UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II
SCUOLA DI DOTTORATO IN
INGEGNERIA INFORMATICA ed AUTOMATICA
DIPARTIMENTO DI INGEGNERIA ELETTRICA
E TECNOLOGIE DELL' INFORMAZIONE

A Control Architecture for *Unmanned Aerial Vehicles* Operating in Human-Robot Team for Service Robotic Tasks

Jonathan Cacace

jonathan.cacace@unina.it

In Partial Fulfillment of the Requirements for the Degree of
PHILOSOPHIAE DOCTOR in
Computer Science and Automation Engineering

March 2016

TUTOR
Prof. Vincenzo Lippiello

COORDINATOR
Prof. Francesco Garofalo

UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

SCUOLA DI DOTTORATO IN
INGEGNERIA INFORMATICA ED AUTOMATICA

Date: **March 2016**

Author: **Jonathan Cacace**

Title: **A Control Architecture for *Unmanned Aerial Vehicles* Operating in Human-Robot Team for Service Robotic Tasks**

Department: **DIPARTIMENTO DI INGEGNERIA ELETTRICA E TECNOLOGIE DELL' INFORMAZIONE**

Degree: **PHILOSOPHIAE DOCTOR**

...to my parents

Acknowledgments

First, I deeply thank my advisor Prof. Vincenzo Lippiello. With his patience, kindness and invaluable support have been the key point for my professional growth. I have to say a huge "*thank you*" also to Prof. Bruno Sicliano who made possible this Ph.D experience and all my travelings all around the world. I have to thank also my international "local colleagues" of the PRISMA group. They enjoyed with me a lot of time also out of the "standard" situations and working days. I thank also all the professors, young colleagues, students and friends met during these years in summer schools, European projects meetings and in the robotics conferences and seminars. *Grazie*!

The research leading to these results has been supported by the ARCAS and SHERPA collaborative projects, which both have received funding from the European Community Seventh Framework Programme (FP7/2007-2013) under grant agreements ICT-287617 and ICT-600958, respectively.

Napoli, March 2016

Jonathan Cacace

Summary

In the last decade the use of *Unmanned Aerial Vehicles* has exponentially grown especially due to recent research progresses. Thanks to their fast navigation ability these kind of platforms are particularly suitable for different service applications such as the building inspection, the surveillance, the remote manipulation and others. As consequences, the use of these platforms in service robotics is becoming an attractive field and a cutting-edge research topic, and it has been able to attract the attention of several international research groups. In such a scenario, the European project SHERPA (FP7-ICT-600958), under which this thesis has been partially developed, intends to develop a mixed ground and aerial heterogeneous robotic platform leaded by a human operator to assist him in *Search & Rescue* missions in alpine scenarios. Moreover, different other research projects aim to use *UAVs* in service robotics applications such as the remote non-destructive material analysis or cooperative assembly and building construction via aerial manipulation. Those are the cases of European research projects *AIRobot* and *ARCAS* respectively. In this thesis, a hierarchical control architecture suitable for Aerial Robotic Platforms able to perform service tasks autonomously and in closed loop with a human operator is presented. In this context, the problem of controlling a robot has been addressed at different level of abstraction in order to integrate *High-level* functionalities needed to interact in an intelligent way with a human operator, with *Middle-level* autonomous action execution activities and *Low-level* control functionalities. In particular, in the *High-level* module the attention has been focussed on selecting and commanding a robot in a multi-robot system in order to enable a more effective and natural interaction with the use of multimodal voice and gestures commands combination. In the *Middle-level* module the action *planning & execution* problem has been addressed proposing different methodologies for autonomous and semi-autonomous action execution relying on *Mixed-Initiative* interaction. Finally the *Low-level* control problem of a Vertical Take off and Landing *VToL UAV* has been addressed. In this context, the proposed architecture allow the robot to stabilize and track desired trajectories in a robust way through the compensation of external unmodeled disturbances. In addition a hybrid visual servoing with a hierarchical task composition control framework is presented in order to allow the *UAV* eventually equipped with a light manipulator, to interact in a intelligent way with the environment in which it operates via aerial manipulation.
In detail, the outline of this thesis is organized as follows.

- Chapter 1 is a general introduction underlining the need to make a robot autonomous and able to interact with human operators and operate in unstructured scenarios to cope with service robotics and industrial applications requests. The research project that motivates this work are then introduced.
- Chapter 2 addresses the problem of *Human-Robot-Interaction* between a human operator and a multi-flying robot system in hazardous and hostile environment supported by multi-modal interaction.
- Chapter 3 introduces novel approaches for autonomous and semi-autonomous action execution relying on *Mixed-Initiative* interaction in unstructured and cluttered environment considering sliding autonomy between the human operator and the autonomous system of the *UAV*.
- Chapter 4 is focussed on the *Low-level control* of *VToL UAV* platforms presenting novel methodologies for robust stabilization and desired trajectories tracking and aerial manipulation with the use of visual servoing techniques.
- Chapter 5 contains conclusion, remarks and proposals for possible developments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The term *robot* derives from the term *robota* which means executive labour in Slav languages. As well, *robotics* is commonly defined as the science studying the *intelligent connection between perception and action* [95]. These two last definitions show how perfectly a robot fits into the above human being's desires: these last, besides, can be accomplished if and only if the *three fundamental laws* introduced by Asimov are respected. These laws established rules of behaviour to consider as specifications for the design of a robot, and they are namely:

- A robot may not injure a human being or, through inaction, allow a human being to come to harm.
- A robot must obey the orders given by human beings, except when such orders would conflict with the first law.
- A robot must protect its own existence, as long as such protection does not conflict with the first and the second law.

Over the course of centuries, human beings tried to design and to build new machines first to help themselves in the execution of several tasks, and then to completely replace themselves, especially in the most dangerous works. In a short time, this desire about having machines in substitution of human being in physical activities has been caught up as well by the desire to substitute him in decision making tasks. Nowadays, robots are widely used in industrial applications for such works where human being would have more risk for his life, more cost per hour and more stress for his body. The connotation of a robot for industrial applications is that of operating in a *structured environment* whose geometrical characteristics are mostly known a priori.

Hence, operating in scarcely structured or *unstructured environments* – where the geometrical characteristics are not known a priori –, even with or in cooperation with humans, it is not possible without robots with a marked characteristics of autonomy. The expression *advanced robotics* usually refers to this framework, in which the ability in decision making tasks plays a relevant role. Advanced robotics is still a young discipline

and therein several researchers are motivated to investigate solutions which could be the answer to the growing need of autonomous robots for domestic and service applications, but also for new industrial requests.

Therefore, robotic systems of the next decade will be, potentially, a part of everyday life as helpers and elder care companions, assisting surgeons in medical operations, intervening in hazardous or life-critical environments for search and rescue operations and so on. Personal and service robots will thus be found in all domains of our future life, and they represent not only a hope for a more convenient world but also a massive new market for leading-edge technology industry and significant business opportunities, especially for industries. Only a few of the technologies required to build functional personal and service robots already exist at the component level and markets for these products are getting gradually into place. Continuous research and development efforts are required to combine the different technologies, create new products and services, enhance the existing ones for a wide range of possible applications.

Unmanned Aerial Vehicle represents a relatively new kind of robotic system that recently has been used in different robotic applications. These kind of platforms are capable of flying without the presence of a on-board human operator, both via remote teleoperation and autonomous guidance. In particular, the Rotary Wing UAV (*RW-UAV*) is the category of UAV where the flying system is composed by different rotor blades (different configuration exits, from 3 to 16 rotors) that revolve around a fixed mast. The most important skill of these kind of robots is their capacity to hover and perform agile manoeuvring that makes rotary wing *UAVs* well suited to applications like inspections and all the application where precision manoeuvring and the ability to maintain a visual on a single target for extended periods of time is required. In addition, research progresses make possible to plug a light manipulator downward the *UAV* platform in order to perform aerial manipulation tasks. Obviously the autonomous execution of tasks by means of *UAV* platforms is strictly related to the possibility to controlling and commanding it in an easy way.

In the followings two projects that motivates this work are presented:

1.1 Arcas Project

The ARCAS project (Fig. 1.1) proposes the development and experimental validation of the first cooperative free-flying robotic system for assembly and structure construction in order to provide integrated and consolidated scientific foundations for flying robot perception, planning and control. In particular, ARCAS will produce a framework for the design and development of cooperating flying robots for assembly operations. The integration of these functionalities has the aim to pave the way for new applications and services in aerial and space robotics. The building of platforms for the evacuation of people in rescue operations, the installation of platforms in uneven terrains for landing of manned and unmanned VTOL aircraft, the cooperative inspection and maintenance and the construction

of structures, are some examples of aerial robotics' potential.

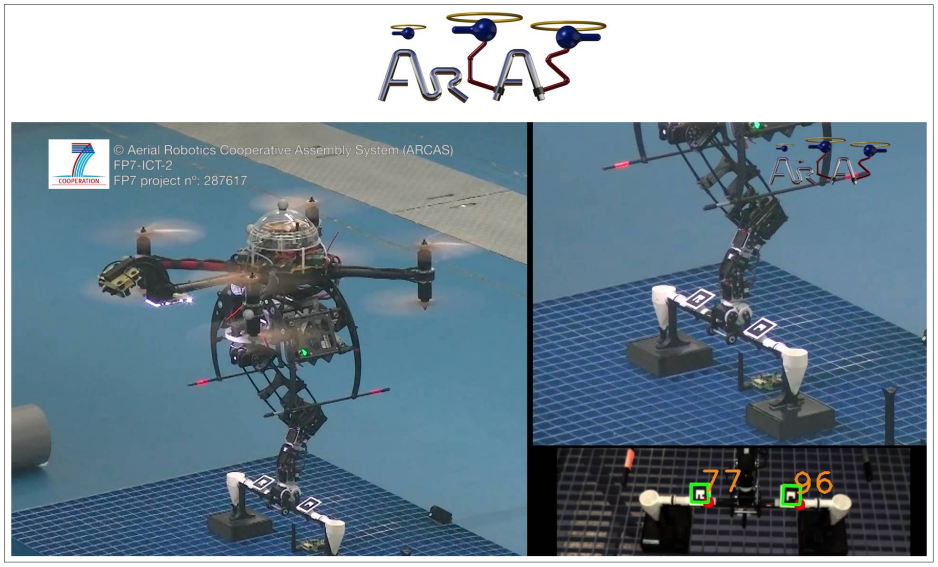

Figure 1.1: The ARCAS project.

The detailed scientific and technological objectives are:

- New methods for motion control of a free-flying robot with mounted manipulator in contact with a grasped object as well as for coordinated control of multiple co-operating flying robots with manipulators in contact with the same object (e.g. for precise placement or joint manipulation);
- New flying robot perception methods to model, identify and recognize the scenario and to be used for the guidance in the assembly operation, including fast generation of 3D models, aerial 3D SLAM, 3D tracking and cooperative perception;
- New methods for the cooperative assembly planning and structure construction by means of multiple flying robots with application to inspection and maintenance activities;
- Strategies for operator assistance, including visual and force feedback, in manipulation tasks involving multiple cooperating flying robots.

The achievement of the research objectives within ARCAS will have an important impact toward the achievements of robust and versatile behaviour of artificial systems in open-ended environments providing intelligent response in unforeseen situation.

The main contribution of this thesis to the ARCAS project is related to the *Low-level* control of the robot (composed by the *UAV* system coupled with its manipulator

arm). In this context, a novel hybrid visual servoing with a hierarchical task-composition control framework for aerial manipulation allowing the robot to physically interact with the operating environment is presented. In addition, a passivity-based control enhanced with an estimator for unmodeled dynamics and external wrench acting on the robot and based on the momentum of the system has been implemented in order to compensate the disturbances effect, such as the movement of the UAV manipulator during the flight.

1.2 Sherpa project

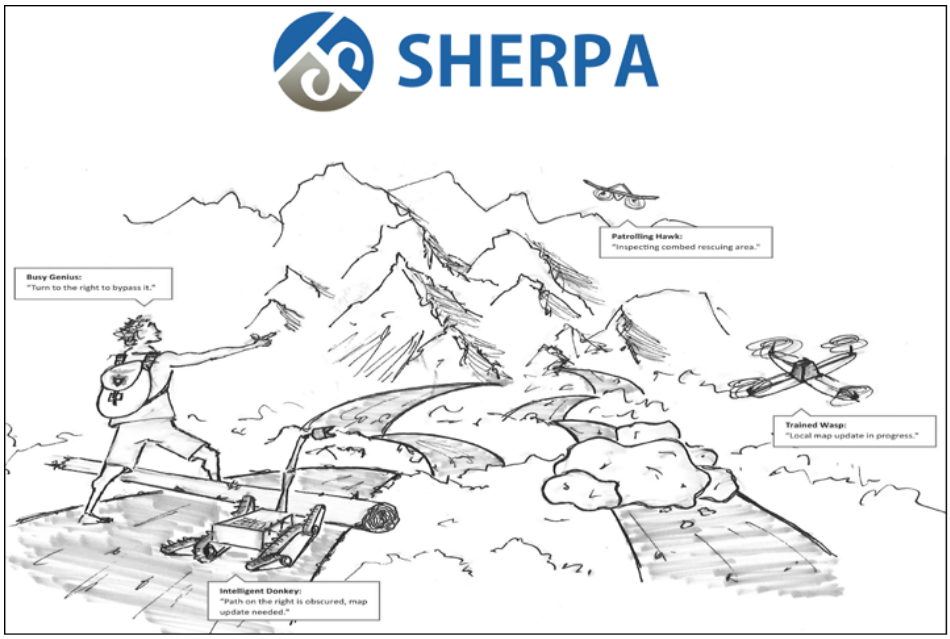

Figure 1.2: The SHERPA project.

The goal of SHERPA project (Fig. 1.2) is to develop a mixed ground and aerial robotic platform to support *Search & Rescue* operations in a real-world and hostile environment like the alpine scenario. The technological platform and the alpine rescuing scenario are the occasion to address a number of research topics about Artificial Intelligence cognition and Low level control. What makes the project potentially very rich from a scientific viewpoint is the heterogeneity and the capabilities to be owned by the different actors of the SHERPA system: the human rescuer is the *busy genius*, working in team with the ground vehicle, as the *intelligent donkey*, and with the aerial platforms, i.e. the *trained wasps* and *patrolling hawks*. Indeed, the research activity focuses on how the*busy genius* and the *SHERPA animals* interact and collaborate with each other, with their own features and capabilities, toward the achievement of a common goal. A mix of advanced control

and cognitive capabilities characterize the SHERPA system, aiming to support the rescuer by improving his awareness of the rescue scene even in tough environments and with the *genius* often *busy* in the rescuing activity (and thus unable to supervise the platform). Thus emphasis is placed on robust autonomy of the platform, acquisition of cognitive capabilities, collaboration strategies, natural and implicit interaction between the *genius* and the *SHERPA animals*, which motivate the research activity. Five benchmarks, inspired by real-world scenarios, drive the research and motivate demonstration activities on realistic testing sites planned during the project.

The main contribution of this thesis to the SHERPA project if framed in the multi-modal human-robot interaction context in order to allow the *busy genius* to interact in an easy and natural way with the robots of the SHERPA platform. In this context, novel methodologies to command a single robot in a multi-robot system using a multi-modal combination of voice, gestures are presented. In addition, *Mixed-Initiative* interaction has been experimented to enhance the autonomous or semi-autonomous action *Planning & Execution* of the robot.

Chapter 2

Human-Robot-Interaction

Human-robot interaction is the study of interactions between humans and robots. It is a multidisciplinary field with contributions from *human-computer interaction*, *artificial intelligence*, *robotics*, *natural language understanding*, and *social sciences*. With recent progresses in the robotics field and the advent of intelligent robots, human-robot interaction has gained importance as a research topic. This interaction has been deemed necessary to design and build effective robot systems which include human users. In this context, this chapter is mainly focussed on the problem of the interaction of a human operator with a heterogeneous multi-flying robot team in hazardous and hostile environment in which the human operator must be able to lead the robotic team in the execution of Search & Rescue missions. To interact with a heterogeneous multi-robot team the human operator must be able to select and command a desired robot in an efficient way during the execution of a collaborative task. In the followings the proposed approaches to support the human operator in selecting and commanding phases are presented.

2.1 Robot Selection Problem

In this context, a system suitable for human multi-robot interaction that supports the operator in the robot selection process is presented. The proposed framework allows a human to issue commands the to a robotic team without an explicit robot selection, in so enabling a more fluent and effective interaction. This work is framed in the operative context of the SHERPA project, which proposes the deployment of a robotic platform for Search & Rescue in an alpine scenario and assumes the presence of a human rescuer that can orchestrate the robots operations with multimodal commands. Implicit robot selection is mainly motivated by fast communication and the difficulties to distinguish different robots of similar shape in a hazardous environment and in adverse weather conditions. In the proposed approach, each robot of the team can evaluate the probability to be referred in an incomplete command, considering its actual capabilities along with geometrical and contextual information. In order to facilitate the collaboration between a human operator

and a multi-robot system, the robot ability to interpret non verbal cues and react accordingly is a crucial issue, in particular when complex interactive tasks are to be executed [139]. In this paper, a framework for implicit selection of robots in human multi-robot interaction is presented. Specifically, we will address a *robot selection problem*, in which the human operator must designate a particular robot of interest as the selected one in order to assign a task. We propose an approach where the human operator can omit to explicitly indicate the intended robot in task assignment, because the robotic team is able to infer the candidate that best match the operator's intention. Our work is framed in the context of SHERPA project [138] whose goal is to develop a mixed ground and aerial robotic platform for search and rescue operations in an alpine environment. A sketch of SHERPA scenario is depicted in Figure 2.1. In this domain a human rescuer leads a team of heterogeneous robots in the search of survivors after an avalanche. The robotic team is mainly composed of aerial vehicles (in this paper we will assume quadcopters) equipped with different types of sensors in order to assist the human operator in the rescue mission. In this context, the human operator is not fully dedicated to the control of the robots, but he is involved in the rescue operations too. In order to enable more effective and natural the interaction with the robots, the human operator is endowed with wearable devices (*gesture control armband*, *headset*, *etc...*) and mobile devices (*tablet*) that allows him to orchestrate the team operations using voice and gestures commands, while receiving back relevant audio/video information about the mission. As for the robotic platforms, we will mainly refer to electrical flying robots whose main limitation is the battery duration. In order to address this issue, the robotic team includes a ground rover that works as a docking station for the drones supporting both landing and battery recharging. In this scenario, depending on the battery charge, we have a continuous reconfiguration of the active members of the robotic team and their actual capabilities. Selecting and commanding individual robots in this setting, without the support of specialized user interface, could represent a challenge. For this reason, we propose a method in which the human operator is able to select the desired robot in a non verbal and implicit manner. Specifically, we propose a method in which each available robot can evaluate the probability to be the one designated by the human for the execution of a command. The single robot evaluation process rely on a multi-layered architecture in which a *Dynamic Bayesian Network* is deployed to infer the human intentions form the state of the robots and a learned contextual and geometrical information. Finally, in order to train the system and test its effectiveness, we defined a simulated interaction scenario where a human operator is to lead a group of drones in a search mission in alpine scenario.

2.1.1 Motivation

We consider the following motivational scenario. The human operator is involved in a rescue mission after an avalanche in adverse weather conditions and decides to exploit the fast flying capabilities of the *UAVs* to retrieve important information by scanning quickly a large area. In order to accomplish this mission, the robot responsible of the scanning

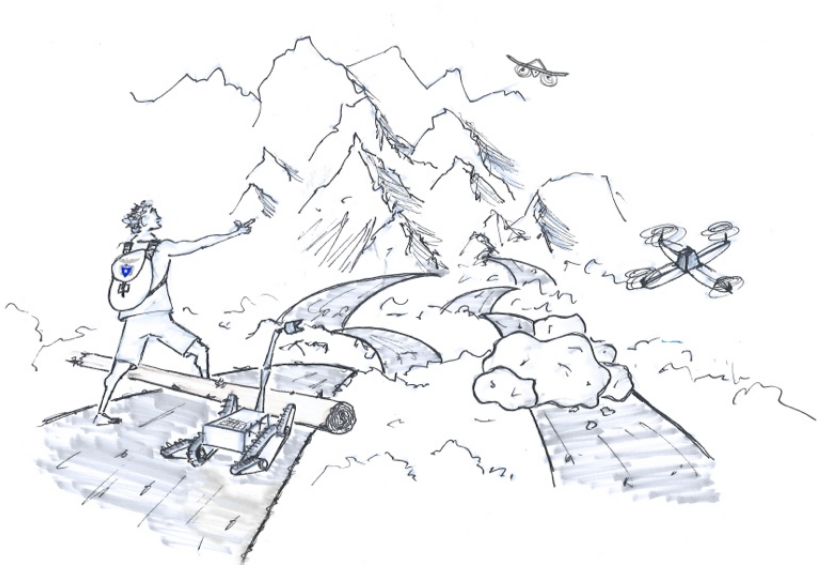

Figure 2.1: A sketch of the SHERPA scenario [138]

must be properly selected and this is usually performed by selecting the desired robot using vocal keywords such as the *code-name* or the *id* of the desired robot (e.g *Red Wasp* or *Wasp Zero*). However, the hard operative domain caused by the adverse weather conditions or the hazardous environment along with the limited time to accomplish the rescue mission affect the psychophysical state of the rescuer and his cognitive capacity. Moreover, even thought the human operator is co-located with the robotic team, the similar shape of different robots could make difficult for him to select the desired robot using verbal communication without running into misunderstanding on the correct robot *id*, in so provoking the selection of robot with different capabilities or resources or in different location. Furthermore, the operative environment presents different unmodelled sound noise sources such as the wind, the propeller of the robots or the helicopters used for the transportation of the rescuer or the victims. This could even increase the failure rate of the automatic speech recognition algorithm affecting the effectiveness of the overall interaction system.

2.1.2 Related Work

Unmanned Aerial Vehicles (*UAVs*) are extensively employed in several service applications such as in industrial building inspection [140][141], surveillance, remote sensing and many others. Related to our scenario, in [142] and [143] Search & Rescue operations are supported by the *UAVs* demonstrating the benefits that they can provide to this domain. In [144] a mixed-initiative system for the supervision of multiple drones suitable

for Search & Rescue mission in alpine scenario is presented. In [145] an *ad-hoc* user interface has been designed to interact with a multi-robot system composed of very large groups of robots. Another approach to robot selection problem is presented in [146]. In this work, a solution that rely on face engagement is proposed where eye contact between the human and the robot is used to select a robot to command. Approaches of this kind are not feasible in our outdoor domain, because of the distance between the operator and the robot and the low visibility conditions. In addition, in our scenario the human operator can command a robot even beyond line-of-sight fly. From a different perspective, in *Multi Robot Task Allocation* (MRTA), the robot selection is obtained considering a feasible assignment of tasks to the robot that optimizes some objective [147]. Notice, that our aim instead is to select the robot that best matches the human operator intention, even when this selection is far from an optimal choice. Probabilistic graphical models like *Dynamic Bayesian Network* (*DBN*) [148][149] has been widely deployed in human robot interaction. In this context, several works focus on activity recognition and human action anticipation performed by humans in order to estimate the plan or intention [150][151]. In contrast, in our approach the *DBN* is used to provide an estimation of the human intention regarding the selection of a robot among a group of possible candidates.

2.1.3 Implicit Robot Selection

In this section we describe the implicit robot selection process. Given the set of all the available robots $R = \{r_1, r_2, ..., r_n\}$, $A_R(t) \subseteq R$ represents the set of the active robots in the rescue mission at time t. The robot $i \in R$ is endowed with the set of capabilities $K_i \subseteq K$, where $K = \{k_1, k_2, ..., k_j\}$ is the set of all the available capabilities, and its state is represented by the pair $s_i =< b_i, f_{s_i} >$, where b is the battery level and f_s is the flying status of the robot. Let $C = \{c_1, c_2, ..., c_k\}$ be the set of all possible commands the human operator can invoke. Given a command $c_j \in C$ that requires the set K_j of capabilities to be properly executed, the probability for the robot $i \in A_R(t)$ to be the referred in the command c_j is $P(r_i) = P(r_i \mid c_j)$. As previously stated, in our approach each robot of the team is able to estimate this probability when a command is requested by the operator. In order to perform this estimation process, we endow the robots with the *Intention Estimation System* depicted in Figure 2.2 that is to estimate the probability $P(r)$ fusing different kind of data. Specifically, upon receiving the new command c_j, the estimator calculates three different factors α, β and γ associated with, respectively, *Capability*, *Geometrical* and *Contextual* information. The final probability value is defined as $P(r \mid c_j) = w_1\alpha + w_2\beta + w_3\gamma$, where w_1, w_2 and w_3 are empirically estimated weights.
In the scenario considered in this paper, we mainly focus on navigation and multimedia commands, such as *take-off*, *land*, *go* along a direction, *stop*, *rotate*, *explore*, *take-a-picture* and *start/stop recording* a video. In addition, in order to assist the human operator in the rescue mission, the robot can be equipped with a *camera*, a *camcorder* or an *ARVA*[1]

[1]The *Appareil de Recherche de Victimes en Avalanche* is an instrument commonly used to find victims of avalanches.

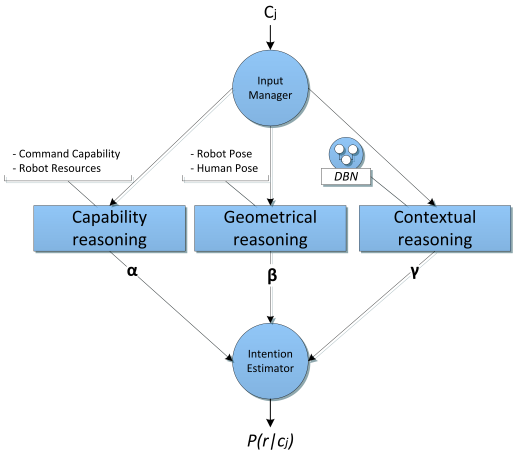

Figure 2.2: The *Intention Estimation System* architecture

sensor. Finally, we define three different operative states of the robot: *idle*, *hovering* and *flying* if the robot is, respectively, hold on the ground, flying but not employed in any mission or navigating toward a destination point.

Capability Reasoning

Each robot should be capable of evaluating its own ability to perform a possible command (e.g. a take-a-picture command should be referred a robot equipped with a camera). For this purpose, the robot estimates an α value taking into account its equipment (capabilities) and the available power (resources). As for the capabilities, each robot r estimates a probability $P(K_r \mid c)$ that a given command c refers to its set of capabilities K_r. In the second case, the robot estimates a probability $P(B_r \mid c)$ that a given command refer to robot r with an estimated power charge B_r after the execution of c, this allows the robot to evaluate the command assuming that the human is aware about the associated energy consumption.

Geometrical Reasoning

This module assesses geometrical relationships between the commands and the poses (position and orientation) of the human and the robot. For instance, the robots in the human field of view could have a higher probability to be selected for specific commands. In order to discover these relationships, we collected a domain specific training set (see Section 2.1.4), where, for each command we consider the orientation of the operator with respect of the selected robot and the absolute distance between the two. Specifically, two different probabilistic values $\beta_1 = P(dist(h,r) \mid c)$ and $\beta_2 = P(fow(h,r) \mid c)$ are evaluated by the robot, where $dist(h,r)$ and $fow(h,r)$ are, respectively, the distance and the

orientation of the human with respect to the robot. The overall β value is calculated as a weighted and normalized sum of β_1 and β_2.

Contextual Reasoning

This layer represents the core of the *Intention Estimation System*. At this level, the probability value is calculated exploiting of a Dynamic Bayesian Network (*DBN*). A Bayesian Networks N is a triplet (V, A, P) where V is a set of random variables, A is a set of arcs and $P = P\{(v \mid \pi_v) : v \in V\}$, $G = (V,A)$ and P represents, respectively, a directed acyclic graph and the set of conditional probabilities of all variables given their parents. Similarly, a *DBN* captures the development of the network over time steps. The proposed *DNB*, illustrated in Figure 2.3, allows the robot to infer on the operator's intention given the contextual information represented in the nodes of the network. In particular, the probability density of the robot r over the command c is calculated considering only the contextual information $cont(r)$ as $P(cont(r) \mid c)$ from the $P(cont(r)_t \mid cont(r)_{t-1}, c)$ provided by the network. The proposed network (see Figure 2.3) in composed of the following nodes:

- *Status node*: the robot operative state, its battery level, and the time elapsed from last received command.
- *Command node*: the last received command;
- *Robot node*: This node represents the probability that the robot is selected by the operator.

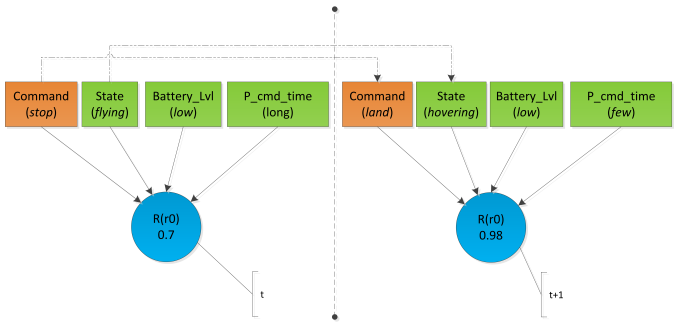

Figure 2.3: Dynamic Bayesian Network. In *orange* the command node, in *green* the status nodes and in *blue* the robot node.

Here, the contextual information is defined by status and command nodes, while the *Robot* node estimates the probability of a robot to be the selected. Notice, that the contextual nodes assume discretized values.

2.1.4 Architecture

The overall system architecture is depicted in Figure 2.4 and defines the interface between a heterogeneous team of robots and the human operator. The *Human Operator* interacts with the *Robot Selection* module via a set of multi-modal commands generated by the *Multimodal HRI* module on the base of the input provided by the human operator via a combination of voice and gestures [154]. Upon received a new command, the *Robot Selection* module is responsible for contacting all the active robots of the team in order to collect their probability estimation of being selected and then select the one with the higher value. Finally, to produce the estimation of the operator's intention, each robot exploits the *Intention Estimation System* described in the previous section. Notice that, in our approach, we assume that each robot evaluates its own probability of being selected, neglecting the status of the other robots. We deliberately decided this simplified setting to better handle the continuous change of robots involved in the rescue mission.

Notice that in our framework, implicit selection is not mandatory, and the human operator can always directly refer to a particular robot in an explicit manner using its code name. Moreover, in order to minimize the errors, once all the estimated probabilities have been collected from the robots, the *Selector* module uses two parameters τ_1 and τ_2 to identify and manage ambiguous selections. In particular, τ_1 provides a threshold on the probabilities for selectable robots, while τ_2 represents a minimum difference between the two best generated results needed to define a selection. When τ_1 and τ_2 are not satisfied, the selection is considered ambiguous and an explicit interaction with the operator can be started to disambiguate the human intention. Using *text-to-speech* technology, the *Multimodal HRI* can ask the human operator to choose among the most probable robots. The values of τ_1 and τ_2 are set by means of the learning process described in the next section.

2.1.5 Training & Testing

In order to train and test the proposed system, different simulated interaction scenarios have been set up (see Figure 2.6). The simulated environment reproduces typical scenarios of our domain, while the operator can navigate the scene in a first person perspective and control a group of maximum 6 drones. In this context, a tester can perform different tasks interacting with the simulator with *command-line* interface to send commands using a joypad to navigate the environment. The drones are differently coloured and the user can refer to them using code names based on colors (*e.g. Red Drone*). In order to get information about the status of the robot and the environment, the user can exploit a basic *user interface* shaped as a non-invasive map displayed in a corner of the monitor. This interface is intended to substitute the human operator's tablet used in real world scenario. An example of this interface is depicted in Figure 2.5, where the battery level, the flying state and the capabilities of the drones are illustrated for the user situation awareness while the drones are represented on the map as coloured spheres. Moreover, the streams

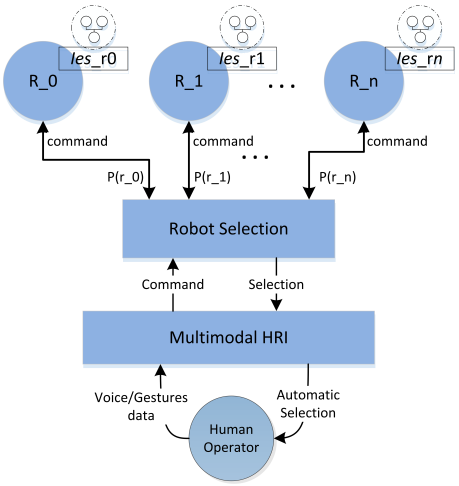

Figure 2.4: System architecture

of the drones' on-board camera can be inspected. In the following, we discuss the system training and evaluation.

Training

In the training session, we generate the training sets needed by the *Intention Estimation System*. For this purpose, three mission scenarios has been defined considering different situations. In the first scenario, represented in Figure 2.6*a*, the user was asked to land the drones on different landing zones represented by the red flat squares in the figure. Both the initial position of the drones and the landing zones are located close to the operator's initial position in order to encourage the user to use commands for line-of-sight navigation. In the second scenario, depicted in Figure 2.6*b*, the drones are all equipped with an on-board camera and the mission of the users is bring the robots in a landing zone located in a farther place with respect to the previous scenario. The aim here is to force the operator to use the on-board camera of the drones in order to avoid obstacles, such as wood or mountains, and follow them during the navigation. Finally in the last scenario, depicted in Figure 2.6*c*, the operator must command the robots over a mountain and use the drone on-board camera to acquire pictures or videos of a predefined area. In this context, the end of a training session is determined by the discharging of the batteries of the drones. In the training phase, we involved 7 different users already expert of the system. Once the *Intention Estimation System* trained, another training session is needed the adapt the τ_1 and τ_2 thresholds. This is obtained by asking the testers to execute another training session in which they validate the framework by accepting or rejecting the results of the selection process.

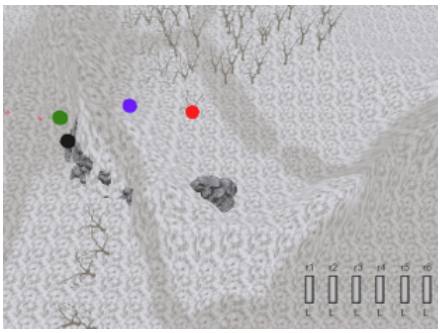

Figure 2.5: User interface for the user situation awareness.

2.1.6 Case Study

The effectiveness of the proposed system has been evaluated involving a group of 10 testers (8 men), who were asked to perform a mission in the simulation scenario depicted in Figure 2.6*d* that combine the scenarios introduced in the training phase. Indeed, here, both navigation by sight and exploration with multimedia data acquisition tasks are considered. Specifically, the goal of the user is to both bring an arbitrary number of robots to a landing zone and to use the capability of the drones to explore a predefined searching area. During the test, the user is asked to confirm the correctness of the selection process results for each command. This way, we can collect the *True Positive* (*tp*), *False Positive* (*fp*) and *True Negative* (*tn*). The performances of the system are then reported in terms of *Precision*, *Recall*, *Accuracy*, *Sensitivity*, and *Specificity*, with the standard definition, i.e. $Precision = \frac{tp}{tp+fp}$, $Recall = \frac{tp}{tp+fn}$ while *Sensitivity* and *Specificity* are the *tp* and *tn* rates, respectively. Moreover, our aim is to assess both the *Intention Estimation System* and the overall system. Therefore, we designed two different test cases, with or without the thresholds check. The results of both tests are reported in Table 2.1 and in Table 2.2, respectively. In the first test case, in the case of a wrong robot selection, tester can also assess the system error as fair alternative of the intended selection, i.e. the selected drone is different form the intended one, but the user considers it as an equivalent choice. The percentage of these mistakes is reported under the *sm* entry in Table 2.1. Instead, in the second case, we consider in the percentage interactions needed to disambiguate a selection (*dialogue* entry in Table 2.2).

Table 2.1: Intention estimation system results

Precision	*Recall*	*Accuracy*	*Sensitivity*	*Specificity*	*sm*
79%	79%	88%	79%	20%	31%

The presented results shows that the intention estimator is able to correctly select the

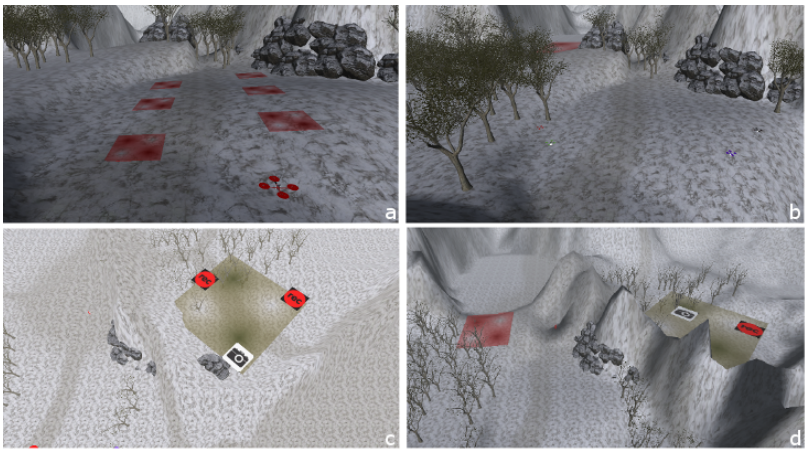

Figure 2.6: Simulated environment for system training and testing.

Table 2.2: System results

Precision	*Recall*	*Accuracy*	*Sensitivity*	*Specificity*	*dialogue*
91%	91%	95%	91%	8%	36%

intended robot with a satisfactory performance. The enhanced performances in Table 2.2, show that, as expected, the dialogue system used in the second test case enhances the precision of the selection process. On the other hand, a high percentage of the selection errors in the first test case are considered not relevant by the user.

2.2 Multimodal HRI with Multiple Drones

In order to enable the robot control a command must be easily generated by the human operator. For this purpose, we present a multimodal interaction framework suitable for a human rescuer that operates in proximity with a set of co-located drones during search missions. Differently from typical human-drone interaction settings, here the operator is not fully dedicated to the drones, but involved in search and rescue tasks, hence only able to provide sparse, incomplete, although high-value, instructions to the robots. This operative scenario requires a human-interaction framework that supports multimodal communication along with an effective and natural mixed-initiative interaction between the human and the robots.

