of the vehicle. This is often done in monitoring, coverage and patrolling tasks (see e.g., Smith et al. (2012)), where the robots are not required aggressive motion. In this regard, a future extension could be to design an ad hoc timing law controller that regulates the circulation speed of the robot, including possible actuation constraints that are generally disregarded, and study also how such constraints limit the path reshaping.

Chapter 5

Conclusions

This manuscript has presented a study on robotic tasks with a human-in-the-loop (RTHL) a topic of research that recently has received great attention from the research community. The relevance of this subject has been illustrated in Chap. 1 with a brief summary of several application domains that involve RTHL. Indeed, the topic is very broad, however the projects that have been presented in Chaps. 2 to 4 provide a general overview by spanning various possibilities for the characteristic aspects of RTHL:

Human: In the three projects the human has different roles, i.e., piloting a simulated vehicle, high-level steering a team of robots and modifying the planned path of a mobile robot.

Robot: Several kinds of robots have been considered, i.e., a motion platform based on an extended anthropomorphic manipulator, a group of UAVS and a single mobile robot.

Feedback: Various sensorial cues have been adopted to provide a feedback to the operator, i.e., vestibular, visual and haptic cues.

Besides giving an overview on RTHL, the three projects are also individually relevant and involve applications and problems that are presently of great interest. In particular, each of the frameworks that have been developed add several novel contributions to prior works and could be extended or generalized to other scenarios. In short:

- In Chap. 2 it was presented and modelled a novel actuated cabin that can be mounted on industrial manipulators to be used as motion platforms in order to extend their motion envelope. This novel 'joint' could also be used without gondola to increase the manipulability of

a tooltip. Lastly, the control algorithm is in general applicable to any redundant manipulator that has to track a trajectory unknown in advance.

- In Chap. 3 it was presented a formation controller that regulates the relative bearings of a group of UAVs while allowing the operator to command synchronised motions of all the UAVs. The framework also extends the classic bilateral (haptic) control of mobile robots by providing a force feedback on these synchronised motions. Lastly, this framework could be generalized to other formations, beyond bearing-formations.

- In Chap. 4 it was presented a framework for shared motion planning, in which the operator modifies online the path travelled by a robot. The robot can then further correct the modified path in order to verify some properties. This framework also extends the classic bilateral (haptic) control of mobile robots by providing an integral force feedback on the evolution of the planned path. Lastly, this framework could be generalized by adding to the planned motion constraints from other tasks, such as grasping a certain object in the environment.

It should also be noted that the three frameworks could also be combined to provide new paradigms of human-robot interaction. For instance, in the shared bearing-formation control architecture the human could steer the UAVs from onboard the motion simulator, which can then be used to provide the operator with a suitable vestibular feedback that increases his/her situation awareness. In this case the vestibular feedback could reproduce the sensation of motion of the 1st UAV (beacon of the formation) or of the whole team. A preliminary study in this direction was presented in Robuffo Giordano et al. (2010a), in which it was evaluated the performance of a pilot that was tasked to steer a single UAV while being onboard the motion simulator, however the authors only combined a visual feedback from a camera onboard the UAV with the vestibular feedback from the motion simulator but without haptic feedback. Another possibility is to combine the bearing formation control with the shared planning framework, by letting the operator operate at the planning level and modifying the planned path for the whole formation of robots (or of a characteristic point of the formation).

Looking now at the three projects together, there are several observations to be made regarding the challenges and properties of RTHL introduced in Sec. 1.4.

Human-in-the-loop One of the main challenges of RTHL is that the unpredictable actions of the human-in-the-loop may complicate or conflict with the task assigned to the robot, and this problem affects the design of the shared control architecture. This is also true for the three projects presented in Chaps. 2 to 4, and the frameworks therein illustrate different approaches to deal with this problem:

Chapter 2: The operator commands, through a motion cueing algorithm, a reference trajectory for the robot that does not account for the the limitations of the robot, such as the joint limits. The controller of the robot is then designed to follow, when possible, the trajectory originated from the operator while coping with the aforementioned limitations.

Chapter 3: In the shared bearing-formation control the user interface is designed so that the commands of the operator do not change the relative bearings of group of UAVs thus not interfering with the task assigned to the robots.

Chapter 4: In the shared planning framework the user commands through a suitable interface modifications of the path traveled by the robot. These transformations are locally locally cancelled by means of a null-space projection to ease tracking for the robot.

Summarizing with some abstraction, the frameworks show how to deal with the unpredictability of the human-in-the-loop on three different levels: 1) on the *robot side*, by delegating to the robot the duty of coping with the exogenous human's signals; 2) on the *human side*, by designing the user interface such that the operator does not interfere with the task of the robot; 3) on the *interconnection* between human and robot, by cancelling or filtering the components of the human's signals that conflict with the task of the robot. These three different strategies entail a higher complexity of different parts of the human-robot system and they could also be combined together. The advantages of the different approaches could be assessed by further comparative studies.

Autonomy, role of the operator and UI Another question with HRLTs is 'how much autonomy should be given to the robots'? The answer to this question is generally left to the designer's choice, but from the three projects that have been tackled it is possible to gain some insight on what this choice entails. Consider first the motion simulator presented in Chap. 2. The shared architecture in this case involves little autonomy of the robot and it is actually very close to a direct control paradigm, because the human directly pilots a simulated vehicle and the robot tracks the trajectory generated by the aforementioned virtual vehicle. In other words, the limited autonomy in this architecture translates to a (almost) direct command interface and it also determines a more involved role of the person who participates to the task because he/she has to (almost) directly control the motion of the robot at all times. In the case of the motion simulator the role of the operator and the command interface are actually implicitly specified by the application itself. Even the involvement of the operator during the task is desirable, because it can help increasing the sensation of 'realistic' simulated motion. However, for other applications it is in general desirable to reduce the commitment of the operator in order to decrease his/her workload and stress level.

