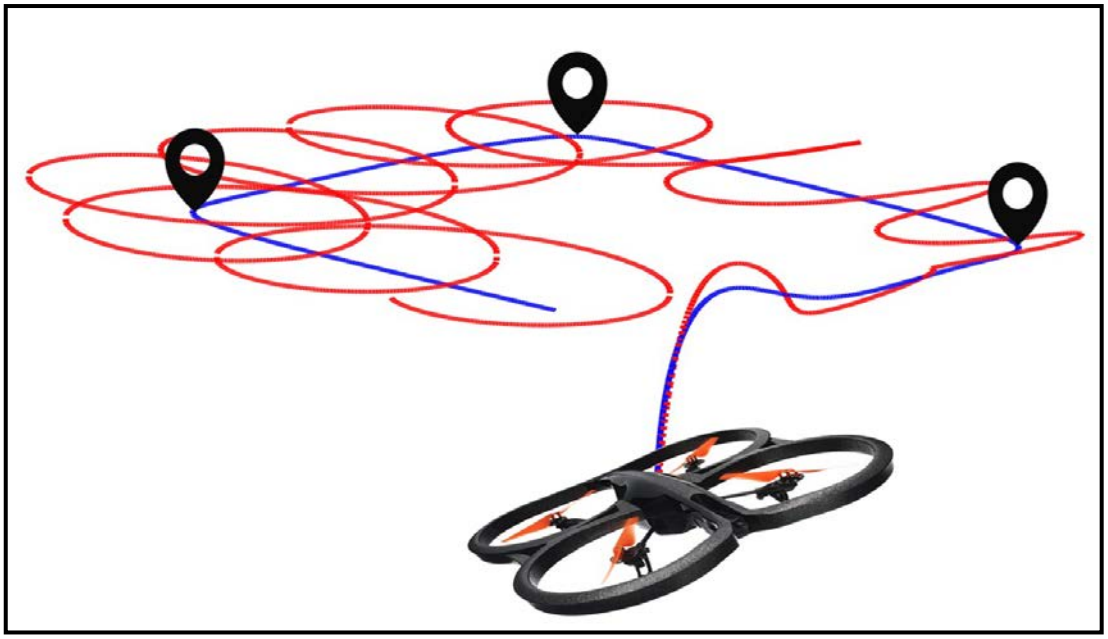
Modeling and control of aerial vehicles using teleoperation with input delay

Belem Isabel Rojas-Ramirez

Par **Belem Isabel ROJAS-RAMIREZ**

Modeling and control of aerial vehicles using teleoperation with input delay

Thèse présentée
pour l'obtention du grade
de Docteur de l'UTC



Soutenue le 4 décembre 2020
Spécialité : Automatique et Robotique : Unité de recherche Heudyasic (UMR-7253)

D2568

Université de Technologie de Compiègne
Laboratoire Heudiasyc UMR CNRS/UTC 7253

Doctoral School **École doctorale de l'UTC**

University Department **Laboratoire Heudiasyc UMR CNRS/UTC 7253**

Thesis defended by **Belem Isabel ROJAS-RAMIREZ**

Defended on **4th December, 2020**

In order to become Doctor from Université de Technologie de Compiègne

Academic Field **Automatic and Robotic**

Spécialité : Automatique et Robotique

Modeling and control of aerial vehicles using teleoperation with input delay

Thesis supervised by	Pedro CASTILLO-GARCIA	Supervisor
	Rogelio LOZANO-LEAL	Co-Supervisor

Committee members

Referees	Pedro GARCIA-GIL	Professor at UPV
	Sergio Rosario SALAZAR-CRUZ	Professor at CINVESTAV
Examiners	Veronique CHERFAOUI	Professor at UTC
	Reine TALJ KFOURY	HDR Junior Researcher at CNRS
	Claude PEGARD	Professor at Université de Picardie Jules Verne
	Juan Antonio ESCARENO-CASTRO	Associate Professor at XLIM
Supervisors	Pedro CASTILLO-GARCIA	HDR Junior Researcher at CNRS
	Rogelio LOZANO-LEAL	Senior Researcher at CNRS

Université de Technologie de Compiègne
Laboratoire Heudiasyc UMR CNRS/UTC 7253

École doctorale **École doctorale de l'UTC**

Unité de recherche **Laboratoire Heudiasyc UMR CNRS/UTC 7253**

Thèse présentée par **Belem Isabel ROJAS-RAMIREZ**

Soutenue le **4 décembre 2020**

En vue de l'obtention du grade de docteur de l'Université de Technologie de Compiègne