2.2.1 Introduction

A multimodal interaction framework suitable for human-UAVs interaction in search and rescue missions is presented in this paper. This work is framed in the context of the SHERPA project [138] whose goal is to develop a mixed ground and aerial robotic platform supporting search and rescue (SAR) activities in a real-world alpine scenario. One of the peculiar aspects of the SHERPA domain is the presence of a special rescue operator, called the *busy genius*, that should cooperate with a team of aerial vehicles in order to accomplish the rescue tasks. Differently from typical human-UAVs interaction scenarios [21, 10], in the place of a fully dedicated operator we have a rescuer which might be deeply involved in the SAR mission, hence only able to provide fast, incomplete, sparse, although high-value, inputs to the robots. In this context, the human should focus his cognitive effort on relevant and critical activities (e.g. visual inspection, precise maneuvering, etc.), while relying on the robotic autonomous system for specialized tasks (navigation, scan, etc.). Moreover, the human should operate in proximity with the drones in hazardous scenarios (e.g. avalanche), hence the required interaction is substantially dissimilar to the one considered in other works where the human and co-located UAVs cooperate in controlled indoor conditions. In this paper, we present the multimodal and mixed-initiative interaction framework we are currently designing for this challenging and novel domain. The multimodal interaction should allow the operator to communicate with the robots in a natural, incomplete, but robust manner exploiting gestures, vocal, or tablet-based commands. Currently, we are manly focusing on a gesture- and speech-based interaction suitable for accomplishing navigation and search tasks in coordination with a set of drones operating in the scene. In order to communicate with the robots, we assume the human equipped with light and wearable devices, such as a headset (speech) and the *Myo Gesture Control Armband*[2] (gestures). Notice that, vision-based interaction/recognition methods are not appropriate in our context. In this domain, we introduce a set of multimodal commands and communication primitives suitable for the accomplishment of cooperative search tasks. In the proposed framework both *command-based* and *joystick-based* interaction metaphors can be exploited and smoothly combined to affect the robots behavior. The proposed framework permits different kinds of interactions, from precise vocal commands (e.g. the operator can ask the robot to "*rotate 3 o'clock*" or "*go up 3 meters*"), to deictic communication where speech and gestures are combined (e.g. the operator can say "*go there*" while pointing). Moreover, while a robot is executing a task, the human can exploit gestures in *joystick-based* metaphor to adjust the generated trajectory or to directly teleoperate the selected drones. Indeed, the interpreted human interventions are continuously integrated within the robotic control loop by a mixed-initiative system that can adjust the drone behaviors according to the operator intentions. In order to test the framework and the associated interaction modalities, we introduce a test-bed where a human operator can orchestrate the operations of simulated drones to search for lost people in an alpine scenario. This case study is used to discuss

[2] https://www.thalmic.com/en/myo/

the functioning and the effectiveness of the proposed interaction system.

2.2.2 Search and Rescue Missions with Multiple Drones

We assume a human operator involved in SAR tasks in an alpine environment with the support of a set of co-located drones (see Fig. 2.7). The mission goal is to find a set of missing persons in a specified area with loose time constraints. In particular, we refer to a winter scenario where the probability of survival decreases dramatically with the time. Moreover, we focus on the search phase of the rescue mission where the rescuer and the drones are already in the operative area. During the search, the operator can issue verbal and/or gestural commands to the drones which are used to extend the rescuers perception by streaming video or images taken with their on-board cameras. We assume a restricted search area of few square kilometers (1 to 10) with a short mission time (less than 15 minutes). As for the drones, we assume a set of small quadrotors with standard specification (flight time 25 min., max. airspeed 15 m/s, max. climb rate 8 m/s, etc.) equipped with standard sensors including an onboard camera used by the operator to remotely inspect the environment. In order to communicate with the drones, we assume the following (light and low-cost) human equipment: a tablet with a user interface, a headset to vocally communicate, and a *Thalmic Myo Armband* device, endowed with Eight Steel EMG and 9 *DOF* IMU, for gesture-based interaction and teleoperation.

Figure 2.7: An illustration of the SHERPA winter scenario.

2.2.3 Mixed-Initiative Multimodal Interaction

In the domain illustrated above, the operator should interact with the robots in a rapid, concise, and natural manner, exploiting verbal and non-verbal communication. Moreover, since the human is not fully dedicated to the drones, the robotic system should support different control modes sliding form an *autonomous* behavior to direct *teleoperation*, passing through the *mixed-initiative* mode, where the user can execute some operations while relying on the autonomous control system for the remaining ones.

2.2.4 Multimodal Mixed-Initiative Interaction Architecture

The operator should be capable of interacting with the system using different modalities (gestures, speech, tablet, etc.) at different levels of abstraction (task, activity, path, trajectory, motion, etc.). These continuous human interventions are to be suitably and reactively interpreted and integrated in the robotics control loops providing a natural and intuitive interaction. The HRI architecture designed to accomplish these requirements is illustrated in Fig. 3.19 whose components are detailed below.

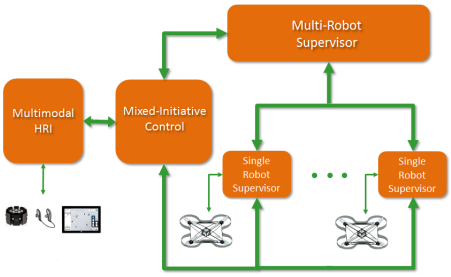

Figure 2.8: The overall HRI architecture

The Multimodal Interaction Module interprets the operator commands/intentions integrating inputs from multiple communication channels. For instance, either speech- or gesture-based commands may be used to stop a drone, while vocal commands in combination with deictic gestures can be used to specify navigational commands (e.g. "*go-there*") to the co-located drones. Notice that commands can be given at different levels of abstraction, from task assignments to direct teleoperation, combining different modalities. On the other hand, beyond line-of-sight control, the human can receive feedback from the drones via a tablet interface, the headphones, and the armband.

Once interpreted, the multimodal human commands are managed by the Mixed-Initiative Control module that interacts with the Single Robot Supervisor and Multirobot Supervisors mediating between the human and the robotic initiative. Indeed, commands can be provided to both single or multirobots, while vague instructions are to be completed and instantiated by the robots supervisory systems according to the operational context. Following a mixed-initiative planning approach [27], the human interventions are integrated into continuous planning and execution processes that reconfigure the robotic activities according to the human intentions and the operative state.

2.2.5 Multimodal Interaction

We employ a multimodal interaction framework that exploits a late fusion approach [89] where single modalities are separately classified and then combined (see Fig. 2.9), this approach permits to introduce new modalities (e.g. tablet-based interaction) in an extensible

Table 2.3: List of primitive commands with modalities

Command	Modalities
Take-Off	speech
Continue	speech/gesture
Land	speech
Rotate #o'clock	speech/gesture and speech
Selection #Drone-Id	speech/gesture and speech
Faster	speech/gesture
Slower	speech/gesture
Rotate Clockwise	speech/gesture
Rotate Anti-clockwise	speech/gesture
Brake	speech/gesture
Go #Direction	speech/gesture/gesture and speech
Search Expanding	speech/gesture
Search Parallel Track	speech/gesture
Search Creeping Line	speech/gesture
Switch	gesture

and modular manner.

In this work, we mainly focus on commands suitable for interacting with the set of co-located drones during navigation and search tasks. In particular, we are concerned with speech- and gestures-based communication with the drones suitable for the following purposes:

- *Selection:* in order to select single or groups of robots involved in the action, both speech (e.g. "*all hawks take off*", "*red hawks land*") and gestures (e.g. "*you go down*") can be used in combination. Specific names (e.g. "*red hawk*", "*blue hawk*", etc.) can identify drones, while deictic gestures (pointing) can be used to select not only drones, but also targets (e.g. "*you go there*").

- *Motion:* a set of commands are used for navigation. Motion directives can be coupled with voice directives. For example, a rotation command (gesture) can be associated with the final orientation (voice) of the drone, while during a movement command (gesture) the operator can specify the exact distance to be covered (voice). When these values are not explicitly provided, default ones are assumed.

- *Search:* a set of commands are used to select the search pattern used to scan an area with a helicopter search [76], i.e. search-expanding, search-parallel-track and search-creeping-line (see Fig. 2.10). Those patterns can be invoked either vocally or by means of specific gestures.

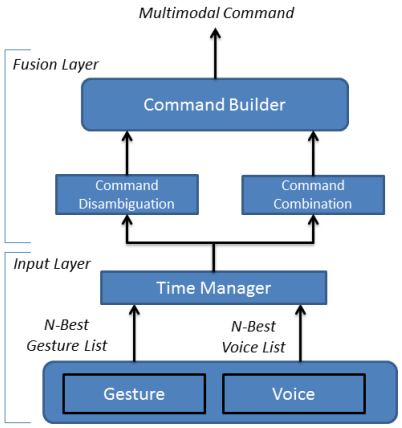

Figure 2.9: Multimodal Interaction System

- *Switch:* meta level commands allow the operator to change the interaction mode, e.g. from command-based to joystick-like interactive control and viceversa.

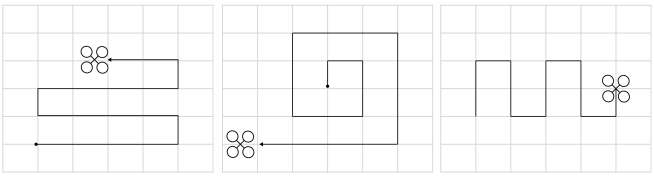

Figure 2.10: Parallel track (Left), Expanding (Center) and Creeping line (Right) search patterns

Table 2.3 summarizes the overall set of primitive commands that we have introduced in our domain. These can be invoked and flexibly combined in a multimodal manner using speech, gestures, or speech and gestures together. In the following, we provide details about the adopted methods for speech/gesture recognition and fusion.

Speech recognition we rely on Julius[3], a two-pass large vocabulary continuous speech recognition (LVCSR) engine. A suitable grammar has been defined to parse the commands of the users. A N-best list of possible interpretations is continuously provided in output.

[3] http://julius.osdn.jp/en*index.php*

Gestures recognition The proposed gesture recognition system exploits the *Thalmic Myo Armband*. This device permits to detect and distinguish several poses of the hand (see Fig. 2.11) from the electrical activity of the muscles of the arm where the band is weared. In addition, the band is endowed with a 9 *DOF* IMU that permits motion capture. Therefore, both hand poses and movements can be detected and used for robot control. In our framework, the position of the hand is used to enable/disable control modalities (*switch* commands), while the movements of the hand are to be interpreted as gestures. Specifically, in this work we introduce the following intuitive switching strategy: when the hand is closed (Fig. 2.11, right) the command-based gesture interpretation is enabled, when the hand is open (Fig.. 2.11, left) the joystick-like control is active. As far as the

Figure 2.11: Different poses of the hand recognized by Thalmic Myo Armband.

hand poses are concerned, we directly rely on the built-in *Myo Armband* classifier. Instead, a robust gesture recognition system based on the armband acceleration measures requires an independent classification method. The main advantage of our method is that few examples are needed for training, while ensuring a robust user-dependent application. Since in our scenario we assume the presence of a trained operator (the busy genius) with a tailored recognition system, this approach is satisfactory. On the other hand, continuous recognition is not supported: the classifier needs the start and the end of the executed gestures. However, as already mentioned, we can exploit the hand pose detected by the *Myo Armband* to enable and disable the classifier, indeed we assume that the samples of a gesture are stored and classified only when the hand is closed (*switch* command). Once the samples of an executed gesture are collected, the gesture classification process works as follows. Initially, two transformations are deployed to filter the acceleration signal generated by the input device. The first one removes noise from the acceleration samples. The second filter allows a uniform sampling independently of the execution speed of the gesture. For this purpose we linearly interpolate m points in order to transform the input signal into a m-pla of samples $\langle \vec{a}_1, \ldots, \vec{a}_m \rangle$ of fixed size with equal distance between them. After this phase, a third transformation makes the gesture invariant from the actual position of the arm of the operator and compensates the gravity force. Specifically, each sample is modified as: $\vec{a}_i' = R_S^W \vec{a}_i - \vec{g}$, with R_S^W the rotation matrix from the sensor reference frame S to the world frame W and $\vec{g}$ the gravity acceleration vector. This way, each gesture G can be associated with a uniform m-pla of samples. Given a training set T_s that collects a set of m-ple for each gesture type, the gesture classification can be directly obtained form an Euclidean distance in the 3D space. Specifically, given the performed gesture G, for each gesture $G_i \in T_s$, we can define a score $s_i = ||G - G_i||$. The best scores

for each gesture type defines the N-best list of matches for the executed gesture. Since we need a reliable classifier, we improved the robustness of the classifier in two ways. First of all, for each executed gesture we generate not only one m-pla, but a set of m-ple representing possible slides of the gesture samples within a time window (see Fig. 2.12). The best match is then used to rank the gesture with respect to the training set. In the second place, a secondary evaluation that takes into account the orientation of the executed gesture is performed when the difference between the last 2-best values is below a suitable threshold. If this is the case, the gestures are ranked again by improving the values of the gestures whose average orientation is closer to the average orientation of the performed gesture.

As for the dataset, we defined 14 different types of gestures (with 10 trials for each type) for both navigation commands and search strategy requests. We evaluated the gesture recognizer by performing 30 trials of each gesture (randomly generated to avoid the learning effect). In Table 2.5, we report the *Precision*, *Recall*, and *Accuracy* for the gestures of the training set.

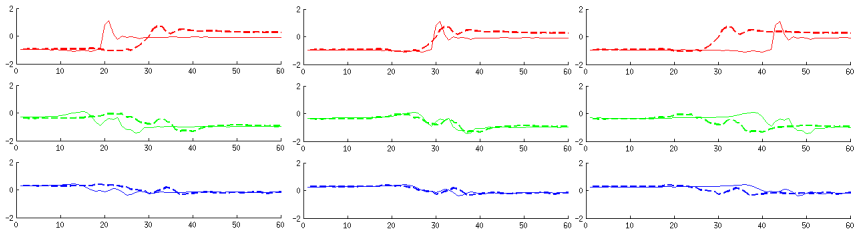

Figure 2.12: Gesture matching at different time (start, middle, end) of the sliding window. Dashed and solid curves represent the recognized and executed gesture respectively.

Multimodal Fusion The *fusion module* combines the results of vocal and/or gesture recognition providing the command interpretation. We deploy a late fusion approach that exploits the confidence values generated by the separated (speech and gestures) classifiers. In order to integrate the classifiers outcomes, the two channels are to be first synchronized. We assume that the first channel that becomes active (speech or gesture) starts a time interval (about 1 sec in our setting) during which any other activity can be considered as synchronized. This way, a vocal command provided during the execution of a gesture, or immediately, after can be fused (or a gesture after a vocal command). When this is the case, contextual rules are used to *disambiguate* the conflicting commands or to *combine* vocal and gesture inputs using the information contained in both the channels. In the first case, or when an explicit rule is not available to disambiguate, the N-best values provided by the speech and gesture recognizers are compared and the interpretation with the maximum value is selected. In the second case, if the classification results are compatible, these are combined according to simple rules (e.g. navigation gesture towards a certain

Table 2.4: Gesture classification results

Gesture	Precision	Recall	Accuracy
Brake	93.3%	96.5%	94.9%
Go Ahead	96.6%	90.6%	93.4%
Go Backward	100%	100%	100%
Go Down	100%	88.3%	93.5%
Go Left	73.3%	100%	86.2%
Go Right	96.6%	85.3%	90.4%
Go Up	100%	96.7%	98.3%
Rotate Anti-Clockwise	100%	96.7%	98.3%
Rotate Clockwise	83.3%	100%	91.38%
Search Creeping Line	80%	82.7%	82.4%
Search Expanding	96.6%	85.3%	90.4%
Search Parallel Track	96.6%	90.6%	93.4%
Faster	96.6%	82.9%	88.9%
Slower	66.6%	90.9%	79.9%
Average	91.4%	91.9 %	91.7 %

Table 2.5: Multimodal classification results

	Precision	Recall	Accuracy
Average:	96.95%	96.4%	96.3%

direction combined with a vocal indication of a distance is interpreted as *Go to #distance*). In Table 2.5, the average classification results - collected by rerunning the classifier evaluation considering the fusion of speech and gestures - show the improvement in robustness due to multimodal disambiguation. Overall, the reliability and the latency of the multimodal interaction system seems compatible with a natural and effective interaction.

2.2.6 Mixed-Initiative Interaction

The operator is allowed to interact with the drones at any time at different levels of abstraction with different interaction metaphores. In this work, we focus on navigation and search activities, hence our main concern is on path/trajectory level interaction. In this context, we introduced two main interaction metaphores: *command-based* and *joystick-based*. In the first case the robots are considered as agents to be coordinated by the operator multimodal commands, in the second case the operator can directly teleoperate a drone using his/her open hand as a virtual joystick. Here, the hand position is used to switch between these two control modes: *hand-closed* for gestural commands, *hand-open* for trajectory adjustments and teleoperation. For instance, the human may ask a drone the execution of a task (e.g. “blue hawks go there”), and once the execution starts, use the *hand-open* mode to manually correct the trajectory. A complete teleoperated control can

be obtained once the current command execution has been stopped by a *brake* command.

Path and Trajectory level interaction The navigation commands introduced so far are associated with drone movements to be suitably planned and executed. We deploy a RRT^* algorithm [37] for path planning, while trajectory planning is based on a 4-th order spline concatenation preserving continuous acceleration. Following the approach in [14], during the execution, the human is allowed to on-line adjust the robot planned trajectory without provoking replanning (*open-hand* mode). Specifically, the planned robot position $a(t)$ can be deviated into a mixed trajectory $m(t) = a(t) + h(t)$, by the $h(t)$ human contribution defined as follows

$$h(\mathrm{t}) = \begin{cases} h(t-1) + human(t) \text{ if } mixed = ON \\ h(t-1) + \Lambda(t) \; otherwise \end{cases}$$

That is, when the mixed-initiative is active the control reference $human(t)$ generated by the operator -via gestures or voice- increases the current displacement $h(t)$; otherwise, when the human intervention is released, the deviated trajectory is smoothly driven back towards the planned one by a linear function $\Lambda(t)$. On the other hand, similarly to [14], if the human deviation goes outside a context-specific workspace, a replanning process starts and the autonomous system generates another path and another trajectory to reach the next waypoint. A vibro-tactile feedback on the armband provides the operator with the perception of the robot deviation with respect to the planned trajectory.

2.2.7 Case Study

The effectiveness of the multimodal interaction framework proposed in this work has been tested in a simulated environment of an alpine scenario (see Fig. 2.14, left). We used *Unity 3D* to simulate a set of drones equipped with an onboard camera. A tablet user interface (see Fig. 2.14, right) allows the operator to monitor the robots position on a map, while receiving video streams for the cameras of the drones on multiple windows; the one associated with the selected drone has a bigger size. In this scenario, a set of victims is randomly positioned within the environment. The mission goal is to find a maximum number of missed persons within a time deadline. When a missed person appears in the field of view of a drone camera, the operator can use the tablet interface to mark his/her position.

Experimental set-up As an pilot study, we designed an experimental set-up with the aim of reproducing a similar setting in the real world. Preliminary experience with our real drones suggested to start form an initial configuration with 2 or 3 UAVs per operator. We focus on a winter scenario, where the rescuers operate under time pressure in a restricted area, therefore we defined a 120×120 m with the goal of searching for 6 missing persons in a maximum time of 6 minutes. The target user is a trained operator, hence we involved

Figure 2.13: Simulated alpine scenario testbed.

a group of 5 expert users (4 males, 1 female). Each subject was asked to perform 3 runs of a mission. For each trial we collected the following data:

- *detected persons*: number of discovered victims;
- *time to detect*: mean time needed to find a victim;
- *selection time*: time spent while monitoring and controlling a drone;
- *operative mode*: time spent per drone for each operative mode (autonomous, mixed-initiative, teleoperation);
- *interaction type*: modality used to invoke commands (voice, gesture, voice and gestures, etc.).

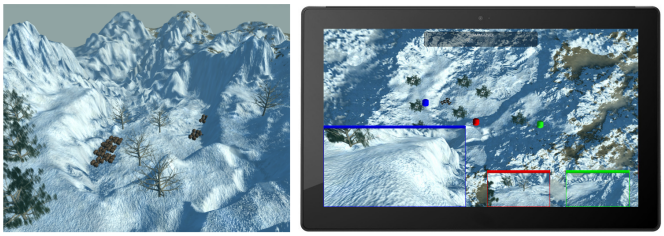

Figure 2.14: Simulated environment (left), tablet interface (right).

Results In Table 2.6, we report the victims found (mean values) using 2 and 3 drones along with the mean, maximum, and minimum time (sec) needed to find a victim. We further detail the user performance in Table 2.7 where we illustrate the success rate in finding *n* (of 6) victims along the mean time (sec) needed. We can observe that the overhead of monitoring and controlling 3 drones seems here compensated by better performances in

search, indeed the 3 drones search outperforms the 2 drones one (5.6 vs. 3.9 victims found, with two-tailed $p < .0001$), while the mean time needed to find the *n*-th victim (e.g. $96,6$ vs. $249,6$ sec. to find the first 3 victims, two-tailed $p < .0001$) is also considerably reduced. Table 2.8 shows how the operators balance the time (sec) dedicated to the 3 drones during the tests. Here, we consider: maximum (*max*), minimum (*min*), middle (*mid*) time dedicated to a drone, and the time spent monitoring all the drones (*all*). The averages of these values and the associated selection percentage, shows a satisfactory balance. Interestingly, the time dedicated to *all* the drones is not negligible, indeed all drones are selected during parallel operations (e.g. "all hawks take off") or to inspect all the cameras at the same time during a high-level scan.

Table 2.6: Mission results: victims found and time to find a victim

	Victims				**Time**			
	avg	min	max	std	avg	min	max	std
2 Drn	3.9	3	5	0.65	217.4	60	350	18.3
3 Drn	5.6	4	6	0.61	157.6	45	350	17.1

Table 2.7: Mission details: time to find *n* victims

2 Drn		1	2	3	4	5	6
Succ. %		100	100	100	80	20	0
Time	avg	85.8	122.7	249.6	296	333.3	–
	std	15.7	22.8	24	13.8	15.2	–
3 Drn		1	2	3	4	5	6
Succ. %		100	100	100	100	93.3	46,6
Time	avg	59	64.3	96.6	166.9	235.5	324.2
	std	6.98	6.13	12.3	25.2	32.5	19.8

We can also analyze whether the multimodal interaction is actually exploited. For this purpose, Figure 2.15 (left) illustrates the distribution of the commands for each modality. Interestingly, we can observe that the gesture-based commands are frequently used to orchestrate the operations of the drones during the search, both as single gestures, or in combination with vocal instructions (used either to complete or to reinforce a command), while purely vocal interactions are less frequent. Finally, in Figure 2.15 (right) we compare the precentage of time spent by the drones in the operative modes (*autonomous*, *teleoperated*, *mixed-initiative*). Here, we can observe that during the tests (3 drone cases) each drone mainly operates in the autonomous mode, with a minor percentage of time spent in interactive adjustments (joystick-based interaction), while the direct teleoperation is a rare control modality. The dominance of the autonomous mode allows the user to sporadically interact with the drones as required in the SHERPA scenario. Overall, this preliminary evaluation suggests that the proposed multimodal interaction framework

Table 2.8: Drone Selection: time dedicated to the drones

		Max	Mid	Min	All
Time	avg	108	92.16	80.64	75.6
	std	17.9	16.3	17.6	31.6
Select.	%	30	25.6	22.4	22

is effective in the 3 drone configuration, indeed, the user monitoring and control effort during search mission seems well balanced among the drones, while both the multimodal interaction and the autonomous control mode seem exploited as expected.

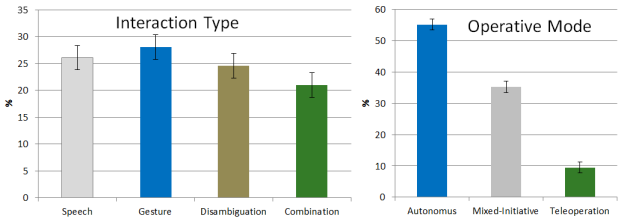

Figure 2.15: Interaction type (left) and operative modes (right)

Chapter 3

Autonomous and semi-autonomous action execution

An important skill of robots that operates in service robotics tasks is the ability to perform assigned tasks with a different level autonomy. This means that the robot is itself responsible of the execution of a commanded action, decomposing the task in different sub-tasks in which, the accomplishment of all sub-tasks lead the robot to accomplish the entire mission. We will refer to this phase as *Planning & Execution*. In case of cluttered and unknown environment, as the one considered in typical service robotics applications, the presence of a human operator in the loop of the action execution process could be really important. For this purpose, in this thesis different approached to the Planning & Execution phase that rely on *Mixed-Initiative* interaction are presented. In *Mixed-Initiative* interaction applications either the computer or the human can take initiative and decide what to do next, in order to achieve better results in the overall mission accomplishment. In the followings three different approaches for the autonomous and semi-autonomous action execution by means of flying robots are presented.

3.1 Aerial service vehicles for industrial inspection: task decomposition and plan execution

In this paragraph a high level architecture designed for an Aerial Service Vehicle (ASV) operating in close interaction with the external environment is presented. This work is framed within the The AIRobots project [121] whose aim is to develop a new generation of unmanned service helicopters, equipped with sensors and end-effectors, and capable not only to fly, but also to achieve robotic tasks in proximity and in contact with the surface (e.g. site inspections, simple manipulations, etc.).

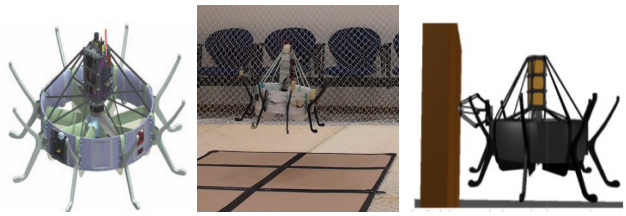

Figure 3.1: Robotic Platform: ducted-fan ASV

In our scenario, the autonomous system should orchestrate a new set of operations like wall approach, docking, undocking, wall scanning etc.. These operations represent different operative modes, each associated with a different controller with specific control laws and performance the high-level control system should be aware of. Each switch from one operative mode to the other should be suitably prepared and planned to keep smooth control trajectories. Since the system flies close to the obstacles in cluttered and unknown environments, fast planning engines are required to generate (or to adjust) trajectories in real-time. On the other hand, the system should be able to regulate the trade off between fast planning and accurateness of the generated trajectories depending on the operative mode and the context. Moreover, since the system operates with the man in the loop, the planning/executive system should be able to manage sliding autonomy, from autonomous to teleoperated mode, depending on the humans' interventions. This applicative domain is challenging and novel and has not been investigated in depth in the UAV literature which is mainly focused on free flight tasks and simultaneous localization, mapping, and path planning problems [124, 128, 137]. Few high-level architectures for UAV can be found in literature [125], but none of these addresses the complexity of the operative domain proposed in this section.

3.1.1 System Requirements and Architecture

The applicative scenario described so far requires a high-level control system with following features:

- The air vehicle operates in close interaction with the environment, hence reactive, adaptive, and flexible planning/replanning capabilities are needed;
- Both autonomous and human-in-the-loop control modalities should be supported to allow human interventions and teleoperation;
- High-level control strategies should be defined taking into account the low-level operative modes and constraints.

In particular, the high-level system should orchestrate the activations of a set of low-level controllers, modeled as hybrid automata [134], switching to the appropriate controller

according to the operative mode and the task (see Fig. 3.2) feeding the selected controller with suitable data (e.g. state and references).

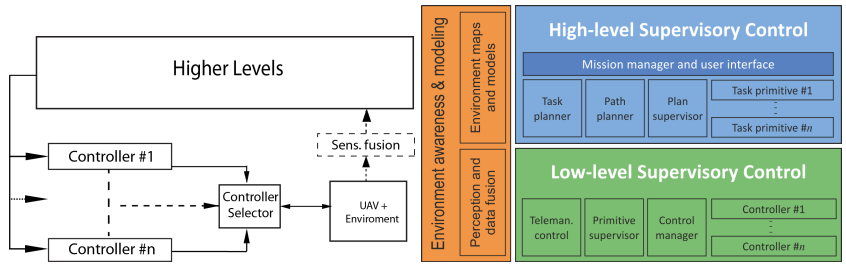

Figure 3.2: Interaction between the high level system and the low-level controllers (left); the high level control system is composed of high-level and low-level supervisory systems.

To match these requirements we proposed the layered architecture depicted in Fig. 3.2. Here, two layers are distinguished: the high-level supervisory system is responsible for user interaction, task planning, path planning, execution monitoring, while the low-level supervisory system manages the low-level execution of control primitives setting the controllers and providing control references. This architecture is detailed in Fig. 3.3.

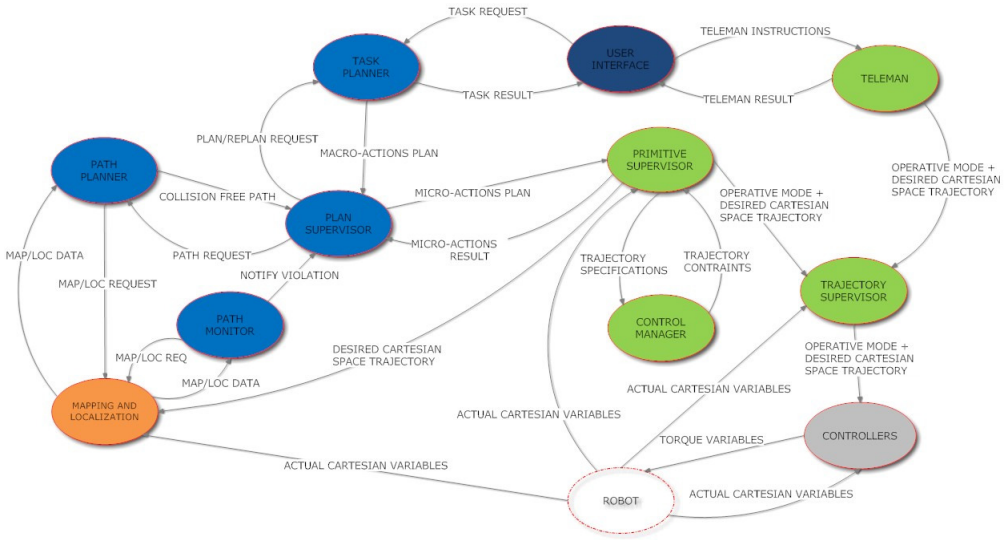

Figure 3.3: High Level Architecture: high level, low level, and reactive level modules are respectively in blue, green, and gray

The *User* module (US) allows us to specify high-level goals (e.g. $Inspect(p)$) or lower level tasks (e.g. $TakeOff$) or to directly teleoperate. Each task/goal is delivered to the TP which expands a task into an abstract plan composed of *macro-actions*. This plan

is then sent to the *Plan Supervisor* (PS) for high-level execution. Each task or macro-action can be interrupted and pre-empted by new tasks provided by the user, provoking task replanning. The PS generates, for each macro-action in the high-level plan, a set of *micro-actions* to be executed by the *Primitive Supervisor* (PR). Each *macro-action* is further decomposed into a sequence of *micro-actions* which are endowed with detailed information about the associated geometrical paths. The PR exploits the *Control Manager* (CM) to select the low-level controller responsible for the micro-action execution. Finally, the PR generates the control trajectory passing it to the *Trajectory Supervisor* (TS) to generate control references at a suitable frequency. The PR exploits concatenations of fifth-order polynomials to provide smooth trajectories between waypoints [131] while ensuring the velocity and tolerance constraints. When a micro-action fails, the PS can either call the PP to generate an alternative path or call the TP to generate a different plan of macro-actions. Furthermore, it can be interrupted by the *Path Monitor* (PM) which checks for trajectory deviations and unexpected obstacles. Finally, the operator can always switch to a manual control mode, in this case the TS should monitor the trajectory provided by the *Teleman*. Once the autonomous control is restored, a replanning process is needed to recover the execution of the current task.

3.1.2 Task Planning and Executive Control

The high-level executive system coordinates task decomposition and plan monitoring. It relies on a PRS engine that manages a BDI-like execution cycle [135] and hierarchical task decomposition. The high-level executive system responds to events generated by the US, PS, or TP itself by committing to handle one pending goal, selecting a method from a plan library, managing the hierarchical decomposition to extract/update the macro-actions plan. Once a plan is generated, the PS should manage the actual execution of each *macro-action* providing the action results to the TP module. During this execution process, user interventions are treated in a uniform way: at any time the user can interrupt/suspend the current task, or the execution of alternative tasks can be invoked. In this case, the executive system reacts by replanning from the current state: it selects alternative methods and generates an alternative plan. This enables mixed initiative task planning [122].

3.1.3 Path Planning and Replanning

The *Path Planner* expands each *macro-action* into a set of *micro-actions* representing a path that respects geometric and operative constraints. The path generation algorithm is based on a Rapidly-exploring Random Tree (RRT) algorithm [130] which is particularly suitable in highly unstructured and dynamic domains. In this work, the RRT algorithm generates collision-free paths composed of sequences of waypoints (x, y, z, θ), where (x, y, z) is a point and θ is the yaw. More specifically, it generates a path as a sequence of (x, y, z) points in a 3D search space (3D grid map), while the yaw θ is obtained as the direction pointing towards the next waypoint. The generated path should satisfy a set of

additional control, safety, and temporal constraints: *Maximum angle* for pitch and yaw; *Minimal distance* from the obstacles (this parameter is also associated with the operative mode and the accuracy of the selected controller); *Maximum Time* for the path generation processes, if the algorithm cannot find a feasible path before the timeout, it should provide the best partial path. Moreover, that RRT path planner can generate several solutions to refine the path, until one of the following conditions are satisfied: *timeout*, i.e. the available time for path planning expires; *interrupt*, i.e. a replanning request or an exogenous event interrupts plan generation; *cost threshold*, i.e. as soon as the current path cost is below a suitable threshold, the generated plan is considered as satisfactory. The path planning refinement process is illustrated in the Algorithm 1 where the path generation process is iterated until the current generated path is not satisfactory. If the $timeout$ occurs before the generation of the first solution, the $solveRTT$ function generates the $path$ that arrives closer to the target.

initialize($path$,$time$);
while $((time < timeout) \wedge (preempted = false) \wedge (pathCost \geq threshold))$ **do**
  $newPath \leftarrow$ solveRRT(q_{init},q_{goal},$timeout$);
  if $C(newPath) < path$ **then**
    $path \leftarrow newPath$;
    $pathCost \leftarrow C(newPath)$;
  end if
end while
return $path$

The path cost is defined as follows:

$$\begin{aligned}\mathbf{c}(path) =& \mathbf{c}_{lng}(path) \cdot p_{lng} + \mathbf{c}_{ang}(path) \cdot p_{ang} + \\ & \mathbf{c}_{way}(path) \cdot p_{way} + \mathbf{c}_{obs}(path) \cdot p_{obs} + \mathbf{c}_{unk}(path) \cdot p_{unk}\end{aligned} \tag{3.1}$$

where the p_i are suitable weights and $\mathbf{c}_i$ are defined as follows. $\mathbf{c}_{lng}(path)$ is a cost associated with the path length; $\mathbf{c}_{ang}(path)$ represents the cost associated with angular (yaw and pitch) variations, by minimizing this cost a straight path should be preferred to a path with angular turns; $\mathbf{c}_{way}(path)$ counts the generated waypoints and allows us to minimize the segments in the path; $\mathbf{c}_{obs}(path)$ is associated with obstacle proximity and penalizes paths close to obstacles; $\mathbf{c}_{unk}(path)$ penalizes paths through -or close to- unexplored cells. Once a path is generated, the path planner defines a set of constraints $cst = (ms, md, et)$ associated with each generated segment. Roughly, for each segment, we set the maximum speed *ms* directly proportional to the obstacle minimal distance *mo* along the corresponding segment; *ms* is also associated with a proportional error *et*, therefore we set *md* as *mo-et* (if this value is not positive, the speed limit is lowered). These constraints *cst* are also accessible to the human operator which can manually reset them. Note, that *cst* are just rough limits used by the CM and the PR to select the right controller and to generate the trajectory associated with the path.

Path replanning is managed with different strategies depending on the time available for path generation. The urgency associated with the replanning activity depends on the

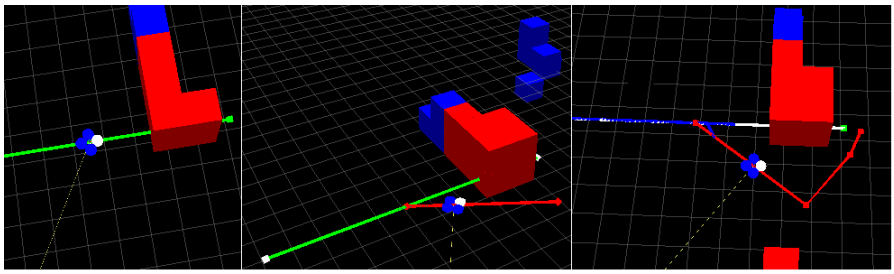

Figure 3.4: (left) *Brake* to avoid collision; (center) *Escape* path to avoid the obstacle; (right) *Replan* a new path generated to reach the target.

position of the collision point p_{obs} and the estimated time to collision t_{ttc}. This one is estimated by considering the obstacle distance d_{obs} along the trajectory and the mean velocity v_{mean} along the path. Given the time to collision t_{ttc}, we introduce two thresholds $T_b < T_e$ used to distinguish the following three cases:

- *Brake.* If $t_{ttc} \leq T_b$ then the obstacle is too close for replanning, hence the PS directly sends a *Brake* command to the PR to stop the robot in *hovering* (Fig. 3.4 up-left).
- *Escape.* If $T_b < t_{ttc} \leq T_e$, the PP is invoked by the PS to find an escape path that allows the robot to avoid the obstacle; the escape trajectory represents a fictituos detour that provides the planner with additional time to generate the new path on-the-fly (Fig. 3.4 up-right).
- *Replan.* If $t_{ttc} > T_e$ then the time is sufficient for safe replanning, hence the PS calls the PP to replan, on-the-fly, a trajectory from a suitable deviation point along the previous path (Fig. 3.4 down).

The PP is called in the case of *Escape* and *Replan*. In the case of *Escape*, the path planning task is simple: it is to select a close and safe target point q_{target} in the free space, far enought to enable safe on-the-fly replanning, and to generate a path to reach it (Fig. 3.4 right). That is, *Escape* provides a path that not only permits to avoid the obstacle, but also provides the time for replanning a new path to the goal. The interesting case is the third one, where the path planning process should find an alternative path that connects the old trajectory with a new one while the robot is flying. The replanning algorithm is illustrated in Algorithm 2. Given the target q_{goal}, the old path $path_{old}$, the collision point q_{obs}, and the t_{ttc} time, the replanning process first estimates the time needed to replan t_{rp} (*estimatedRepTime*); then it selects a waypoint wp_{rp}, along the old path $path_{old}$, from which it is possible to safely calculate the deviation $path_{new}$ from $path_{old}$ (*selectDeviationWP*); finally, upon setting a suitable threshold (*setThreshold*), the replanning process calls RRT_-refine to generate the new path $path_{new}$ from the deviation waypoint wp_{rp} to the target q_{goal}. $path_{new}$ should allow the PR to generate a new trajectory connecting the old one with a smooth deviation from wp_{rp}.