Consider now the projects presented in Chaps. 3 and 4. In both cases, the underlying idea of the frameworks is to give to the robots more autonomy and elevate the role of the operator at a higher level, i.e., high-level steering of a group of UAVs and motion planning. The first observation is that with the shift in the role of the human it becomes necessary to delegate to the robots the execution of larger parts of the task, e.g., 1) in the shared bearing-formation control, the regulation of the formation is done by the robots, and 2) in the shared planning the mobile robot autonomously ensures that the planned path is collision free and regular. Besides this transfer of authority, the increased autonomy of the robots and the new role of the human operator is also reflected by a change in the UIs.

Namely, the commands provided by the human are not anymore direct motion references for the robots, but motions of a higher-level system, e.g., 1) in the shared bearing-formation control, the operator gives motion commands to the formation of robots as a single entity, and 2) in the shared planning framework the operator commands modification of the planned path. In the frameworks that have been developed for these projects the new command paradigms have been achieved by designing suitable maps from the degrees of freedom of the input device used by the operator to motions of the high-level system. The main observation in regard to these

new command paradigms is that they can provide easier and more intuitive interfaces to the operator with respect to traditional direct control. This observation was confirmed by the numerous simulations and experiments which have shown that the operator was able alone to steer a formation of robots and to modify a planned path with good skill and without any prior training. Indeed, in the projects that have been completed the focus was not on human factors, therefore further studies should be conducted to evaluate the benefits of this paradigm shift for the performance of the operator. However this appears to be a promising direction to attenuate several issues of RTHL, such as the need of training and commitment of the human-in-the-loop.

Another observation is that the feedback given to the human must be modified accordingly to the change of his/her role and of the command interface. The feedback in this case should not give an information of the state of the individual robots but rather of the state of the system controlled by the operator. In this respect, in the bilateral architectures of Chaps. 3 and 4 the force feedback was rendered on the input channels provided to the operator (e.g., in the shared formation-control the feedback is given on the translational, rotational and scaling channels). This change towards a 'high-level feedback' can drastically reduce the amount of data that the operator has to interpret, e.g., by providing a global information of a team of robots rather than on every individual robot (see framework of Chap. 3). In practice, the simulations and experiments that have been discussed in Chaps. 3 and 4 have shown that the novel haptic feedback paradigms were very helpful to the operator. However, as already discussed for the command interfaces, it will be necessary to conduct additional studies focused on human-factors in order to evaluate how this high-level feedback affects situation awareness, workload and performance of the human.

The last observation regards the question of *human-to-robot ratio* in RTHL. It was discussed in Sec. 1.4 that the difficulty of the tasks assigned to the human and the complexity of the UIs often mandate the presence of multiple persons to manage a single robot, thus making the use of large teams of robots not viable in practice. The framework presented in Chap. 3 shows that, even though controlling independently and simultaneously several robots is unfeasible for a single person, shifting the role of the operator and making him/her control only some global aspect of the group of robots as a whole is a promising solution to decreasing the human-to-robot ratio. Further studies could be needed to assess the performance and workload of a single person controlling multiple robots with this new

high-level command interface and feedback.

Bibliography

D. A. Abbink, M. Mulder, F. C. T. van der Helm, M. Mulder, and E. R. Boer. Measuring neuromuscular control dynamics during car following with continuous haptic feedback. *IEEE Trans. on Systems, Man, & Cybernetics. Part B: Cybernetics*, 41(5):1239–1249, 2011.

D. A. Abbink, M. Mulder, and E. R. Boer. Haptic shared control: smoothly shifting control authority? *Cognition, Technology & Work*, 14(1):19–28, 2012.

J. J. Abbott, P. Marayong, and A. M. Okamura. Haptic virtual fixtures for robot-assisted manipulation. In *Robotics Research*, volume 28 of *Springer Tracts in Advanced Robotics*, pages 49–64. Springer Berlin Heidelberg, 2007.

S. K. Agrawal and Y. Jin. A three-wheel vehicle with expanding wheels: differential flatness, trajectory planning, and control. In *2003 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, volume 2, pages 1450–1455, 2003.

G. Antonelli. Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems. *IEEE Trans. on Robotics*, 25 (5):985–994, 2009.

G. Antonelli, F. Arrichiello, and S. Chiaverini. Experiments of formation control with multirobot systems using the null-space-based behavioral control. *IEEE Trans. on Control Systems Technology*, 17(5):1173–1182, 2009.

F. Arrichiello, S. Chiaverini, G. Indiveri, and P. Pedone. The null-space based behavioral control for a team of cooperative mobile robots with actuator saturations. *2009 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 5911–5916, 2009.

C. A. Avizzano, F. Barbagli, and M. Bergamasco. Washout filter design for a motorcycle simulator. In *2000 IEEE Int. Conf. on Systems, Man, and Cybernetics*, volume 2, pages 995–1000, 2000.

C. A. Avizzano, P. Tripicchio, L. Joale, and M. Bergamasco. Design of a motion based sailing simulator. In *2010 IEEE Int. Symp. on Robots and Human Interactive Communications*, pages 1–7, 2010.

S. D. Beard, S. E. Reardon, E. L. Tobias, and B. L. Aponso. Simulation system fidelity assessment at the vertical motion simulator. In *American Helicopter Society (AHS) Annual Forum and Technology Display*, Phoenix, AZ, USA, 2013.

T. Bellman, M. Otter, J. Heindl, and G. Hirzinger. Real-time path planning for an interactive and industrial robot-based motion simulator. In *Proc. of the 2nd Motion Simulator Conference*, 2007.

T. Bellmann, J. Heindl, M. Hellerer, R. Kuchar, K. Sharma, and G. Hirzinger. The DLR robot motion simulator part I: Design and setup. In *2011 IEEE Int. Conf. on Robotics and Automation*, pages 4694–4701, 2011a.

T. Bellmann, M. Otter, and G. Hirzinger. The DLR robot motion simulator part II: Optimization based path-planning. In *2011 IEEE Int. Conf. on Robotics and Automation*, pages 4702–4709, 2011b.