Discipline **Automatique et Robotique**

Modeling and control of aerial vehicles using teleoperation with input delay

Thèse dirigée par Pedro CASTILLO-GARCIA directeur
Rogelio LOZANO-LEAL co-directeur

Composition du jury

Rapporteurs	Pedro GARCIA-GIL	professeur à l'UPV
	Sergio Rosario SALAZAR-CRUZ	professeur au CINVESTAV
Examinateurs	Veronique CHERFAOUI	professeure à l'UTC
	Reine TALJ KFOURY	chargée de recherche HDR au CNRS
	Claude PEGARD	professeur à l'Université de Picardie Jules Verne
	Juan Antonio ESCARENO-CASTRO	MCF au XLIM
Directeurs de thèse	Pedro CASTILLO-GARCIA	chargé de recherche HDR au CNRS
	Rogelio LOZANO-LEAL	directeur de recherche au CNRS

Modeling and control of aerial vehicles using teleoperation with input delay

Abstract

Unmanned Aerial Vehicles (UAVs) are receiving increasing interest from industry and academia due to their wide application in search and rescue, infrastructure inspection, surveillance, among others. This thesis focuses on research in the area of teleoperation systems for quadrotor vehicles. Throughout this thesis, a teleoperation system for a quadrotor vehicle was developed. In this system, the user interface is based on a virtual telepresence approach. Control algorithms were developed and implemented within the master and slave systems.

The first part of this thesis consists of developing mathematical models of the dynamics of a quadrotor aircraft. Most works currently found in the literature for quadrotors are based on classical approaches such as Euler angles. These representations can lead to problems such as discontinuities, singularities, gimbal-locks, and highly non-linear equations. An alternative to these classical representations are unit quaternions. These have the advantages of the lack of singularities and gimbal lock effects.

The second part of this work was dedicated to the development of a quadrotor teleoperation system. This system consists of a virtual user interface in a local environment and a quadrotor in a remote environment. A User Datagram Protocol (UDP) communication was used to communicate both environments. The user manipulates a virtual drone in the local environment and a real drone follows the position and orientation references in a remote environment. The user receives virtual feedback on the states of the real vehicle in the virtual environment. Results of the implementation of the proposed teleoperation system in real time are presented.

The last part of this thesis addresses the delay problem in the teleoperation system. Delays due to system latency and the distance between environments were modeled as a delayed control input. Then, a predictor-based controller was developed in order to maintain the stability of a drone's flight. This approach was applied to the classical Euler-Lagrange model and to the quaternion-based model in order to analyze performance. Simulations of both models with delayed inputs are presented.

Keywords: modeling, control, virtual telepresence, simulation, time delay systems, input delay

Laboratoire Heudiasyc UMR CNRS/UTC 7253
Site de l'Innovation – CS 60319 - 57 avenue de Landshut – 60200 Compiègne Cedex – France

Modeling and control of aerial vehicles using teleoperation with input delay

Résumé

UAVs suscite un intérêt croissant de la part de l'industrie et des universités en raison de leur large application dans la recherche et le sauvetage, l'inspection des infrastructures, la surveillance, entre autres. Cette thèse se concentre sur la recherche dans le domaine des systèmes de téléopération pour véhicules quadrirotor. Tout au long de cette thèse, un système de téléopération pour un véhicule quadrirotor a été développé. Dans ce système, l'interface utilisateur est basée sur une approche de téléprésence virtuelle. Des algorithmes de contrôle ont été développés et mis en œuvre dans les systèmes maître et esclave.

La première partie de cette thèse consiste à développer des modèles mathématiques de la dynamique d'un avion quadrotor. La plupart des travaux actuellement trouvés dans la littérature pour les quadrotors sont basés sur des approches classiques telles que les angles d'Euler. Ces représentations peuvent conduire à des problèmes tels que des discontinuités, des singularités, des verrous à cardan et des équations hautement non linéaires. Une alternative à ces représentations classiques sont les quaternions unitaires. Ceux-ci présentent les avantages du manque de singularités et d'effets de verrouillage de la nacelle.

La deuxième partie de ce travail a été consacrée au développement d'un système de téléopération à quatre rotors. Ce système se compose d'une interface utilisateur virtuelle dans un environnement local et d'un quadrotor dans un environnement distant. Une communication UDP a été utilisée pour communiquer les deux environnements. L'utilisateur manipule un drone virtuel dans l'environnement local et un vrai drone suit les références de position et d'orientation dans un environnement distant. L'utilisateur reçoit un retour virtuel sur les états du véhicule réel dans l'environnement virtuel. Les résultats de la mise en œuvre du système de téléopération proposé en temps réel sont présentés.