$q_c \leftarrow$ getPosition();
$t_{rp} \leftarrow$ estimatedRepTime(q_c, q_{goal}, $path_{old}$, q_{obs});
$wp_{rp} \leftarrow$ selectDeviationWP(q_c, q_{obs}, $path_{old}$, t_{rp});
$threshold \leftarrow$ setThreshold(wp_{rp}, q_{goal}, t_{rp}, t_{ttc});
$path_{new} \leftarrow$ Refine_RRT(wp_{rp} , q_{goal}, $threshold$, t_{rp});
return $path_{new}$

To select the deviation waypoint wp_{rp} we defined the following strategy. Given the estimated time needed to replan t_{rp}, we estimate the robot position q_{pr} at time t_{rp} (assuming that it keeps following the old path $path_{old}$ during replanning), if there exists a waypoint wp in $path_{old}$ that follows p_{rp} and precedes q_{obs} (keeping a suitable range the we assume as *maxRange*), then we select wp as the deviation waypoint wp_{rp}, otherwise, q_{rp} is on the path segment that intersects the obstacle, hence we select wp_{rp} as the point q_m in the middle of the segment that connects q_{rp} and a point q'_{obs} which is at *maxRange* distance from q_{obst}. In Fig. 3.4 (center), we find an example of replanning from a waypoint after the collision detection (left).

3.1.4 3D Mapping

The environment for mapping and path planning is a 3D grid-map of cells which is runtime generated given the robot pose and the 3D point clouds extracted from the cameras. We deploy the well known pin-hole camera model [127]. Pose estimation of the UAV is needed to identify the 3D position of the projected camera points in the world reference frame. Our pose is either obtained by using libviso2 [126] coupled with a Kalman filter or, alternalively, by directly deploying an optitrack motion capture system. Given the pose, the associated point cloud map should be suitably processed into a 3D occupancy grid. This is obtained by discretizing the vehicle's workspace with elementary cubes of equal size. In our case, we employed a vehicle of $50 \times 50 \times 20$ cm hence, we used cubes of 10 cm. For each cube we stored: the number of inliers (3D triangulated points) fell into the cube volume, the last camera position which an inlier had been collected, and the state of the cube. The number of inliers represents the number of different points from which the same obstacle has been detected. The last camera position is required in case of hovering, to avoid that the same image feature generates dome wrong inliers, while it is possible that the same outlier is achieved from different points of view. Each cell can be associated with one of the following values: $free$, *occupied*, *obstacle*, *target*, *ignored* or *unknown*. Initially each cube is set to $free$. When a 3D point is detected to belong to a given cube, the value of the corresponding cube is set to *occupied*. When the number of points inside a cube reaches a given treshold, the state is set to *obstacle*. On the other hand, when a target is identified, the corresponding cube is set to *target*. Moreover, from each position that had generated a valid target view point, all the cubes laying along the optical rays are set to *ignored*. For wide environments, a sparse representation of the occupancy grid map is associated with a spatial/temporal vanishing criterion. This determines whether an occupancy cube is sill reliable or if it has to be discarded (depending on the distance

travelled by the vehicle or on the time last after its previous update). In fact, due to the drift of the vehicle pose estimation, obstacles which have been observed a long time before or far from the current position cannot be considered reliable anymore in the current map representation, therefore they should be refreshed. With these solutions the reliability and scalability of the map representation can be suitably tuned.

3.1.5 Experimental Results

In this section, we present experimental results on planning, replanning, and obstacle avoidance, both in real-world scenarios and in simulated environments.

Real-world planning and execution. Our architecture has been tested in a real scenario of dimension $400 \times 400 \times 300\ cm^3$ considering the two environments depicted in Fig. 3.5 (up and down). In the two testing scenarios, the task was the following: inspect a target point in pose $(380, 350, 50, 90)$ from the pose $(40, 40, 50, 0)$ with maximum and minimum speed set at $0.3\,m/s$ and $0.1\,m/s$ respectively. The obstacles are detected on the fly and this can provoke task/path replanning, escape, or brake. For each scenario, we executed each test 10 times collecting mean, max, min, and standard deviation (STD) of: time spent during planning (Tp), time spent in replanning (Tr), number of replanning episodes (Nr), length of the executed path (Lp), and total time for execution (including replanning time) (Te). For computation and simulation we used an Intel Core Duo, 1.40GHz, 3GB ram, Ubuntu 10.04. The high-level architecture was developed in ROS. As for 3D mapping, we used cameras ueye with hardware synchronized images, compressed on-board using atom 1.6 GHz, and sent to a ground station. The stereo images are streamed at around 15 Hz at the ground station. The vision algorithm can track around 120 image features correspondences on 4 images working at the streaming frequency. Each camera provides images with resolution of 752×480 and an angle of view of around $50°$.

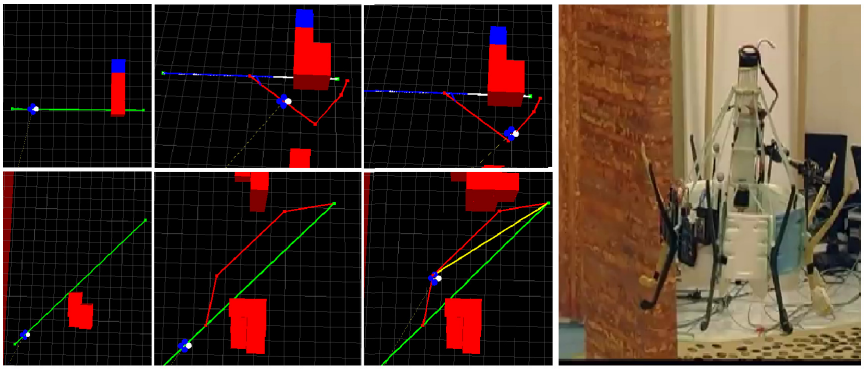

Figure 3.5: Replanning: generated and executed path (left) real platform during plan execution (right).

	Test 1				Test 2			
	Mean	**STD**	**Max**	**Min**	**Mean**	**STD**	**Max**	**Min**
Tp	0.075	0.014	0.08	0.04	0.017	0.002	0.03	0.01
Tr	0.614	0.41	1.20	0.01	0.067	0.04	0.11	0.005
Te	60.5	10.12	75	42	49.9	8.18	60	40
Lp	14.4	1.54	18	12	13.18	1.11	15	11

Table 3.1: Planning and execution results (in seconds) in the real scenario.

Res/Env	**LL**		**LH**		**HL**		**HH**	
	Mean	**STD**	**Mean**	**STD**	**Mean**	**STD**	**Mean**	**STD**
Tp	0.21	0.11	0.39	0.03	0.25	0.10	0.31	0.14
Tr	0.12	0.03	0.07	0.01	0.20	0.04	0.23	0.03
Te	308.39	3.1	211.88	2.4	718.57	5.2	720.45	7.6
Lp	79.09	13.76	78.04	9.63	86.79	12.65	85.24	13.12
Nr	0.9	0.21	0.3	0.12	3.4	1.71	2.5	1.10

Table 3.2: Planning and execution results(time in seconds, length in meters)

Tab. 2 reports the results for the two scenarios (Test 1 and Test 2 in Fig. 3.5). For both these settings, initially, the obstacles are not visible, hence the generated plan is simple and planning time is low (Fig. 3.5 (left)). Once the obstacles are discovered on the fly, replanning is needed to adjust on-line the trajectory. Replanning and execution time are slightly higher in the first scenario which is more complex. Instead, Tr seems negligible when compared with Te. The final trajectory length (Te) is similar in both the settings and comparable with the distance between the starting and target point, hence the final trajectory seems not affected by the continuous replanning process. In these tests, Tp and Tr are mainly due to path and trajectory planning (task planning is negligible). We never experienced brake or escape episodes. Overall, the system task/path planning performance seems compatible with the operative scenario requirements.

Simulated planning and execution. We tested our planning and execution system in simulated environments. To test continuous replanning, we considered a larger space of dimension $100 \times 100 \times 50$ m^3 with 4 and 9 obstacles. To decouple replanning from map bulding, we assumed a know map associated with a visibility horizon (not visible obstacles are detected on the fly causing replanning). For each test, the task was to inspect a target point in pose $(90,90,5,90)$ starting from *hovering* in the pose $(5,5,5,0)$ (in meters); the robot maximum and minimum velocity was set at $0.5 m/s$ and $0.1 m/s$ respectively. By changing the visibility horizon (green cells in Fig. 3.5) of the planner (15 or 25 m) and the complexity of the environment (4 or 9 obstacles) we obtained 4 scenarios. Tab. 3.2 collects means and STD of 10 tests for each entry (time and length are in *sec*. and *m*, LL, HL, etc. are for Low complexity and Low visibility, High complexity and Low visibility).

	Physical Inspection				Visual Inspection			
	Mean	**STD**	**Max**	**Min**	**Mean**	**STD**	**Max**	**Min**
Tp	0.798	0.012	0.019	0.009	0.734	0.47	1.25	0.42
Tm	0.324	0.17	1.07	0.12	0.329	0.22	0.57	0.3
Tpp	0.473	0.27	0.71	0.14	0.405	0.07	0.49	0.39

Table 3.3: Physical inspection and visual inspection

Here, we can see that Tp increases with the obstacles (HL,HH) and decreases with short visibility (LL,HL). Indeed, in these cases the planning problem is simpler. However, short visibility is associated with additional replanning time which, in turn, decreases with the number of obstacles. The lower the replanning time, the lower is the execution time and the shorter the executed path. A similar effect is due to visibility: short visibility causes frequent replanning events (Nr) and longer paths (Lp) and execution times (Te). Furthermore, the variance is enhanced with short visibility that enhances the uncertainty. In these tests, the task planning time is usually negligible (Tp and Tr mainly due to path and trajectory). Also in this case, we never experienced brakes or escapes.

Simulated inspection. As for operations closer to the surface, we considered two typical inspection scenarios: physical (Pi) and visual inspection (Vi). In both these cases the system has to move in a pose which faces a vertical surface hovering at a close and fixed distance (approach), in this case 50 cm. As for Pi (see Fig. 3.6, left), the robot executes a docking maneuver (docking) and slides (keeping the contact) along a linear trajectory (p-inspect) of 225 cm. In the case of Vi, an inspection trajectory (v-inspect) should be planned and executed. Here, the goal is to scan a 150×100 cm^2 surface with step 50 cm distant 50 cm from the wall (see Fig. 3.6, right).

In Tab. 3.3, we collect the results of 10 tests for each scenario considering planning time (Tp) divided in trajectory (Tm) and path planning (Tpp) time (task planning is negligible). For each test and scenario, both path and task planning times are compatible with the operative scenario requirements.

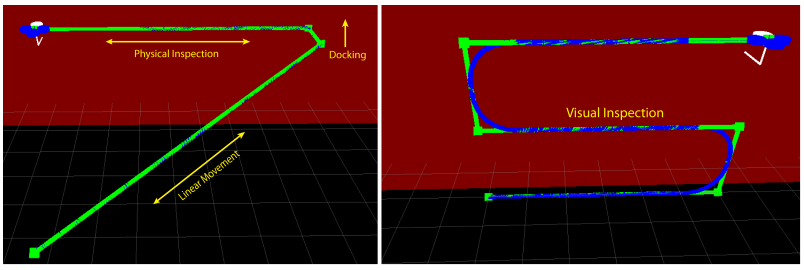

Figure 3.6: Physical inspection (left) and visual inspection (right)

3.2 A Mixed-Initiative Control System for an Aerial Service Vehicle supported by force feedback

In this section, a mixed-initiative control framework for unmanned aerial vehicles (UAVs) that enables sliding autonomy is proposed. An autonomous system that can plan and execute robotic tasks is considered while a human operator plays the role of a supervisor providing interventions when necessary. In this context, the force feedback is used to provide the feeling of the deviation between the real position of the robot and the one planned by its autonomous system. The higher the distance between the robot and the planned trajectory, the higher should be the force that the system provides to the human to bring the robot back to the planned path. The purpose is to provide the human operator with a direct perception of the displacement from the planned trajectory and to give an intuitive feeling about the direction to command in order to get back to the planned trajectory. We tested the system at work in virtual and real environments considering simple navigation tasks to be achieved in a mixed initiative control mode. We compared the performance of human operators with the proposed feedback enabled and disabled. The collected results show the effectiveness of the proposed approach.
Robot control based on haptic devices has been widely investigated in the human-robot interaction (HRI) literature. Force feedback has been used for precise remote control of manipulators [109] or to support obstacle avoidance during teleoperation [110][115]. Haptic interfaces have also been proposed to make the user able to perceive the presence of virtual objects in an augmented reality setting [111]. Analogously, in [112] artificial force fields are used supporting mobile robot navigation in virtual environments. More closely related to our approach, in [111] an event-based direct control with force feedback has been proposed to represent the difference between the actual velocity of the robot and the desired one in ground mobile robot navigation. Haptic feedback methods supporting the teleoperation of a unmanned helicopter are presented in [113], with the focus on collision avoidance. In [114] a novel haptic framework is proposed to teleoperate multiple UAVs, where an operator can remotely command some UAVs while haptically perceiving the state of the other ones.
The proposed framework is deployed in the *Aerial Service Robotics* (ASV) [106] domain. In this context, UAVs should perform not only typical navigation operations (like take-off, hover, waypoint fly, land, etc.), but also tasks in proximity and in contact with surface (e.g. docking, undocking, sample picking, wall inspection, etc.). In this domain the robot operates in cluttered environments, in proximity and in contact with the surfaces, hence both autonomous and teleoperated control should be provided. Indeed, even tough the low-level control of the robot is robust and able to estimate and compensate exeternal disturbances [119], the human can help the robot in managing difficult/unexpected events and manipulation tasks, while the robot can autonomously plan sequences of tasks, map, localize, and navigate across the environment. The system should enable a docile and smooth sliding from an autonomous mode to a teleoperated one and vice-versa. For this

purpose, we propose a framework that combines mixed-initiative planning [120] and haptic feedback.

3.2.1 HRI Control Architecture

We assume the HRI control architecture (see Figure 3.20) divided into two layers: the *High-level Supervisory Control* layer (HLS) is responsible for user interaction, and task/path planning and execution monitoring, while the *Low-level Supervisory Control* (LLS) layer manages the low-level execution of the action primitives. The operator interacts with both the HLS and LLS: at HLS he can activate high-level tasks through the *User Interface* module, while at the LLS the *teleoperation* module allows the human operator to directly control the robot movements. If the human provides high-level goals to the HLS, the system works in the autonomous mode deploying planning and execution engines to achieve the tasks. On the other hand, if the system is in the teleoperated mode, the human can directly and exclusively control the vehicle. The proposed HRI system allows us to combine these two control modalities enabling to smoothly slide from fully autonomous control to fully teleoperated control and vice versa. In the following a detailed description of these control modalities is provided.

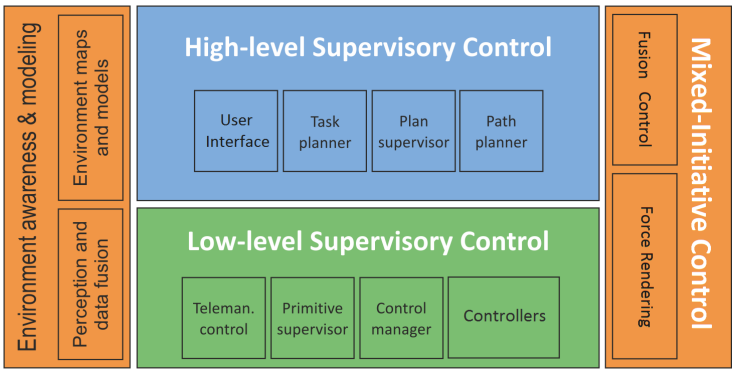

Figure 3.7: Supervisory System and Mixed Initiative Control

Planning and execution At the higher level of abstraction, the operator can specify the task the robot has to perform. This task is decomposed into a set of operations and commands that are sent to the low-level supervisory system that controls the robot actuators. The robot executive system structured as a *Belief-Desire-Intention* (*BDI*) architecture [117]. This module relies on a PRS engine and is responsible for goal management, task decomposition tracking the current high-level environmental and executive state while managing any interrupts of the active plan. Indeed, the operator can interact at any time by sending new goals or interrupting the current action execution. Once the plan is generated, its execution is managed by the *Plan Supervisor* that exploits the *Path Planner* to

generate a path between two points in a limited interval of time. In our framework, the path generation algorithm is based on a version of the Rapidly-exploring Random Tree (RRT) algorithm [130]. The generated path should satisfy a set of additional control, safety, and temporal constraints (e.g. planning and execution timeouts). The low-level *Primitive Supervisor* receives from the *Plan Supervisor* a list of micro actions associated with a sequence of waypoints, each tagged with constraints, i.e.: minimal distance from obstacles and maximum velocity. These constraints along with the micro operations are then used by the *Control Manager* to select the right controller (e.g. trading-off velocity and precision). Given the controller and the waypoints, the *Trajectory Planner* can then generate and monitor the control trajectory. Since the robot operates in cluttered unknown environments and in proximity of the surfaces, the generated trajectory is to be continuously monitored generating replanning or recovering depending on the time available to react [107][152].

Trajectory Planner The trajectory planner generates a trajectory in terms of position, velocity, acceleration, and jerk processing all the waypoints and constraints generated by the path planner. The trajectory is generated exploiting a 4-*th* order spline concatenation method that preserves continuous acceleration. In the proposed architecture this trajectory can be directly modified by the human interventions, i.e. the trajectory planner can be continuously invoked to adjust the current trajectory. In Figure 3.8 is shown an example of position, velocity and acceleration generated by the trajectory planner in which multiple trajectory replanning have been invoked.

Human-in-the-loop The human operator is involved in both the high-level and low-level control loop. At the lower level of interaction, he can interact with the robot through a remote controller either by direct teleoperation or by adjusting the planned trajectory followed by the robot. In this latter case, the human and the autonomous control data have to be suitably integrated. Figure 3.9 shows the scheme of the mixed-initiative control.

We assume that, in the mixed initiative mode, the operator controls the robot in velocity. Therefore, the remote controller generates a relative position command h_C that is added to a_C, generated by the autonomous planner. These functions are described as follow:

- $a_C(t)$: the position command (x_t, y_t, z_t) at time t generated by the autonomous system;
- $h_C(t)$: the relative position command $(x_{m_t}, y_{m_t}, z_{m_t})$ at time t, generated by the human operator.

The h_C function is calculated as follows:

$$h_C(\mathrm{t}) = \begin{cases} h_C(t-1) + joypad(t) \text{ if } mixedControl = ON \\ h_C(t-1) + \Lambda(t) \; otherwise, \end{cases}$$

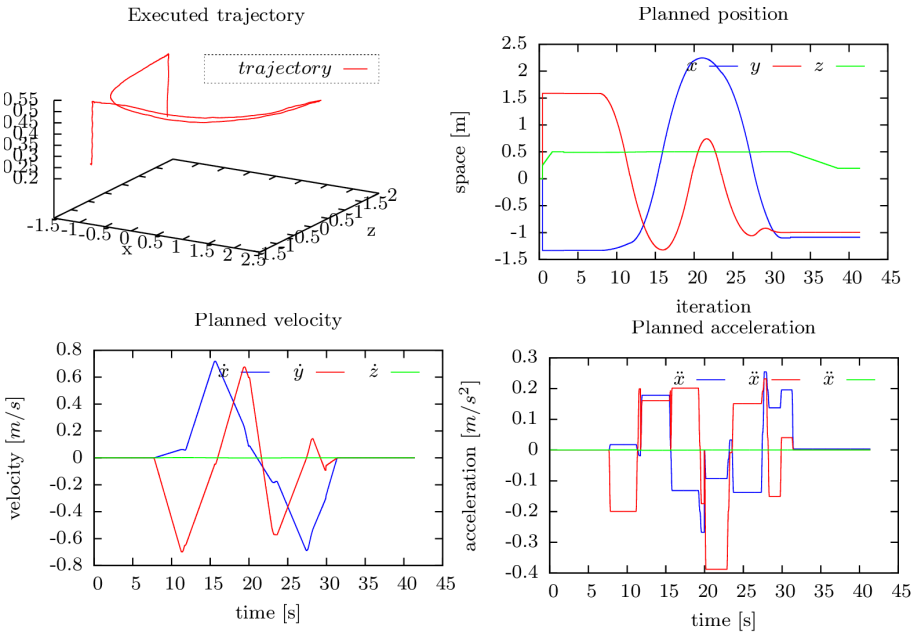

Figure 3.8: Grountruth, position, velocity and acceleration executed trajectory

where $joypad(t)$ represents the data generated by the human operator through the remote controller at time t, $(x_{j_t}, y_{j_t}, z_{j_t})$, and $\Lambda(t)$ is a linear function that increases or decreases the value of $h_C(t)$ to drive the $h_C(t)$ towards the one provided by the autonomous control. In the mixed-initiative mode, autonomous and human contributions are composed to obtain only one position command. Therefore, the human interventions can move the robot away from the planned trajectory in the direction indicated by the remote controller (see Figure 3.10 $left$). When the operator releases the remote controller, the autonomous mode is enabled and the robot is gradually brought towards the planned trajectory following the Λ function. During the teleoperation, the human operator can feel the force feedback through the remote controller. This force represents the state of the robot in terms of direction and distance from planned path.

The human operator can move the robot within a spherical region centered in the current autonomously planned position (green globe in Figure 3.10 $left$). This sphere represents the time-varying workspace of the user operator. When the robot is controlled in the autonomous mode, if an obstacle falls into the planned trajectory, the control system of the robot generates another path to reach the destination point (replanning). An analogous replanning process is started when the human operator moves the robot out of this confidence region. In this case, the autonomous system generates another path to reach the next waypoint.

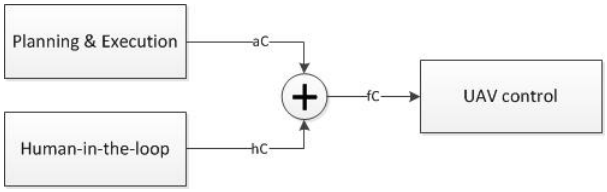

Figure 3.9: Mixed-Initiative control scheme.

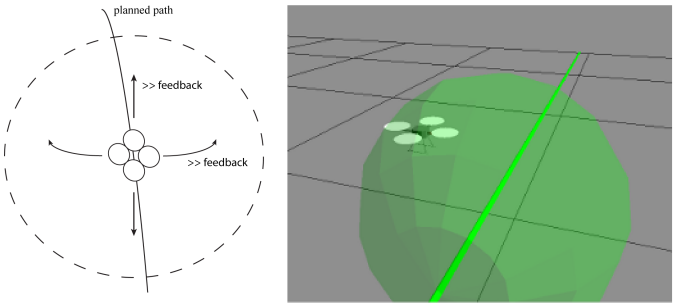

Figure 3.10: (*Left*)Deviation from the planned path, (*Right*) force feedback idea.

3.2.2 Force Rendering

The human operator interacts with the robot through a 3-DOF hapitc interface. During the mixed-initiative interaction mode, the operator receives a tactile feedback in response to the movement direction of the robot. This feedback is continuously calculated taking into account both the remote controller position and position of the robot with respect to the planned trajectory. The basic idea is that the force feedback should provide the feeling of the deviation between the real position of the robot and the one planned by its autonomous system. The higher the distance between the robot and the planned trajectory, the higher should be the force that the system provides to bring the robot back to the planned way. When a suitable threshold is reached the old plan is broken and a new one generated.

In our setting, the force feedback is a force vector in the 3D space updated with a frequency of 100Hz. The total feedback is a combination of two types of forces: the *logical force*, which is calculated according to the distance of the robot from the planned path, and the *collision preventing force* that helps the human operator to avoid obstacles in the environment. The final rendered force is given by the sum of these components.

Logical Force The logical force is calculated according to the distance of the robot from the planned trajectory. In particular, the distance value is calculated considering the point where the robot should be at that time along the trajectory. The higher the displacement from the planned trajectory, the higher is the stiffness of the remote controller. When the robot is aligned with the trajectory the force should be at zero, while, the maximum

stiffness should be provided when the robot reaches the maximum allowed distance. More specifically, the application point is the logical position of the haptic device in the robot reference system, while the endpoint of the force vector is the point on the planned path. This endpoint provides the direction vector.

Let $\vec{f_l} \in \mathbb{R}$ the force vector sent to the haptic device, this is defined by the function: $\vec{f_l} = g(\vec{x}_p, \vec{x}_r)$ where g is a function of two parameters: $\vec{x}_r$, representing the position of the robot reached by teleoperation, and $\vec{x}_p$, the position of the robot according to the planning system. Specifically, $\vec{x}_p = (x_{1_p}, x_{2_p}, x_{3_p})$, $\vec{x}_r = (x_{1_r}, x_{2_r}, x_{3_r})$. The i-th component of the force vector is given by: $f_i = k(\delta) \cdot (\hat{x}_{i_p} - \hat{x}_{i_r})$ where $k(\delta)$ is a function of δ and $\delta = ||\vec{x}_p - \vec{x}_r||$:

$$k(\delta) = k_{max} \cdot \left(\frac{1}{1 + e^{-((\delta - w_1)/w_2)}} - \frac{1}{1 + e^{w_1/w_2}} \right). \tag{3.2}$$

In particular $k(\delta)$ is a sigmoidal function, where δ is the distance of the robot from the planned trajectory, and k_{max}, w_1, w_2 are:

- k_{max} is the maximum value of the stiffness applied by the haptic controller;
- w_1 defines the position of the inflection point of the sigmoidal function;
- w_2 defines the growth rate of the sigmoidal function.

The relationship between the distance and the stiffness of the controller is shown in Figure 3.11, that contains two different graphics. In the right one, the green curve represents how far (in meters) is the robot from the planned path. In the left one, the blue curve represents the corrisponding stiffness of the grip of the controller under the human touch.

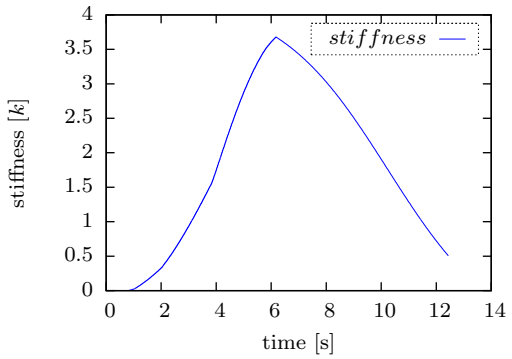

Figure 3.11: Stiffness with respect to time.

Figure 3.10 (*right*) illustrates our force feedback concepts. In this case, the UAV follows a planned path while the human operator moves it along eight directions inside the workspace (dashed circle); this movement creates a displacement between the real robot position and planned one increasing the force feedback.

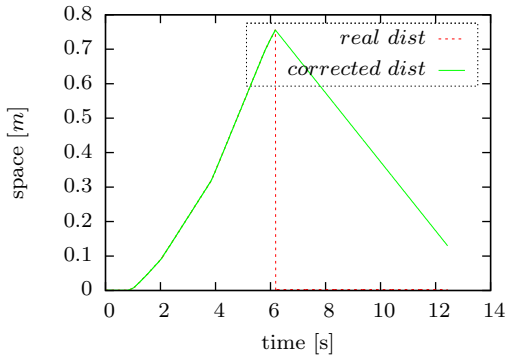

Figure 3.12: Real and corrected distance from the planned path with respect to time.

Replanning If a human operator moves the robot outside the allowed workspace, a replanning phase is invoked. In this case, the force feedback should present a discontinuity from a maximum value (maximum allowed displacement) to zero (the old path is erased while the new one starts form the current pose). The associated abrupt change of the force feedback is smoothed by assuming a linear decrease of the distance between the position of the robot and the planned trajectory (see Figure 3.12).

Collision-preventing Force The collision-preventing force is computed according to the robot direction and the distances of the obstacles. In this case, the force vector moves the robot away from obstacles, but close to the planned trajectory. For this purpose, in addition to the *Logical force* we introduce another force component that should suggest to the human operator the best way to avoid the obstacles keeping the robot in trajectory. We assume that a force feedback is provoked only if an obstacle is detected inside the robot workspace. The force associated with a single obstacle is the vector $\vec{f}_{ob_i} \in \mathbb{R}^3$ defined as follows. Let d_i be the Euclidean distance between the i-th obstacle and the robot: $d_i = ||x_r - x_{ob}||$, where $\vec{x}_r$ is the position of the robot while x_{ob} is the position of the obstacle. Let $\delta(x_p, x_r)$ be the Euclidean distance between the position of the robot x_p and the planned point on the trajectory $\vec{x}_r$. The force generated by the i-th obstacle is given by:

$$\vec{f}_{ob_i} = -\rho(d_i) \cdot \delta(x_p, x_r) \cdot \hat{d}_{ob_i},$$

where $\hat{d}_{ob_i}$ is the unit vector pointing from the robot to the i-th obstacle and the function $\rho(d_i)$ is defined as $\rho(d_i) = e^{-d_i r_w}$, with r_w radius of the spherical workspace.
For n obstacles in the workspace, the force vector sent to the haptic controller is given by:

$$\vec{f}_{ob} = \frac{\sum_{i=1}^{n} \vec{f}_{ob_i}}{\sum_{i=1}^{n} \rho(d_i)}.$$

Therefore, the total force sent to the haptic controller is a composition of the logical forces and the collision-preventing force, as shown below:

$$\vec{f}_{tot} = \alpha \cdot \vec{f}_l + (1-\alpha) \cdot \vec{f}_{ob} \qquad \text{where } 1 \leq \alpha \leq 0$$

3.2.3 Implementation

We implemented a control system for a generic UAV. The haptic device we used is the consumer 3D touch *Novint Falcon*, with 3-DOF and programmable under linux operating system. In Figure 3.13 we illustrate the main components of the implemented mixed-initiative framework. The system is implemented in C++ programing language, using ROS[2] as high-level middleware software. In particular, we have used the ROS package hector-quadrotor [116] that allows us to develop a Cartesian controller for an UAV, and Gazebo[3] as dynamics simulator.

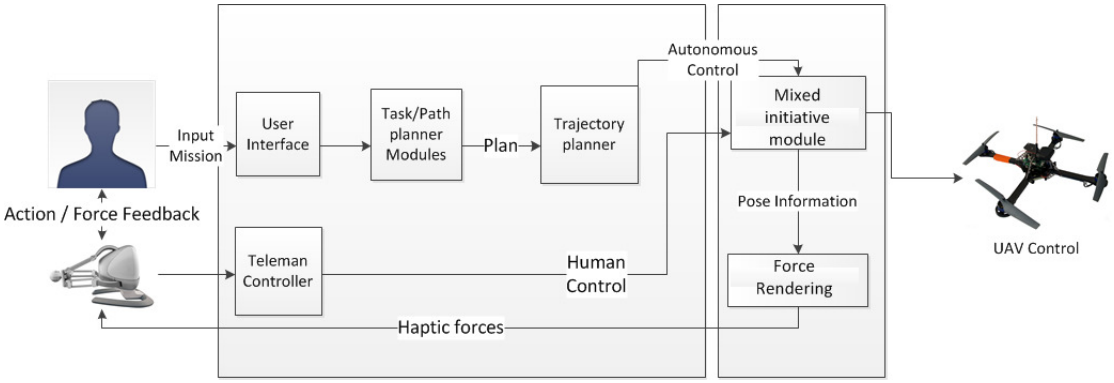

Figure 3.13: Mixed-initiative planning and execution with force feedback.

Figure 3.14 illustrates how the force vector f (left) is applicate to the hatpic device (right). Here, P_p is the point where the robot should be in the planning path, while R_p is the real position of the robot. The vector $\vec{f}$ is calculated with respect to the distance between R_p and P_p points. Once calculated, the force vector $\vec{f}$ can be applied to the haptic device which pushes the grip of the controller suggesting the operator to move the robot from the R_p point to the P_p point.

3.2.4 Experiments

We tested the effectiveness of the proposed system both in a virtual environment and in a real scenario.

In the following, we detail and discuss each test case.

Simulated tests We designed 3 scenarios (see Figure 3.15) where the user has to move from a starting position to a destination point following a planned trajectory; along the

[2]http://www.ros.org/
[3]http://gazebosim.org/

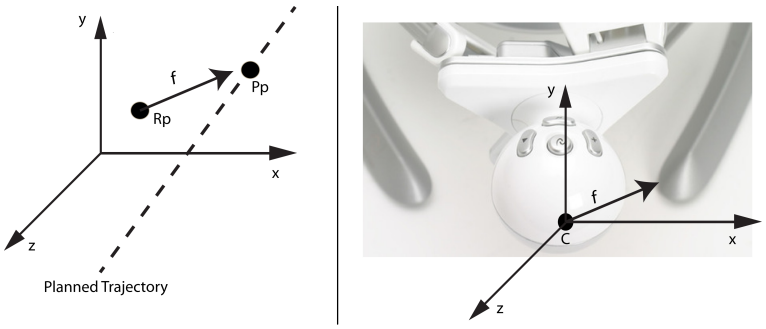

Figure 3.14: Force feedback on the haptic device.

path we introduced some deviation opportunities (marked as blue balls) and the user task is to reach the destination point avoiding obstacles and trying to pass through the maximum number of target waypoints. In this setting, we can assess the operator performance with and without the assistance of the mixed initiative system. We involved a group of 20 students and each subject was asked to repeat the experiment 3 times in the virtual 3D environment, both in teleoperation and with the support of the mixed initiative system. Every user performed the test in different order, counterbalancing the experiments with and without force feedback in order to address the problem of *learning* effect. Moreover, there was a break after each single test performance.

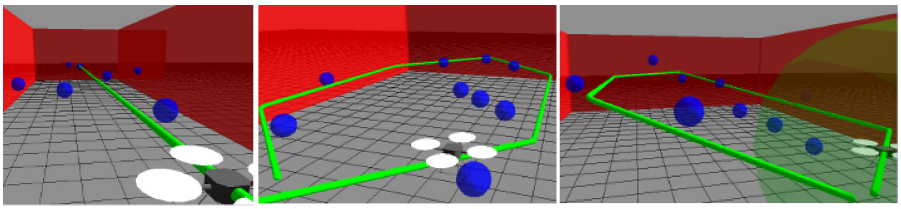

Figure 3.15: Simulated test scenarios.

Corridor scenario The first scenario simulates a corridor with a linear segment of 14 meters as a planned path and 7 landmarks to be reached, as depicted in Figure 3.15*(left)*. For each test, we measured the following variables:

- *Score*: number of reached landmarks.
- *Distance*: minimal distance of the robot from the walls.
- *Length*: length of the executed path.
- *Time*: time to reach the goal position.

In Table 3.4 we compare the values collected with and without the use of the force feedback illustrating for each entry, the minimum, the maximum, the mean value, and the standard deviation. In addition, we report the results of the t-test (2-tailed), to show the significance of the data.

Table 3.4: Corridor scenario

Score	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	5	6.6	6.06	0.5	0.0002285
No forces	3	6.3	5.19	0.8	
Distance m.	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	0.8	1.3	1.01	0.1	$<.0001$
No forces	0.4	0.9	0.68	0.13	
Path Length m.	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	19	20	19.3	0.32	0.0423755
No forces	18.6	21	19.6	0.62	
Nav. time sec.	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	68	72.4	69.9	1.3	0.089711
No forces	68	71.6	69.4	0.9	

Planar scenario The second scenario is depicted in Figure 3.15*(center)*. Here, the robot has to autonomously execute a circular path, while the human operator can deviate from the predefined trajectory in order to reach the greatest number of target waypoints. Furthermore, since a large deviation could break the planned path inducing a replanning activity, we also considered the number of replanning episodes. Indeed, path replanning activity should be minimized to minimize computation, deviations, and to keep a simple executed trajectory. In this experiment we collected the following variables:

- *Score*: number of landmarks that the user reaches.
- *Replan*: number of replanning episodes.
- *Length*: length of the total executed path.
- *Time*: time to complete the test.

In this environment we introduced twelve landmarks, each at the same altitude of the generated path. This coplanar positioning of targets allows the operator to reach the target without the need of changing its altitude. Analogously to the previous test, the minimum, maximum, and mean value for each variable and for each user are depicted in Table 3.5.

Table 3.5: Planar Scenario

Score	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	7.6	10.6	9.27	0.8	<.0001
No forces	4	7	5.4	1.16	
Re-planning	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	0	0.9	0.24	0.3	<.0001
No forces	0	2	1.1	0.59	
Path Length m.	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	27	32	28.68	1.42	0.07625
No forces	26.6	32.3	29.8	1.2	
Nav. time sec.	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	97.6	101	98.9	1.1	0.05846
No forces	98.5	103.6	100.6	1.45	

Non-planar scenario Likewise the previous scenario, in this test the robot executes a circular path in the environment illustrated in Figure 3.15*(right)*. However, in this case the waypoints are not positioned at the same altitude therefore it is difficult for the user to teleoperate the robot with the visual feedback only.

Table 3.6: Non-planar scenario

Score	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	7.6	10	8.98	0.66	<.0001
No forces	4	6.6	4.8	0.82	
Re-planning	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	0	0.6	0.13	0.2	<.0001
No forces	0	2.3	1.4	0.52	
Path Length m.	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	25.5	31.3	28.3	1.8	0.004773
No forces	25.3	30.3	27.52	1.51	
Nav. time sec.	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	97	101.3	97.98	1.22	0.000102
No forces	95.3	105.3	98.97	2.44	

Real Scenario We tested the system in an indoor Arena endowed with an OptiTrack motion tracking system. The vehicle platform is an Asctec Pelican quadrotor producted by *Ascending Technologies* with standard sensors equipment. In this context, the initial trajectory was a linear path of 3.5 meters, navigated in both directions with a movement

of 7.0 meters. Analogously to the previous test cases, the user tasks is to reach some targets outside the planned trajectory. In particular, we introduced two virtual landmarks at different heights, which becomes visible to the operator when the robot distance becomes less then 1.2 meters.

In this experiment, we involved a group of 3 students and each of them executed the test 2 times. Analogously to the non-planar case, the results in Table 3.7 supports the hypothesis that the force feedback allows us to enhance the score and to reduce replanning. In addition, for one of these experiments we reported in Figure 3.17 the planned and commanded position data generated by the autonomous system and the mixed-initiative module, respectively.