C. Belta and V. Kumar. Abstraction and control for groups of robots. *IEEE Trans. on Robotics*, 20(5):865–875, 2004.

A. Berthoz, W. Bles, H. H. Bülthoff, B. J. Grácio, P. Feenstra, N. Filliard, R. Huhne, A. Kemeny, M. Mayrhofer, M. Mulder, H.-G. Nusseck, P. Pretto, G. Reymond, R. Schlusselberger, J. Schwandtner, H. Teufel, B. Vailleau, M. M. van Paassen, M. Vidal, and M. Wentink. Motion scaling for high-performance driving simulators. *Human-Machine Systems, IEEE Transactions on*, 43(3):265–276, 2013.

K. Beykirch, F. M. Nieuwenhuizen, H. Teufel, H.-G. Nusseck, J. Butler, and H. H. Bülthoff. Control of a lateral helicopter side-step maneuver on an anthropomorphic robot. In *AIAA Modeling and Simulation Technologies Conference and Exhibit*, pages 1–11, 2007.

K. Beykirch, F. M. Nieuwenhuizen, H. J. Teufel, H.-G. Nusseck, and H. H. Bülthoff. A roll-lateral helicopter side-step maneuver on the MPI motion simulator. In *AHS 64th Annual Forum*, pages 1–8, 2008.

L. Biagiotti and C. Melchiorri. *Trajectory Planning for Automatic Machines and Robots*. Springer, 2008. ISBN 978-3540856283.

A. Bicchi, M. Buss, M. O. Ernst, and A. Peer. *The Sense of Touch and Its Rendering: Progress in Haptics Research*. Springer Tracts in Advanced Robotics, 2008.

W. Bles and E. Groen. The DESDEMONA motion facility: Applications for space research. *Microgravity Science and Technology*, 21(4):281–286, 2009.

J. E. Bobrow, S. Dubowski, and J. S. Gibson. Time-optimal control of robotic manipulators along specified paths. *The International Journal of Robotics Research*, 4(3):3–17, 1985.

H. Boessenkool, D. A. Abbink, C. J. M. Heemskerk, F. C. T. van der Helm, and J. G. W. Wildenbeest. A task-specific analysis of the benefit of haptic shared control during telemanipulation. *IEEE Transactions on Haptics*, 6(1):2–12, 2013.

O. Brock and O. Khatib. Elastic strips: A framework for motion generation in human environments. *The International Journal of Robotics Research*, 21(12):1031–1052, 2002.

F. Bullo, E. Frazzoli, M. Pavone, K. Savla, and S. L. Smith. Dynamic vehicle routing for robotic systems. *Proceedings of the IEEE*, 99(9): 1482–1504, 2011.

J. Burke, R. R. Murphy, E. Rogers, V. J. Lumelsky, and J. Scholtz. Final report for the DARPA/NSF interdisciplinary study on human-robot interaction. *IEEE Trans. on Systems, Man, & Cybernetics. Part C: Applications and Reviews*, 34:103–112, May 2004a.

J. L. Burke, R. R. Murphy, M. D. Coover., and D. L. Riddle. Moonlight in miami: Field study of human-robot interaction in the context of an urban search and rescue disaster response training exercise. *Human–Computer Interaction*, 19(1-2):85–116, 2004b.

J. Burki-Cohen, T. H. Go, and T. Longridge. Flight simulator fidelity for total air line pilot training and evaluation. In *Proc. of the AIAA Modeling and Simulation Technologies Conference*, 2001.

F. Caccavale and B. Siciliano. Quaternion-based kinematic control of redundant spacecraft/manipulator systems. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 1, pages 435–440, 2001.

D. W. Casbeer, R. W. Beard, T. W. McLain, S.-M. Li, and R. K. Mehra. Forest fire monitoring with multiple small UAVs. In *2005 American Control Conference*, pages 3530–3535, 2005.

T. F. Chang and R. V. Dubey. A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators. *IEEE Trans. on Robotics and Automation*, 11(2):286–292, 1995.

P. Cheng, J. Keller, and V. Kumar. Time-optimal UAV trajectory planning for 3D urban structure coverage. In *2008 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2750–2757, 2008.

P. Chiacchio, S. Chiaverini, L. Sciavicco, and B. Siciliano. Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy. *The International Journal of Robotics Research*, 10(4):410–425, 1991.

S. Chiaverini. Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Trans. on Robotics and Automation*, 13(3):398–410, 1997.

S. Chiaverini, O. Egeland, and R. K. Kanestrom. Achieving user-defined accuracy with damped least-squares inverse kinematics. In *Proc. of the 5th Int. Conf. on Advanced Robotics*, volume 1, pages 672–677, 1991.

S. Chiaverini, G. Oriolo, and I. D. Walker. Kinematically redundant manipulators. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, pages 245–285. Springer, 2008.

M. Cognetti, P. Stegagno, A. Franchi, G. Oriolo, and H. H. Bülthoff. 3-D mutual localization with anonymous bearing measurements. In *2012 IEEE Int. Conf. on Robotics and Automation*, pages 791–798, St. Paul, MN, May 2012.

B. Conrad and S. F. Schmidt. A study of techniques for calculating motion drive signals for flight simulators. Technical Report NASA CR-114345, NASA, 1971.

J. Cortés, S. Martínez, F. Karataş, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Trans. on Robotics and Automation*, 20 (2):243–255, 2004.

J. W. Crandall, M. A. Goodrich, D. R. O. Jr., and C. W. Nielsen. Validating human-robot interaction schemes in multitasking environments. In *IEEE Trans. on Systems, Man, & Cybernetics. Part A: Systems & Humans*, volume 35, pages 438–449, 2005.

A. K. Das, R. Fierro, V. Kumar, P. Ostrowski, J. Spletzer, and C. J. Taylor. A vision-based formation control framework. *IEEE Trans. on Robotics and Automation*, 18(5):813–825, 2002.

C. De Boor. On calculating with B-splines. *Journal of Approximation Theory*, 6(1):50–62, 1972.

A. De Luca, G. Oriolo, and C. Samson. Feedback control of a nonholonomic car-like robot, 1997.

J. E. Dieudonne, R. V. Parrish, and R. E. Bardusch. An actuator extension transformation for a motion simulator and an inverse transformation applying newton-raphson's method. NASA TN D-7067, National Aeronautics and Space Administration, 1972.