La dernière partie de cette thèse aborde le problème des retards dans le système de téléopération. Les retards dus à la latence du système et à la distance entre les environnements ont été modélisés comme une entrée de commande retardée. Ensuite, un contrôleur basé sur des prédicteurs a été développé afin de maintenir la stabilité du vol d'un drone. Cette approche a été appliquée au modèle classique d'Euler-Lagrange et au modèle basé sur les quaternions afin d'analyser les performances. Des simulations des deux modèles avec des entrées retardées sont présentées.

Mots clés : modélisation, contrôle, téléprésence virtuelle, simulation, systèmes de temporisation, entrée retarde

Acknowledgements

I want to give thank you to the institutions that supported this work from the beginning. To the *Consejo Nacional de Ciencia y Tecnología (CONACYT)* for the scholarship program that supported me. I want to also thank the *Heudiasyc (Heuristique et DIAgnostique des SYstemes Complexes)* laboratory and the Ecole Doctorale Sciences pour l'Ingenieur at the Université de Technologie de Compiègne (UTC), that made this work possible with their PhD programs. Also to the LABEX MS2T and ROBOTEX projects, that provided the laboratory with funding for material and equipment that was essential to the development of my PhD.

I want to specially thank my PhD directors Pedro Castillo García, Chargé de Recherches CNRS, and Rogelio Lozano Leal, Directeur de Recherches CNRS, for their invaluable advice, support, and counseling in this project from its beginning to its ending. I also want to thank Professor Indira Thouvenin at the Université de Technologie de Compiègne for her support and advice at the beginning of this project.

I want to expressly thank Professor Pedro García Gil at the Departamento de Ingeniería de Sistemas y Automática from the Universitat Politècnica de València, and Professor Sergio Salazar Cruz at Centro de Investigación y de Estudios Avanzados del IPN, for accepting to be referees of my work. Their notations and advise were very appreciated to improve this thesis.

I also want to thank Véronique Cherfaoui, Professeur des Universités at the Université de Technologie de Compiègne; Reine Talj, Chargée de Recherches CNRS; Claude Pegard, Professor at Université de Picardie Jules Verne, and Juan Escareno Castro, Maître de conférence at Institut de recherche XLIM from the Université de Limoges Recherche for accepting to be part of my jury and giving very helpful commentaries.

Thanks to my colleagues Ángel Alatorre and Hernán Abaunza for guiding me to some topics to develop this project. I also thanks to my colleagues Julio Betancourt, Alexis Offerman and Cristino de Souza for sharing moments and experiences.

Acronyms

A

AR Augmented Reality. xi, 11–13

D

DoF degrees of freedom. 21, 27, 48

F

FDEs Functional Differential Equations. 16, 65
FL-AIR Framework Libre Air. 56, 60
FOV Field of View. 11, 12
FPV First Person View. 11
FSA Finite Spectrum Assignment. 24
FTC Fundamental Theorem of Calculus. 65, 73, 75, 90, 91, 100

G

GPS Global Positioning System. 5, 6
GUI Graphical User Interface. 7

H

HMD Head-Mounted Display. 11

I

IMU Inertial Measurement Unit. 60
IP Internet Protocol. 60

L

LoS Line-of-Sight. 2, 3, 10, 12, 13
LTI Linear Time-Invariant. 24, 25

M

MPC Model Predictive Control. 21
MR Mixed Reality. 11–13

N

NUIs Natural User Interfaces. 10, 12

O

ODEs Ordinary Differential Equations. 65, 67, 68

P

PDE Partial Differential Equations. 24
PID Proportional Integral Derivative. 51, 52

R

ROS Robot Operating System. 10, 14
ROV's Remote Operated Vehicles. 5

S

SaR Search and Rescue. 1, 2, 8

T

TCP Transmission Control Protocol. 22
TDSs Time-Delay Systems. 65, 67, 68

U

UAV Unmanned Aerial Vehicle. 1, 2, 8, 9, 11, 12, 33, 50, 56
UAVs Unmanned Aerial Vehicles. v, vi, xiii, 2, 5, 6, 8, 25, 45, 47, 48, 101
UCD User-Centered Design. xi, 10
UDP User Datagram Protocol. v, vi, 22, 47, 56, 60
UGV's Unmmanned Ground Vehicles. 5

V

VR Virtual Reality. 11–13

Contents

List of Figures

Outline of the thesis

This thesis contains five chapters organized as follows.