Figure 3.16: PRISMA Flight Arena

Table 3.7: Real Scenario

Score	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	1.5	2	1.8	0.28	0.014004
No forces	0	1	0.5	0.5	
Re-planning	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	0	0.5	0.16	0.28	0.0425
No forces	1	3	2.16	1.04	
Path Length m.	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	7.8	9.5	8.86	0.92	0.1869505
No forces	8.6	10.8	9.7	1.1	
Nav. time sec.	**Min**	**Max**	**Mean**	**Std**	**t-test**
Forces	46	51	48.6	2.51	0.037178
No forces	54	77	65	11.53	

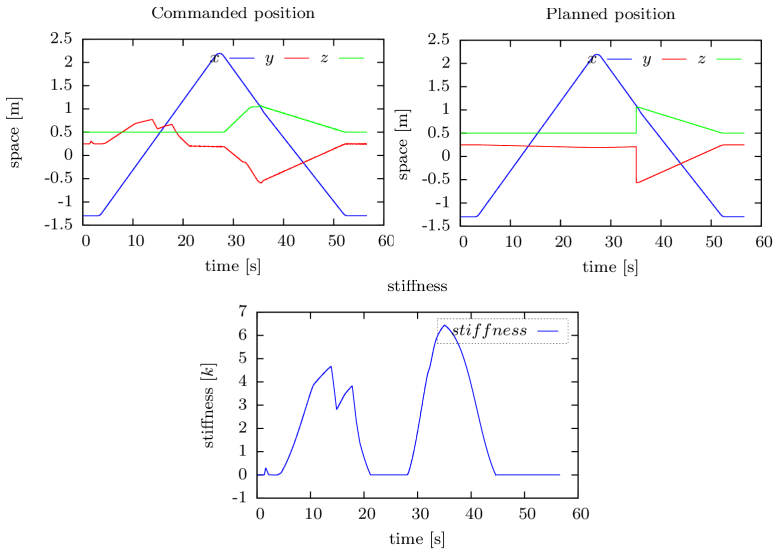

Figure 3.17: Real scenario trajectories

3.2.5 Result analysis

For each scenario, the score of the system endowed with the force feedback is greater than the no-force one. This is more evident in the second and in the third scenario of the simulated environment. In these cases, the replanning episodes are also less frequent, while navigation time and path lengths seem not affected. These results suggest that the system endowed with force feedback allows the operator to better assess how to deviate from trajectories without breaking the plan (replanning episodes). The effect of this support seems emphasized in the non-planar environment. In this case, the haptic feedback works as a sort of sensory substitution: it provides a direct sensation of the displacements which cannot be easily estimated in the virtual 3D environments. This seems confirmed by the fact that the help of the haptic assistance is attenuated in the planar environment and not evident in the corridor scenario. The same results has been noticed in the real scenario. Finally, the t-test values show the significance of the collected data.

3.3 Mixed-Initiative Planning and Execution for Multiple Drones in Search and Rescue Missions

3.3.1 Introduction

We present a mixed-initiative system for multiple drones suitable for search and rescue activities in a real-world alpine scenario. This work is framed within the SHERPA project

[138]. Differently from typical human-multidrones interaction scenarios [21, 81, 43, 10, 59], in this work we assume a human operator that is co-located with the robots and not fully dedicated to their supervision and control. In this context, the human level of involvement in supporting the robots behavior is not ensured: as a member of the rescue team involved in the search and rescue activities, the human operator might be capable to directly operate the robots, or involved in a specific task, hence only able to provide sketchy and sparse inputs. This scenario requires a framework that supports adjustable autonomy, from explicit teleoperation to a complete autonomy for the robots, and an effective and natural mixed-initiative interaction between the human and the robots [75].

The framework presented in this work should allow a single human operator to supervise and orchestrate the operations of a set of UAVs by means of a natural multimodal communication (using gestures, speech, joypad, tablet interface, etc.) supported by adjustable autonomy. In the proposed approach, we assume a high-level supervisory system that can compose and execute structured robotic tasks while the human rescuer can provide interventions when necessary. These interventions range from abstract task assignments for the multi-drone system (e.g. new areas to explore, search strategies definition, paths to follow, etc.) to navigation adjustments (e.g. deviations from planned paths or trajectories) or precise maneuvering of single robots (e.g. inspection of cluttered environments). More specifically, the proposed human-robot interaction framework combines a multimodal interaction module with a layered mixed-initiative supervisory system. The latter is composed of a multirobot supervisory system interacting with single robot supervisors. For each supervisor, the executive control cycle is managed by a BDI (Belief Desire Intention) system that orchestrates task planning, switching, decomposition, and execution. The robotic activities are represented as hierarchical tasks which are continuously instantiated and supervised by the executive system depending on the environmental events and the human requests. In this setting, the operator is allowed to continuously interact with the supervisory systems at different levels of abstraction (from high-level tasks assignment/switching to path/trajectory adjustments) while these human interventions are interpreted, monitored, and integrated exploiting the planning and execution control loops. Indeed, following a mixed-initiative planning and execution approach, these interventions can be associated with system reconfigurations which are managed by replanning activities [27, 18, 94, 11]. However, in our setting, different planning/replanning engines are strictly intertwined in order to address mission, path, and control constraints [16]. In order to evaluate the effectiveness of the proposed system, we designed a simulated rescue and search case study where a human operator interacts with a set of UAVs in order to accomplish typical searching tasks [76, 9] in an alpine scenario.

3.3.2 Search and Rescue Mission with UAVs

Following standard guidelines for search and rescue [77, 20, 76] and theory of optimal search [97], we assume the following search phases for the rescue mission: (1) define the search area for the targets; (2) define sub-areas for assignment of search pattern; (3) assign

specific search patterns to cover each sub-area; (4) define a sequence for the search patterns execution; (5) execute the chosen sequence of patterns, marking the positions of the victims found. During the execution of the mission each of these steps can be dynamically rearranged by the human expert depending on the context. In particular, the human experts should be able to refine the search areas and the associated priority value depending on the development of the mission and the new information gathered. Analogously, if the exploration is supported by UAVs, these areas can be dynamically assigned/reassigned to robots. For this purpose, we introduce some primitives to set exploration paths and areas and the associated search methods (see Table 3.8). In our setting, a search path is represented by a set of waypoints $p = \{(x_1, y_1), \ldots, (x_n, y_m)\}$ in the 2D map, instead, a search area is specified by a center and a radius $a = \langle (x, y), r \rangle$ (more complex search areas can be easily included). Search areas and paths are also associated with a priority value P_i depending on the estimated probability of finding targets in that area. The search areas can be assigned with an exploration method that instantiates one of the search patterns suggested by the NATO search and rescue manual for helicopter search [76] (here extended to drones as in [9]):

- *Sector Search (SS)*: that covers the center of the search area and permits a view of the search area from many angles (Figure 3.18, A).
- *Parallel Track Search (PTS)*: used for a uniform search coverage if the search area is large and the approximate location of the survivor is known (Figure 3.18, B).
- *Creeping Line Search (CLS)*: used when the search area is narrow and the probable location of the survivor can be on either side of the search track (Figure 3.18, C).
- *Expanding Square Search (ESS)*: used when the search area is small and the position of the survivor is known within a close limit (Figure 3.18, D).

In our setting, we assume that each pattern can be instantiated by assigning an area of search and a specific step of expansion (or an angle in the case of SS). We introduce a cost function $C_a(a, sp, u)$ that estimates the cost of the search pattern sp applied to the search area a for the drone u, analogously a cost function $C_p(p, u)$ is to assess the cost of a search path p for the drone u. In this context, once a set of search areas $A = \{a_1, \ldots, a_n\}$ and search paths $P = \{p_1, \ldots, p_m\}$ have been specified by the human expert (step (1) and (2)), that human operator should interact with the autonomous system in order to assign and instantiate the exploration tasks to the drones (step (3) and (4)) and then monitoring and orchestrating the execution (step (5)). Notice that these search assignments may be rearranged depending on the current state of the mission and the drones along with their capabilities.

3.3.3 HRI Architecture

The human operator should interact with the robots in a simple and intuitive manner, focusing the cognitive effort on relevant and critical activities (e.g. visual inspection, precise

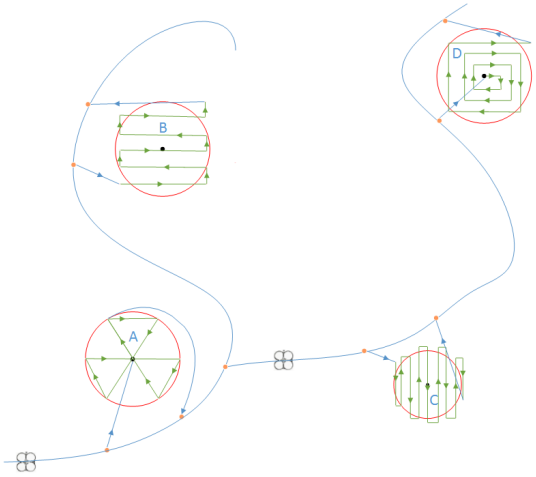

Figure 3.18: Exploration strategies for two drones searching the environments.

maneuvering, etc.) while relying on the robotic autonomous system for routinized operations and behaviors (task decomposition, path planning, waypoint navigation, obstacle avoidance, etc.). In this context, the robotic architecture must be capable of managing different control modes: *Autonomous*, i.e. the robot can plan and execute a complex task without the human support; *Manual*, i.e. each robot can be directly teleoperated by a human; *Mixed-Initiative*, i.e. the user can execute some operations, while the autonomous system reacts or reconfigures itself accordingly. For this purpose, we designed a modular architecture suitable for supervising and orchestrating the activities of both groups of robots and single robots. The operator should be capable of interacting with the system using different modalities (joypad, gestures, speech, tablet, etc.) at different levels of abstraction (task, activity, path, trajectory, motion, etc.). These continuous human interventions should be suitably and reactively integrated in the robotics control loops providing a natural and intuitive interaction.

The architecture of the HRI system presented in this section is depicted in Figure 3.19; in the following we illustrate each component.

Multimodal Interaction. The *multimodal module* allows the operator to interact with the robots using speech, gestures, joypad, tablet, etc.. The integration of different modalities permits a natural, flexible, and robust communication between the human and the system. The *speech* modality is used to control the robot both in mixed-initiative and manual control. We focused on instructions concerning movement, selection, and exploration commands (a subset of these can be found in Table 3.9 and Table 3.8). Gesture-based communication (e.g. pointing, directional signals, etc.) may be used to complete navigational commands with deictic communication (e.g. *go-there*) during proximity in-

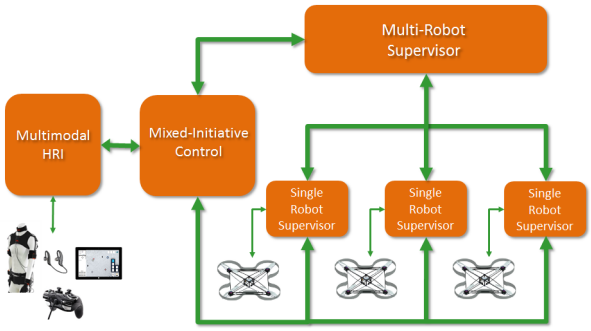

Figure 3.19: The overall HRI architecture.

teraction with the co-located drones. Joypad-based control is mostly used to manually teleoperate the robots or to adjust the execution of specific tasks. For instance, the operator is allowed to modify the robot speed, orientation or elevation through the joypad, without changing the robot task. A *display/tablet* allows the operator to keep the states of the robot, its current task and textual/graphical feedback on the environment and the operative state. Some informations, like quick notifications, may be sent to headphones, while more complex data have to be displayed. For example, a suitable map of the environment is used to select the areas and path to explore and the waypoints to reach (see Figure 3.23). Informations coming from the different channels have to be integrated to produce a single interpretation for a task, command, or query; for this purpose, we rely on a multimodal interaction framework based on a late fusion approach [89].

Multi Robot Supervisory System. The *Multi Robot Supervisory System* (MRS) is to delegate tasks to the *Single Robot Supervisory System* (SRS) and monitor their execution with respect to multi-robot integrity, resource, and mission constraints. In particular, in our context the MRS should complete and delegate the abstract, incomplete, and sketchy tasks provided by the operator. For example, the operator may only specify a set of areas to be explored without specific assignments for the single drones or assign a task that cannot be accomplished by a drone, given its current state and equipment. In particular, for each robot the MRS should track the pose, the tasks, subtasks, and actions under execution and power/battery information. Particular tasks are also associated with additional information, like the path followed or the particular region a robot is monitoring. On the other hand, the robots have to make decisions alone without continuously asking confirmations or details to the human. For this reason, the system should be able to delegate and monitor simple, but abstract commands, like *ScanArea* and *SearchPath*, which are then decomposed in detailed subtasks. Complex delegation system for UAVs are provided in the literature (e.g. [43]), since in this work our focus is on mixed-initiative human-robot interaction, we will rely on a simple, but reactive MRS managed by a BDI (Belief Desire

Intention executive system) [117] executive system (see Figure 3.20, upper layer), implemented by a PRS engine, that interacts with a hierarchical task planner [25, 74]. Notice that the BDI paradigm is particularly suited for our mixed-initiative system because it provides a flexible, reactive, and adaptive executive engine that is also intuitive for the human.

Single Robot Supervisory System. The SRS can continuously receive tasks from both the MRS and the Operator. The interaction with the latter is mediated by the *Mixed Initiative Control* (MIC) module that supervises the coherence of the human behavior with respect to the robotic behavior at different levels of abstraction (multi/single-agent task, path, trajectory); moreover, it manages the communication between the robots and the human (e.g. task accepted/refused, task accomplished, failures notifications, human decision request, etc.). This communication should be suitably filtered depending on the task and the human operative state, for this purpose the deployment of a multimodal dialogue manager is envisaged [54], however, in this work we will assume a simpler approach where the notifications are provided to the user in a rule-based fashion (depending on the task and the current state of the operator). The SRS (see Figure 3.20, second and third layer) is subdivided into two layers: the *High-level Supervisory Control* layer (HLS) which is responsible for user interaction, goal management, task/path planning and execution monitoring, while the *Low-level Supervisory Control* (LLS) layer that manages the low-level execution of the action primitives. Analogously to the MRS, also the HLS is orchestrated by a BDI-based executive system interacting with a hierarchical task planner for task decomposition. In this case, the executive system interacts also with a path planner to instantiate navigation commands and search strategies. (more details are provided in the section about mixed-initiave planning and execution). The executive engine provides goal management, task decomposition tracking the current high-level environmental and executive state. Moreover, it manages any interrupts of the active plan. In this setting, the operator can interact at any time by sending new goals or interrupting the current action execution. Once a complete plan is generated, its execution is managed by the *Plan Supervisor*. The low-level *Primitive Supervisor* receives from the *Plan Supervisor* a list of micro actions associated with a sequence of waypoints, each tagged with constraints, i.e.: minimal distance from obstacles and maximum velocity. These constraints along with the micro operation are then used by the *Control Manager* to select the right controller (e.g. trading-off velocity and precision). Given the controller and the waypoints, the *Trajectory Planner* can then generate and monitor the control trajectory.

3.3.4 Mixed Initiative Planning and Execution

In the architecture presented above, the human is allowed to continuously interact with the system at different levels of abstraction. The system supports smoothly sliding from fully autonomous control to fully teleoperated control and vice versa; this interaction is integrated into a continuous planning and execution process that reconfigures the robots

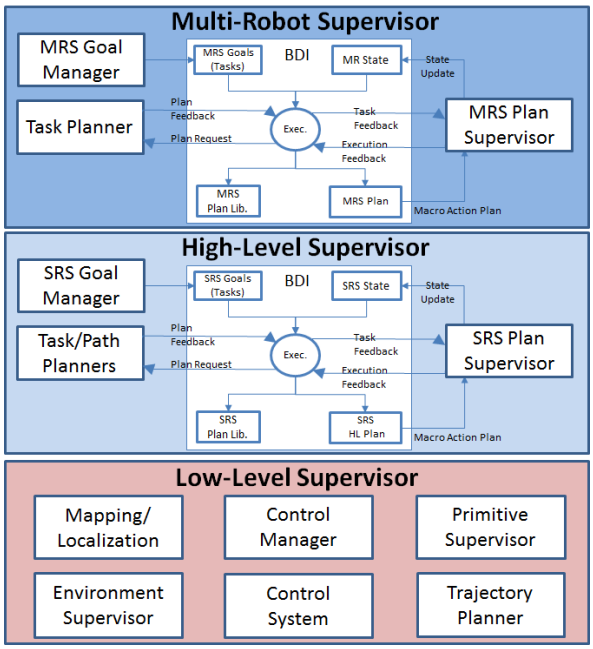

Figure 3.20: Multi Robot and Single Robot Supervisory Systems.

activities according to the human intentions and the operative state. In the following, we provide some details about this process.

Task level interaction. At the higher level of abstraction, the operator can specify the high-level tasks the robot has to perform. Each task is hierarchically represented and can be decomposed into a set of operations and commands that are sent to the lower-level supervisory system. The executive cycle of both the MRS and SRS are managed by a PRS engine that provides goal management and task decomposition; moreover it tracks the current high-level environmental and the executive state handling any interrupts of the active plan. The operator is integrated in this loop and can interact at any time by sending new goals, changing tasks, or interrupting the current action execution. Depending on the task, the executive system can also call a Hierarchical Task Planner to complete or optimize the task decomposition process. In particular, we rely on the Human Aware Task Planner (HATP) framework [74], a SHOP-like Hierachical planner [78] that can explicitly represent the human interventions. Note that the hierarchical planning paradigm - in combination with the BDI framework - is particularly suited for this domain since it allows the user to monitor and modify the plan at different abstraction levels (task, activity, path, trajectory, motion, etc.) supporting both situation awareness and an intuitive interaction with the generated plan structure. In our setting, the HATP planner is mainly invoked for the resolution of abstract tasks like, e.g. $ExploreMap$, $ScanAreas(A)$, $SearchPath(P)$,

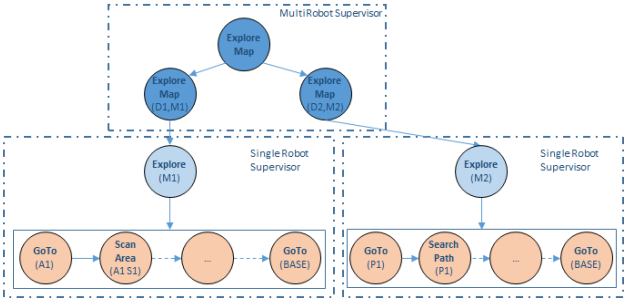

Figure 3.21: Example of a hierachical plan where the MRS task decomposition is then refined by the SRSs.

etc. (a list of possible tasks/subtasks in our domain can be found in Table 3.8). Each task can be further specified with the explicit assignment of the parameters, e.g. the robot r and the area a to be scanned can be explicitly provided (i.e. $ScanArea(r,a)$) along with the p search pattern (i.e. $ScanArea(r,a,p)$). Notice that, if the UAV is not explicitly defined by the operator, the MRS system should generate the assignments trying to maximize the overall mission reward (see Figure 3.21); otherwise, the task can be directly provided to the SRS of a specific robot, in this case the MSR should only check for constraints violations. In the HATP, each operation A_k^a for a robot a can associated with a duration D_k^a and a cost function C_k^{ctxt}. For instance, in our scenario, the estimated cost of scanning an area a with the search pattern sn for a robot u, evaluated in the current context, is the sum of the cost of reaching the area $C_r(u,a)$ and the cost of scanning it $C_a(a,sp,u)$ minus the reward gathered for the area exploration which is proportional to the associated priority P_a. As far as multi-robot constraints are concerned, HATP allows us to define specific *social rules* associated with a cost for their violation $\langle S_k, P_k^{ctxt} \rangle$. In our domain, we only penalize plans where robots explore the same areas or paths and unbalanced distributions of the search effort for the available drones. Therefore, each plan P is associated with a cost:

$$Cost(P) = \Sigma_{a_i \in P} C_{a_i}^{ctxt} + \Sigma_{s_k \in P} P_{s_k}^{ctxt},$$

where a_i is an action of the plan P, s_k is a social rule. In this context, the planner should provide a feasible plan P associated with the minimal cost obtained before a specific time-out. Indeed, the executive system invokes the planner providing a latency; if a solution cannot be generated within the planning latency a default task is executed to recover from the plan failure. In our case, the timeout is defined by context- and task-based rules, for instance if a robot is landed or hovering the planning latency can be extended (up to 5 sec. in our tests), instead, during the flight it can be reduced (max 1 sec. in our tests), otherwise, if the mission time or the energy is below a suitable threshold only a reactive recovery behavior is allowed. More complex policies can be easily introduced and assessed.

Task	Description
SetSearchArea	Add/delete/modify a search area
SetSearchPath	Add/delete/modify a search path
SetSearchPattern	Change the exploration method
SearchPath	Search along the path
ScanArea	Scan the area with a pattern
ExploreMap	Explore the map
GoTo	Move towards a direction or an area
AbortMission	abort the overall mission

Table 3.8: List of mission level tasks.

Command	Description
Up	Take off or increase of the altitude
Land	Move the robot to the ground
Down	Decrease the altitude
Left	Move the robot to the left
Right	Move the robot to the right
Forward	Move the robot ahead
Backward	Move the robot to the back
Away	Move the robot away from the target
Closer	Move the robot towards the target
Faster	Increase the speed of the robot
Slower	Decrease the speed of the robot
Go	Move the robot to a specific position
Rotate	Rotate the robot with a specific angle
Switch On/Off	Turn on/off the UAV engines
Brake	The robot brakes

Table 3.9: List of navigation commands.

Path and Trajectory level interaction. The primitive tasks introduced above (e.g. $Explore(a,u,p)$, $GoTo(a)$, etc.) are associated with drone movements to be suitably planned and executed. We deploy an RRT^* algorithm [37] for the generation of obstacle-free paths in the 3D space. Given the waypoints and constraints (proximity of the obstacles) provided by the path planner, a trajectory planner generates a trajectory in terms of position, velocity, acceleration, and jerk. The trajectory is generated exploiting a 4-*th* order spline concatenation method that preserves continuous acceleration. In the proposed architecture, this trajectory can be directly modified by the human interventions, i.e. the trajectory planner can be continuously invoked to adjust the current trajectory. Indeed, in the mixed-initiative mode, autonomous and human contributions are composed to obtain only one position command. This way, the human interventions can move the robot away from the planned trajectory. However, when the human intervention is released the autonomous mode is enabled and the robot is gradually brought towards the planned trajectory without the need of replanning. More specifically, we assume that, in the mixed initiative mode, the operator can control the robot in velocity. In this setting, the human generates a relative position command $h_C(t) = (x_{m_t}, y_{m_t}, z_{m_t})$ that is added to the $a_C = (x_t, y_t, z_t)$ which is generated by the trajectory planner. The h_C function is calculated as follows:

$$h_C(\mathrm{t}) = \begin{cases} h_C(t-1) + human(t) \text{ if } mixed = ON \\ h_C(t-1) + \Lambda(t) \ otherwise \end{cases}$$

where $human(t)$ represents the control reference generated by the human operator (through the joypad, gestures, voice, etc.) at time t while $\Lambda(t)$ is a linear function that increases or decreases the value of $h_C(t)$. It is used to drive the $h_C(t)$ towards the one provided by the autonomous control when the joypad is released (see [14] for an analogous approach). Moreover, we assume that the human operator can move the robot within a spherical region centered in the current planned position (see Figure 3.22). This sphere represents the context-dependent workspace of the user operator. A replanning process (analogous to the one used for obstacle avoidance) is started when the human operator moves the robot out of this sphere. In this case, the autonomous system generates another path and trajectory to reach the next waypoint. Additional details can be found in [14]. Note that path and trajectory replanning can also elicit task replanning if the conditions associated with the execution of the current tasks are not valid anymore (e.g. preconditions, energy, resource, and time constraints); these consistency conditions are continuously assessed by the PRS executive systems (single and multi-robot).

In Table 3.9, we can find some examples of commands that the operator can provide to the system in a multimodal manner (joypad, speech, gestures, etc.) to interact with the robots or to directly controlling them. Note that these commands can be sketchy and context-dependent, for instance, if the robot is in *idle state* the *Up* is for *take off*, otherwise it will increase the UAV altitude. The navigational commands (*left, right, forward, backward*) are robot dependent (e.g. *left* moves the robot to the left side of its camera) and can be abstract (the actual movement can be instantiated by the system) or more specific (e.g. *left* $1m$). The *faster/slower* commands change the robot speed during the execution of a

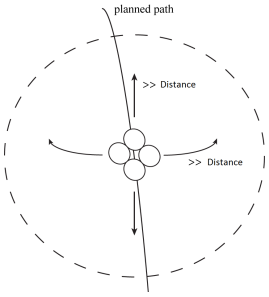

Figure 3.22: Spherical envelope for trajectory adjustments without replanning.

command (they have no effect if the robot is idle); each invocation of these commands will increase/decrease the actual speed to a given percentage up to a limit value. The *go* command moves the robot towards a specific location associated either to coordinates stored on the map or to a symbolic location (either already provided in the map or marked by the operator during the mission execution).

3.3.5 Simulation and Evaluation

A simulated alpine scenario has been defined with different configurations in order to evaluate the system performance. In this section, we illustrate the scenario and some initial tests we carried out in order to assess the system at work during a typical search and rescue mission in the three modalities: manual control, mixed initiative, and fully autonomous. In this setting, we started to consider only joypad and tablet based interactions.

Platform. We assume a set of simulated quadrotors with the specification of the Asctec Pelican (flight time 20 min., max. airspeed 16 m/s, max. climb rate 8 m/s, max. payload 650 g, etc.) equipped with standard sensors. The overall software system has been developed in ROS under linux uduntu 12.04. The environment has been simulated using the Unity3D game engine.

Environment. The simulated scene, depicted in Figure 3.23, includes several situations in which an hiker might have lost its way. We considered both summer and winter features. The scene comprises missed hikers and some associated items, either lost by the hikers or irrelevant objects. These objects can help the operator in the search operations (clues), but also divert him/her away from the right direction.

Test and scenario description. The performance of the system has been evaluated considering three control modalities. In the first test the system works in the autonomous mode, lacking any interaction with the operator. In the second test case, the operator

Figure 3.23: Specification of search paths and areas in a simulated environment (tablet interface). The red segments are possible paths followed by missed hikers, darker areas are associated with a higher priority value.

Figure 3.24: Simulated environment: starting point (base) for the two drones in our tests.

Figure 3.25: User interface during the simulated mission in the mixed initiative mode. The camera streaming of the controlled drone is full screen, the other drone camera and the environmental map are the smaller windows on the right. Text messages for the two drones are illustrated at the bottom.

may only use the joypad, while the high-level supervisory system is disabled both at the multi-robot and at the single robot level. In this scenario, when one of the robots is not directly operated it waits in the hovering state. In the third case, the operator is supported by the overall system and can work in the mixed initiative control mode. The main aim here is to illustrate the mixed initiative framework at work in a typical rescue scenario comparing its performance with respect to the ones obtained in the other two modalities. In Figure 3.23, we illustrate the map provided to the user at the beginning of each test: it represents the search environment where paths, areas, and likelihood values of finding survivors are represented. The interface employed for the tests is depicted in Figure 3.25, here the video streaming of the controlled robot is full screen, while smaller windows show the video streaming of the other robot and the environmental map with the robots positions. Messages from each robot are also provided though the interface; the information available to the user during the tests depends on the control mode as illustrated in Table 3.10.

Each test starts with the robots positioned in a fixed point and should end in that position (see Figure 3.24). During the tests, we assumed a perfect positioning system and a perfect object/human detection system when the target is in the camera field of view (50-degree) within a range of $30m$.

In the autonomous case, we assume that the mission is planned by the MSR-level task planner at the start and then reactively adjusted by the autonomous system during the exploration, depending of the detected objects. When a relevant object is detected, the robot is to replan in order to explore the surrounding area with a predefined scan path. Analogously, when the battery power falls below a suitable threshold, the robot should replan in order to come back to the initial position.

Test Type	Autonomous	Manual	Mixed-Initiative
Information available	- Num. of survivor - Loc. of areas	- Num. of survivor - Loc. of areas	- Num. of survivor - Loc. of areas
Messages Shown	- None	- Time elapsed - Battery warning - Survivor conf.	- Salient object - Survivor alert - Exploration start/end - Time elapsed - Failure request.
Control	- Autonomous	- Teleoperated	- Mixed-Initiative

Table 3.10: Information available to the user during the tests.

In the teleoperated mode, the robots are directly controlled through a tablet and a joypad used to define direction, speed, orientation of the robot and of the camera. During the tests, the user receives only two messages: the confirmation of the effective survivor sighting and a warning about the battery level, if it drops below a fixed threshold.

In the *mixed initiative* scenario, the user can interact with the MRS and SRS during the mission using the tablet and the joypad. The graphical information provided by the tablet interface is similar to the one of the *teleoperated* case, but additional textual information is provided (see Table 3.10). Analogously to the *autonomous* mode, also in this case the mission is planned in advance, but the human is allowed to provide interventions at the plan and the trajectory level. On the other hand, the system alerts the operator when salient clues are detected by a robot. The operator can then inspect the clues and decide whether to check the area. The operator can always inspect the current operative and environmental state: including robots position, speed, state, task, plan. These notices can be supplied through different channels, like audio notifications or messages on a tablet. In this setting we decided for the text notification on the tablet.

Test set-up. In our tests, we considered the scenario depicted in Figure 3.23 to be explored by 2 robot. The mission goal is to find 15 persons within 10 minutes. The testing area is a simulated environment of $160 \times 140\ m^2$ with 9 areas to be explored and 9 paths. 9 clues (one for victims) and 21 irrelevant objects (distractors) are randomly distributed on the environment. We defined 3 different dispositions of survivors and objects within the scene. In the first case (test A), all the targets are located inside the areas and paths with a uniform distribution. In the second (test B) and third case (test C), the 66% and 13% of the targets is located inside the areas/paths with a uniform distribution, while the remaining are uniformly positioned in the rest of the scene. Each target can be associated with 1 clue which is positioned within a range of $20m$. In the teleoperation and mixed-initiative modes, each modality has been executed 12 times, by 4 users (3 tests for each mode after 2 session of training).

Targets	**Min**	**Max**	**Mean**	**Std**	**t-test**
Mixed-Initiative	12	14	12.00	0.84	0.1009915
Autonomous	11	12	11.25	0.5	
Survivors	**Min**	**Max**	**Mean**	**Std**	**t-test**
Mixed-Initiative	12	14	12.00	0.84	0.0001655
Teleoperation	6	8	7	0.71	

Table 3.11: Mixed-initiative mode vs. automous and teleoperated mode (test A).

Targets	**Min**	**Max**	**Mean**	**Std**	**t-test**
Mixed-Initiative	11	12	11.40	0.55	0.034469
Autonomous	7	13	9.20	2.28	
Survivors	**Min**	**Max**	**Mean**	**Std**	**t-test**
Mixed-Initiative	11	12	11.40	0.55	$<.0001$
Teleoperation	4	7	5.8	1.3	

Table 3.12: Mixed-initiative mode vs. automous and teleoperated mode (test B).

Targets	**Min**	**Max**	**Mean**	**Std**	**t-test**
Mixed-Initiative	7	9	7.80	0.84	$<.0001$
Autonomous	2	3	2.6	0.55	
Survivors	**Min**	**Max**	**Mean**	**Std**	**t-test**
Mixed-Initiative	7	9	7.80	0.84	$<.0001$
Teleoperation	3	5	3.6	0.89	

Table 3.13: Mixed-initiative mode vs. automous and teleoperated mode (test C).

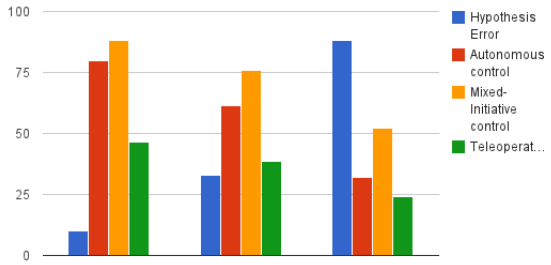

Figure 3.26: Percentage of success in target detection with respect to the control modalities.

Results. In Figure 3.26, we illustrate the percentage of survivors found in the three control modalities with respect to the different test cases (A, B, C). As expected, the teleoperated mode is not effective, indeed not only the parallel search of the two drones cannot be exploited in this case, but also the lack of task/path guidance reduces the overall situation awareness, hence the number of correct detections is significantly lower than in the other modes. On the other hand, since we assume a reliable human/object detection system, the autonomous mode is very effective when the initial hypothesis is accurate. In this case, the two robots can find about 80% of survivors, scanning all the areas. However, the success rate rapidly drops when the initial hypothesis becomes less accurate; indeed, the autonomous system is not flexible enough to diverge from the planned activities. Instead, the mixed-initiative mode seems more effective than the autonomous mode for each of the test cases and this advantage seems emphasized when the initial hypothesis becomes wrong. Indeed, in the worst case, the mixed-initiative mode behaves significantly better (57.78%) than the autonomous (32.25%) and the teleoperated ones (24.12%). The significance of these results is illustrated in the Tables 3.11, 3.12 and 3.13 where we compare the mixed-initiative, autonomous, and teleoperated results for each of the 3 tests cases. The comparison of the mixed-initiative performance in the three testing scenarios can be found in Table 3.14.

In the mixed-initiative mode, we also analyzed the human interventions for the different cases. In Table 3.15 we can observe that, as expected, the percentage of the time spent in teleoperation (joypad usage) increases with the complexity of the domain, indeed, when the initial hypotheses are wrong the user should intensify the interventions diverging from the planned activities. This is directly correlated with the increment of the low-level interventions (e.g. trajectory corrections or teleoprated search) and task replanning episodes. The latter can be directly invoked by the user or indirectly elicited by

Targets	**Min**	**Max**	**Mean**	**Std**	**t-test**
Mixed-Initiative (test A)	13.2	12	12.00	0.84	0.0019205
Mixed-Initiative (test B)	11	12	11.40	0.55	
Survivors	**Min**	**Max**	**Mean**	**Std**	**t-test**
Mixed-Initiative (test B)	11	12	11.40	0.55	$<.0001$
Mixed-Initiative (test C)	7	9	7.80	0.84	
Survivors	**Min**	**Max**	**Mean**	**Std**	**t-test**
Mixed-Initiative (test A)	13.2	12	12.00	0.84	0.000459
Mixed-Initiative (test C)	7	9	7.80	0.84	

Table 3.14: Mixed-Initiative performance w.r.t. the accuracy of the initial hypothesis.

external events (e.g. object detection) or constraint violations (e.g. low energy, resource conflicts, etc.). Note that the high-level interventions seem more sparse because are usually associated with strategic decisions (e.g. new areas to be explored). Notice also that since the operator can easily provide direct adjustments during the mixed-initiative mode without provoking replanning, the time spent in teleoperation remains high for each of the cases analysed in Table 3.15.

		Test A	**Test B**	**Test C**
Joypad usage (%)	**Mean**	30.67%	43.07%	53.04%
	Std	1.53	2.64	2.34
Low Level Int.	**Mean**	5.4	6.4	6.8
	Std	1.34	2.19	1.92
Task Replanning	**Mean**	10.0	12.0	13.8
	Std	0.83	1.87	3.49

Table 3.15: Human interventions and task replanning episodes during the 3 test scenarios in the mixed initiative mode.

Chapter 4

Low Level Control

In this Chapter the *Low-level* control problem of an *UAV* is addressed. If in higher levels of the architecture the goal is to select which action perform, at this layer the aim is to define how to do that action, calculating the *low-level* control input to send to the robot. Since we are mainly interested to *UAV* platforms, we could consider these input as the position or velocity of the *UAV* reference frame or directly the propeller velocities. In the followings two approaches to *low-level* control of a single *UAV* are presented. In the first approach, the *UAV* is equipped with a 6-*DOF* arm to allowing aerial manipulation, that is a key task for service robotics field. In this approach a vision based method method for aerial grasping and plugging of structured bars. In the second method a robust position control method that allow the quadrotor to react to external unmodeled disturbances is presented. This approach is particularly interesting for our alpine domain, in which the continuous and variable wind of the environment make the platform difficult to control.

4.1 Hybrid Visual Servoing with Hierarchical Task Composition for Aerial Manipulation

4.1.1 Introduction

In the last years new application domains in the field of aerial service robotics have been addressed by researchers from different disciplines, *e.g.* surveillance, inspection, agriculture, delivering, etc. Sophisticated prototypes have been developed with the capacity to physically interact with the environment [61]. The modeling and control of an unmanned aerial vehicle (UAV) able of interacting with the environment to accomplish simple robotic-manipulation tasks have been proposed in [63], which is based on a force-position control law designed through a feedback linearizing technique.

With the improvement of the batteries and the miniaturization of motors and servos, new high-performance UAV prototypes endowed with a robot arm —called unmanned aerial manipulators (UAMs)— have been designed. A control algorithm which is able to

exploit all the degrees of freedom (DoFs) of a UAM is proposed in [32], where the execution of tasks with physical interaction with the environment has been achieved. However the employed UAM is completly actuated only along one direction and has no redundancy. In [50, 48] the dynamic model of a UAM and a Cartesian impedance control have been designed providing a desired relationship between external wrench and the system motion. However, redundancy is exploited in a rigid way. Aerial manipulation tasks executed with a UAM endowed with a 2-DoFs robot arm have been presented in [39], where an adaptive sliding-mode controller has been adopted. A control solution considering valve turning with a dual-arm UAM has been proposed in [41]. In these works no vision and redundancy are employed for the task execution.

The use of vision for the execution of aerial robotic tasks is a widely adopted solution to cope with unknown environments. In [66, 71] new image-based control laws endowing a UAM with the capability of automatically positioning parts on target structures have been proposed, where the system redundancy and underactuation of the vehicle base have been explicitly taken into account. A task-oriented control law for aerial surveillance has been proposed in [93], where a camera is attached to the end-effector of the robot arm to perform visual servoing towards a desired target. However in these sections redundancy is employed in a rigid way and the interaction between dependent tasks is not considered.

In this section a hybrid image- and position-based visual servoing via a hierarchical task-composition control is presented for the control of a UAM. The presence of redundancy in a UAM system allows combining a number of subtasks with a new hierarchical-task formulation. Different subtasks can be designed both in the Cartesian space, *e.g.* obstacle avoidance, manipulation tasks, and in the image space of the camera, *e.g.* field-of-view constraints, as well as in the arm joint space, *e.g.* center-of-gravity balancing, joint-limits avoidance, manipulability, etc. Moreover, the underactuation of the aerial vehicle base has been systematically taken into account within a new recursive formulation. A number of practical tasks have been designed requiring only few DoFs to be accomplished, hence allowing an accurate profiling of the system behavior. The study of the task Jacobian singularity and a smooth task activation mechanism are also presented.