K. L. Doty, C. Melchiorri, E. M. Schwartz, and C. Bonivento. Robot manipulability. *IEEE Trans. on Robotics and Automation*, 11(3):462–468, 1995.

J. L. Drury, L. Riek, and N. Rackliffe. A decomposition of UAV-related situation awareness. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 88–94, Salt Lake City, UT, USA, 2006.

J. W. Durham, A. Franchi, and F. Bullo. Distributed pursuit-evasion without global localization via local frontiers. *Autonomous Robots*, 32(1): 81–95, 2012.

M. R. Endsley. Design and evaluation for situation awareness enhancement. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 32(2):97–101, 1988.

T. Eren, W. Whiteley, A. S. Morse, P. N. Belhumeur, and B. D. O. Anderson. Sensor and network topologies of formations with direction, bearing, and angle information between agents. In *42nd IEEE Conf. on Decision and Control*, volume 3, pages 3064–3069, 2003.

T. Eren, W. Whiteley, and P. Belhumeur. Using angle of arrival (bearing) information in network localization. In *45th IEEE Conf. on Decision and Control*, pages 4676–4681, San Diego, CA, Jan. 2006.

EUROP. Robotic vision - to 2020 and beyond, 2009. *The Strategic Research Agenda for Robotics in Europe*, EUROP,2009.

N. Faiz, S. K. Agrawal, and R. M. Murray. Trajectory planning of differentially flat systems with dynamics and inequalities. *Journal of Guidance, Control, and Dynamics*, 24(2):219–227, 2001.

T. Faulwasser, V. Hagenmeyer, and R. Findeisen. Optimal exact path-following for constrained differentially flat systems. In *2011 IFAC World Congress*, pages 9875–9880, Sept. 2011.

D. Ferrazzin, F. Salsedo, and M. Bergamasco. The MORIS simulator. In *Robot and Human Interaction, 1999. RO-MAN '99. 8th IEEE International Workshop on*, pages 136–141, 1999.

W. R. Ferrell and T. B. Sheridan. Supervisory control of remote manipulation. *IEEE Spectrum*, 4(10):81–88, 1967.

J. Fink, N. Michael, S. Kim, and V. Kumar. Planning and control for cooperative manipulation and transportation with aerial robots. *The International Journal of Robotics Research*, 30(3):324–334, 2010.

M. Fliess, J. Lévine, P. Martin, and P. Rouchon. Flatness and defect of nonlinear systems: Introductory theory and examples. *International Journal of Control*, 61(6):1327–1361, 1995.

S. L. Fraga, J. Borges de Sousa, and F. L. Pereira. User-assisted trajectory generation- of underwater vehicles. In *Proceedings of OCEANS 2003*, volume 2, pages 696–701, 2003.

A. Franchi, L. Freda, G. Oriolo, and M. Vendittelli. The sensor-based random graph method for cooperative robot exploration. *IEEE/ASME Trans. on Mechatronics*, 14(2):163–175, 2009.

A. Franchi, G. Oriolo, and P. Stegagno. Probabilistic mutual localization in multi-agent systems from anonymous position measures. In *49th IEEE Conf. on Decision and Control*, pages 6534–6540, Atlanta, GA, Dec. 2010.

A. Franchi, C. Masone, H. H. Bülthoff, and P. Robuffo Giordano. Bilateral teleoperation of multiple UAVs with decentralized bearing-only formation control. In *2011 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2215–2222, San Francisco, CA, Sep. 2011a.

A. Franchi, P. Robuffo Giordano, C. Secchi, H. I. Son, and H. H. Bülthoff. A passivity-based decentralized approach for the bilateral teleoperation of a group of UAVs with switching topology. In *2011 IEEE Int. Conf. on Robotics and Automation*, pages 898–905, Shanghai, China, May 2011b.

A. Franchi, C. Masone, V. Grabe, M. Ryll, H. H. Bülthoff, and P. Robuffo Giordano. Modeling and control of UAV bearing-formations with bilateral high-level steering. *The International Journal of Robotics Research, Special Issue on 3D Exploration, Mapping, and Surveillance*, 31(12): 1504–1525, 2012a.

A. Franchi, C. Masone, and P. Robuffo Giordano. A synergetic high-level/reactive planning framework with application to human-assisted navigation. In *2012 IEEE IROS Work. on Real-time Motion Planning: Online, Reactive, and in Real-time*, Vilamoura, Portugal, Oct. 2012b.

A. Franchi, C. Secchi, M. Ryll, H. H. Bülthoff, and P. Robuffo Giordano. Shared control: Balancing autonomy and human assistance with a group of quadrotor UAVs. *IEEE Robotics & Automation Magazine, Special Issue on Aerial Robotics and the Quadrotor Platform*, 19(3):57–68, 2012c.

A. Franchi, C. Secchi, H. I. Son, H. H. Bülthoff, and P. Robuffo Giordano. Bilateral teleoperation of groups of mobile robots with time-varying topology. *IEEE Trans. on Robotics*, 28(5):1019–1033, 2012d.

A. R. Girard, A. S. Howell, and J. K. Hedrik. Border patrol and surveillance missions using multiple unmanned air vehicles. In *2004 IEEE Conf. on Decision and Control*, volume 1, pages 620–625, 2004.

M. A. Goodrich and A. C. Schultz. Human-robot interaction: a survey. *Foundations and Trends in Human-Computer Interaction*, 1(3):203–275, 2007.

V. Grabe, H. H. Bülthoff, and P. Robuffo Giordano. On-board velocity estimation and closed-loop control of a quadrotor UAV based on optical flow. In *2012 IEEE Int. Conf. on Robotics and Automation*, pages 491–497, St. Paul, MN, May 2012.

V. Grabe, M. Riedel, H. H. Bülthoff, P. Robuffo Giordano, and A. Franchi. The TeleKyb framework for a modular and extendible ROS-based quadrotor control. In *6th European Conference on Mobile Robots*, Barcelona, Spain, Sep. 2013.