With respect to our previous work [12], a new advanced formulation is derived with the capability to guarantee decoupling of independent tasks (not only orthogonal as in the previous work), the stability analysis of the new proposed control law is discussed together with the derivation of all the task Jacobian matrices (in [12] the Jacobian matrices of the uncontrollable variables are missing), and finally both simulation and experimental results are provided to evaluate the effectiveness of the new proposed control law.

4.1.2 Reference frames and camera model

Let $o_b \in \mathbb{R}^3$ and $R_b \in SO(3)$ be the position and rotation matrix, respectively, of the body reference frame $\{\mathscr{B} : O_b - x_b y_b z_b\}$, which is fixed with the UAV base, with respect to the inertial reference frame $\{\mathscr{I} : O - xyz\}$ (see Fig. 4.1). The triple $\phi = (\varphi, \vartheta, \psi)$ of Euler roll-pitch-yaw angles is considered for the representation of the vehicle orientation, which

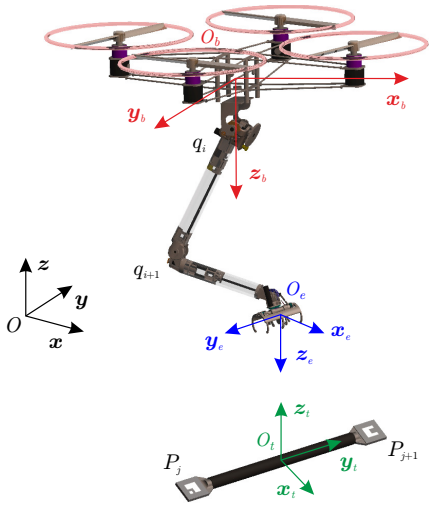

Figure 4.1: Reference frames.

in matrix form can be expressed as $R_b(\phi)$.

A standard VToL UAV, *e.g.* a quadrotor, is an underactuated system with only 4 DoFs. In fact, the linear motion of the vehicle is generated by modifying the attitude, and thus the total propeller thrust generates a linear acceleration in the desired direction. Therefore, the roll and pitch rotations are constrained by imposing a desired linear motion.

A robot arm with v DoFs attached to the UAV base is considered. Let $q = \begin{bmatrix} q_1 & \dots & q_v \end{bmatrix}^\top$ be the arm joint vector describing the arm configuration, where q_i is the ith joint variable, with $i = 1, \dots, v$, and let $\{\mathscr{E} : O_e - x_e y_e z_e\}$ be the reference frame fixed with the arm end-effector (*e.g.* gripper). Notice that, by considering the DoFs of the vehicle base, even if a VToL UAV is an underactuated system, the addition of enough DoFs of the robot arm can generate a task-redundant system ($n = 4 + v$ total DoFs), *e.g.* with respect to the positioning of the gripper.

Finally, the vehicle base is endowed with a downward looking camera. Without loss of generality, the camera reference frame is considered coincident with $\mathscr{B}$. The pin-hole camera model is employed, *i.e.* by denoting with $p^b = \begin{bmatrix} x^b & y^b & z^b \end{bmatrix}^\top$ the position of an observed point P expressed with respect to $\mathscr{B}$ (see Fig. 4.1), the optical projection of P onto the normalized image plane of the camera determines the so-called image feature vector s:

$$s = \begin{bmatrix} X \\ Y \end{bmatrix} = \frac{1}{z^b} \begin{bmatrix} x^b \\ y^b \end{bmatrix}. \tag{4.1}$$

4.1.3 Object pose estimation

We assume that m known visual markers are attached to the target object. Let $\{\mathscr{T} : O_t - x_t y_t z_t\}$ be a reference frame fixed with the object, and o_j^t and R_j^t, with $j = 1,\ldots,m$, be the known position and rotation matrix, respectively, of the reference frame attached to the jth marker in $\mathscr{T}$.

By using a visual tracker [26, 2], the pose of any visible marker can be measured with respect to the camera. Then, the pose of the target object can be reconstructed from the measurement of each visible marker as follows:

$$o_t^b = o_j^b - {R_j^b}^\top o_j^t \tag{4.2}$$

$$R_t^b = R_j^b {R_j^t}^\top . \tag{4.3}$$

However, if more markers are simultaneously visible, a more robust and accurate solution can be achieved by combining the available information. Each marker contour is divided in several points (*e.g.* the corners and the middle points of the edges). Assuming the 3D model of the marker is known, we define the 3D coordinates of each contour point l relative to its marker frame as p_l^j. Then, each 3D point of a marker is associated to the corresponding object frame $\mathscr{T}$ with

$$p_l^t = R_j^t p_l^j + o_j^t . \tag{4.4}$$

Once all contour correspondences are associated to the object frame, a Perspective-n-Point method [46] is used to obtain the camera pose with respect to the object. A RANSAC outlier rejection mechanism is integrated to remove point correspondences resulting from imaging artifacts that might be inconsistent with the computed transformation [28].

4.1.4 Dynamic task priority control

The kinematic redundancy in the system allows control of the UAM end-effector by simultaneously achieving a number of secondary tasks. Indeed, redundancy means that the same gripper pose can be reached with several, even infinite, system configurations. Hence, the system can be suitably reconfigured by using internal motions, *i.e.* without affecting the gripper pose, to satisfy mechanical constraints (*e.g.* arm joint limits) and several subtasks (*e.g.* field of view, arm manipulability, robot-arm center of gravity control).

Let $x = \begin{bmatrix} o_b^\top & \phi^\top & q^\top \end{bmatrix}^\top$ be the system state vector, and $\omega_b = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^\top$ the angular velocity of the body frame. To easily address the system under actuation we extract from x only the controlled variables in the new vector Notice that the angular velocity ϖ can be measured with a standard onboard inertial measurement unit (IMU) under the assumption of a classical time-scale separation between the attitude controller (faster control loop, *e.g.* up to 1 kHz) and the velocity controller (slower control, in our case at camera frame rate of 25 Hz).

Moreover, let $\sigma_0 = f_0(x) \in \mathbb{R}^{\mu_0}$ be the variables of a configuration-dependent main task; hence the following differential relationship holds:

$$\dot{\sigma}_0 = \frac{\partial f_0(x)}{\partial x}\dot{x} = J_0(x)\upsilon + \overline{J}_0(x)\varpi, \tag{4.5}$$

where $J_0(x) \in \mathbb{R}^{\mu_0 \times n}$ and $\overline{J}_0(x) \in \mathbb{R}^{\mu_0 \times 2}$ are the main task Jacobian matrices of the controlled and uncontrolled state variables, respectively. By inverting (4.5) and considering a regulation problem of σ_0 to the desired value σ_0^*, hence by defining $\widetilde{\sigma}_0 = \sigma_0^* - \sigma_0$ as the main task error, the following velocity command can be considered:

$$\upsilon^* = J_0^{\dagger}(\Lambda_0\widetilde{\sigma}_0 - \overline{J}_0\varpi), \tag{4.6}$$

where $\Lambda_0 \in \mathbb{R}^{\mu_0 \times \mu_0}$ is a positive-definite gain matrix, and $J_0^{\dagger}$ is the generalized inverse of J_0, which has been assumed to be full-rank. By substituting (4.6) into (4.5), the following exponentially stable error dynamics is achieved

$$\dot{\widetilde{\sigma}}_0 = -\Lambda_0\widetilde{\sigma}_0. \tag{4.7}$$

In case of $\mu_0 < n$ a second lower-priority subtask $\sigma_1 = f_1(x) \in \mathbb{R}^{\mu_1}$ can be added with the following command:

$$\upsilon^* = J_0^{\dagger}\Lambda_0\widetilde{\sigma}_0 + (J_1N_0)^{\dagger}\Lambda_1\tilde{\sigma}_1 - \overline{J}_{0|1}\varpi, \tag{4.8}$$

with J_1 the Jacobian matrix of the second subtask, which is assumed to be full-rank, and

$$\overline{J}_{0|1} = (J_1N_0)^{\dagger}\overline{J}_1 + (I_n - (J_1N_0)^{\dagger}J_1)\overline{J}_{0|0}, \tag{4.9}$$

where $N_0 = I_n - J_0^{\dagger}J_0$ is the projector onto the null space of J_0, with I_n the n-dimension identity matrix, $\overline{J}_{0|0} = J_0^{\dagger}\overline{J}_0$, $\Lambda_1 \in \mathbb{R}^{\mu_1 \times \mu_1}$ is a positive definite gain matrix. The Jacobian matrix $\overline{J}_{0|1}$ allows the compensation of the variation of the ϖ. Notice that matrix J_1N_0 is full-rank only if the two tasks are *orthogonal* ($J_1J_0^{\dagger} = O_{\mu_1 \times \mu_0}$) or *independent* (not orthogonal and $\text{rank}(J_0^{\dagger}) + \text{rank}(J_1^{\dagger}) = \text{rank}([J_0^{\dagger}\ J_1^{\dagger}])$). See [4] for more details. Substituting (4.8) into (4.5) and by noticing that N_0 is idempotent and Hermitian, hence $(J_1N_0)^{\dagger} = N_0(J_1N_0)^{\dagger}$, the dynamics of the main task (4.7) is again achieved and so the exponential stability is proven. To study the behavior of the secondary task σ_1 we can differentiate the subtask variables as follows

$$\dot{\sigma}_1 = \frac{\partial f_1(x)}{\partial x}\dot{x} = J_1(x)\upsilon + \overline{J}_1(x)\varpi. \tag{4.10}$$

Then, by substituting (4.8) and by assuming that the tasks are at least independent, the following error dynamics is achieved

$$\dot{\widetilde{\sigma}}_1 = -J_1J_0^{\dagger}\Lambda_0\widetilde{\sigma}_0 - J_1(J_1N_0)^{\dagger}\Lambda_1\tilde{\sigma}_1 + \left(J_1J_0^{\dagger}\overline{J}_0 + J_1(J_1N_0)^{\dagger}(\overline{J}_1 - J_1J_0^{\dagger}\overline{J}_0) - \overline{J}_1\right) \tag{4.11}$$

$$\varpi = -J_1 J_0^{\dagger} \Lambda_0 \widetilde{\sigma}_0 - \Lambda_1 \tilde{\sigma}_1, \tag{4.12}$$

where we used the property $J_1 (J_1 N_0)^{\dagger} = I_{\mu_1}$. Finally the dynamics of the error system can be written as follows

$$\begin{bmatrix} \dot{\widetilde{\sigma}}_0 \\ \dot{\tilde{\sigma}}_1 \end{bmatrix} = \begin{bmatrix} -\Lambda_0 & O_{\mu_0 \times \mu_1} \\ -J_1 J_0^{\dagger} \Lambda_0 & -\Lambda_1 \end{bmatrix} \begin{bmatrix} \widetilde{\sigma}_0 \\ \widetilde{\sigma}_1 \end{bmatrix}, \tag{4.13}$$

that is characterized by a Hurwitz matrix, hence the exponential stability of the system is guaranteed. Moreover, we can notice a term representing the coupling effect of the main task on the secondary task. In case of orthogonal tasks this term is zero, *i.e.* $\dot{\widetilde{\sigma}}_1 = -\Lambda_1 \tilde{\sigma}_1$, and the behavior of the main and that of the secondary tasks are decoupled.

By generalizing (4.8) to the case of η prioritized subtasks, the following general velocity command can be formulated:

$$\upsilon^* = J_0^{\dagger} \Lambda_0 \widetilde{\sigma}_0 + \sum_{i=1}^{\eta} (J_i N_{0|\dots|i-1})^{\dagger} \Lambda_i \widetilde{\sigma}_i - \overline{J}_{0|\dots|\eta} \varpi, \tag{4.14}$$

with the recursively-defined compensating matrix

$$\begin{aligned} \overline{J}_{0|\dots|\eta} = {} & (J_\eta N_{0|\dots|\eta-1})^{\dagger} \overline{J}_\eta \\ & + (I_n - (J_\eta N_{0|\dots|\eta-1})^{\dagger} J_\eta) \overline{J}_{0|\dots|\eta-1}, \end{aligned} \tag{4.15}$$

where $N_{0|\dots|i}$ is the projector onto the null space of the *augmented Jacobian* $J_{0|\dots|i}$ of the ith subtask, with $i = 0, \dots, \eta - 1$, which are respectively defined as follows

$$J_{0|\dots|i} = \begin{bmatrix} J_0^{\top} & \cdots & J_i^{\top} \end{bmatrix}^{\top} \tag{4.16}$$

$$N_{0|\dots|i} = (I_n - J_{0|\dots|i}^{\dagger} J_{0|\dots|i}). \tag{4.17}$$

The previous stability analysis can be straightforwardly extended to the general case of η subtasks.

The hierarchical formulation (4.14) guarantees that the execution of all the higher-priority tasks from 0 (main task) to $i-1$ will not be affected by the ith subtask and by the variation of the uncontrolled state variables. In other words, the execution of the ith task is subordinated to the execution of the higher priority tasks present in the task stack, *i.e.* it will be fulfilled only if suitable and enough DoFs are available, while the complete fulfillment of the main task, instead, is always guaranteed. However, with this new formulation for all reciprocally annihilating or independent tasks, a fully decoupling of the error dynamics is guaranteed.

The task composition and priority can be modified at runtime as needed, *i.e.* by activating or deactivating subtasks as well as by changing the priority order of the current active tasks already present in the task stack. However, in order to avoid discontinuity of the control input, a smooth transition between different task stacks has to be considered. This goal can be achieved by adopting a time-vanishing *smoothing term* when a new task

stack is activated. Without loss of generality, we suppose that the transition phase starts at $t = 0$, *i.e.* the rth task stack has to be deactivated and substituted by the new one $(r+1)$th. During the transition the velocity command is computed as follows:

$$\upsilon^*(t) = \upsilon^*_{r+1}(t) + e^{-\frac{t}{\tau}}(\upsilon^*_r(0) - \upsilon^*_{r+1}(0)), \tag{4.18}$$

where τ is a time constant determining the transition phase duration, and υ^*_k is the velocity command corresponding to the kth task stack. When t becomes sufficiently greater than τ, the rth task stack is fully removed and a new transition can start. Notice that the smoothing term $e^{-\frac{t}{\tau}}(\upsilon^*_r(0) - \upsilon^*_{r+1}(0))$ is bounded and exponentially vanishing, hence it will not affect the stability of the proposed control law. The time constant τ has to be smaller than the inverse of the maximum eigenvalue of the gain matrices Λ_i to ensure a short transient time response in comparison with the nullifying time of the task errors $\widetilde{\sigma}_i$.

4.1.5 Hybrid visual servoing and system behavior control for aerial manipulation tasks

In this section several elementary tasks useful for the composition of an aerial manipulation (grasping/plugging) task by exploiting visual measurements will be proposed. Besides tasks allowing the control of the gripper pose with respect to the observed objects, tasks able to guarantee the camera FoV constraint, to minimize the effect of the motion of the arm on the vehicle positional stability, and that addresses the issue of the joint mechanical limits are proposed. These tasks are not all orthogonal and/or independent but they are essential for the specific mission purposes. The priority of the proposed tasks can be arranged on the basis of the desired behaviour, even if some general constraints have to be considered. For example, the FoV constraint is essential because the loss of the observed object from the camera image will determine the failure of the whole mission (depending on the available camera optics, this problem could be less significant). On the other hand, the constraint on the joint limits is very invasive because it affects all the arm joints. For this reason it is advisable to move this task initially to the lower priority and increase it only when some joint limits are being approached.

4.1.6 Gripper position and orientation

Let o^*_e be the desired position of the gripper, *e.g.* suitable to perform the object grasping in the considered case study. This value can be computed by combining the pose measurement of the target object provided by the visual system, with the desired relative displacement of the gripper with respect to the target object in the grasping configuration. The corresponding position error is defined as $e_p = o^*_e - o_e$, and the task function is chosen equal to its square norm, yielding

$$\sigma_p = e_p^\top e_p, \tag{4.19}$$

where the desired task variable is $\sigma_p^* = 0$ (*i.e.* $\widetilde{\sigma}_p = -\sigma_p$). The corresponding task Jacobian matrix is

$$J_p = 2e_p^\top \begin{bmatrix} I_3 & S(R_b o_e^b)\iota_z & J_{q,P} \end{bmatrix}, \tag{4.20}$$

where $S(\cdot)$ is the skew-symmetric matrix representing the vectorial product, $\iota_z = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^\top$, and

$$J_q = \begin{bmatrix} J_{q,P} \\ J_{q,O} \end{bmatrix} = \begin{bmatrix} R_b & O_3 \\ O_3 & R_b \end{bmatrix} J_q^b, \tag{4.21}$$

with $J_q^b(q)$ the arm Jacobian matrix with respect to $\mathscr{B}$, with $J_{q,P}$ and $J_{q,O}$ $(3 \times \nu)$-matrices. The corresponding task Jacobian matrix of the uncontrolled state variables is

$$\overline{J}_p = 2e_p^\top S(R_b o_e^b) \begin{bmatrix} \iota_x & \iota_y \end{bmatrix}, \tag{4.22}$$

where $\iota_x = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^\top$, $\iota_y = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^\top$.

Notice that if o_e is also measured by using visual markers attached to the gripper, the camera calibration error and the arm direct kinematics error will not affect the grasping accuracy in a similar way as in an image-based approach. Moreover, with the proposed choice of σ_p, only one DoF is required to execute this subtask, because only the norm of e_p will be nullified, *i.e.* the motion of the gripper during the transient is constrained on a sphere of radius $\|e_p\|$. However, the corresponding task Jacobian J_p becomes singular when $e_p \to 0$. Nevertheless, in the task composition the generalized-inverse $J_p^\dagger$ is multiplied by σ_p. Hence, if J_p is full-rank, its determinant goes to zero only linearly when $e_p \to 0$, but σ_p goes to zero squarely.

Let $\{\eta_e, epsilon_e\}$ and $\{\eta_e^*, \varepsilon_e^*\}$ be the unit quaternions corresponding to R_e and to its desired value R_e^*, respectively. The corresponding orientation error can be expressed as

$$e_o = \eta_e \varepsilon_e^* - \eta_e^* \varepsilon_e - S(\varepsilon_e^*)\varepsilon_e. \tag{4.23}$$

The task function is chosen equal to

$$\sigma_o = e_o^\top e_o, \tag{4.24}$$

with the desired task variable $\sigma_o^* = 0$ (*i.e.* $\widetilde{\sigma}_o = -\sigma_o$), while the corresponding task Jacobian matrix is

$$J_o = 2e_o^\top \begin{bmatrix} O_3 & \iota_z & J_{q,O} \end{bmatrix}. \tag{4.25}$$

The Jacobian matrix of the uncontrolled state variables is

$$\overline{J}_o = 2e_o^\top \begin{bmatrix} \iota_x & \iota_y \end{bmatrix}. \tag{4.26}$$

Remarks similar to the position case concerning the number of required DoFs, the singularity of the task Jacobian matrix, and the direct visual measurement of the gripper orientation can be repeated straightforwardly.

4.1.7 Camera field of view

Let $s_c \in \mathbb{R}^2$ be the image feature vector of the projection of the observed markers centroid $o_c^b = \begin{bmatrix} x_c^b & y_c^b & z_c^b \end{bmatrix}^\top$ onto the normalized image plane, *i.e.*

$$s_c = \begin{bmatrix} X_c \\ Y_c \end{bmatrix} = \frac{1}{z_c^b} \begin{bmatrix} x_c^b \\ y_c^b \end{bmatrix}. \tag{4.27}$$

The FoV subtask consists in constraining s_c within a maximum distance with respect to a desired position s_c^* in the normalized image plane (*e.g.* the center of the image) by moving the vehicle base, *i.e.* the camera point of view (notice that we assumed the camera mounted on the vehicle base). Without loss of generality, any point of the observed target can be chosen to be controlled in the image. To achieve this goal, the following task function is considered:

$$\sigma_c = e_c^\top e_c, \tag{4.28}$$

where $e_c = s_c^* - s_c$, and the desired task variable is $\sigma_c^* = 0$ (*i.e.* $\widetilde{\sigma}_c = -\sigma_c$), while the corresponding task Jacobian is

$$J_c = \begin{bmatrix} 2e_c^\top L_p R_b^\top & 2e_c^\top L_o R_b^\top \iota_z & O_{1\times v} \end{bmatrix}, \tag{4.29}$$

where

$$L_p = \frac{1}{z_c^b} \begin{bmatrix} -1 & 0 & X_c \\ 0 & -1 & Y_c \end{bmatrix}, \tag{4.30}$$

$$L_o = \begin{bmatrix} X_c Y_c & -(1+X_c^2) & Y_c \\ 1+Y_c^2 & -X_c Y_c & -X_c \end{bmatrix}. \tag{4.31}$$

Finally, the Jacobian of the uncontrolled state variables is

$$\bar{J}_c = 2e_c^\top L_o R_b^\top \begin{bmatrix} \iota_x & \iota_y \end{bmatrix}. \tag{4.32}$$

Notice that only one DoF is required to accomplish this subtask. In fact, the distance of o_c^b with respect to the desired optical ray corresponding to s_c^* is controlled. However, the corresponding task Jacobian matrix J_c is singular when $e_c \to 0$, but since it is not strictly required to accomplish the main mission that the target object is exactly in the desired position of the image (*e.g.* the center), this subtask can be activated only when σ_c exceeds a *safety threshold*.

4.1.8 Center of gravity

The weight of the robot arm can generate an undesired torque on the vehicle base depending on the configuration. In particular, the arm motion statically perturbs the system attitude and position when the center of gravity (CoG) of the arm p_g is not aligned with the CoG of the vehicle base along the gravitational line of action ι_z. Without loss of generality, the CoG of the vehicle base is assumed to be in O_b. Hence, by denoting with

$e_g = \left((p_g - o_b)^\top \imath_z\right) \imath_z - (p_g - o_b)$ the error between the desired and the current position of the arm's CoG, the designed task function is defined as follows:

$$\sigma_g = {e_g}^\top e_g, \tag{4.33}$$

with the desired task variable $\sigma_g^* = 0$ (*i.e.* $\widetilde{\sigma}_g = -\sigma_g$). The corresponding task Jacobian matrix can be computed from the corresponding Jacobian represented. In fact, p_g^b is only a function of the arm joint configuration defined as

$$p_g^b = \frac{1}{m} \sum_{i=1}^{v} m_i p_{gi}^b, \tag{4.34}$$

where m_i and p_{gi}^b are the mass and the position of the CoG of the ith arm link, respectively, and $m = \sum_{i=1}^{v} m_i$.

The CoG of a partial chain of links can be represented, with respect to $\mathscr{B}$, from the link j to the end-effector, yielding

$$r_{gj}^b = \frac{1}{m} R_j^b \sum_{i=j}^{v} m_i p_{gi}^b, \tag{4.35}$$

where R_j^b is the rotation matrix between the jth arm link and $\mathscr{B}$. Finally, the differential relationship between p_g and the arm joint configuration is

$$\dot{p}_g^b = J_g^b \dot{q}, \tag{4.36}$$

where $J_g^b \in \mathbb{R}^{3\times v}$ is the CoG Jacobian expressed in $\mathscr{B}$ and defined as follows

$$J_g^b = \begin{bmatrix} j_{g1}^b & \cdots & j_{gv}^b \end{bmatrix}, \tag{4.37}$$

with j_{gi}^b the ith joint Jacobian formulated from the partial CoG as follows

$$j_{gj}^b = \frac{\sum_{i=j}^{v} m_i}{m} S(R_j^b \imath_z) r_{gj}^b. \tag{4.38}$$

Finally, the corresponding task Jacobian is defined as

$$J_g = 2 e_g^\top \begin{bmatrix} O_{1\times 3} & S(R_b p_g^b)\imath_z & R_b J_g^b \end{bmatrix}, \tag{4.39}$$

Notice that only one DoF is required and similar considerations as in the previous tasks definitions on the singularity of the Jacobian matrix when $e_g \to 0$ can be done.

4.1.9 Joint-limits avoidance constraint

A possible solution to avoid mechanical joint limits is to make attractive the central position of the joint ranges. Let $e_q = q^* - q$ be the corresponding error, where $q^* = q_L + \frac{1}{2}(q_H - q_L)$, with $q_L = \begin{bmatrix} q_{1L} & \cdots & q_{vL} \end{bmatrix}^\top$ and $q_H = \begin{bmatrix} q_{1H} & \cdots & q_{vH} \end{bmatrix}^\top$ the vectors

of the low and high joint limits, respectively. The corresponding square distance can be used as a task function to push the system to reach a safe configuration as follows

$$\sigma_l = {e_q}^\top \Lambda_l e_q, \tag{4.40}$$

where Λ_l is the following weighting matrix needed to normalize the control action with respect to the joint range

$$\Lambda_L = \text{diag}\{(q_{1H} - q_{1L})^{-2} \quad \dots \quad (q_{vH} - q_{vL})^{-2}\}. \tag{4.41}$$

The desired task variable is $\sigma_l^* = 0$ (*i.e.* $\widetilde{\sigma}_l = -\sigma_l$), and the corresponding task Jacobian matrix is

$$J_l = \begin{bmatrix} O_{1\times 4} & -2\Lambda_L {e_q}^\top \end{bmatrix}. \tag{4.42}$$

Notice that the uncontrolled state variables do not affect this subtask, *i.e.* the corresponding Jacobian is the null matrix.

Due to the higher priority tasks, some joint could reach anyway its limit. However, when a joint is approaching a mechanical limit, the corresponding component of the task function can be extracted from the previous subtask to form a new isolated subtask that can be activated on the top of the task stack. With this policy, if mechanically viable, the system will reconfigure its internal DoFs to achieve all the remaining subtasks until the dangerous condition will disappear and the original priority will be restored.

4.1.10 Simulation results

The proposed approach has been tested in the simulator developed in the ARCAS project (www.arcas-project.eu), which is based on GAZEBO physics engine (http://gazebosim.org) (see Fig. 4.2). A quadrotor with a weight of 5 kg and endowed with a downward looking camera at 25 Hz and a 6-DoFs robot arm plus a gripper has been employed. The camera has been positioned 50 cm ahead the vehicle base with an inclination of 30 deg with respect to the vertical axis in a way to observe the grasping manoeuvre without self-occlusion. The target object is a bar endowed with two visual markers at the ends. A UAM velocity control has also been employed [67, 72].

The assigned task is composed of two phases:

- *approaching phase* — the UAM starts from a distance of about 125 cm and has to move the gripper to a pre-grasping pose at 10 cm over the grasping pose;
- *grasping phase* — once the intermediate pose has been reached with an error less than a suitable threshold (2 cm for the position and 2 deg for the orientation), the target pose is moved towards the final grasping pose in 10 s; the closing of the gripper is then commanded when the final pose has been reached with a good accuracy (1 cm for the position and 1 deg for the orientation).

Four task-stack configurations have been simulated:

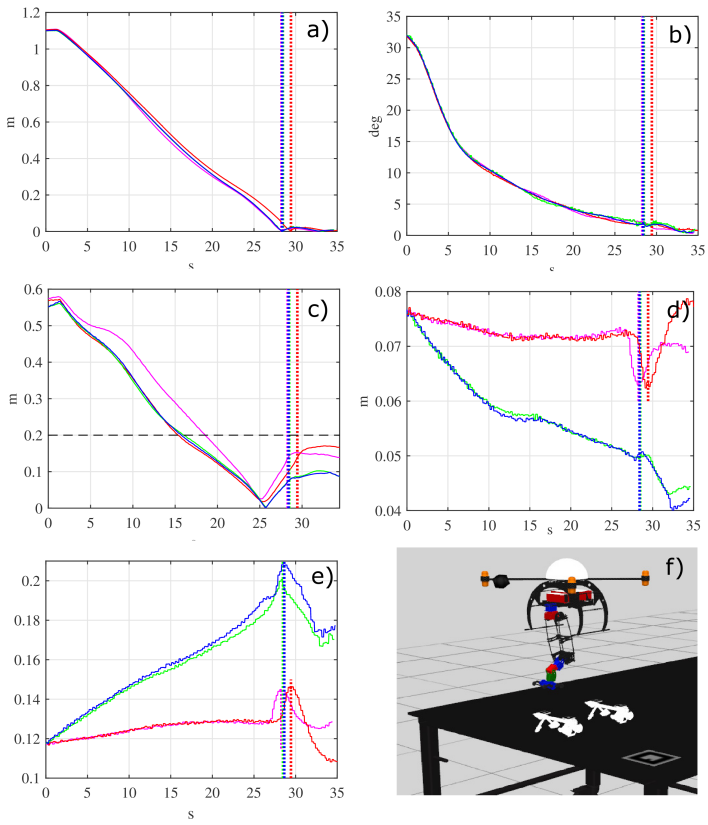

Figure 4.2: Results of the simulated grasping task (case 1) in magenta, case 2) in red, case 3) in green, case 4) in blue): a) norm of the position error $\|e_p\|$; b) norm of the orientation error; c) norm of the FoV error $\|e_s\|$ with a 20 ± 2 cm activation/deactivation threshold; d) norm of the CoG error $\|e_g\|$; e) minimum distance from the joint limits normalized to the joint range; f) the GAZEBO simulator. The vertical lines indicate the conclusion of the approaching phase, while the end of each trajectory indicates the grasping time.

1. only main tasks for position and orientation control (Section 4.1.6) active;
2. as case 1) plus FoV subtask (Section 4.1.7);
3. as case 2) plus CoG subtask (Section 4.1.8);
4. as case 3) plus joint limits subtask (Section 4.1.9).

The achieved results are shown in Fig. 4.2 with different colors for the four considered control behaviors. A dashed vertical line is employed to highlight the end of the approaching phase and the starting of the grasping phase. Notice that the approaching and

the grasping phases in all the considered case studies have different durations depending obviously on the selected control behavior, *i.e.* the active tasks stack.

Figure 4.2*a* shows the time history of the position error in norm during the task execution for each case study. For all cases, a smooth nullification of the pose error is observed. In particular, during the approaching phase the position error decreases almost linearly due to the saturation of the maximum vehicle cruise velocity (10 cm/s in these case studies). Figure 4.2*b* shows the time history of the norm of the orientation error. The initial orientation is only a few degrees far from the final grasping pose, hence the error goes under the threshold in few seconds in all the case studies. Notice how the behavior of both the position and orientation errors are similar in all the cases coherently with the hierarchical task combination adopted in the proposed formulation, *i.e.* the activation of subtasks cannot affect the behavior of the higher priority tasks.

Figure 4.2*c* shows the results achieved with the activation of the camera FoV subtask. In detail, this subtask is dynamically activated and deactivated by comparing the error norm with a double threshold, *i.e.* with a suitable hysteresis (20 ± 2 cm) to avoid chattering phenomena. By taking into account the camera pose with respect to $\mathscr{B}$, the desired position of the image features centroid has been chosen equal to $s_c^* = \left[0, -0.1\right]^\top$. The achieved results show how except for case 1), *i.e.* when this subtask is activated, the FoV error is improved without affecting the movement of the gripper.

Figure 4.2*d* shows the time histories of the error in norm of the CoG subtask. For the chosen initial arm configuration the distance of the CoG with respect to the vehicle gravitational axis is 7.6 cm. In cases 1) and 2) this distance remains almost constant, while when the CoG subtask is active, *i.e.* for cases 3) and 4), the behavior is always improved without affecting the tasks with a higer priority in the stack.

Finally, in the last case study 4) also the joint-limits avoidance constraint is activated. Differently with respect to the previous cases, as it is shown in Fig. 4.2*e* (zero indicates the reaching of a joint limit, while 0.5 indicates that all joints are in the middle of the joint range), the task is not completely fulfilled, even if a clear increase of the distance with respect to the closest joint limit is guaranteed. This behavior is mainly due to the conflict with other subtasks that have a higher priority in the tasks stack. As described before, it is possible to increase the priority of this task in the stack when a joint limit is excessively close in a way to guarantee mechanical safety at the expense of other tasks.

4.1.11 Experimental results

The UAM employed for the experimental tests has been developed in the ARCAS project. It is a multi-rotor aircraft with eight rotors in coaxial configuration with a 105 cm tip-to-tip wingspan, height of 50 cm, 13-inches propellers, and a total mass of 8.2 kg including batteries and the 6-DoFs robotic arm (see Fig. 4.3). The employed autopilot has been developed by CATEC (www.catec.aero) and allows also the control of the robot arm. A model-based design methodology [92] established on MATLAB/SIMULINK code generation tools has been adopted. The UAM has been endowed with an i7 ASCTEC MAS-

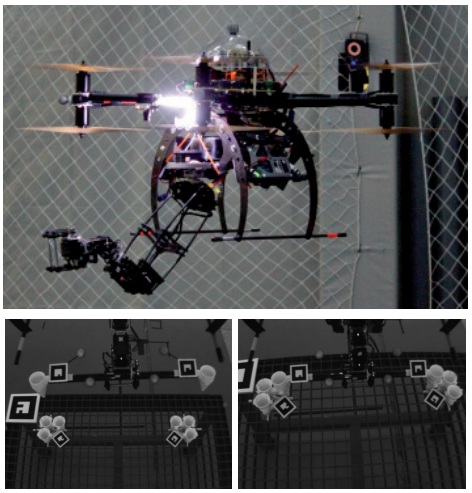

Figure 4.3: Bonebraker UAM employed during the experiments (top), and images from the onboard camera during the approaching phase (bottom-left) and at the plugging instant (bottom-right).

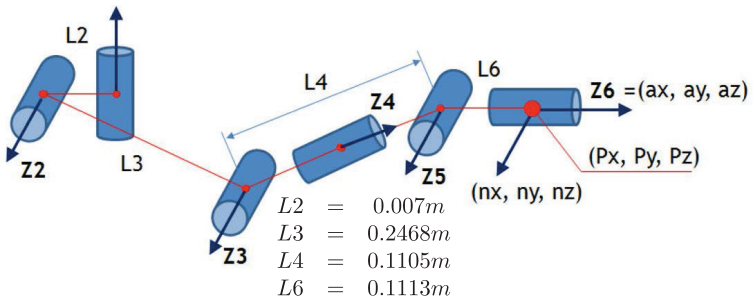

Figure 4.4: Reference frames following Denavit-Hartenberg convention.

TERMIND on-board for costly computing code, such as image processing. A motion capture system running at 100 Hz has been used as the positioning system, while the attitude is measured with the onboard IMU. A 6-DoFs manipulator [17] running at 50 Hz is attached below the vehicle base. The robotic manipulator direct kinematic model is obtained by using the well known Denavit-Hartenberg convention (see Fig. 4.4 and Table 4.1).

A high-definition camera running at 14 Hz has been positioned as in the simulation case study. The calibration of the vision system has been divided in two steps. First, the camera intrinsic parameters are obtained with several views of a calibration pattern (*i.e.* a chessboard). Secondly, the extrinsic parameters are obtained using the motion capture system to precisely localize the platform body frame ($\mathscr{B}$) and an object in the

Table 4.1: Arm Denavit-Hartenberg parameters.

Link i	θ_i	d_i	a_{i-1}	α_{i-1}
1	θ_1	0	0	0
2	θ_2	0	-$L2$	$\pi/2$
3	$\theta_3+\pi/2$	0	$L3$	0
4	θ_4	$L4$	0	$\pi/2$
5	θ_5	0	0	$-\pi/2$
6	θ_6	$L6$	0	$\pi/2$

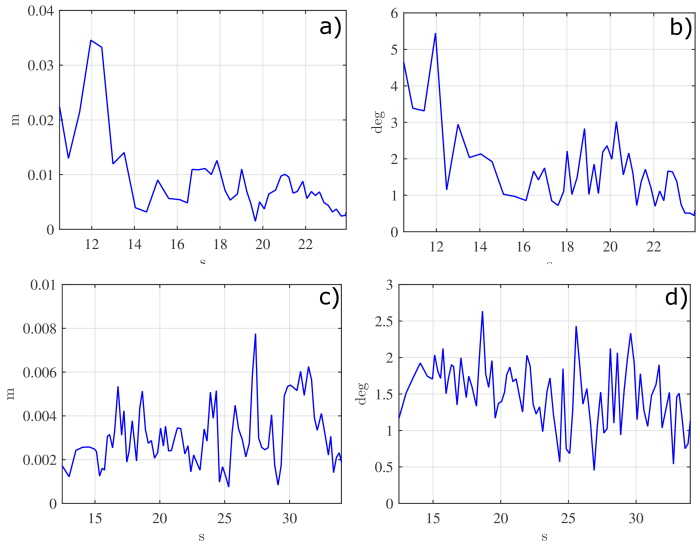

Figure 4.5: Norm of the object position errors with respect to the ground-truth during grasping (a) and plugging (c) maneuvers. The corresponding orientation errors are shown in (b) and (d).

scene (which corresponds to a marker). By knowing the pose of the camera attached to the quadrotor body frame, we can trivially obtain the frame transformation between the camera and the object. However, the estimation of the error between the camera and the optical frames is also required. The marker detector is employed estimating the marker pose with respect to the optical frame. Then, a pose average of the difference between the camera and the optical frames is computed with respect to the object.

Figure 4.5 shows the error between the detected bar and the ground truth poses during grasping and plugging tasks.

The experimental task consists in plugging a bar endowed with two clipping systems at the ends into a fixed base[1], as shown in the bottom part of Fig. 4.3. As for the simulated

[1] Several grasping and plugging experiments of different type of bars are shown in the video attached to this section.

case studies, the mission has been decomposed into two steps: the approaching phase, to move the bar over the plugging base at a distance of 5 cm, and the final plugging phase. During this latter phase the FoV task is turned off because the constraint is always satisfied by the system mechanical configuration and the adopted optics. The task requires high accuracy both in position and orientation, *i.e.* about 1 cm for the position and 1 deg for the orientation, that has to be guaranteed stable in time to avoid undesired collisions. To cope with this requirement, the bar and the plugging base have been endowed with visual markers. Hence, the positioning error has been computed by using the measurements of the bar and of the base mitigating the effects of the calibration errors.

The achieved results are shown in Fig. 4.6. Figures 4.6*a* and 4.6*b* show the time history of the norm of the position and orientation errors, respectively. The vertical dashed line indicates the end of the approaching phase and the beginning of the plugging phase. The plugging instant corresponds with the end time of the plots. One can observe how the initial errors are quite high because the system starts from a distance of about 40 cm from the goal position, and with a significant orientation error too, however for both errors the target accuracy has been reached in a fast and stable way.

The time history of the norm of the FoV error $\|e_s\|$ is shown in Fig. 4.6*c*, from which one can observe how this subtask is suitably executed, hence the system is able to prevent the loss of the visual markers from the camera FoV.