P. R. Grant and L. D. Reid. Motion washout filter tuning: rules and requirements. *J. of Aircraft*, 34(2):145–151, 1997.

E. L. Groen and W. Bles. How to use body tilt for the simulation of linear self motion. *Journal of Vestibular Research*, 14(5):375–385, 2004.

S. Haddadin, A. Albu-Schäffer, A. D. Luca, and G. Hirzinger. Collision detection and reaction: A contribution to safe physical human-robot interaction. In *2008 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3356–3363, 2008.

M. Hägele, K. Nilsson, and J. N. Pires. Industrial robotics. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, pages 963–986. sv, 2008.

J. Hauser and R. Hindman. Aggressive flight maneuvers. In *1997 IEEE Conf. on Decision and Control*, volume 5, pages 4186–4191, 1997.

J. Heindl, M. Otter, H. Hirschmueller, M. Fromberger, F. Siegert, and H. Heinrich. The robocoaster simulation platform, path and video generation for an authentic mars flight simulation. In *Proceedings ISR-2006, Joint Conference on Robotics / ROBOTIK 2006*, München, 2006.

L. J. Hettinger and M. W. Haas. *Virtual and adaptive environments: applications, implications and human performance issues*. LEA, 2009.

G. Hirzinger, B. Brunner, J. Dietrich, and J. Heindl. Rotex-the first remotely controlled robot in space. In *1994 IEEE Int. Conf. on Robotics and Automation*, volume 3, pages 2604–2611, 1994.

N. Hogan. Controlling impedance at the man/machine. In *1989 IEEE Int. Conf. on Robotics and Automation*, pages 1626–1631, Scottsdale, AZ, May 1989.

P. F. Hokayem and M. W. Spong. Bilateral teleoperation: An historical survey. *Automatica*, 42(12):2035–2057, 2006.

A. Howard, L. E. Parker, and G. S. Sukhatme. Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection. *The International Journal of Robotics Research*, 25(5-6): 431–447, 2006.

A. Isidori. *Nonlinear Control Systems, 3rd edition.* Springer, 1995. ISBN 3540199160.

E. N. Johnson, A. J. Calise, R. Sattigeri, Y. Watanabe, and V. Madyastha. Approaches to vision-based formation control. In *43th IEEE Conf. on Decision and Control*, pages 1643–1648, Paradise Island, Bahamas, Jan. 2004.

Y. Jungwon, B. Novandy, C.-H. Yoon, and K.-J. Park. A 6-DOF gait rehabilitation robot with upper and lower limb connections that allows walking velocity updates on various terrains. *IEEE/ASME Trans. on Mechatronics*, 15(2):201–215, 2010.

W. Käding and F. Hoffmeyer. The advanced Daimler-Benz driving simulator. In *SAE Technical Paper 950175*, 1995.

K. Kant and S. W. Zucker. Toward efficient trajectory planning: The path-velocity decomposition. *The International Journal of Robotics Research*, 5(3):72–89, 1986.

L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. on Robotics and Automation*, 12(4):566–580, 1996.

J. Kelly and G. S. Sukhatme. Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *The International Journal of Robotics Research*, 30(1):56–79, 2011.

Z. A. Khan and S. K. Agrawal. Control of longitudinal flight dynamics of a flapping-wing micro air vehicle using time-averaged model and differential

flatness based controller. In *2007 American Control Conference*, pages 5284–5289, 2007.

C. A. Kitts and I. Mas. Cluster space specification and control of mobile multirobot systems. *IEEE/ASME Trans. on Mechatronics*, 14(2):207–218, 2009.

L. Kneip, A. Martinelli, S. Weiss, D. Scaramuzza, and R. Siegwart. Closed-form solution for absolute scale velocity determination combining inertial measurements and a single feature correspondence. In *2011 IEEE Int. Conf. on Robotics and Automation*, pages 4546–4553, Shanghai, China, May 2011.

K. Kosuge and Y. Hirata. Human-robot interaction. In *2004 IEEE Int. Conf. on Robotics and Biomimetics*, pages 8–11, Aug. 2004.

R. Kristiansen, E. Oland, and D. Narayanachar. Operational concepts in UAV formation monitoring of industrial emissions. In *2012 IEEE Int. Conf. on Cognitive Infocommunications (CogInfoCom)*, pages 339–344, 2012.

J. Lächele, A. Franchi, H. H. Bülthoff, and P. Robuffo Giordano. SwarmSimX: Real-time simulation environment for multi-robot systems. In I. Noda, N. Ando, D. Brugali, and J. Kuffner, editors, *3rd Int. Conf. on Simulation, Modeling, and Programming for Autonomous Robots*, volume 7628 of *Lecture Notes in Computer Science*, pages 375–387. Springer, 2012.

T. M. Lam, V. D'Amelio, M. Mulder, and M. V. Paassen. UAV teleoperation using haptics with a degraded visual interface. In *2006 IEEE Int. Conf. on Systems, Man, and Cybernetics*, volume 3, pages 2440–2445, 2006a.

T. M. Lam, M. Mulder, and M. M. V. Paassen. Haptic feedback for UAV tele-operation - force offset and spring load modification. In *2006 IEEE Int. Conf. on Systems, Man, and Cybernetics*, volume 2, pages 1618–1623, 2006b.

T. M. Lam, H. W. Boschloo, M. Mulder, and M. M. V. Paassen. Artificial force field for haptic feedback in UAV teleoperation. *IEEE Trans. on Systems, Man, & Cybernetics. Part A: Systems & Humans*, 39(6):1316–1330, 2009.

F. Lamiraux, D. Bonnafous, and O. Lefebvre. Reactive path deformation for nonholonomic mobile robots. *IEEE Trans. on Robotics*, 20(6):967–977, 2004.

S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. TR 98-11 Computer Science Dept. Iowa State University, Oct. 1998.

S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006. ISBN 0521862051. Available at `http://planning.cs.uiuc.edu`.

S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *The International Journal of Robotics Research*, 20(5):378–400, 2001.

D. Lee and M. W. Spong. Bilateral teleoperation of multiple cooperative robots over delayed communication network: theory. In *2005 IEEE Int. Conf. on Robotics and Automation*, pages 360–365, Barcelona, Spain, Apr. 2005.

D. Lee, A. Franchi, P. Robuffo Giordano, H. I. Son, and H. H. Bülthoff. Haptic teleoperation of multiple unmanned aerial vehicles over the internet. In *2011 IEEE Int. Conf. on Robotics and Automation*, pages 1341–1347, Shanghai, China, May 2011.

D. J. Lee and K. Huang. Passive-set-position-modulation framework for interactive robotic systems. *IEEE Trans. on Robotics*, 26(2):354–369, 2010.

D. J. Lee, A. Franchi, H. I. Son, H. H. Bülthoff, and P. Robuffo Giordano. Semi-autonomous haptic teleoperation control architecture of multiple unmanned aerial vehicles. *IEEE/ASME Trans. on Mechatronics, Focused Section on Aerospace Mechatronics*, 18(4):1334–1345, 2013.

P. Leven and S. Hutchinson. A framework for real-time path planning in changing environments. *The International Journal of Robotics Research*, 21(12):999–1030, 2002.

N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1997. ISBN 1558603484.

A. A. Maciejewski and C. A. Klein. Numerical filtering for the operation of robotic manipulators through kinematically singular configurations. *J. Robot Syst.*, 5(6):527–552, 1985.

R. Mahony, T. Hamel, and J.-M. Pflimlin. Nonlinear complementary filters on the special orthogonal group. *IEEE Trans. on Automatic Control*, 53 (5):1203–1218, 2008.

D. Manocha and J. F. Canny. Detecting cusps and inflection points in curves. *Comp. Aided Geom. Design*, 9:1–24, 1992.

G. L. Mariottini, F. Morbidi, D. Prattichizzo, N. Vander Valk, N. Michael, G. Pappas, and K. Danilidis. Vision-based localization for leader-follower formation control. *IEEE Trans. on Robotics*, 25(6):1431–1438, 2009.

C. Masone. Mechanical design and control of the new 7-DOF CyberMotion simulator. In *5th Human Centered Motion Cueing Workshop at VTI*, Göteborg, Sweden, 2011.

C. Masone, P. Robuffo Giordano, and H. H. Bülthoff. Mechanical design and control of the new 7-DOF CyberMotion simulator. In *2011 IEEE Int. Conf. on Robotics and Automation*, pages 4935–4942, 2011.

C. Masone, A. Franchi, H. H. Bülthoff, and P. Robuffo Giordano. Interactive planning of persistent trajectories for human-assisted navigation of mobile robots. In *2012 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2641–2648, Vilamoura, Portugal, Oct. 2012a.

C. Masone, A. Franchi, H. H. Bülthoff, and P. Robuffo Giordano. Shared trajectory planning for human-in-the-loop navigation of mobile robots in cluttered environments. In *5th Int. Work. on Human-Friendly Robotics*, Bruxelles, Belgium, Oct. 2012b.

C. Masone, P. Robuffo Giordano, H. H. Bülthoff, and A. Franchi. Semi-autonomous trajectory generation for mobile robots with integral haptic shared control. Accapted at 2014 IEEE Int. Conf. on Robotics and Automation, 2014a.

C. Masone, P. Robuffo Giordano, H. H. Bülthoff, and A. Franchi. A real-time framework for semi-autonomous motion planning and control of mobile robots with integral haptic feedback. submitted to IEEE Trans. on Robotics, 2014b.

A. S. Matveev, H. Teimoori, and A. Savkin. A method for navigation of an autonomous vehicle for border patrol. In *2010 American Control Conference*, pages 6187–6190, 2010.

J. S. McCarley and C. D. Wickens. Human factors implications of UAVs in the national airspace. Technical report, University of Illinois, 2005.

D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE Int. Conf. on Robotics and Automation*, pages 2520–2525, 2011.

C. D. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, 2001.

N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar. The GRASP multiple micro-UAV testbed. *IEEE Robotics & Automation Magazine*, 17(3): 56–65, 2010.

V. Mistler, A. Benallegue, and N. K. M'Sirdi. Exact linearization and noninteracting control of a 4 rotors helicopter via dynamic feedback. In *10th IEEE Int. Symp. on Robots and Human Interactive Communications*, pages 586–593, Bordeaux, Paris, France, Sep. 2001.

N. Moshtagh, N. Michael, A. Jadbabaie, and K. Daniilidis. Vision-based, distributed control laws for motion coordination of nonholonomic robots. *IEEE Trans. on Robotics*, 25(4):851–860, 2009.

M. Mulder, J. A. Mulder, and H. G. Stassen. Cybernetics of tunnel-in-the-sky displays. I. straight trajectories. In *IEEE Int. Conf. on Systems, Man, and Cybernetics*, volume 5, pages 1082–1087, 1999a.

M. Mulder, J. A. Mulder, and H. G. Stassen. Cybernetics of tunnel-in-the-sky displays. II. curved trajectories. In *IEEE Int. Conf. on Systems, Man, and Cybernetics*, volume 5, pages 1088–1093, 1999b.

M. Mulder, D. A. Abbink, and E. R. Boer. Sharing control with haptics: Seamless driver support from manual to automatic control. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 54 (52):786–798, 2012.

R. R. Murphy. Human-robot interaction in rescue robotics. *IEEE Trans. on Systems, Man, & Cybernetics. Part C: Applications and Reviews*, 34 (2):138–153, May 2004a.

R. R. Murphy. Trial by fire [rescue robots]. *IEEE Robotics & Automation Magazine*, 11(3):50–61, 2004b.

R. R. Murphy and D. Schreckenghost. Survey of metrics for human-robot interaction. In *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 197–198, 2013.

R. R. Murphy, S. Tadokoro, D. Nardi, A. Jacoff, P. Fiorini, H. Choset, and A. M. Erkmen. Search and rescue robotics. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, pages 1151–1173. Springer, 2008.