The CoG subtask has been employed with an activation/deactivation threshold of $15 \pm$ 2cm. However, it is never activated because the high-priority FoV subtask determines arm configurations already compatible with the CoG subtask. In fact, the alignment error of the CoGs is lower than 4 cm.

Finally, Figure 4.6*d* shows the minimum distance computed over all the arm joints from the corresponding joint limits normalized to the joint range (zero indicates the reaching of a joint limit, while 0.5 indicates that all joints are in the middle of the joint range). Even if this is the lower priority task, a safety distance of more than 20% of the joint ranges, in the worst case, is always preserved.

Introduzione low level

4.2 Passivity-based Control of VToL UAVs with a Momentum-based Estimator of External Wrench and Unmodeled Dynamics

4.2.1 Introduction

Service robotics applications are day by day making more use of VToL UAVs to pursue different actions. From passive tasks like inspection [141, 60], surveillance and monitoring [80], remote sensing and so on, such aerial vehicles are now beginning to be employed in active tasks like grasping [86] and manipulation [7, 31, 41, 42, 51, 47, 65, 70, 83, 85, 98, 103]. This change of perspective requires the UAV to operate in changing and un-

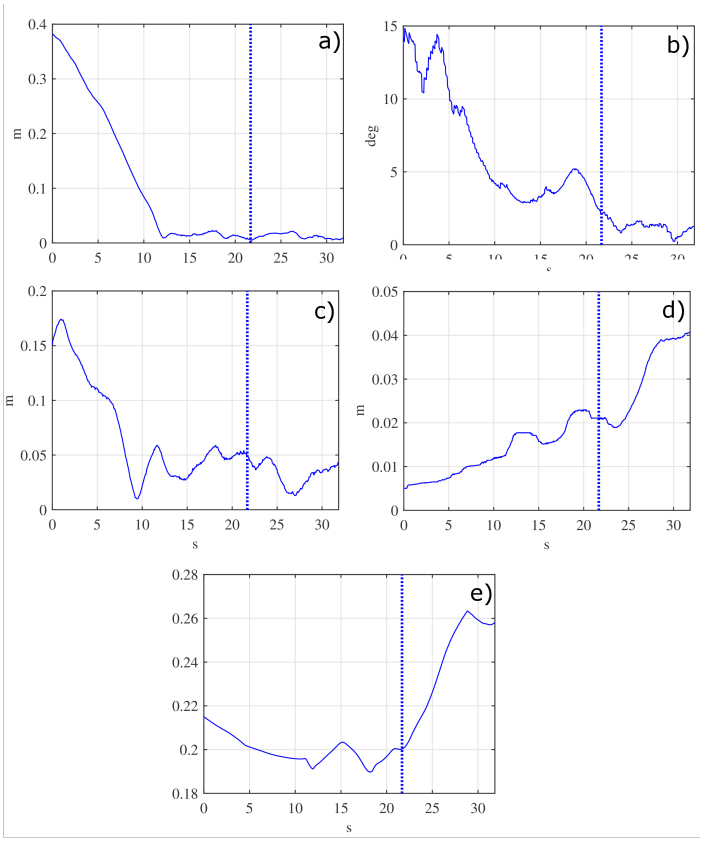

Figure 4.6: Experimental results of the plugging task: a) norm of the position error $\|e_p\|$; b) norm of the orientation error; c) norm of the FoV error $\|e_s\|$; d) norm of the CoG error $\|e_g\|$ with a 15 ± 2 cm activation/deactivation threshold; e) minimum distance from the joint limits normalized to the joint range. The vertical dotted lines indicate the time instant when the approaching phase is concluded, while the each trajectory ends at the plugging time.

structured scenarios. To this purpose, the controller has to deal with unknown parameters (i.e., the battery charge level), the transportation of unknown payloads, aerodynamic effects that are usually neglected during the control design phase, and the interaction with the environment.

In this section, a passivity-based control of VToL UAVs is presented. The classical hierarchical architecture separating the (fast) rotational and the (slow) translational dynamics [79] is employed. The controller ensures a closed-loop mechanical impedance behaviour for the translational part of the VToL UAV, while the rotational part does not rely upon exact cancellation of nonlinearities, conferring in this way robustness to the attitude part. The collision identification technique based on the momentum of the system proposed in [22] has been suitably modified in this context to play as an external wrench and unmodeled dynamics estimator. The estimation is taken into account by the controller to compensate forces and moments arising from wind, aerodynamics effects not taken into account in the model, external wrench caused by a robotic arm attached to the vehicle's base during aerial manipulation tasks, unknown carried payloads, physical interactions, and so on. The residual errors between the estimated external wrench, the unmodeled dynamics and the real ones are seen as perturbations in the closed-loop system. As long as the closed-loop system bandwidth –tunable through the control gains on the basis of the available robotic platform and the controller sample time– is able to cope with such time-varying residual errors, the overall performance benefits from the proposed architecture as theoretically and experimentally evaluated.

As far as authors know, the novelty of this section is the combination of a passivity-based control for VToL UAVs together with an external wrench and unmodeled dynamics estimator, a rigorous stability proof under certain assumptions, and the consequent experimental validation. As a result, the aerial platform is able to perform tasks without a precise knowledge about the dynamic parameters and the external disturbances: this is absolutely useful in the forthcoming aerial service robotics applications, e.g. aerial manipulation, where interaction with the environment is required. Moreover, with respect to the current state of the art in which adaptive and integral actions are employed to cope with the aforementioned problems, less parameters have to be tuned in the proposed architecture, where instead the gains assume precise physical meanings.

The outline of the section is as follows. Next section presents the related work. The dynamic model of a quadrotor is presented in Section 4.2.3. The momentum-based external wrench and unmodeled dynamics estimator is revised in Section 4.2.5. The control is introduced in Section 4.2.6. The stability proof of the proposed controller combined with the compensation of the estimated terms is addressed in Section 4.2.7. Performed experiments are described in Section 4.2.11. Conclusion and future work are finally provided.

4.2.2 Related work

Regarding aerial manipulation, two approaches can be in principle thought to control an aerial manipulator (UAV with an attached robotic arm endowed with a gripper). The for-

mer approach considers the UAV and the robotic arm as a unique entity, and thus the controller is designed on the basis of such complete dynamic model [51, 47, 40]. The latter approach considers instead the UAV and the robotic arm as two separate and independent systems: the effects of the arm on the aerial vehicle are then considered as external disturbances and viceversa. This might be useful in case the dynamics of the arm is not enough to compensate the UAV position error and/or in case the arm does not allow torque control (i.e., servomotors) [91]. The here presented section is oriented towards the latter approach: it has been thus considered the control of the single UAV subject to external disturbances and time-varying parameters. Therefore, many different approaches address problems related to the stabilization and tracking of desired trajectories with a VToL UAV. The most widely used controller takes into account a hierarchical architecture [79, 57] highlighting a time-scale separation between the translational (slow time-scale) and angular (fast time-scale) dynamics of the aerial vehicle. Other approaches rely upon backstpping [56], impedance [33] and optical flow [49] techniques. However, in general, a precise knowledge of system dynamics is required to perform a feedback linearization of both fast and slow time-scale parts of the system. Hence, several of the above mentioned controllers implement an integral action to resist against external disturbances and cope with unknown and time varying parameters. Recently, adaptive controls have been employed to counteract such disturbances [85, 88, 5, 6, 23, 13]. A nonlinear force observer has been introduced in [104] to estimate disturbances applied to a quadrotor. A sliding mode observer has been instead employed in [55] to impose more robustness on the closed-loop system. Since passivity-based controllers do not rely on the exact compensation of the considered model, they are expected to be more robust with respect to parameters uncertainties. Port-Hamiltonian methods have been developed in [104, 58, 105], a passive backstepping in [34], and passivity-based attitude controls in [24, 30], in particular without angular velocity measurement in [53, 101].

In this section, the passivity-based control proposed in [96, 8] is adapted to be suitable for a VToL UAV system as described in Section 4.2.6. Moreover, a compensation of external wrench and unmodeled dynamics is here introduced to further reduce aerodynamic effects and external disturbances. A similar architecture has been introduced by the authors in [90] where an impedance controller is instead employed without providing a rigorous stability proof. Under certain assumptions, this issue is overcome by the current section.

4.2.3 Modeling

The most popular configurations of VToL UAVs employed in the above defined scenarios are the *quadrotor* and the *hexarotor*, which are platforms equipped with four or six propellers, respectively, aligned in the same direction. Hence, these aerial vehicles are underactuated mechanical systems having six degrees of freedom but only four independent control inputs. Without loss of generality, in the remainder of this section, the chosen VToL UAV is a quadrotor.

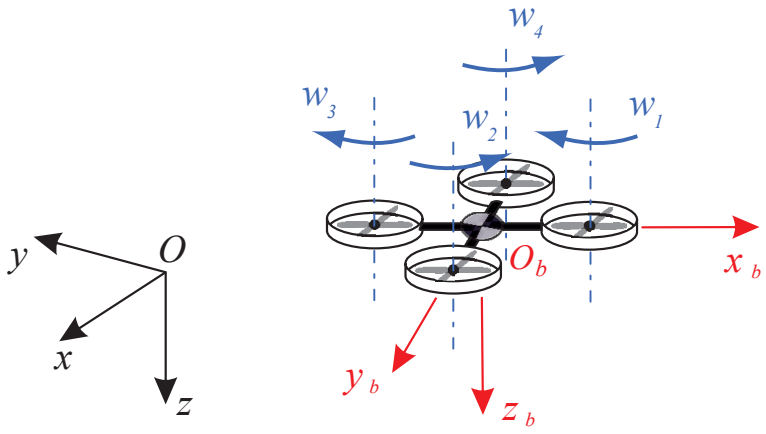

Figure 4.7: Quadrotor and related frames. In black, the inertial frame Σ_i. In red, the body frame Σ_b. In blue, the propellers speed and the label of each motor.

Define a world-fixed inertia reference frame Σ_i and a body-fixed reference frame Σ_b placed at the UAV's center of mass (see Fig. 4.7). The absolute position of the UAV with respect to Σ_i is denoted by $p_b = \begin{bmatrix} x & y & z \end{bmatrix}^{\mathrm{T}}$. Using the roll-pitch-yaw Euler angles, $\eta_b = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^{\mathrm{T}}$, the attitude of the UAV is defined by the rotation matrix $R_b(\eta_b) \in \mathrm{SO}(3)$, expressing the rotation of Σ_b with respect to Σ_i, given by [95]

$$R_b(\eta_b) = \begin{bmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix},$$

where $s_\times$ and $c_\times$ are abbreviations for sine and cosine, respectively.

Let $\dot{p}_b$ and ω_b denote the absolute translational and angular velocities of the UAV, respectively, while $\dot{p}_b^b$ and ω_b^b describe the absolute translational and angular velocities of the aerial vehicle expressed in Σ_b, respectively. Denoting with $\dot{\eta}_b$ the time derivative of η_b, the following equations hold [95]

$$\dot{p}_b = R_b(\eta_b)\dot{p}_b^b, \tag{4.43a}$$
$$\omega_b = T_b(\eta_b)\dot{\eta}_b, \tag{4.43b}$$
$$\omega_b^b = R_b(\eta_b)^{\mathrm{T}}\omega_b = Q(\eta_b)\dot{\eta}_b, \tag{4.43c}$$

where $T_b(\eta_b)$ is the (3×3) transformation matrix between the time derivative of η_b and the correspondent ω_b, while $Q(\eta_b) = R_b(\eta_b)^{\mathrm{T}} T_b(\eta_b)$ maps the time derivative of η_b into the UAV angular velocity expressed with respect to Σ_b. The detailed expression of $Q(\eta_b)$ is

$$Q(\eta_b) = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix},$$

with a singularity at $\theta = \pm\pi/2$.

The dynamic equations related to the UAV can be retrieved by using the Newton-Euler formulation [36]

$$m\ddot{p}_b^b = -mS(\omega_b^b)\dot{p}_b^b + mR_b(\eta_b)^{\mathrm{T}} g + f_b^b + f_u^b(\cdot), \tag{4.44a}$$

$$\dot{R}(\eta_b) = R(\eta_b)S(\omega_b) \tag{4.44b}$$

$$I_b\dot{\omega}_b^b = -S(\omega_b^b)I_b\omega_b^b + \tau_b^b + \tau_u^b(\cdot), \tag{4.44c}$$

where $\ddot{p}_b^b$ is the absolute translational acceleration of the UAV expressed with respect to Σ_b; m is the mass of the vehicle; I_b is the (3×3) constant inertia matrix of the UAV expressed with respect to Σ_b; $\dot{\omega}_b^b$ is the absolute angular acceleration of the UAV expressed with respect to Σ_b; $S(\cdot)$ denotes the skew-symmetric matrix; $g = \begin{bmatrix}0 & 0 & g\end{bmatrix}^{\mathrm{T}}$ is the (3×1) gravity vector with $g = 9.81\mathrm{m/s}^2$; f_b^b and τ_b^b are the (3×1) forces and torques input vectors, respectively, expressed in Σ_b; $f_u^b(\cdot)$ and $\tau_u^b(\cdot)$ are two (3×1) vectors denoting unknown forces and moments, respectively, acting on the vehicle –aerodynamic and buoyancy effects, flapping dynamics [82], parametric uncertainties, imbalances caused by batteries and/or on-board sensors, motion of a robotic arm (or moving sensors, e.g. a laser scanner on a pan-tilt mechanism) mounted on the aerial platform, wind gusts, interaction with the environment, etc.– and whose dependencies on $(\dot{p}_b, \ddot{p}_b, \omega_b^b, \dot{\omega}_b^b, R(\eta_b), t)$, where t denotes the time variable, have been omitted for brevity.

The detailed expressions of both the input forces f_b^b and torques τ_b^b depend on the configuration of the considered aerial vehicle. Most of the VToL UAVs are underactuated systems with six degrees of freedom and four main control inputs. Hence, many UAVs can be characterized by three input control torques $\tau_b^b = \begin{bmatrix}\tau_\phi & \tau_\theta & \tau_\psi\end{bmatrix}^{\mathrm{T}}$ and one input control force $f_b^b = \begin{bmatrix}0 & 0 & u\end{bmatrix}^{\mathrm{T}}$, where u denotes the thrust perpendicular to the propellers rotation plane. In the quadrotor case of Fig. 4.7, the relationship between the thrust, the control torques, and the squared propellers speed w_i^2, with $i = 1,\ldots,4$, is [56]

$$u = \rho_u(w_1^2 + w_2^2 + w_3^2 + w_4^2), \tag{4.45a}$$

$$\tau_\phi = l\rho_u(w_2^2 - w_4^2), \tag{4.45b}$$

$$\tau_\theta = l\rho_u(w_3^2 - w_1^2), \tag{4.45c}$$

$$\tau_\psi = cw_1^2 - cw_2^2 + cw_3^2 - cw_4^2, \tag{4.45d}$$

where l is the distance between each propeller and the center of mass of the quadrotor, $\rho_u > 0$ and $c > 0$ are the thrust and drag factors, respectively. It is worth noticing that many aerodynamics effects are neglected through this representation. However, in case of hovering, or at least low-speed motions, the relationships in (4.45) can be considered as a valid approximation [79].

Folding (4.43) and the relative time derivatives into (4.44), and considering the expression of f_b^b yield the following dynamic model, useful for control design purposes, expressed with respect to Σ_i, and representing a wide range of VToL UAVs configura-

tions:

$$m\ddot{p}_b = mg - uR_b(\eta_b)i_3 + f_u(\cdot), \tag{4.46a}$$

$$M(\eta_b)\ddot{\eta}_b = -C(\eta_b,\dot{\eta}_b)\dot{\eta}_b + Q(\eta_b)^{\mathrm{T}}\tau_b^b + \tau_u(\cdot), \tag{4.46b}$$

where $i_3 = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^{\mathrm{T}}$; $M(\eta_b) = Q(\eta_b)^{\mathrm{T}} I_b Q(\eta_b)$ is the (3×3) symmetric and positive definite inertia matrix (provided that $\theta \neq \pm\pi/2$), and $C(\eta_b,\dot{\eta}_b) = Q^{\mathrm{T}} S(Q\dot{\eta}_b) I_b Q + Q^{\mathrm{T}} I_b \dot{Q}$ is the (3×3) Coriolis matrix, in which the dependencies have been dropped and $\dot{Q}(\eta_b)$ represents the time derivative of $Q(\eta_b)$.

Mentioning that only Euclidean norms are taken into account in the remainder of the section, the following assumptions are considered.

- **Assumption 1.** The aerial vehicle does not pass through the singularities. The allowable configuration space for the yaw-pitch-roll angles η_b is thus $\mathscr{Q}_\eta = \{\eta_b \in \mathbb{R}^3 | \theta \neq \pi/2 + k\pi,\ k = \ldots,-1,0,1,\ldots\}$.

- **Assumption 2.** Unknown forces $f_u(\cdot)$ and moments $\tau_u(\cdot)$ depend only on the time variable t and they are continuously differentiable and bounded with respect to it. Therefore, the following inequalities hold

$$\|f_u\| \leq B_1 < \infty, \tag{4.47a}$$

$$\|\dot{f}_u\| \leq B_2 < \infty, \tag{4.47b}$$

$$\|\ddot{f}_u\| \leq B_3 < \infty, \tag{4.47c}$$

$$\|\tau_u\| \leq B_4 < \infty, \tag{4.47d}$$

$$\|\dot{\tau}_u\| \leq B_5 < \infty, \tag{4.47e}$$

$$\|\ddot{\tau}_u\| \leq B_6 < \infty, \tag{4.47f}$$

  where B_i, with $i = 1,\ldots,6$, are positive constants.

It is also worth to recall the following property.

- **Property 1.** Considering the expression in (4.46b), the following property holds [95]

$$\dot{\eta}_b^{\mathrm{T}} \left(\dot{M}(\eta_b) - 2C(\eta_b,\dot{\eta}_b)\right)\dot{\eta}_b = 0, \tag{4.48}$$

  where $\dot{M}(\eta_b)$ represents the time derivative of $M(\eta_b)$. If the Coriolis matrix is represented through the Christoffel symbols, then for any arbitrary (3×1) vector v the following quadratic form holds

$$\dot{v}^{\mathrm{T}} \left(\dot{M}(\eta_b) - 2C(\eta_b,\dot{\eta}_b)\right)\dot{v} = 0.$$

4.2.4 Discussion about the employed assumptions

The impact of the employed assumptions, made to simplify the control design, is deeply analysed in the following.

Assumption 1 is restrictive only from a mathematical point of view. The singularity about the pitch angle is related to the employed angular representation and it is not a physical singularity; moreover, notice that a pitch angle of $\pm\pi/2$ does not happen because acrobatic manoeuvres (i.e., death loops) are not within the goals of this work, the initial conditions are chosen far from that singularity condition and the controller will be shown to be stable. In addition, since only two points in the configuration space are not allowed, this case might also be handled from a practical point of view during the implementation of the programming code. It goes without saying that a non minimal representation for the rotations might be in principle employed, i.e. unit quaternions [102, 100, 99]. The related control laws guarantee almost global asymptotic stability [2]. In any case, both Euler angles and quaternions representations suffer of the so-called *unwinding phenomenon* [64] if the control laws are not properly designed. In this section, through the use of Assumption 1, the problem is related to the yaw angle [3]. Nevertheless, the concept about the hybrid-dynamic path-lifting algorithm proposed in [64] can be easily implemented as a solution for both Euler angles and quaternions representations.

With reference to Assumption 2, notice that the motivations about neglecting the dependence of the unknown forces and moments from the aerial vehicle's angular attitude, angular velocity and translational accelerations are taken from [36]. The independence of f_u and τ_u from $\ddot{p}_b$ and $\dot{\omega}_b^b$ can be justified since, in general, the density of the body of the aerial platform is much more relevant than the one of the environment fluid. The independence from ω_b^b is better justified when the unknown generalized forces apply near the aerial vehicle's center of mass and the motion reaction forces resulting from the rotation of the aerial platform can be neglected with respect to the ones produced by eventual linear movements. The independence from $\dot{p}_b$ is the most restrictive one since it is supposed that the aerial vehicle moves very slowly and for almost all the task it is in hovering. Such an assumption is much more justified in aerial manipulation tasks. However, on the one hand, such condition simplifies the derivation of the control law and its stability proof; on the other hand, during experimental validation in Section 4.2.11, the hovering condition is overcome and the performance of the control law is evaluated despite the employed assumption. The independence from the vehicle's attitude $R_b(\eta_b)$ is valid when the aerodynamic forces do not depend on the aerial platform orientation. This happens essentially on the basis of the vehicle's shape. In case of VToL UAVs such assumption is thus very reasonable due to the the fact that lift forces are not so sensitive with respect to the attack angles. In conclusion, thanks to Assumption 2, the unknown forces and moments are only time depending and their boundedness is not so much restrictive, but instead properly physically justified as underlined in [36].

[2] In the unit quaternion case, the problem is that, roughly speaking, different quaternions may represent the same physical attitude of the related rigid body [99].

[3] As an example, defining the yaw angle between $[0, 2\pi]$ and stabilizing the yaw around 0, it may happen that for some values of the yaw around 0 the controller tries to make an undesired complete rotation of the aerial vehicle.

4.2.5 Momentum-based estimator of external wrench and unmodeled dynamics

The (6×1) generalized momentum vector of the system (4.46) can be defined as

$$q = \begin{bmatrix} mI_3 & O_3 \\ O_3 & M(\eta_b) \end{bmatrix} \begin{bmatrix} \dot{p}_b \\ \dot{\eta}_b \end{bmatrix}, \tag{4.49}$$

where I_n and O_n are $(n \times n)$ identity and zero matrices, respectively. From the expressions of $M(\eta_b)$ and $C(\eta_b, \dot{\eta}_b)$ and from Property 1, it is possible to prove that the following expression holds

$$\dot{M}(\eta_b, \dot{\eta}_b) = C(\eta_b, \dot{\eta}_b) + C(\eta_b, \dot{\eta}_b)^{\mathrm{T}}. \tag{4.50}$$

By using (4.46) and (4.50), the time derivative of the generalized momentum vector (4.49) is

$$\dot{q} = \begin{bmatrix} -uR_b(\eta_b)i_3 + f_u(t) + mg \\ Q(\eta_b)^{\mathrm{T}}\tau_b^b + \tau_u(t) + C(\eta_b, \dot{\eta}_b)^{\mathrm{T}}\dot{\eta}_b \end{bmatrix}. \tag{4.51}$$

The goal of the proposed estimator is to achieve a linear relationship between the dynamics of the estimated external wrench, unmodeled effects and the real ones. Hence, in the Laplace's domain, such relationship has the following expression

$$\begin{bmatrix} \widehat{f}_u(s) \\ \widehat{\tau}_u(s) \end{bmatrix} = G(s) \begin{bmatrix} f_u(s) \\ \tau_u(s) \end{bmatrix}, \tag{4.52}$$

where s is the complex variable in the Laplace's domain, $\widehat{f}_u$ and $\widehat{\tau}_u$ are the (3×1) vectors of the estimated unknown forces and moments, respectively, while $G(s)$ is a (6×6) diagonal matrix of transfer functions in which the i-th element, with $i = 1, \ldots, 6$, has the following expression

$$G_i(s) = \frac{\omega_{n,i}^2}{s^2 + 2\zeta_i\omega_{n,1}s + \omega_{n,i}^2}, \tag{4.53}$$

where $w_{n,i}$ and ζ_i are the desired natural frequency and damping of the designed estimator, respectively, for the i-th component.

In order to get (4.53) component-wise in (4.52), the expression of the estimated external wrench and unmodeled dynamics $r(t) = \begin{bmatrix} \widehat{f}_u^{\mathrm{T}} & \widehat{\tau}_u^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ in the time domain is defined as follows

$$\begin{aligned} r(t) = K_1 \Bigg(\int_o^t -r(\sigma) + K_2 \Bigg(q(\sigma) - \int_0^t \Bigg(r(\sigma) + \\ - \begin{bmatrix} -uR_b(\eta_b)i_3 + mg \\ Q(\eta_b)^{\mathrm{T}}\tau_b^b + C(\eta_b, \dot{\eta}_b)^{\mathrm{T}}\dot{\eta} \end{bmatrix} \Bigg) d\sigma \Bigg) d\sigma \Bigg), \end{aligned} \tag{4.54}$$

where it is assumed that [4] $q(0) = r(0) = \dot{r}(0) = 0$, while $K_1 = \mathrm{diag}\{K_{1,1}, K_{1,2}\}$ and $K_2 = \mathrm{diag}\{K_{2,1}, K_{2,2}\}$ are (6×6) positive definite diagonal matrices, in which $K_{i,j}$, $i, j = \{1, 2\}$,

[4] This condition means that, in the practice, the estimator has to start before the take-off of the UAV.

is a (3×3) positive definite diagonal matrix. Considering (4.46) and (4.51), the dynamics of (4.54) is

$$\ddot{r} + K_1 \dot{r} + K_1 K_2 r = K_1 K_2 \begin{bmatrix} f_u \\ \tau_u \end{bmatrix}, \tag{4.55}$$

that in Laplace domain is equivalent to the 6 transfer functions in (4.52). Once the natural frequencies and the damping factors in (4.53) have been designed, the components of the gains K_1 and K_2 in (4.54) can be computed as follows

$$\begin{aligned} k_{1,i} k_{2,i} &= \omega_{n,i}^2 \\ k_{1,i} &= 2\zeta_i \omega_{n,i} \end{aligned}$$

where $i = 1, \ldots, 6$, and $k_{1,i}$ and $k_{2,i}$ are the i-th elements of K_1 and K_2, respectively.

Notice that, in ideal case,

$$\begin{matrix} \zeta_i \to 1 \\ \omega_{n,i} \to \infty \end{matrix} \Longrightarrow r(t) = \begin{bmatrix} \widehat{f}_u \\ \widehat{\tau}_u \end{bmatrix} \approx \begin{bmatrix} f_u \\ \tau_u \end{bmatrix},$$

where $i = 1, \ldots, 6$, which means that the gains should be taken as large as possible in the practice.

The quantities required to compute r are the UAV orientation η_b and the related time derivative $\dot{\eta}_b$, the vehicle translational velocity $\dot{p}_b$, the commanded input torques τ_b^b, the thrust u and the knowledge about the UAV inertia matrix I_b and mass m. The quantities η_b and $\dot{\eta}_b$ can be retrieved by the on-board IMU sensor, while $\dot{p}_b$ can be estimated by using GPS and/or visual data [68, 69]. The thrust u and the input torques τ_b^b are given by the passivity-based controller (see Section 4.2.6). The UAV inertia I_b and mass m should be instead known a-priori. Notice that no inversion of the inertia matrix $M(\eta_b)$ is required, and also no knowledge about the absolute position p_b of the UAV is needed. Moreover, notice that, with respect to [22], a second-order transfer function has been considered to better weaken the effects of high-frequencies noise (e.g., introduced by both the IMU sensor and the estimation of $\dot{p}_b$) that overcomes the selected bandwidth designed through the choice of $w_{n,i}$, with $i = 1, \ldots, 3$. Notice that with small modifications to (4.54), it is possible to reach a transfer function in (4.52) of the desired order.

4.2.6 VToL UAVs passivity-based control

The time scale separation highlighted in the classical hierarchical controllers [79, 57] is traduced in a inner-outer loop control architecture. Namely, the inner loop is devoted to control the fast time-scale angular part, while the outer loop tackles the slow time-scale position tracking part. Because of the underactuation of the system, only 4 components can be provided by an external planner. Since p_b and ψ are flat outputs for the system (4.46) [29], the planner gives as inputs to the controller the desired position trajectory of the UAV, described by the (3×1) vectors p_d, $\dot{p}_d$ and $\ddot{p}_d$, and the desired yaw trajectory, described by ψ_d, $\dot{\psi}_d$ and $\ddot{\psi}_d$. Hence, the desired pitch and roll components are implicitly computed on the basis of the planned UAV position and yaw.

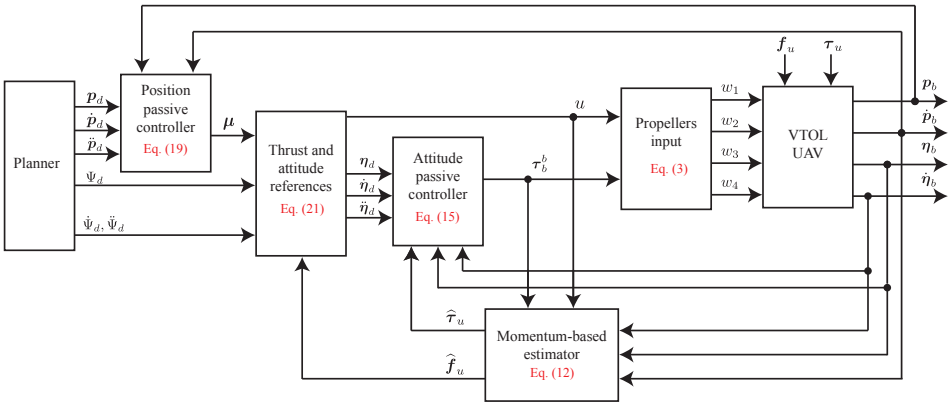

Figure 4.8: Block scheme of the proposed control architecture. In red, the corresponding equations in the section related to each block.

For the inner loop, let $\eta_d = \begin{bmatrix}\phi_d & \theta_d & \psi_d\end{bmatrix}^{\mathrm{T}}$ be the (3×1) vector of the UAV desired attitude with $\dot{\eta}_d$ and $\ddot{\eta}_d$ its time derivatives. Define the following (3×1) reference vector for the attitude velocity

$$\dot{\eta}_r = \dot{\eta}_d - e_\eta, \tag{4.56}$$

where $e_\eta = \eta_b - \eta_d$ is the (3×1) angular tracking error and $\nu > 0$ a coupling parameter. The following passivity-based control input can be then defined for the inner loop

$$\begin{aligned}\tau_b^b =& Q(\eta_b)^{-\mathrm{T}}\left(M(\eta_b)\ddot{\eta}_r + C(\eta_b,\dot{\eta}_b)\dot{\eta}_r - \widehat{\tau}_u\right.\\ &\left. - D_o v_\eta - K_o e_\eta\right),\end{aligned} \tag{4.57}$$

where D_o and K_o are (3×3) positive definite diagonal gain matrices, $\dot{e}_\eta = \dot{\eta}_b - \dot{\eta}_d$, $\ddot{\eta}_r = \ddot{\eta}_d - \dot{e}_\eta$ and $v_\eta = \dot{e}_\eta + e_\eta$. Considering (4.46b), notice that no cancellation of dynamic model terms is performed through (4.57).

The outer loop has then to provide the desired thrust and the reference values of the pitch and roll angles. Define a (3×1) virtual input acceleration vector μ devoted to the position tracking part and that will be designed in the following. It should be possible to retrieve the thrust and the desired attitude angles values for the inner loop from the virtual control input μ. For this reason, it is imposed that

$$\mu = -\frac{u}{m}R_b(\eta_d)i_3 + g + \frac{1}{m}\widehat{f}_u, \tag{4.58}$$

representing the desired acceleration vector with respect to Σ_i, in which the magnitude is the thrust u produced by the propellers, while the orientation is given by the desired UAV attitude. Properly designing mu, inverting (4.58) it is then possible to retrieve the desired values for the thrust and the attitude angles that are in turn exploited as references for the inner control loop. Therefore, let $e_p = p_b - p_d$, $\dot{e}_p = \dot{p}_b - \dot{p}_d$, $\ddot{e}_p = \ddot{p}_b - \ddot{p}_d$ and

$\ddot{e}_\eta = \ddot{\eta}_b - \ddot{\eta}_d$ be the (3×1) tracking errors, and let $\widetilde{f} = f_u - \widehat{f_u}$ be the estimated force error. Replacing both $\eta_b = \eta_d + e_\eta$ and $f_u = \widehat{f_u} + \widetilde{f}$ in (4.46a), recalling (4.58), yields

$$\ddot{p}_b = \mu + \frac{u}{m}\delta(\eta_d, e_\eta) + \frac{1}{m}\widetilde{f}, \tag{4.59}$$

where $\delta(\eta_d, e_\eta) = \begin{bmatrix}\delta_x & \delta_y & \delta_z\end{bmatrix}^{\mathrm{T}}$ is the following (3×1) interconnection vector

$$\delta = \begin{bmatrix} s_{\phi_d}s_{\psi_d} - s_\phi s_\psi - c_\phi c_\psi s_\theta + c_{\phi_d}c_{\psi_d}s_{\theta_d} \\ c_\psi s_\phi - c_{\psi_d}s_{\phi_d} + c_{\phi_d}s_{\theta_d}s_{\psi_d} - c_\phi s_\theta s_\psi \\ c_{\theta_d}c_{\phi_d} - c_\theta c_\phi \end{bmatrix}, \tag{4.60}$$

with $\phi = e_\phi + \phi_d$, $\theta = e_\theta + \theta_d$ and $\psi = e_\psi + \psi_d$. The virtual control input μ can now be chosen as

$$\mu = \ddot{p}_d - \frac{1}{m}\left(D_p\dot{e}_p + K_p e_p\right), \tag{4.61}$$

where D_p and K_p are two (3×3) positive definite diagonal gain matrices.

Folding (4.57) and (4.61) into (4.46b) and (4.59), respectively, yields the following closed-loop equations

$$m\ddot{e}_p + D_p\dot{e}_p + K_p e_p = u\delta(\eta_d, e_\eta) + \widetilde{f}, \tag{4.62a}$$

$$M(\eta_b)\dot{\nu}_\eta + \left(C(\eta_b, \dot{\eta}_b) + D_o\right)\nu_\eta + K_o e_\eta = \widetilde{\tau}, \tag{4.62b}$$

where $\dot{\nu}_\eta = \ddot{e}_\eta + \nu\dot{e}_\eta$ and $\widetilde{\tau} = \tau_u - \widehat{\tau}_u$. The right side of equation (4.62a) acts like an external force on the first subsystem and depends on both the UAV attitude error and the estimated unknown forces error. The right side of equation (4.62b) is the residual of the estimated moments and acts as a disturbance on the second subsystem. Thus, the expressions in (4.62) establish passive relationships between the reconstruction errors of generic unknown disturbances and the tracking errors. In particular, for equation (4.62b), as underlined in [84], there exists a passive mapping between $\widetilde{\tau}$ and ν_η, at least in hovering case.

- **Remark 1.** Notice that the relationship in (4.62a) is equivalent to a generalized mechanical impedance reacting to the external disturbance given by $(u\delta(\eta_d, e_\eta) + \widetilde{f})$ with the same mass m of the aerial vehicle, and with a stiffness and damping that are programmable through the choice of the gain matrices K_p and D_p, respectively.

- **Remark 2.** The gains that have to be tuned in the proposed controller are namely: K_1 and K_2 for the estimator; K_p and D_p for the UAV translational part; ν, K_o and D_o for the angular one. A discussion about how to choose ν is done in [19], while the physical meanings of K_p and D_p are given in Remark 1. K_o and D_o might have similar meanings of programmable stiffness and damping of a torsional spring. The translational part is the slowest one due to the time-scale separation and because it depends on the attitude error. Hence, once the desired stiffness K_p and damping D_p have been chosen, it is possible to retrieve the closed-loop bandwidth of the

controlled system. Then, the natural frequency and damping factor of the estimator can be tuned on the basis of this choice. In particular, they have to be at least larger than those designed for the UAV translational part so as not to weaken the closed-loop designed performance.

To recap, the proposed architecture is depicted in the block scheme of Fig. 4.8. After computing the position tracking errors e_p and $\dot{e}_p$, and knowing the feedforward acceleration $\ddot{p}_d$, the virtual control input μ can be computed as in (4.61). The desired thrust u and the reference pitch and roll can be computed by inverting (4.58) as follows [79]

$$u = m\sqrt{\bar{\mu}_1^2 + \bar{\mu}_2^2 + (\bar{\mu}_3 - g)^2}, \tag{4.63a}$$

$$\phi_d = \sin^{-1}\left(m\frac{\bar{\mu}_2\cos\psi_d - \bar{\mu}_1\sin\psi_d}{u}\right), \tag{4.63b}$$

$$\theta_d = \operatorname{atan2}\left(\bar{\mu}_1\cos\psi_d + \bar{\mu}_2\sin\psi_d, \bar{\mu}_3 - g\right), \tag{4.63c}$$

where $\bar{\mu} = \begin{bmatrix}\bar{\mu}_1 & \bar{\mu}_2 & \bar{\mu}_3\end{bmatrix}^{\mathrm{T}} = \mu - (1/m)\widehat{f}_u$, with $\widehat{f}_u$ given by (4.54), while the desired yaw ψ_d is given by the planner. A second-order low-pass digital filter should be employed to reduce noise and compute both first and second derivatives of η_d [79], so as to get $\dot{\eta}_d$ and $\ddot{\eta}_d$, and hence compute in turn the attitude tracking errors e_η and $\dot{e}_\eta$. The control input vector τ_b^b is computed as in (4.57), in which $\widehat{\tau}_u$ is given by (4.54). Having both the thrust u and the actuation torques τ_b^b, the squared propellers speeds w_i^2 of the VToL UAV, with $i = 1,\ldots,4$, are computed by inverting (4.45).

- **Remark 3.** Notice that in case $\bar{\mu} = \mu - (1/m)\widehat{f}_u = g$, equation (4.63b) is indeterminate. This exact condition is very difficult to happen in the practice but it can not be a priori excluded. From a physical point of view, such numeric singularity means a desired acceleration for the UAV equal to the gravity: this can be achieved with a zero thrust, i.e. turning off the propellers as it is evident from (4.45a). When the propellers are turned off, any values for the pitch and the roll are not reachable since the control is obviously not in action. In the practice, such a particular and uncommon condition can be nonetheless easily managed from a software point of view once the thrust is calculated as in (4.63a). It is worth remarking that no problems happened during the experiments, some of which are described in Section 4.2.11.

- **Remark 4.** Although it is of less interest, the passivity-based approach here proposed can be employed also without considering the compensation of external wrench and unmodeled dynamics, i.e. neglecting the term $\widehat{\tau}_u$ in (4.57) and with $\bar{\mu} = \mu$ in (4.63). As highlighted in [3], the use of integral/adaptive actions, as well as of external disturbances observers, might in some cases worsen and not improve the controller performance. Therefore, in the reminder of the section and during the experiments, it will be checked under which conditions the compensation of the estimated terms improves the performance of the sole passivity-based controller.