R. R. Murphy, K. L. Dreger, S. Newsome, J. Rodocker, E. Steimle, T. Kimura, K. Makabe, , F. Matsuno, S. Tadokoro, and K. Kon. Use of remotely operated marine vehicles at minamisanriku and rikuzentakata japan for disaster recovery. In *2011 IEEE Int. Symp. on Safety, Security and Rescue Robotics*, pages 19–25, Kyoto, Japan, Nov. 2011.

R. M. Murray, M. Rathinam, and W. Sluis. Differential flatness of mechanical control systems: A catalog of prototype systems. In *Proceedings of the 1995 ASME International Congress and Exposition*, 1995.

L. Nehaoua, H. Arioui, and S. Mammar. Review on single track vehicle and motorcycle simulators. In *2011 19th Mediterranean Conference on Control Automation (MED)*, pages 940–945, 2011.

B. J. Nelson and P. K. Khosla. Strategies for increasing the tracking region of an eye-in-hand system by singularity and joint limit avoidance. *The International Journal of Robotics Research*, 14(3):255–269, 1995.

G. Niemeyer, C. Preusche, and G. Hirzinger. Telerobotics. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, pages 741–757. Springer, 2008.

F. M. Nieuwenhuizen and H. H. Bülthoff. The MPI CyberMotion simulator: A novel research platform to investigate human control behavior. *Journal of Computing Science and Engineering*, 7(2):122–131, 2013.

M. J. V. Nieuwstadt and R. M. Murray. Real-time trajectory generation for differentially flat systems. *International Journal on Robust and Nonlinear Control*, 8:995–1020, 1998.

H.-G. Nusseck, H. J. Teufel, F. M. Nieuwenhuizen, and H. H. Bülthoff. Learning system dynamics: Transfer of training in a helicopter hover simulator. In *Proc. of the AIAA Modeling and Simulation Technologies Conference*, pages 1–11, 2008.

O. A. A. Orqueda and R. Fierro. Robust vision-based nonlinear formation control. In *2006 American Control Conference*, pages 1422–1427, Minneapolis, MN, Jun. 2006.

T. Ortmaier, M. Groger, D. H. Boehm, V. Falk, and G. Hirzinger. Motion estimation in beating heart surgery. *IEEE Trans. on Biomedical Engineering*, 52(10):1729–1740, 2005.

J. Peng and S. Akella. Coordinating multiple robots with kinodynamic constraints along specified paths. *The International Journal of Robotics Research*, 24(4):295–310, 2005.

Q.-C. Pham. Fast trajectory correction for nonholonomic mobile robots using affine transformations. In *2011 Robotics: Science and Systems*, June 2011.

Q.-C. Pham and Y. Nakamura. Regularity properties and deformation of wheeled robot trajectories. In *2012 IEEE Int. Conf. on Robotics and Automation*, pages 3212–3217, May 2012.

L. Pollini, M. Innocenti, and A. Petrone. Novel motion platform for flight simulators using an anthropomorphic robot. *AIAA Journal of Aerospace Computing, Information, and Communication*, 5:175–196, 2008.

P. Pretto, H.-G. Nusseck, H. J. Teufel, and H. H. Bülthoff. Effect of lateral motion on drivers' performance in the MPI motion simulator. In *Driving Simulation Conference Europe*, 2009.

M. Quigley, M. A. Goodrich, and R. W. Beard. Semi-autonomous human-UAV interfaces for fixed-wing mini-UAVs. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 2457–2462 vol.3, 2004.

S. Quinlan and O. Khatib. Elastic bands: Connecting path planning and control. In *1993 IEEE Int. Conf. on Robotics and Automation*, pages 802–807, Atlanta, GA, May 1993.

A. Renzaglia, L. Doitsidis, A. Martinelli, and E. B. Kosmatopoulos. Multi-robot three dimensional coverage of unknown areas. *The International Journal of Robotics Research*, 31(6):738–752, 2012.

M. Riedel, A. Franchi, H. H. Bülthoff, P. Robuffo Giordano, and H. I. Son. Experiments on intercontinental haptic control of multiple UAVs. In *12th Int. Conf. on Intelligent Autonomous Systems*, pages 227–238, Jeju Island, Korea, Jun. 2012.

P. Robuffo Giordano, H. Deusch, J. Lächele, and H. H. Bülthoff. Visual-vestibular feedback for enhanced situational awareness in teleoperation of UAVs. In *AHS International 66th Annual Forum*, pages 2809–2818, 2010a.

P. Robuffo Giordano, C. Masone, J. Tesch, M. Breidt, L. Pollini, and H. H. Bülthoff. A novel framework for closed-loop robotic motion simulation - part I: Inverse kinematics design. In *2010 IEEE Int. Conf. on Robotics and Automation*, pages 3876–3883, 2010b.

P. Robuffo Giordano, C. Masone, J. Tesch, M. Breidt, L. Pollini, and H. H. Bülthoff. A novel framework for closed-loop robotic motion simulation - part II: Motion cueing design and experimental validation. In *2010 IEEE Int. Conf. on Robotics and Automation*, pages 3896–3903, 2010c.

P. Robuffo Giordano, A. Franchi, C. Secchi, and H. H. Bülthoff. Bilateral teleoperation of groups of UAVs with decentralized connectivity maintenance. In *2011 Robotics: Science and Systems*, Los Angeles, CA, Jun. 2011a.

P. Robuffo Giordano, A. Franchi, C. Secchi, and H. H. Bülthoff. Experiments of passivity-based bilateral aerial teleoperation of a group of UAVs with decentralized velocity synchronization. In *2011 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 163–170, San Francisco, CA, Sep. 2011b.

E. J. Rodríguez-Seda, J. J. Troy, C. A. Erignac, P. Murray, D. M. Stipanović, and M. W. Spong. Bilateral teleoperation of multiple mobile agents: Coordinated motion and collision avoidance. *IEEE Trans. on Control Systems Technology*, 18(4):984–992, 2010.

L. B. Rosenberg. Virtual fixtures: Perceptual tools for telerobotic manipulation. In *1993 IEEE Virtual Reality Annual Int. Symp.*, pages 76–82, Seattle, WA, 1993.