4.2.7 Stability proof

This section is devoted to show the stability of the whole control scheme made up by the momentum-based estimator of external wrench and unmodeled dynamics, and the passivity-based controller. It is worth mentioning that only marginal stability can be ensured since perturbation terms on the right sides of (4.62) might be nonvanishing. Moreover, it is also shown how adding the compensation of the estimated terms may help in reducing the asymptotic bounds of the closed-loop systems.

Let $x_1 = \begin{bmatrix} e_p^{\mathrm{T}} & \dot{e}_p^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ and $x_2 = \begin{bmatrix} e_\eta^{\mathrm{T}} & \dot{e}_\eta^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ be two (6×1) vectors denoting the state of the closed-loop system equations (4.62a) and (4.62b), respectively, which can also be arranged in the following way

$$\dot{x}_1 = \alpha_1(m, x_1, K_p, D_p) + \beta_1(u, m, \eta_d, e_\eta, \widetilde{f}), \tag{4.64a}$$
$$\dot{x}_2 = \alpha_2(\nu, x_2, \eta_b, \dot{\eta}_b, K_o, D_o) + \beta_2(\eta_b, \widetilde{au}), \tag{4.64b}$$

where

$$\alpha_1 = \begin{bmatrix} \dot{e}_p \\ -(1/m)D_p\dot{e}_p - (1/m)K_p e_p \end{bmatrix},$$
$$\alpha_2 = \begin{bmatrix} \dot{e}_\eta \\ -M^{-1}\left(\nu\dot{e}_\eta + (C + D_o)\,\nu_\eta + K_o e_\eta\right) \end{bmatrix},$$
$$\beta_1 = \begin{bmatrix} 0_3 \\ (u/m)delta + (1/m)\widetilde{f} \end{bmatrix},$$
$$\beta_2 = \begin{bmatrix} 0_3 \\ M^{-1}\widetilde{au} \end{bmatrix},$$

in which dependences have been dropped and 0_n is the $(n \times 1)$ null vector. Let define the *nominal* systems as the closed-loop equations (4.64) without the perturbation terms $\beta_1(u, m, \eta_d, e_\eta, \widetilde{f})$ and $\beta_2(\eta_b, \widetilde{au})$

$$\dot{x}_1 = \alpha_1(m, x_1, K_p, D_p), \tag{4.65a}$$
$$\dot{x}_2 = \alpha_2(\nu, x_2, \eta_b, \dot{\eta}_b, K_o, D_o). \tag{4.65b}$$

The following further assumption is considered.

- **Assumption 3.** The planned translational acceleration norm is bounded as

$$\ddot{p}_d \leq \|\ddot{p}_d\|_{max} = B_7. \tag{4.66}$$

Two main theorems will be employed in the following.

Theorem 1. *Consider the generic perturbed system*

$$\dot{x} = f(t, x) + g(t, x). \tag{4.67}$$

Let $x = 0$ be a globally exponentially stable equilibrium point of the nominal system

$$\dot{x} = f(t,x). \tag{4.68}$$

Let $V(t,x)$ be a Lyapunov function of (4.68) *satisfying the following inequalities*

$$\gamma_1\|x\|^2 \leq V(t,x) \leq \gamma_2\|x\|^2, \tag{4.69a}$$

$$\frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(x) \leq -\gamma_3\|x\|^2, \tag{4.69b}$$

$$\|\frac{\partial V}{\partial x}\| \leq \gamma_4|x\|, \tag{4.69c}$$

where $V(t,x)$ is defined in $[0,\infty) \times D = \{\|x\| < \infty\}$ and $\gamma_i > 0$, with $i = 1,\ldots,4$. Suppose the perturbation term in (4.67) *satisfies the uniform bound*

$$\|g(t,x)\| \leq \Delta < \infty, \tag{4.70}$$

for all $t \geq t_0$. Then, for all $\|x(t_0)\| < \infty$, the solution $x(t)$ of the perturbed system (4.67) *satisfies*

$$\|x(t)\| \leq \xi e^{-\rho(t-t_0)}\|x(t_0)\|, \quad \forall t_0 \leq t < t_i, \tag{4.71a}$$

$$\|x(t)\| \leq B, \qquad \forall t \geq t_i, \tag{4.71b}$$

for some finite time t_i, where

$$\xi = \sqrt{\frac{\gamma_2}{\gamma_1}},\ \rho = \frac{(1-\varepsilon)\gamma_3}{2\gamma_2},\ B = \frac{\Delta\gamma_4}{\varepsilon\gamma_3}\sqrt{\frac{\gamma_2}{\gamma_1}},$$

with $\varepsilon < 1$.

Proof. See [38], Lemma 5.2. □

Theorem 2. *Consider a generic perturbed system like* (4.67)*. Let $x = 0$ be a globally exponentially stable equilibrium point of the nominal system* (4.68)*. Let $V(t,x)$ be a Lyapunov function of* (4.68) *satisfying inequalities* (4.69)*. Suppose the perturbation term in* (4.67) *satisfies the following inequality*

$$\|g(t,x)\| \leq \Gamma_1(t)\|x\| + \Gamma_2(t), \tag{4.72}$$

where both $\Gamma_1(t)$ and $\Gamma_2(t)$ are nonnegative and continuous terms for all $t \geq t_0$. Moreover, $\Gamma_2(t)$ has to be also bounded for all $t \geq t_0$, while $\Gamma_1(t)$ must satisfy the following inequality

$$\int_{t_0}^{t} \Gamma_1(t)dt \leq b_1(t-t_0) + b_2, \tag{4.73}$$

for some nonnegative constants b_1 and b_2, with

$$b_1 < \frac{\gamma_1\gamma_3}{\gamma_2\gamma_4}. \tag{4.74}$$

Then, for any initial condition of the state $x(t_0)$, the solution of the perturbed system (4.67) *satisfies the following bound*

$$\|x(t)\| \leq b_3, \tag{4.75}$$

with

$$b_3 = \max\left\{\xi\frac{\gamma_2}{\gamma_1}\|x(t_0)\|, \frac{\xi\gamma_4}{2\rho\gamma_2}b_4\right\},$$

where

$$\begin{aligned}\xi &= e^{\frac{\gamma_4 b_2}{2\gamma_1}},\\ \rho &= \frac{1}{2}\left(\frac{\gamma_3}{\gamma_2} - b_1\frac{\gamma_4}{\gamma_1}\right),\\ b_4 &= \sup_{t\geq t_0}\Gamma_2(t).\end{aligned}$$

Proof. See [38], Lemma 5.7. □

By exploiting the two theorems introduced above, a two-steps procedure is employed to prove the stability of (4.64). First, the stability of (4.64b) is verified and the ultimate bound is found for the solution $x_2(t)$, with $t \geq t_0$ and $t_0 \geq 0$ a generic starting time instant. Then, the stability of (4.64a) is verified considering also the interconnection with the angular closed-loop equation (4.64b) given by e_η. However, before starting with these proofs, the boundedness of the errors of the momentum-based estimator is provided.

4.2.8 Boundedness of the external wrench and unmodeled dynamics estimation errors

A bound for the error of the momentum-based estimator of the external wrench and unmodeled dynamics is provided in this subsection. The detailed analysis is carried out for the estimated moments: a similar procedure is valid for the estimated forces.

Since $\widetilde{au} = au_u - \widehat{au}_u$, the following equations hold

$$\widehat{au}_u = au_u - \widetilde{au}, \quad \dot{\widehat{au}}_u = \dot{au}_u - \dot{\widetilde{au}}, \quad \ddot{\widehat{au}}_u = \ddot{au}_u - \ddot{\widetilde{au}}. \tag{4.76}$$

Equation (4.55) can be written in the following way for what concerns the moments' part

$$\ddot{\widehat{au}}_u + K_{1,2}\dot{\widehat{au}}_u + K_{1,2}K_{2,2}\widehat{au}_u = K_{1,2}K_{2,2}au_u, \tag{4.77}$$

and substituting (4.76) into (4.77) yields

$$\ddot{\widetilde{au}} + K_{1,2}\dot{\widetilde{au}} + K_{1,2}K_{2,2}\widetilde{au} = \ddot{au}_u + K_{1,2}\dot{au}_u, \tag{4.78}$$

which is the considered perturbed systems representing the evolution of the error estimate of the unknown moments. The right side term in (4.78) denotes a disturbance against the convergence of the error estimate to zero. For sole constant unknown moments au_u, the

right side term in (4.78) vanishes, meaning a perfect estimate. Otherwise, due to delay introduced by the estimation dynamics in (4.55), the error remains anyway bounded as proven by the following corollary.

Corollary 1. *The error* $\widetilde{r} = \begin{bmatrix} \widetilde{au}^{\mathrm{T}} & \widetilde{f}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ *is bounded while estimating unknown perturbations satisfying* (4.47)*. In particular, considering* $x_\tau = \begin{bmatrix} \widetilde{au}^{\mathrm{T}} & \dot{\widetilde{au}}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$*, the following ultimate bound holds*

$$\|x_\tau(t)\| \leq \xi_1 e^{-\rho_1(t-t_0)} \|x_\tau(t_0)\|, \quad \forall t_0 \leq t < t_1, \tag{4.79a}$$

$$\|x_\tau(t)\| \leq B_8, \qquad \forall t \geq t_1, \tag{4.79b}$$

for some positive constants ξ_1*,* ρ_1 *and* B_8 *defined in the proof. Moreover, in case of constant unknown forces* f_u *and moments* au_u*, the equilibrium points* $x_f = \begin{bmatrix} \widetilde{f}^{\mathrm{T}} & \dot{\widetilde{f}}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} = 0_6$ *and* $x_\tau = 0_6$ *are globally exponentially stable.*

Proof. See 4.2.13. □

From (4.79) it is possible to have

$$\|\widetilde{au}\| \leq B_9 = \max\{\xi_1 \|x_\tau(t_0)\|, B_8\}, \tag{4.80}$$

for all $t \geq t_0$, which represents the ultimate bound for the moments estimation error. Therefore, since the proof in 4.2.13 is performed for the estimated moments but the same procedure holds for the estimated forces, it is possible to conclude that

$$\|\widetilde{f}\| \leq B_{10}, \tag{4.81}$$

with $B_{10} > 0$ depending on B_2 and B_3 (see (4.47b)-(4.47c)).

- **Remark 5.** Taking into account the proof in 4.2.13, if the unknown moments au_u are constants, the right side term in (4.78) vanishes. Hence $B_5 = B_6 = 0 \Rightarrow B_8 = \|\widetilde{au}\| = 0$, for all $t \geq t_1$.

4.2.9 Stability of the closed-loop equation (4.64b)

As shown by the following corollary, the closed-loop equation (4.64b) is stable for bounded perturbations and exponentially stable for constant unknown moments.

Corollary 2. *Under the given assumptions, considering the dynamic model of a generic VToL UAV for the angular part* (4.46b)*, the designed control law* (4.57) *and the compensation of the estimated moments in* (4.54)*, the state error* x_2*, whose dynamics is given by the closed loop system* (4.64b)*, is ultimately bounded as*

$$\|x_2(t)\| \leq \xi_2 e^{-\rho_2(t-t_0)} \|x_2(t_0)\|, \quad \forall t_0 \leq t < t_2, \tag{4.82a}$$

$$\|x_2(t)\| \leq B_{11}, \qquad \forall t \geq t_2, \tag{4.82b}$$

for some positive constants ξ_2*,* ρ_2 *and* B_{11} *defined in the proof. In particular, in case of constant unknown moments* au_u*, the equilibrium point* $x_2 = 0_6$ *is exponentially stable.*

Proof. See 4.2.13. □

The following remark concludes the analysis.

- **Remark 6.** As underlined in Section 4.2.8, when only constant unknown moments au_u are present, the norm $\|\widetilde{au}\|$ goes asymptotically to zero. Hence, B_9 is zero as well as B_{11} for all $t \geq t_2$. Therefore, the closed-loop system (4.64b) becomes exponentially stable.

4.2.10 Stability of the closed-loop equation (4.64a)

As shown by the following corollary, the closed-loop equation (4.64a) is stable for bounded perturbations and exponentially stable for constant unknown moments and forces.

Corollary 3. *Under the given assumptions, considering the dynamic model of a generic VToL UAV for the translational part* (4.46a)*, the designed control law* (4.61) *and the compensation of the estimated unknown forces in* (4.54)*, the state error x_1, whose dynamics is given by the closed loop system* (4.64a)*, is ultimately bounded as*

$$\|x_1(t)\| \leq B_{17}, \tag{4.83}$$

with B_{17} a finite positive bound given in the proof. In particular, in case of constant unknown forces f_u and moments au_u, the equilibrium point $x_1 = 0_6$ is exponentially stable.

Proof. See 4.2.13. □

4.2.11 Experiments

4.2.12 Set-up and technical details

Experiments have been performed by using an Asctech Pelican quadrotor. Both the controller and the estimator have been implemented onboard at 100 Hz on an ATOM CPU with a patched RTAI real-time kernel UBUNTU OS. An OptiTrack motion-capture system has been employed to track both the position and translational velocity of the quadrotor. A ground station made up of a personal computer with UBUNTU OS is in charge of the WiFi communication between the OptiTrack system and the quadrotor as well as for the operator telemetry.

The mass m and the inertia I_b of the vehicle that have been considered in the controller are 1.2 kg and $\mathrm{diag}(3.4, 3.4, 4.7) \cdot 10^{-3}$ kgm^2, respectively. The vehicle parameters in (4.45) are $l = 0.21$ m, $\rho_u = 1.8 \cdot 10^{-5}$ Ns^2/rad^2 and $c = 8 \cdot 10^{-7}$ Nms^2/rad^2.

Following Remark 2, the gains of the controller have been tuned as follows: $K_p = \mathrm{diag}(25, 25, 100)$, $D_p = \mathrm{diag}(10, 10, 20)$, for the translational part; $K_o = \mathrm{diag}(625, 625, 225)$, $D_o = \mathrm{diag}(50, 50, 30)$ for the angular part. The factor ν has been set to 100. Regarding the estimator, instead, the natural frequency and the damping factor have been tuned to 7 rad/s and 1, respectively, for all the force and moment components.

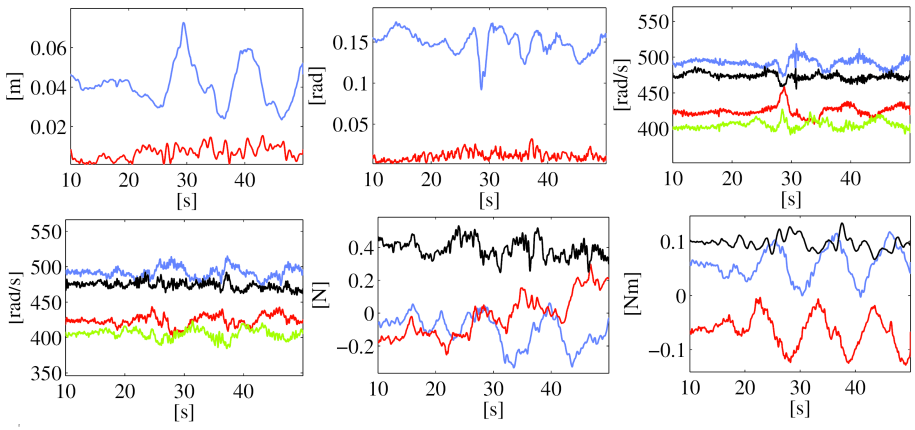

Figure 4.9: Case study A, position, attitude norm errors, commanded propeller velocity with and without compensation. Force and moments estimation in the case of compensation

4.2.13 Case studies

Several case studies are considered in the following. The hovering and tracking performance of the passivity-based control are shown. It will be highlighted how the sole passivity-based control (see Remark 4) is able to perform all the tasks with a good accuracy, but that the compensation is crucial when unmodelled dynamics terms and unexpected situations become relevant. A video of the presented case studies and other different situations can be found in the multimedia attachment[5].

Case study A

In this first case study, the quadrotor tracks three times a given circular trajectory with a constant speed of 0.5 m/s. The circle is planned in the x, y plane at a constant altitude of 1 m from the ground by choosing three different points. The resulting radius is about 0.83 m. After the take-off, the quadrotor reaches the first point of the circle and then executes the planned trajectory. At the end, the landing operation is commanded. In the following analysis represented in Fig. 4.9, the take-off, the landing and the first-point reaching phases are not shown.

The comparison between the norms of the position error in the case of the passivity-based control with and without the compensation of the external wrench and unmodeled dynamics and the attitude error norms are depicted in Fig. 4.9. It is possible to notice how the sole passivity-based control is able to successfully track the circle. The average position error norm is about 5 cm which could be acceptable in several practical tasks.

[5] Also available at: http://youtu.be/iHKtHF0LF-w

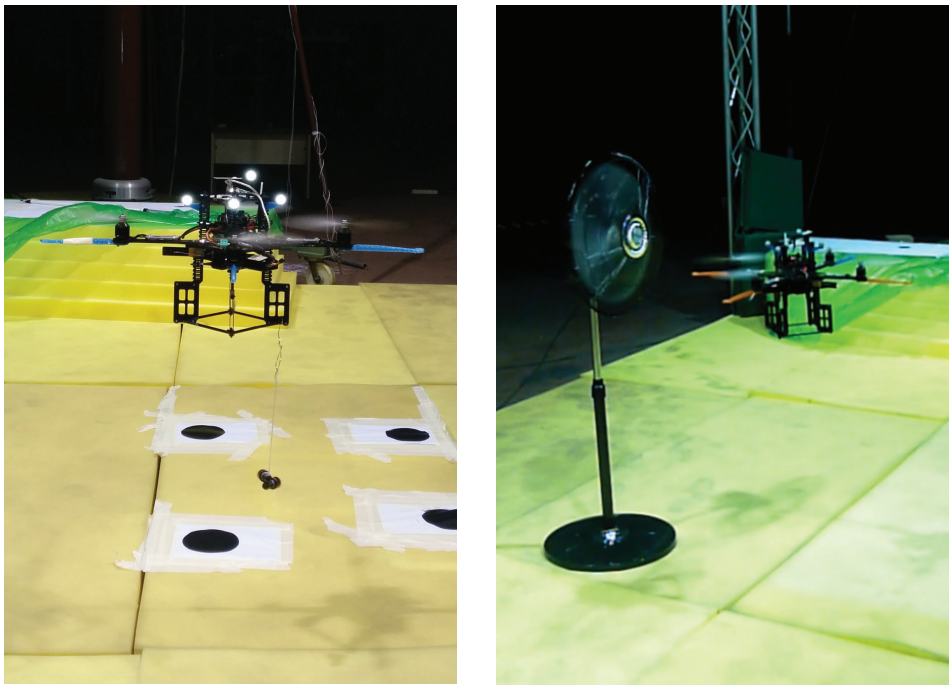

Figure 4.10: Left: quadrotor with the attached pendulum. Right: quadrotor in front of the fan.

However, the performance is substantially improved by using the information provided by the estimator of unknown forces and moments: the norm of the position error decreases to less than 2 cm. From Fig.s 4.9-4.9 it is possible to notice that small uncertainties are present. For instance, the considered inertia I_b might be inaccurate and the estimated force along the z axis might be an indicator about either a missing amount in the considered mass of the Pelican or that the commanded thrust is not perfectly equal to the actual one. The commanded propellers inputs in the two considered cases are represented in Fig.s 4.9-4.9.

Case study B

In this second case study, the same circular trajectory of the previous situation is considered. However, an external load has been physically added and not considered in the controller. In particular, after the take-off, a pendulum has been attached, through a hook (see Fig. 4.10), to the bottom of the quadrotor and far from the vertical axis of the vehicle of about 15 cm. The pendulum has a mass of about 0.15 Kg and a length of 0.21 m.

The effect of the additional load is visible in Fig.s 4.11-4.11, where now the estimated force reflects the presence of the additional mass of the pendulum. Moreover, comparing

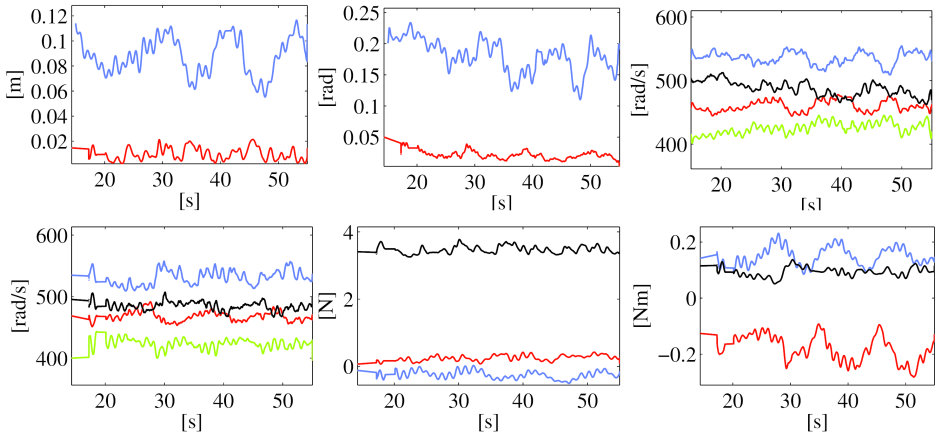

Figure 4.11: Case study B, position, attitude norm errors, commanded propeller velocity with and without compensation. Force and moments estimation in the case of compensation

Fig. 4.9 with Fig. 4.11 it is possible to notice the effect of the oscillations of the pendulum during the circular trajectory resulting in the presence of significant unknown moments. These disturbances affect the performance of the controller. Namely, in the sole passivity-based control, the average position error norm is about 9 cm (see Fig. 4.11), while the average attitude error norm is about 10 degrees (see Fig. 4.11). However, in any case, such control exhibits robust properties in presence of unmodelled and unpredicted effects. The performance is increased by exploiting the compensation provided by the estimator as it possible to see in Fig.s 4.11-4.11. The commanded propellers inputs are represented in Fig.s 4.11-4.11.

Case study C

In this last case study, the quadrotor is subject to an external disturbance caused by a fan (see Fig. 4.10). This last is placed at about 1.1 m from the ground and at a distance of about 20 cm from the aerial vehicle in the x, y-plane. The quadrotor takes off at a height of about 0.6 m, then it reaches the altitude of 1.8 m passing in front of the fan. Then, it decreases again its altitude to 0.6 m (passing again through the wind flow generated by the fan) and finally goes in front of the fan at 1.1 m from the ground, simulating a persisting disturbance. After few seconds, the landing action is commanded. Each rectilinear path along the z axis is performed at a constant speed of 0.5 m/s. The take-off and the landing phases are neglected in the plots of Fig. 4.12.

Again, in general, by looking at Fig.s 4.12-4.12, it is possible to notice that the sole passivity-based control is stable even in the presence of both time-varying and constant

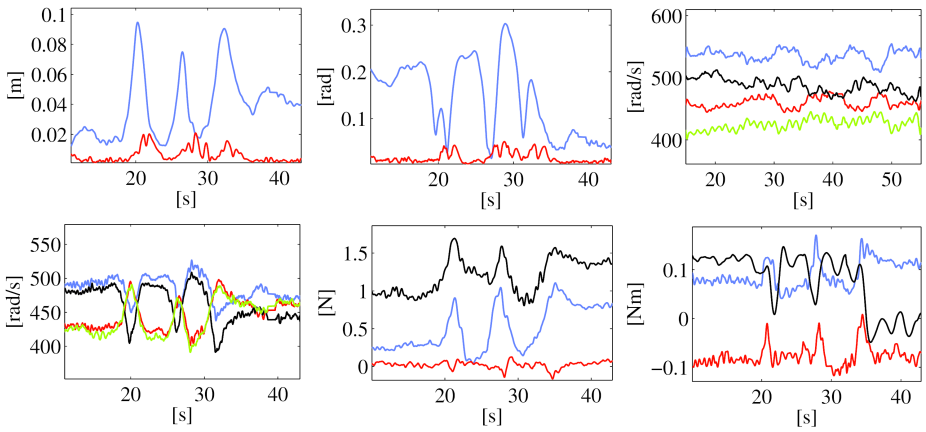

Figure 4.12: Case study C, position, attitude norm errors, commanded propeller velocity with and without compensation. Force and moments estimation in the case of compensation

disturbances: the performance is poor and can be recovered exploiting the compensation provided by the estimator. Fig.s 4.12-4.12 show the estimated forces and moments in the compensation case. It is possible to notice the first passage in front of the fan at about 20 s. Notice that the estimated forces are expressed with respect to Σ_i and the fan is aligned to the x axis of the inertial frame. The second passage in front of the fan is at about 25 s. Afterwards, the quadrotor stays in front of the fan from the time instant at 31 s until the landing command is given. At that point, it is possible to notice that the estimated forces are almost constant (about 0.8 N along the x axis), while the estimated moments wave due to small oscillations of the aerial platform caused by small turbulent aerodynamic effects on the UAV. This last causes also a small estimated force along z axis. The commanded propellers inputs are represented in Fig.s 4.12-4.12.

Proof of Corollary 1

Proof. Considering Theorem 1, the nominal system associated to (4.78) is

$$\alpha_\tau(x_\tau) = \dddot{\widetilde{au}} + K_{1,2}\ddot{\widetilde{au}} + K_{1,2}K_{2,2}\dot{\widetilde{au}} + K_{1,2}K_{2,2}\widetilde{au} = 0_6. \tag{4.84}$$

The origin $x_\tau = 0_6$ of the linear system (4.84) is globally exponentially stable since (4.84) is a second-order differential linear equation with $K_{1,2}$ and $K_{2,2}$ positive definite matrices. Therefore, the following function

$$V_1(x_\tau) = \frac{1}{2} x_\tau^{\mathrm{T}} P_1 x_\tau, \tag{4.85}$$

is a Lyapunov function for (4.84), in which P_1 is a (6×6) positive definite symmetric matrix solving the following equation

$$P_1A_1 + A_1^{\mathrm{T}}P_1 + Lambda_1 = O_6, \tag{4.86}$$

for any (6×6) definite positive symmetric matrix $Lambda_1$, with A_1 the linear matrix associated to (4.84), depending on $K_{1,2}$ and $K_{2,2}$. Through this choice of $V_1(x_\tau)$, the bounds in (4.69) are verified as follows [38]

$$\gamma_1 = \underline{\lambda}_{P_1},\ \gamma_2 = \overline{\lambda}_{P_1},\ \gamma_3 = \underline{\lambda}_{\Lambda_1},\ \gamma_4 = 2\overline{\lambda}_{P_1}, \tag{4.87}$$

where $\overline{\lambda}_{\cdot}$ and $\underline{\lambda}_{\cdot}$ are the maximum and minimum eigenvalues, respectively, of the associated matrix.

Taking into account (4.47e) and (4.47f), the bound in (4.70) of the perturbation term $\ddot{u}_u + K_{1,2}\dot{u}_u$ is satisfied with

$$\Delta = \overline{\lambda}_{K_{1,2}}B_5 + B_6, \tag{4.88}$$

for all $t \geq t_0$.

Then, considering (4.71), for all $\|x_\tau(t_0)\| < \infty$, the solution $x_\tau(t)$ of the perturbed system (4.78) satisfies (4.79) for some finite time t_1, depending on (4.88) and $\|x_\tau(t_0)\|$, where

$$\xi_1 = \sqrt{\frac{\overline{\lambda}_{P_1}}{\underline{\lambda}_{P_1}}},\ \rho_1 = \frac{(1-\varepsilon_1)\underline{\lambda}_{\Lambda_1}}{2\overline{\lambda}_{P_1}},$$

with $\varepsilon_1 < 1$ and

$$B_8 = \frac{2\overline{\lambda}_{P_1}(\overline{\lambda}_{K_{1,2}}B_5 + B_6)}{\varepsilon_1\underline{\lambda}_{\Lambda_1}}\sqrt{\frac{\overline{\lambda}_{P_1}}{\underline{\lambda}_{P_1}}}. \tag{4.89}$$

The same procedure can be applied for the estimated forces. □

It is worth noticing that B_8 can be decreased trough a proper choice of the gains $K_{1,2}$ and $K_{2,2}$ and of the matrix $Lambda_1$.

Proof of Corollary 2

Proof. Theorem 1 is taken into account for the demonstration. In order to show that $x_2 = 0_6$ is a globally exponentially stable equilibrium point of the nominal closed-loop equation (4.65b), the inequalities in (4.69) have to be satisfied [38]. Therefore, consider the following candidate Lyapunov function inspired by [19]

$$V_2(t,x_2) = \frac{1}{2}x_2^{\mathrm{T}}P_2x_2, \tag{4.90}$$

Thanks to Sylvester's criterion, it is possible to verify that the quadratic form in (4.90) is positive definite and vanishes only when $x_2 = 0_6$. Inequality (4.69a) is then proved with $\gamma_1 = \frac{1}{2}\underline{\lambda}_{P_2}$ and $\gamma_2 = \frac{1}{2}\overline{\lambda}_{P_2}$. In order to verify (4.69b), the following inequality holds

$$\frac{\partial V_2}{\partial t} + \frac{\partial V_2}{\partial x_2}\alpha_2(\nu, x_2, \eta_b, \dot{\eta}_b, K_o, D_o) \leq -x_2^{\mathrm{T}}Lambda_2x_2,$$

with

$$Lambda_2 = \begin{bmatrix} \nu K_o + \nu^2 D_o & O_3 \\ O_3 & D_o \end{bmatrix}.$$

It is possible to easily check that $Lambda_2$ is positive definite and then (4.69b) is verified through $\gamma_3 = \underline{\lambda}_{\Lambda_2}$. Finally, inequality (4.69c) is proved with $\gamma_4 = \overline{\lambda}_{P_2}$.

Taking into account (4.80), the uniform bound in (4.70) is proved as follows

$$\|\beta_2(\eta_b, \widetilde{au})\| \leq \overline{\lambda}_{M^{-1}} \|\widetilde{au}\| \leq \overline{\lambda}_{M^{-1}} B_9 < \infty. \tag{4.91}$$

Then, considering (4.71), for all $\|x_2(t_0)\| < \infty$, the solution $x_2(t)$ of the perturbed system (4.64b) satisfies (4.82) for some finite amount of time $t_2 \geq t_1$, depending on (4.91) and $\|x_2(t_0)\|$, where

$$\xi_2 = \sqrt{\frac{\overline{\lambda}_{P_2}}{\underline{\lambda}_{P_2}}}, \quad \rho_2 = \frac{(1-\varepsilon_2)\underline{\lambda}_{\Lambda_2}}{\overline{\lambda}_{P_2}},$$

with $\varepsilon_2 < 1$ and

$$B_{11} = \frac{\overline{\lambda}_{P_2}\overline{\lambda}_{M^{-1}} B_9}{\varepsilon_2 \underline{\lambda}_{\Lambda_2}} \sqrt{\frac{\overline{\lambda}_{P_2}}{\underline{\lambda}_{P_2}}}. \tag{4.92}$$

□

The following two remarks conclude the proof.

- **Remark 7.** As highlighted in Remark 4, it is not ensured in principle that the compensation of estimated generalized forces improves the performance of the sole passivity-based controller. In order to check whether the compensation is convenient or not, the case where the estimations are not employed is considered. In such a case, the perturbation term in (4.64b) appears to be

  $$\beta_2'(\eta_b, au_u) = \begin{bmatrix} 0_3 \\ M(\eta_b)^{-1} au_u \end{bmatrix}.$$

  Corollary 2 still holds, but now inequality (4.91), which is necessary to prove hypothesis (4.70), is modified as follows

  $$\|\beta_2'(\eta_b)\| \leq \overline{\lambda}_{M^{-1}} \|au_u\| \leq \overline{\lambda}_{M^{-1}} B_4 = \Delta' < \infty,$$

  where (4.47d) has been taken into account. On the one hand, in case only constant unknown moments au_u are present, estimated, and compensated, Δ' is always greater than 0, since $B_4 > 0$, while B_9 in (4.91) is zero from Remark 6. Hence, $B_{11}' > B_{11} = 0$: the bound with the compensation is less than the case without the compensation meaning that the performance of the controller is improved when a feedback of the unknown moments estimation is provided in (4.57). On the other hand, when time-varying unknown moments au_u are present, estimated, and compensated, $B_{11} < B_{11}' \Longleftrightarrow \overline{\lambda}_{M^{-1}} B_9 < \Delta'$, meaning that compensation of the estimated

moments is convenient in (4.57) when $B_9 < B_4$. This is in general verified when au_u is slow-time varying and thanks to a proper choice of both estimator bandwidth and matrix $Lambda_1$ in (4.86).

- **Remark 8.** Notice that inequality (4.82b) might be used to verify that the controller maintains the selected Euler angles in a singularity-free zone (see Assumption 1). Nevertheless, the mathematical derivation is cumbersome due to the complicated expression of B_{11} in (4.92). Experiments performed in Section 4.2.11 seem anyway very promising from this point of view.

Proof of Corollary 3

Proof. Theorem 2 is taken into account for the demonstration. The nominal closed-loop system (4.65a) has a unique exponentially equilibrium point $x_1 = 0_6$, since (4.65a) is a linear system with an associated (6×6) state matrix $A_2(x_1)$ which is Hurwitz, since $m > 0$ and both K_p and D_p are positive definite diagonal matrices. Therefore, the following function

$$V_3(x_1) = \frac{1}{2} x_1^{\mathrm{T}} P_3 x_1, \tag{4.93}$$

is a Lyapunov function for (4.65a), in which P_3 is a (6×6) positive definite symmetric matrix solving the following equation

$$P_3 A_2 + A_2^{\mathrm{T}} P_3 + Lambda_3 = O_6, \tag{4.94}$$

for any (6×6) definite positive symmetric matrix $Lambda_3$. Through this choice of $V_3(x_1)$, inequalities (4.69) are verified with the following choice of the bounds [38]

$$\gamma_1 = \underline{\lambda}_{P_3},\ \gamma_2 = \overline{\lambda}_{P_3},\ \gamma_3 = \underline{\lambda}_{\Lambda_3},\ \gamma_4 = 2\overline{\lambda}_{P_3}. \tag{4.95}$$

Taking into account (4.47a), (4.58), (4.61), (4.66), (4.81) and the equation $\widehat{f_u} = f_u - \widetilde{f}$, it is possible to give the following ultimate bound to the thrust

$$\begin{aligned} |u| &= m\|\ddot{p}_d - g - \frac{1}{m} D_p \dot{e}_p - \frac{1}{m} K_p e_p - \frac{1}{m} \widehat{f_u}\| \\ &\leq B_{12} + B_{13}(\|e_p\| + \|\dot{e}_p\|) \\ &\leq B_{12} + B_{13}\sqrt{2}\|x_1\|, \end{aligned} \tag{4.96}$$

where $B_{12} = B_1 + m(g + B_7) + B_{10}$ and $B_{13} = \max\{\overline{\lambda}_{K_p}, \overline{\lambda}_{D_p}\}$. The ultimate bound for the term $delta(\eta_d, e_\eta)$ in $\beta_1(u, m, \eta_d, e_\eta, \widetilde{f})$ is

$$\|delta(\eta_d, e_\eta)\| \leq B_{14}\|e_\eta\|, \tag{4.97}$$

with $B_{14} > 0$. By recalling (4.60) and exploiting the following general relationships

$$\begin{aligned}
\sin(a+b) &= \sin(a) + 2\sin(b/2)\cos(a+b/2) \\
\cos(a+b) &= \cos(a) - 2\sin(b/2)\sin(a+b/2) \\
|\sin(a)| &\leq |a|,\ |\sin(a)| \leq 1,\ |\cos(a)| \leq 1 \\
\prod_{i=1}^{n} |a_i| &\leq \frac{1}{2}\sum_{i=1}^{n} |a_i|,\ \text{for } |a_i| \leq 1,
\end{aligned}$$

inequality (4.97) can be verified providing first a bound to $|\delta_x|$, $|\delta_y|$ and $|\delta_z|$ and then considering $\|delta(\eta_d, e_\eta)\| = \sqrt{\delta_x^2 + \delta_y^2 + \delta_z^2}$. Notice that $|\sin(a)| \leq |a|$ and $|\sin(a)| \leq 1$ are employed with arguments e_Υ and $e_\Upsilon + \Upsilon_d$, respectively, with $\Upsilon = \{\phi, \theta, \psi\}$. Hence, taking into account (4.81), (4.96) and (4.97), the following ultimate bound can be written for the perturbation term $\beta_1(u, m, \eta_d, e_\eta, \widetilde{f})$

$$\|\beta_1\| \leq \frac{1}{m}(B_{10} + B_{13}B_{14}\sqrt{2}\|e_\eta\|\|x_1\| + B_{12}B_{14}\|e_\eta\|). \tag{4.98}$$

Comparing (4.72) and (4.98), it is possible to recognize that $\Gamma_1(t) = (B_{13}B_{14}\sqrt{2}/m)\|e_\eta\|$, while $\Gamma_2(t) = (1/m)(B_{10} + B_{12}B_{14}\|e_\eta\|)$. Notice that both $\Gamma_1(t)$ and $\Gamma_2(t)$ are nonnegative and continuous terms for all $t \geq t_0$. Moreover, $\Gamma_2(t)$ is bounded for all $t \geq t_0$ since

$$\Gamma_2(t) = \frac{B_{10} + B_{12}B_{14}\|e_\eta\|}{m} \leq \frac{B_{10} + B_{12}B_{14}B_{15}}{m},$$

in which (4.82) has been considered, with $B_{15} = \max\{\xi_2\|x_2(t_0)\|, B_{11}\}$. In order to verify (4.73), denoting with $Y = (B_{13}B_{14}/m)\sqrt{2}$, the following inequality holds for $\Gamma_1(t)$

$$\begin{aligned}
Y\int_{t_0}^{t} \|e_\eta\|dt &= Y\left(\int_{t_0}^{t_2} \|e_\eta\|dt + \int_{t_2}^{t} \|e_\eta\|dt\right) \\
&\leq Y\left(\xi_2 B_{16}\|e_\eta(t_0)\| + B_{11}(t - t_2)\right) \\
&< Y\left(\xi_2 B_{16}\|e_\eta(t_0)\| + B_{11}(t - t_0)\right)
\end{aligned} \tag{4.99}$$

where $B_{16} = (1/\rho_2)e^{\rho_2 t_0}(e^{-\rho_2 t_0} - e^{-\rho_2 t_2})$. Hence, inequality (4.73) is verified with $b_1 = YB_{11} > 0$ and $b_2 = YB_{16}\xi_2\|e_\eta(t_0)\|$. Notice that b_2 is always positive and bounded. Therefore, taking into account (4.73) and (4.99), fixing the desired gains K_p and D_p, noticing the dependency from the mass of the vehicle m and the bound B_{11}, it is then always possible to choose a matrix $Lambda_3$ such that inequality (4.74) is verified.