I. Sa and P. Corke. Replanning with RRTs. In *2006 IEEE Int. Conf. on Robotics and Automation*, May 2006.

J. Scholtz and S. Bahrami. Human-robot interaction: development of an evaluation methodology for the bystander role of interaction. In *2003 IEEE Int. Conf. on Systems, Man and Cybernetics*, volume 4, pages 3212–3217, 2003.

M. Schwager, B. J. Julian, and D. Rus. Optimal coverage for multiple hovering robots with downward facing cameras. In *2009 IEEE Int. Conf. on Robotics and Automation*, pages 3515–3522, Kobe, Japan, May 2009.

M. Schwager, B. Julian, M. Angermann, and D. Rus. Eyes in the sky: Decentralized control for the deployment of robotic camera networks. *Proceedings of the IEEE*, 99(9):1541–1561, 2011.

C. W. Schwarz. Two mitigation strategies for motion system limits in driving and flight simulators. *IEEE Trans. on Systems, Man, & Cybernetics. Part A: Systems & Humans*, 37(4):562–568, 2007.

C. Secchi, A. Franchi, H. H. Bülthoff, and P. Robuffo Giordano. Bilateral teleoperation of a group of UAVs with communication delays and switching topology. In *2012 IEEE Int. Conf. on Robotics and Automation*, pages 4307–4314, St. Paul, MN, May 2012.

K. M. Seiler, S. P. Singh, S. Sukkarieh, and H. Durrant-Whyte. Using Lie group symmetries for fast corrective motion planning. *The International Journal of Robotics Research*, 31(2):151–166, 2011.

R. V. O. Shakernia and S. Sastry. Formation control of nonholonomic mobile robots with omnidirectional visual servoing and motion segmentation. In *2003 IEEE Int. Conf. on Robotics and Automation*, pages 584–589, Taipei, Taiwan, Sep. 2003.

T. B. Sheridan. *Telerobotics, Automation, and Human Supervisory Control*. MIT Press, Cambridge, MA, USA, 1992.

T. B. Sheridan and W. L. Verplanck. Human and computer control of undersea teleoperators. Technical report, MIT Man-Machine Laboratory, 1978.

Z. Shiller. On singular time-optimal control along specified paths. *IEEE Trans. on Robotics and Automation*, 10(4):561–566, 1994.

B. Siciliano. Kinematic control of redundant robot manipulators: A tutorial. *Journal of Intelligent and Robotic Systems*, 3(3):201–212, 1990.

B. Siciliano. On the use of quaternions for robot interaction control tasks. In *Proceedings of the 5th Int. Symposium on Methods and Models in Automation and Robotics*, 1998.

B. Siciliano and J.-J. Slotine. A general framework for managing multiple tasks in highly redundant robotic systems. In *Advanced Robotics, 1991. 'Robots in Unstructured Environments', 91 ICAR., Fifth International Conference on*, volume 2, pages 1211–1216, 1991.

B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics: Modelling, Planning and Control*. Springer, 2009.

R. Siuikat. The new dynamic driving simulator at DLR. In *Driving Simulator Conference 2005*, pages 374–381, Orlando, FL, USA, 2005.

M. Sivak. The information that drivers use: is it indeed 90% visual? *Perception*, 25(9):1081–1089, 1996.

J.-J. Slotine and H. S. Yang. Improving the efficiency of time-optimal path-following algorithms. *IEEE Trans. on Robotics and Automation*, 5 (1):118–124, 1989.

R. Smith, M. Schwager, S. L. Smith, D. Rus, and G. Sukhatme. Persistent ocean monitoring with underwater gliders: Towards accurate reconstruction of dynamic ocean processes. In *2011 IEEE Int. Conf. on Robotics and Automation*, pages 1517–1524, 2011.

S. L. Smith, M. Schwager, and D. Rus. Persistent robotic tasks: Monitoring and sweeping in changing environments. *IEEE Trans. on Robotics*, 28 (2):410–426, 2012.

F. Soyka, P. Robuffo Giordano, K. Beykirch, and H. H. Bülthoff. Predicting direction detection thresholds for arbitrary translational acceleration profiles in the horizontal plane. *Experimental Brain Research*, 209(1): 95–107, 2011.

F. Soyka, P. Robuffo Giordano, M. Barnett-Cowan, and H. H. Bülthoff. Modeling direction discrimination thresholds for yaw rotations around an earth-vertical axis for arbitrary motion profiles. *Experimental Brain Research*, 220(1):89–99, 2012.

D. Stewart. A platform with six degrees of freedom. *Proceedings of the Institution of Mechanical Engineers*, 180(1):371–386, 1965.

M. C. Stone and T. D. DeRose. A geometric characterization of parametric cubic curves. *ACM Trans. Graph.*, 8(3):147–163, 1989.

S. Thrun. Toward a framework for human-robot interaction. *Human–Computer Interaction*, 19(1-2):9–24, 2004.

K. Waldron and J. Schmiedeler. Kinematics. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, pages 9–33. Springer, 2008.

M. Wentink, W. Bles, R. Hosman, and M. Mayrhofer. Design & evaluation of spherical washout algorithm for Desdemona simulator. In *AIAA Modeling and Simulation Technologies Conference and Exhibit*, San Francisco, CA, USA, 2005.

J. G. W. Wildenbeest, D. A. Abbink, C. J. M. Heemskerk, F. C. T. van der Helm, and H. Boessenkool. The impact of haptic feedback quality on the performance of teleoperated assembly tasks. *IEEE Transactions on Haptics*, 6(2):242–252, 2013.

E. Yoshida and F. Kanehiro. Reactive robot motion using path replanning and deformation. In *2006 IEEE Int. Conf. on Robotics and Automation*, pages 5456–5462, May 2006.

T. Yoshikawa. Manipulability of robotic mechanisms. *The International Journal of Robotics Research*, 4(2):3–9, 1985.

S. Yuta, H. Asama, S. Thrun, E. Prassler, and T. Tsubouchi. *Field and Service Robotics*, volume 24 of *Springer Tracts in Advanced Robotics*. Springer, 2006.