Then, considering (4.75), for any initial condition of the state $x_1(t_0)$, the solution of the closed-loop system (4.64a) satisfies (4.83) with

$$B_{17} = \max\left\{\xi_3\frac{\overline{\lambda}_{P_3}}{\underline{\lambda}_{P_3}}\|x_1(t_0)\|, \frac{\xi_3\overline{\lambda}_{P_3}}{\rho_3\overline{\lambda}_{P_3}}B_{18}\right\},$$

where

$$\xi_3 = e^{\frac{-Y\overline{\lambda}_{P_3} - B_{16}\xi_2\|e\eta(t_0)\|}{\underline{\lambda}_{P_3}}},$$
$$\rho_3 = \frac{1}{2}\left(\frac{\underline{\lambda}_{\Lambda_3}}{\overline{\lambda}_{P_3}} - 2YB_{11}\frac{\overline{\lambda}_{P_3}}{\underline{\lambda}_{P_3}}\right),$$
$$B_{18} = \sup_{t\geq t_0}\Gamma_2(t).$$

□

The following remark concludes the analysis.

- **Remark 9.** Notice that if only constant unknown moments au_u are present, estimated, and compensated, then B_{11} is zero from Remark 6. As a consequence, $b_1 = 0$ for all $t \geq t_2$. Inequality (4.74) is thus verified for any value of $Lambda_3$, K_p and D_p, while $x_1(t)$'s bound in (4.83) depends only on B_{10}, which is due to the force estimation process. Therefore, if only constant unknown forces f_u are present, estimated, and compensated, then $B_{10} = 0$ thanks to a similar consideration as in Remark 6, and $x_1(t)$ goes asymptotically to zero. Furthermore, similar considerations can be done as in Remark 7 to show that compensation of estimated forces is convenient.

Chapter 5

Conclusions and future works

A brief recap about the methods presented in this thesis and the achieved results will be the object of the current chapter. Proposals for future research directions will be discussed as well.

5.1 Main Results

In this thesis an architecture to control an *Unmanned Aerial Vehicle* at different level of abstractions and operating in human robot teams has been presented. The main motivation of this work can be found in the SHERPA project, an European founded project whose goal is to develop a heterogeneous mixed flying and ground robotic team lead by a human operator to perform *Search & Rescue* operations in alpine scenario. In this context, the hard operative conditions caused by the hazardous environment and the adverse weather conditions and the limited amount of time needed to successfully complete the rescue mission make necessary endow the robot of different intelligent and autonomous skills to simplify the role of the human operator in the robot commanding an controlling phases. The proposed architecture has been split in three different layers:

- *High-level:* at this level the human operator must be able to select and command in a natural and easy way one of the robot of the team. In this context, the main result can be considered in the field of the *Human-Robot Interaction*. In particular, a system that supports human multi-robot interaction during the execution of collaborative interactive tasks by facilitating the robot selection process is presented. In the proposed approach the human operator is able to implicitly designate the robot responsible for the execution of a command, because each robot of the team can estimate the probability of being involved in the task from its current status and the operative context. An automatic selection process mechanism that uses human intention estimation for the robot selection has been designed. To estimate the human intention a three layer architecture has been proposed to fuse different kind of information such as the robot capabilities, geometrical and contextual information. In

addition, to allow the selection and commanding phase a multimodal human-robot interaction framework suitable for a rescuer that cooperates with a set of drones while searching for missing persons in an alpine scenario is presented.

- *Middle-level:* this module considered the autonomous and semi-autonomous action execution in service robotics. In this context, a system that combines a set of Artificial Intelligence methods, such as HTN planning, BDI execution, RRT path planning/replanning is proposed, showing its performance and feasibility in a industrial plant inspection case study. In addition, *Mixed-Initiative Interaction* techniques in *Planning & Execution* phase are investigated in order to better assist the human operator. For this purpose, a mixed initiative control system for SAVs that combines continuous mixed initiative planning and haptic feedback is presented. The proposed framework allows the operator to adjust the planned trajectory (or to force replanning) receiving a force feedback associated with the feeling of the robot displacement with respect to the generated path (and to the presence of obstacles). Moreover, the force feedback gives an intuitive perception of the action to take in order to get back to the autonomous mode and the planned trajectory.

- *Low-level:* in the low-level layer of the architecture the actuation control input for the single *UAV* is generated in order to perform the command decided in the higher levels. For this purpose a method for react in a robust way to external unmodeled disturbances is presented. This case fit perfectly our domain due to the variable wind conditions in high altitude mountains. In particular, a momentum-based estimator of external wrench and unmodeled dynamics has been employed to control a VToL UAV together with a passivity-based control. The algorithm permits to successfully perform hovering and tracking tasks with a good accuracy. The robustness of the control has been tested in presence of unmodeled dynamic parameters and external disturbances. However, even if the stability of the controller is preserved, the performance might be poor in this last case. For this reason, the presence of an estimator of forces and moments becomes crucial in presence of uncertainties. Finally, a hybrid servoing framework for the autonomous interaction of an UAV equipped with a multi-DOF arm for aerial manipulation tasks is presented.

5.2 Proposals for the future

Likewise performed in the previous section of this chapter, the directions for future researches can be also analysed from different point of view.
For what concerting the *Human-Robot Interaction* of the human operator with the robots of the team, the proposed methods is only evaluated in a simulated environment, for this purpose, due to the promising results already obtained, an experimentation on real world scenario must be performed.
Regarding the *Low-level* proposed methodologies, future work will be focused on prob-

lems related to outdoor scenarios: in particular, the effects on introducing an estimated translational velocity rather than using the one provided by a visual tracker. Finally, future work will be focused on a more deep integration between the different levels of the proposed architecture, namely between the *High-level* modules and the *Low-level* ones in order to endow the *Planning & Execution* phase with different heterogeneous kind of information in order to increase the awareness of the robot with respect its internal situation and allow better autonomous action execution performance.

Bibliography

[1] AIRobots, *Eu collaborative project ict-248669, "airobots", www.airobots.eu*, 2010.

[2] A. Amor-Martinez, A. Ruiz, F. Moreno-Noguer, and A. Sanfeliu, *On-board real-time pose estimation for UAVs using deformable visual contour registration*, 2014 IEEE International Conference on Robotics and Automation, May 2014, pp. 2595–2601.

[3] G. Antonelli, *On the use of adaptive/integral actions for 6-degrees-offreedom control of autonomous underwater vehicles*, IEEE Journal of Oceanic Engineering **32** (2007), no. 2, 300–312.

[4] G Antonelli, *Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems*, IEEE Transactions on Robotics **25** (2009), no. 5, 985–994.

[5] G. Antonelli, F. Arrichiello, S. Chiaverini, and P. Robuffo Giordano, *Adaptive trajectory tracking for quadrotor MAVs in presence of parameter uncertainties and external disturbances*, Proceedings 2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronic (Wollongong, AU), 2013, pp. 1337–1342.

[6] G. Antonelli, E. Cataldi, P. Robuffo Giordano, S. Chiaverini, and A. Franchi, *Experimental validation of a new adaptive control scheme for quadrotors MAVs*, 2013 IEEE/RSJ IInternational Conference on Intelligent Robots and Systems (Tokyo, J), 2013, pp. 3496–3501.

[7] G. Arleo, F. Caccavale, G. Muscio, and F. Pierri, *Control of quadrotor aerial vehicles equipped with a robotic arm*, 21st Mediterranean Conference on Control and Automation (Crete, GR), 2013.

[8] H. Berghuis and H. Nijmeijer, *A passivity approach to controller-observer design for robots*, IEEE Transactions on Robotics and Automation **9** (1993), no. 6, 740–754.

[9] S. Bernardini, M. Fox, and D. Long, *Planning the behaviour of low-cost quadcopters for surveillance missions*, Proc. of ICAPS 2014, 2014.

[10] E. Bitton and K. Goldberg, *Hydra: A framework and algorithms for mixed-initiative uav-assisted search and rescue*, Proc. of CASE 2008, 2008, pp. 61–66.

[11] Michael Brenner and Bernhard Nebel, *Continual planning and acting in dynamic multiagent environments*, Journal of Autonomous Agents and Multiagent Systems **19** (2009), no. 3, 297–331.

[12] Luca Rosario Buonocore, Jonathan Cacace, and Vincenzo Lippiello, *Hybrid visual servoing for aerial grasping with hierarchical task priority control*, 23st Mediterranean Conference on Control Automation (MED), June 2015, pp. 651–657.

[13] D. Cabecinhas, R. Cunha, and C. Silvestre, *A nonlinear quadrotor trajectory tracking controller with disturbance rejection*, Control Engineering Practice **26** (2014), 1–10.

[14] J. Cacace, A. Finzi, and V. Lippiello, *A mixed-initiative control system for an aerial service vehicle supported by force feedback*, Proc. of IROS 2014, 2014, pp. 1230 – 1235.

[15] J. Cacace, A. Finzi, V. Lippiello, G. Loianno, and D. Sanzone, *Aerial service vehicles for industrial inspection: Task decomposition and plan execution*, Proc. of EIA-AIE 2013, 2013.

[16] Jonathan Cacace, Alberto Finzi, Vincenzo Lippiello, Giuseppe Loianno, and Dario Sanzone, *Aerial service vehicles for industrial inspection: task decomposition and plan execution*, Appl. Intell. **42** (2015), no. 1, 49–62.

[17] R. Cano, C. Pérez, F. Pruano, A. Ollero, and G. Heredia, *Mechanical design of a 6-DOF aerial manipulator for assembling bar structures using UAVs*, 2nd RED-UAS 2013 Workshop on Research, Education and Development of Unmanned Aerial Systems (Compiegne, F), 2013.

[18] A. Carbone, A. Finzi, A. Olandini, F. Pirri, and G. Ugazio, *Augmenting situation awareness via model-based control in rescue robots*, Proc. of IROS 2005, IEEE, 2005, pp. 3699–3705.

[19] W. Chung, Li-C. Fu, and Su-H. Hsu, *Motion control*, Springer Handbook of Robotics (B. Siciliano and O. Khatib, eds.), Springer, 2008, pp. 133–159.

[20] CSAR, *Canadian national search and rescue manual*, Department of National Defence, 2000.

[21] M.L. Cummings, S. Bruni, S. Mercier, and P. J. Mitchell, *Automation architecture for single operator, multiple uav command and control*, Intern. Command and Control Journal **1** (2007), no. 2, 1–24.

[22] A. De Luca, A. Albu-Schaffer, S. Haddadin, and G. Hirzinger, *Collision detection and safe reaction with the DLR-III lightweight manipulator arm*, IEEE/RSJ International Conference on Intelligent Robots and Systems (Beijing, C), 2006, pp. 1623–1630.

[23] Z.T. Dydek, A.M. Annaswamy, and E. Lavretsky, *Adaptive control of quadrotor UAVs: A design trade study with flight evaluations*, IEEE Transactions on Control Systems Technology **21** (2013), no. 4, 1400–1406.

[24] O. Egeland and J.-M. Godhavn, *Passivity-based adaptive attitude control of a rigid spacecraft*, IEEE Transactions on Automatic Control **39** (1994), no. 4, 842–846.

[25] Kutluhan Erol, James Hendler, and Dana S. Nau, *Umcp: A sound and complete procedure for hierarchical task-network planning*, Proc. of AIPS-1994, 1994, pp. 249–254.

[26] M. Fiala, *ARTag, a fiducial marker system using digital techniques*, 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, June 2005, pp. 590–596.

[27] A. Finzi and A. Orlandini, *Human-robot interaction through mixed-initiative planning for rescue and search rovers*, Proc. of AI*IA 2005, 2005, pp. 483–494.

[28] Martin A. Fischler and Robert C. Bolles, *Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography*, Commun. ACM **24** (1981), no. 6, 381–395.

[29] P.M.M. Fliess, J. Levine, and P. Rouchon, *Flatness and defect of nonlinear systems: Introductory theory and examples*, International Journal of Control **61** (1995), no. 6, 1327–1361.

[30] S. Formentin and M. Lovera, *Flatness-based control of a quadrotor helicopter via feedforward linearization*, 2011 50th IEEE International Conference on Decision and Control and European Control Conference (Orlando, FL, USA), 2011, pp. 6171–6176.

[31] F. Forte, R. Naldi, A. Macchelli, and L. Marconi, *Impedance control of an aerial manipulator*, Proceedings of 2012 American Control Conference (Montreal, CDN), 2012, pp. 3839–3844.

[32] F. Forte, R. Naldi, and Marconi L. Macchelli, A., *On the control of an aerial manipulator interacting with the environment*, IEEE International Conference on Robotics and Automation, May 2014, pp. 4487–4492.

[33] M. Fumagalli and R. Carloni, *A modified impedance control for physical interaction of UAVs*, 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (Tokyo, J), 2013, pp. 1979–1984.

[34] C. Ha, Z. Zuo, F.B. Choi, and D.J. Lee, *Passivity-based adaptive backstepping control of quadrotor-type UAVs*, Robotics and Autonomous Systems **62** (2014), no. 9, 1305–1315.

[35] B. Hannaford, L. Wood, D.A. McAffee, and H. Zak, *Performance evaluation of a six-axis generalized force-reflecting teleoperator*, IEEE Transactions on Systems Man and Cybernetics (1991).

[36] M.-D. Hua, T. Hamel, P. Morin, and C. Samson, *A control approach for thrust-propelled underactuated vehicles and its application to VTOL drones*, IEEE Transactions on Automatic Control **54** (2009), no. 8, 1837–1853.

[37] Sertac Karaman and Emilio Frazzoli, *Sampling-based algorithms for optimal motion planning*, Int. J. Rob. Res. **30** (2011), no. 7, 846–894.

[38] H.K. Khalil, *Nonlinear systems*, Prentice Hall, Upper Saddle River, NJ, 2002.

[39] Suseong Kim, Seungwon Choi, and H.J. Kim, *Aerial manipulation using a quadrotor with a two dof robotic arm*, IEEE/RSJ International Conference on Intelligent Robots and Systems, Nov 2013, pp. 4990–4995.

[40] M. Kobilarov, *Nonlinear trjectory control of multi-body aerial manipulators*, Journal of Intelligent and Robotic Systems **73** (2014), no. 1–4, 679–692.

[41] C. Korpela, M. Orsag, and P. Oh, *Hardware-in-the-loop verification for mobile manipulating unmanned aerial vehicles*, Journal of Intelligent and Robotic Systems **73** (2014), no. 1–4, 725–736.

[42] _____, *Towards valve turning using a dual-arm aerial manipulator*, 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (Chicago, IL, USA), 2014, pp. 3411–3416.

[43] David Landén, Fredrik Heintz, and Patrick Doherty, *Complex Task Allocation in Mixed-Initiative Delegation: A UAV Case Study*, PRIMA 2010, vol. 7057, 2012, pp. 288–303.

[44] S. M. Lavalle, *Rapidly-exploring random trees: A new tool for path planning*, Tech. report, 1998.

[45] G.S. Lee S., Sukhatme, G.J. Kim, and Chan-Mo Park, *Haptic control of a mobile robot: a user study*, Proc. of IROS 2002, 2002.

[46] V. Lepetit, F. Moreno-Noguer, and P. Fua, *EPnP: An accurate O(n) solution to the PnP problem*, International Journal of Computer Vision **81** (2009), no. 2, 155–166.

[47] Ruggiero F. Lippiello, V., *Cartesian impedance control of a UAV with a robotic arm*, 10th International IFAC Symposium on Robot Control (Dubrovnik, HR), 2012, pp. 704–709.

[48] ———, *Exploiting redundancy in cartesian impedance control of uavs equipped with a robotic arm*, IEEE/RSJ International Conference on Intelligent Robots and Systems, Oct 2012, pp. 3768–3773.

[49] V. Lippiello, G. Loianno, and B. Siciliano, *MAV indoor navigation based on a closed-form solution for absolute scale velocity estimation using optical flow and inertial data*, 50th IEEE Conference on Decision Control and European Control Conference (Orlando, FL), 2011, pp. 3566–3571.

[50] V. Lippiello and F. Ruggiero, *Cartesian impedance control of a UAV with a robotic arm*, 10th IFAC Symposium on Robot Control, Sep 2012, pp. 704–709.

[51] ———, *Exploiting redundancy in Cartesian impedance control of UAVs equipped with a robotic arm*, 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (Vilamoura, P), 2012, pp. 3768–3773.

[52] V. Lippiello and B. Siciliano, *Wall inspection control of a VTOL unmanned aerial vehicle based on a stereo optical flow*, 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (Vilamoura, P), 2012, pp. 4296–4302.

[53] F. Lizarralde and J. T. Wen, *Attitude control without angular velocity measurement: A passivity approach*, IEEE Transactions on Automatic Control **41** (1996), no. 3, 468–472.

[54] L. Lucignano, F. Cutugno, S. Rossi, and A. Finzi, *A dialogue system for multimodal human-robot interaction*, Proc. of ICMI 2013, 2013, pp. 197–204.

[55] Benallegue A. Madani, T., *Sliding mode observer and backstepping control for a quadrotor unmanned aerial vehicles*, Proceedings of the 2007 American Control Conference (New York City, NY), 2007, pp. 5887–5892.

[56] T. Madani and A. Benallegue, *Backstepping control for a quadrotor helicopter*, IEEE/RSJ International Conference on Intelligent Robots and Systems (Beijing, C), 2006, pp. 3255–3260.

[57] R. Mahony and T. Hamel, *Robust trajectory tracking for a scale model autonomous helicopter*, International Journal of Robust and Nonlinear Control **14** (2004), no. 12, 1035–1059.

[58] R. Mahony, S. Stramigioli, and J. Trumpf, *Vision based control of aerial robotic vehicles using the port Hamiltonian framework*, 50th IEEE Conference on Decision Control and European Control Conference (Orlando, FL), 2011, pp. 3526–3532.

[59] J. Malasky, L. M. Forest, A. C. Khan, and J. R. Key, *Experimental evaluation of human-machine collaborative algorithms in planning for multiple uavs*, IEEE International Conference on Systems, Man and Cybernetics, 2005, pp. 2469 – 2475.

[60] L. Marconi, F. Basile, G. Caprari, R. Carloni, P. Chiacchio, C. Huerzeler, V. Lippiello, R. Naldi, N. Janosch, B. Siciliano, S. Stramigioli, and E. Zwicker, *Aerial service robotics: The AIRobots perspective*, 2nd International Conference on Applied Robotics for the Power Industry (Zurich, CH), 2012.

[61] L. Marconi, F. Basile, G. Caprari, R. Carloni, P. Chiacchio, C. Hurzeler, V. Lippiello, R. Naldi, J. Nikolic, B. Siciliano, S. Stramigioli, and E. Zwicker, *Aerial service robotics: The AIRobots perspective*, 2nd International Conference on Applied Robotics for the Power Industry, Sept 2012, pp. 64–69.

[62] L. Marconi, L. Basile, G. Caprari, R. Carloni, P. Chiacchio, C. Huerzeler, V. LIPPIELLO, R. Naldi, N. Janosch, B. Siciliano, S. Stramigioli, and E. Zwicker, *Aerial service robotics: The airobots perspective*, 2nd International Conference on Applied Robotics for the Power Industry, 2012.

[63] L. Marconi and R. Naldi, *Control of aerial robots: Hybrid force and position feedback for a ducted fan*, IEEE Control Systems **32** (2012), no. 4, 43–65.

[64] C.G. Mayhew, R.G. Sanfelice, and A.R. Teel, *On quaternion-based attitude control and the unwinding phenomenon*, 2011 American Control Conference (San Francisco, CA, USA), 2011, pp. 299–304.

[65] I. Maza, J. Kondak, M. Bernard, and A. Ollero, *Multi-UAV cooperation and control for load transportation and deployment*, Journal of Intelligent and Robotics Systems **57** (2010), no. 1, 417–449.

[66] Lippiello V. Siciliano B. Mebarki, R., *Exploiting image moments for aerial manipulation control*, ASME Dynamic Systems and Control Conference, Oct 2013, p. V001T01A003.

[67] ———, *Image-based control for dynamically cross-coupled aerial manipulation*, IEEE/RSJ International Conference on Intelligent Robots and Systems, Sept 2014, pp. 4827–4833.

[68] R. Mebarki, J. Cacace, and V. Lippiello, *Velocity estimation of an UAV using visual and IMU data in a GPS-denied environment*, 11th IEEE International Symposium on Safety, Security, and Rescue Robotics (Linkoping,S), 2013.

[69] R. Mebarki and V. Lippiello, *Image moments-based velocity estimation of UAVs in GPS denied environments*, 12th IEEE International Symposium on Safety, Security, and Rescue Robotics (Toyako-cho, Hokkaido, J), 2014.

[70] R. Mebarki, V. Lippiello, and B. Siciliano, *Exploiting image moments for aerial manipulation control*, ASME Dynamic Systems and Control Conference (Palo Alto, CA), 2013.

[71] Rafik Mebarki and Vincenzo Lippiello, *Image-based control for aerial manipulation*, Asian Journal of Control **16** (2014), no. 3, 646–656.

[72] _____, *Image moments-based velocity estimation of uavs in gps denied environments*, IEEE International Symposium on Safety, Security, and Rescue Robotics, Oct 2014, pp. 1–6.

[73] Johannes Meyer, Alexander Sendobry, Stefan Kohlbrecher, Uwe Klingauf, and Oskar von Stryk, *Comprehensive simulation of quadrotor uavs using ros and gazeb*, 3rd Int. Conf. on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR), 2012.

[74] Vincent Montreuil, Aurlie Clodic, Maxime Ransan, and Rachid Alami, *Planning human centered robot activities*, Proc. of SMC 2007, 2007, pp. 2618–2623.

[75] R. R. Murphy, J. Casper, M. Micire, and J. Hyams, *Mixed-initiative control of multiple heterogeneous robots for urban search and rescue*, IEEE Transactions on Systems Man and Cybernetics - Part A Systems and Humans (2000), 19972003.

[76] NATO, *Atp-10 (c). manual on search and rescue. annex h of chapter 6*, 1988.

[77] NATSAR, *Australian national search and rescue manual*, Australian National Search and Rescue Council, 2011.

[78] Dana S. Nau, Hctor Muoz-Avila, Yue Cao, Amnon Lotem, and Steven Mitchell, *Total-order planning with partially ordered subtasks.*, Proc. of IJCAI 2001, 2001, pp. 425–430.

[79] K. Nonami, F. Kendoul, S. Suzuki, and W. Wang, *Autonomous flying robots. unmanned aerial vehicles and micro aerial vehicles*, Springer-Verlag, Berlin Heidelberg, D, 2010.

[80] P.Y. Oh and W.E. Green, *CQAR: Closed Quarter Aerial Robot design for reconnaissance, surveillance and target acquisition tasks in urban areas*, International Journal of Computational Intelligence **1** (2004), no. 4, 353–360.

[81] Anbal Ollero, Simon Lacroix, Luis Merino, Jeremi Gancet, Johan Wiklund, Volker Remuss, Iker Veiga Perez, Luis G. Gutirrez, Domingos Xavier Viegas, Miguel Angel Gonzlez Benitez, Anthony Mallet, Rachid Alami, Raja Chatila, Gnter Hommel, F. J. Colmenero Lechuga, Begoa C. Arrue, Joaquin Ferruz, Jos Ramiro Martinez de Dios, and Fernando Caballero, *Multiple eyes in the skies: architecture and perception issues in the comets unmanned air vehicles project.*, IEEE Robot. Automat. Mag. **12** (2005), no. 2, 46–57.

[82] S. Omari, M.-D. Hua, G. Ducard, and T. Hamel, *Nonlinear control of VTOL UAVs incorporating flapping dynamics*, 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (Tokyo, J), 2013, pp. 2419–2425.

[83] M. Orsag, C.M. Korpela, and P.Y. Oh, *Modeling and control of MM-UAV: Mobile manipulating unmanned aerial vehicle*, Journal of Intelligent and Robotic Systems **69** (2013), no. 1–4, 227–240.

[84] C. Ott, *Cartesian Impedance Control of Redundant and Flexible-Joint Robots*, Springer Tracts in Advanced Robotics, vol. 49, Springer-Verlag, Berlin Heidelberg, D, 2008.

[85] I. Palunko, P. Cruz, and R. Fierro, *Agile load transportation. Safe and efficient load manipulation with aerial robots*, Robotics and Automation Magazine **19** (2012), no. 3, 69–79.

[86] P. Pounds, D. Bersak, and A. Dollar, *Grasping from the air: Hovering capture and load stability*, 2011 IEEE International Conference on Robotics and Automation (Shanghai, CN), 2011, pp. 2491–2498.

[87] A. S. Rao and M. P. Georgeff, *Deliberation and its role in the formation of intentions*, Proc. of UAI, 1991, pp. 300–307.

[88] A. Roberts and A. Tayebi, *Adaptive position tracking of VTOL UAVs*, IEEE Transactions on Robotics **27** (2011), no. 1, 129–142.

[89] S. Rossi, E. Leone, M. Fiore, A. Finzi, and F. Cutugno, *An extensible architecture for robust multimodal human-robot communication*, Proc. of IROS 2013, 2013, pp. 2208–2213.

[90] F. Ruggiero, J. Cacace, H. Sadeghian, and V. Lippiello, *Impedance control of VToL UAVs with a momentum-based external generalized forces estimator*, 2014 IEEE International Conference on Robotics and Automation (Hong Kong, C), 2014, pp. 2093–2099.

[91] F. Ruggiero, M.A. Trujillo, R. Cano, H. Ascorbe, A. Viguria, C. Peréz, V. Lippiello, A. Ollero, and B. Siciliano, *A multilayer control for multirotor UAVs equipped with a servo robot arm*, 2015 IEEE International Conference on Robotics and Automation (Seattle, WA, USA), 2015, (accepted for publication).

[92] D. Santamaria, F. Alarcon, A. Jimenz, A. Viguria, M. Béjar, and A. Ollero, *Model-based design, development and validation for UAS critical software*, Journal of Intelligent & Robotic Systems **65** (2012), no. 1–4, 103–114.

[93] A. Santamaria-Navarro, V. Lippiello, and J. Andrade-Cetto, *Task priority control for aerial surveillance*, 12th IEEE International Symposium on Safety, Security, and Rescue Robotics, Oct 2014, pp. 1–6.

[94] B. Sellner, F. Heger, L. Hiatt, R. Simmons, and S. Singh, *Coordinated multi-agent teams and sliding autonomy for large-scale assembly*, Proceedings of the IEEE **94** (2006), no. 7.

[95] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*, Springer, London, UK, 2008.

[96] J.J. Slotine and W. Li, *On the adaptive control of robot manipulators*, International Journal of Robotics Research **6** (1987), no. 3, 49–59.

[97] L.D. Stone and J.D. Kettelle, *Theory of optimal search*, Topics in Operations Research Series, I N F O R M S: Institute for Operations Research & the Management Sciences, 1989.

[98] K. Suseong, C. Seungwon, and H.J. Kim, *Aerial manipulation using a quadrotor with a two DOF robotic arm*, 2013 IEEE International Conference On Intelligent Robots and Systems (Tokyo, J), 2013, pp. 4990–4995.

[99] A. Tayebi, *Unit quaternion-based output feedback for the attitude tracking problem*, IEEE Transactions on Automatic Control **52** (2008), no. 6, 1516–1520.

[100] A. Tayebi and S. McGilvray, *Attitude stabilization of a VTOL quadrotor aircraft*, IEEE Transactions on Control Systems Technology **14** (2006), no. 3, 562–571.

[101] P. Tsiotras, *Further passivity results for the attitude control problem*, IEEE Transactions on Automatic Control **43** (1998), no. 11, 1597–1600.

[102] J.T.-Y. Wen and K. Kreutz-Delgado, *The attitude control problem*, IEEE Transactions on Automatic Control **36** (1991), no. 10, 1148–1162.

[103] Hyunsoo Yang and Dongjun Lee, *Dynamics and control of quadrotor with robotic manipulator*, 2014 IEEE International Conference on Robotics and Automation (Hong Kong, C), 2014, pp. 5544–5549.

[104] B. Yüksel, C. Secchi, H. Bülthoff, and A. Franchi, *A nonlinear force observer for quadrotors and application to physical interactive tasks*, 2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (Besançon, F), 2014, pp. 433–440.

[105] Secchi C. Bülthoff H. Yüksel, B. and A. Franchi, *Reshaping the physical properties of a quadrotor through IDA-PBC and its application to aerial physical interaction*, 2014 IEEE International Conference on Robotics and Automation (Honk Kong, C), 2014, pp. 6258–6265.

[106] EU Collaborative Project ICT-248669, "AIRobots", www.airobots.eu.

[107] J. Cacace, A. Finzi, V. Lippiello, G. Loianno, D. Sanzone, "Aerial Service Vehicles for Industrial Inspection: Task Decomposition and Plan Execution", 26th Int. Conf. on Industrial, Engineering and other Applications of Applied Intelligent Systems, 2014.

[108] J. Cacace, A. Finzi, V. Lippiello, G. Loianno, D. Sanzone, "Aerial Service Vehicles for Industrial Inspection: Task Decomposition and Plan Execution", Applied Intelligence: Advances in Applied Artificial Intelligence, pp 302-311, 2014.

[109] B. Hannaford, L. Wood, D.A. McAffee, H. Zak, "Performance evaluation of a six-axis generalized force-reflecting teleoperator", IEEE Trans. on Systems Man and Cybernetics, vol. 21, n. 3, pp. 620-633, 1911.

[110] S. Lee, G.S. Sukhatme, G.J. Kim, C.-M. Park, "Haptic control of a mobile robot: a user study", 2002 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 2867-2874, 2002.

[111] I. Elhajj, N. Xi, Y. H. Liu, "Real-time control of internet based teleoperation with force reflection", 2000 IEEE Int. Conf. on Robot. and Autom., pp. 3284-3289, 2000.

[112] D. Xiao, R. Hubbold, "Navigation guided by artificial force fields", Proc. of the SIGCHI Conf. on Human Factors in Computing Systems, pp. 179-186, 1998.

[113] T. M. Lam, M. Mulder, M. M. (Ren) van Paassen, "Haptic Feedback for UAV Tele-operation - Force offset and spring load modification", IEEE Int. Conf. on Systems, Man and Cybernetics, pp. 1618-1623, 2006.

[114] D. Lee, A. Franchi, P. R. Giordano, H. I. Son, H. H. Bulthoff, "Haptic Teleoperation of Multiple Unmanned Aerial Vehicles over the Internet", 2011 IEEE Int. Conf. on Robot. and Autom., pp. 1341-1347, 2011.

[115] R. Carloni, V. Lippiello, M. D'Auria, M. Fumagalli, A.Y. Mersha, S. Stramigioli, B. Siciliano, "Robot Vision: Obstacle-Avoidance Techniques for Unmanned Aerial Vehicles", IEEE Robot. & Autom. Magazine, vol 20, n. 4, pp. 22-31, 2013.

[116] J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf, O. Von Stryk, "Comprehensive Simulation of Quadrotor UAVs using ROS and Gazebo", Third Int. Conf., SIMPAR 2012, pp. 400-411, 2012.

[117] A. S. Rao, M. P. Georgeff, "Deliberation and its Role in the Formation of Intentions", Proc. of the 7th Conf. on Uncertainty in Artificial Intelligence, pp. 300-307, 1991.

[118] S. M. Lavalle, "Rapidly-Exploring Random Trees: A New Tool for Path Planning", Computer Science Dept., Iowa State University, 1998.

[119] F. Ruggiero, J. Cacace, H. Sadeghian, V. Lippiello, "Impedance Control of VToL UAVs with a Momentum-based External Generalized Forces Estimator", 2014 IEEE Int. Conf. on Robot. and Autom., pp. 2093-2099, 2014.

[120] A. Finzi, A. Orlandini, "Human-Robot Interaction Through Mixed-Initiative Planning for Rescue and Search Rovers", 2005 Proc. of the 9th conf. on Advances in Artificial Intelligence, pp 483-494.

[121] EU Collaborative Project ICT-248669, "AIRobots", *www.airobots.eu*.

[122] J. Allen and G. Ferguson, "Human-Machine Collaborative Planning", *NASA Workshop on Planning and Scheduling for Space*, 2002.

[123] H. Bay, A. Ess, T. Tuytelaars and L. Van Gool, "Speeded-up robust features (SURF)", *Computer Vision and Image Understanding*, vol.110 no. 3, pp. 346–359, 2008.

[124] M. Bloesch, S. Weiss, D. Scaramuzza, and R. Siegwart, " Vision Based MAV Navigation in Unknown and Unstructured Environments", *ICRA*, pp. 21–28, 2010.

[125] P. Doherty, J. Kvarnström and H. Fredrik,"A temporal logic-based planning and execution monitoring framework for unmanned aircraft systems", *AAMAS*, pp. 332–377, 2009.

[126] A. Geiger, J. Ziegler and C. Stiller, "StereoScan: Dense 3d Reconstruction in Real-time", *IEEE Intelligent Vehicles Symposium*, pp. 963–968, 2011.

[127] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, 2nd ed. Cambridge U.K.: Cambridge Univ. Press, 2004.

[128] S. Hrabar," Vision-Based 3D Navigation for an Autonomous Helicopter", *Ph.D. Dissertation, University of S. California*, 2006.

[129] F. Ingrand, M. P. Georgeff and A. S. Rao, "An architecture for Real-Time Reasoning and System Control", *IEEE Expert: Intelligent Systems and Their Applications*, pp. 34-44, 1992.

[130] S. M. Lavalle,"Rapidly-Exploring Random Trees: A New Tool for Path Planning", *Computer Science Dept., Iowa State University, Tech. Rep*, 1998.

[131] S. E. Macfarlane and E. A. Croft,"Jerk-bounded manipulator trajectory planning: design for real-time applications", *IEEE Transactions on Robotics*, vol. 19, pp.42–52, 2003.

[132] M. Morales, L. Tapia, R. Pearce, S. Rodriguez and N. Amato, "A machine learning approach for feature sensitive motion planning", *Int. Workshop on the Algorithmic Foundations of Robotics*, 2004.

[133] S. Holmes, G. Klein and D.W. Murray, "A square root unscented Kalman filter for visual monoSLAM", *ICRA*, pp. 3710–3716, 2008.

[134] R. Naldi, M. Marconi, and L. Gentili, "Modelling and control of a flying robot interacting with the environment", Journal of IFAC, 47(12):2571–2583, 2011

[135] A. S. Rao and M. P. Georgeff,"Deliberation and its Role in the Formation of Intentions", *UAI*, pp. 300–307, 1991.

[136] Robotics Operating System, "ROS", *www.ros.org*.

[137] A. Stentz," Optimal and efficient path planning for unknown and dynamic environments", *Int. J. of Robotics and Automation*, 10(3):89–100, 1995.

[138] L. Marconi, C. Melchiorri, M. Beetz, D. Pangercic, R. Siegwart, S. Leutenegger, R. Carloni, S. Stramigioli, H. Bruyninckx, P. Doherty, A. Kleiner, V. Lippiello, A. Finzi, B. Siciliano, A. Sala, and N. Tomatis, "The sherpa project: smart collaboration between humans and ground-aerial robots for improving rescuing activities in alpine environments", in *IEEE Int. Symp. on Safety, Security, and Rescue Robotics (SSRR), College Station*, Texas, USA, Nov. 5-8 2012.

[139] C. Breazeal, C.D. Kidd, A. L Thomaz, G. Hoffman, M. Berlin, "Effects of nonverbal communication on efficiency and robustness in human-robot teamwork ", in the *Conf. on Int. Robots and Systems, 2005*.

[140] J. Cacace, A. Finzi , V. Lippiello, G. Loianno, D. Sanzone, "Aerial service vehicles for industrial inspection: task decomposition and plan execution", *Applied Intelligence January 2015, Volume 42, Issue 1, pp 49-62*

[141] V. Lippiello, B. Siciliano, "Wall inspection control of a VTOL unmanned aerial vehicle based on a stereo optical flow" *Int. Conf. on Int. Robots and Systems, Vilamoura 2012*

[142] S. Waharte, N. Trigoni, "Supporting Search and Rescue Operations with UAVs", *Int. Conf. on Emerging Security Technologies EST 2010*.

[143] M.A Goodrich, J. L. Cooper, J.A Adams, C. Humphrey, R. Zeeman, B. G. Buss, "Using a Mini-UAV to Support Wilderness Search and Rescue: Practices for Human-Robot Teaming", *Int. Workshop on Safety, Security and Rescue Robotics, 2007*.

[144] G. Bevacqua, J. Cacace, A. Finzi, V. Lippiello, "Mixed-Initiative Planning and Execution for Multiple Drones in Search and Rescue Missions", *Int. Conf. on Automated Planning and Scheduling, Jerusalem 2015*.

[145] J. McLurkin, J. Smith, J. Frankel, D. Sotkowitz, D. Blau, B. Schmidt, "Speaking Swarmish: Human-Robot Interface Design for Large Swarms of Autonomous Mobile Robots", *AAAI Spring Symposium, March 28, 200*

[146] A. Couture-Beil, R. T. Vaughan, G. Mori "Selecting and Commanding Individual Robots in a Multi-Robot System", *Canadian Conf. on Computer and Robot Vision 2010*.

[147] G. A. Korsah, A. Stentz, M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation", *Int. Journal of Robotics Research October 2013 vol. 32 no. 12 1495-1512*

[148] N. Friedman, D. Geiger, M. Goldszmidt, "Bayesian Network Classifiers", Machine Learning Volume 29 Issue 2-3, Nov./Dec. 131-163 1997.

[149] K. P. Murphy, "Machine Learning: A Probabilistic Perspective", *The MIT Press, 2012*.

[150] O.C. Schrempf, U.D. Hanebeck, A.J. Schmid, H. Worn, "A novel approach to proactive human-robot cooperation", *Int. Workshop on Robot and Human Interactive Communication, 2005*

[151] D. W. Albrecht, I. Zukerman, A. E. Nicholson, "Bayesian Models for Keyhole Plan Recognition in an Adventure Game", *Journal User Modeling and User-Adapted Interaction Volume 8 Issue 1-2, 1998 Pages 5 - 47*

[152] J. Cacace, A. Finzi, V. Lippiello, "A Mixed-Initiative Control System for an Aerial Service Vehicle Supported by Force Feedback", *Int. Conf. on Intelligent Robots and Systems, Chicago, USA, Sep. 2014*.

[153] J. R. Cauchard, J. L. E, K. Y. Zhai, J. A. Landay, "Drone & Me: An Exploration Into Natural Human-Drone Interaction", *Int. Joint Conference on Pervasive and Ubiquitous Computing, Osaka JAP, Sep. 2015*.

[154] S. Rossi, E. Leone, M. Fiore, A. Finzi and F. Cutugno, "An extensible architecture for robust multimodal human-robot communication", *Int. Conf. on Intelligent Robots and Systems Tokyo 2013*