Chapter 1 describes teleoperation systems in general and the main derived problems. Also, the research problem is limited to the development of a remote operation interface of a quadrotor and the time delay in the system. The state of the art of these two problems related to UAVs is presented.

Chapter 2 introduces the classical models of quadrotors: Euler-Lagrange and Newton-Euler where the rotational dynamics are modeled using the Euler angles. A model based on unit quaternions is also described that eliminates the gimbal lock problem. A status feedback controller for this model is developed.

Chapter 3 describes the development of a remote operating interface of a quadrotor based on a virtual environment. The virtual and real aerial vehicle controllers are described. Also a mapping of coordinates of position and orientation of the virtual environment is presented. Experimental results in real-time are shown.

Chapter 4 describes the challenges of systems with delay compared to systems without delay. A controller based on the fundamental theorem of calculus is developed. This controller is applied to the Euler-Lagrange model. It is then applied to the quaternion-based model in order to observe its performance. Simulations of these applications are shown.

Chapter 5 presents some conclusions derived from the development of this thesis. Some problems that could be addressed in the future are also suggested.

1 Introduction

The topic of teleoperation on aerial vehicles has been important for many researchers in the last years. Teleoperation systems consist of an air vehicle in a remote environment, a human operator at a local site or workstation, and a communication channel to transmit commands and signals. The idea is for the air vehicle to follow the movements dictated by the human operator through a master device.

Nowadays, it is relatively easy and cheap to acquire a quadrotor aircraft. The versatility of these aircrafts has fueled the interest of researchers for the development of novel teleoperation interfaces. The design of those interfaces are dedicated to facilitating drone piloting in different applications: exploration missions, Search and Rescue (SaR), handling of hazardous materials, etc., ensuring the success of the task and without putting the human operator at risk. Also, technological developments such as gestures tracking, haptic and virtual devices have allowed an improvement of the teleoperation systems for aerial robots in the sense of telepresence.

Unmanned Aerial Vehicle (UAV) teleoperation system has the main objective of reducing or eliminating the negative effects of dangerous environments, keeping human life safe. It is also important to reduce the cognitive load of the human pilot and ensure the success of the task. However, several questions arise around a teleoperation system. First, the design of an operator interface that is easy to use and intuitive. Second, ensure the stability of the air robot. Third, to solve the problem of time delay inherent in the teleoperation system. Therefore, a challenge of a teleoperation system is to design an easy-to-use interface for any user and to maintain the stability of the robotic system against delays.

1.1 Problem statement

In some UAV applications, such as SaR or exploring missions in unknown and dangerous environments where human lives depend on the success of the mission, it is necessary to keep the human pilot in the control-loop [1–3]. For this reason, the development of teleoperation systems for aerial vehicles is an area of research in development. These systems have two main requirements that guarantee a successful operation: stability and transparency [4–6]. The stability of the system must be independent of human behavior and is linked to the type of robot. Transparency is related to the design of the interface so that the operator has the feeling of being in the remote environment. Therefore, the teleoperation problem could be divided into three main axis.

The first axis concerns the development of the mathematical models that describe aerial vehicles dynamics. In previous years, multiple methodologies have been proposed and studied by many researchers. These approaches combine mechanical and physical theories such as Newton's equations of motion and the laws of energy conservation. These equations can use different mathematical descriptions of rotations and translations sequences such as Euler angles, rotation matrices, quaternions, among others.

Since the dynamics of aerial vehicles are generally unstable, complex, nonlinear, and underactuated, the design of control and navigation algorithms becomes difficult. Many works have proposed approaches that simply mathematical models to avoid the dynamic nature of the vehicle. However, this put limitations on the real capacities of UAVs. Non-conventional approaches such as using quaternions to describe the vehicle dynamics can provide mathematical simplicity without sacrificing accurateness and generality, this becomes meaningful in the second part of the problem.

The second axis is to provide a teleoperation interface of a quadrotor vehicle for an inexperienced user. This system must ensure the user's task regardless of the dynamics of the vehicle and the movements of the pilot. Also a level of telepresence should be ensured without the user having the vehicle in their Line-of-Sight (LoS). Different works have developed interfaces using on force feedback, haptic, visual, among others, based on natural user interfaces to raise the level of telepresence. However, few works study the latency of the system due to the use of feedback devices.

The third axis refers to the delay due to the latency of the system. The communication medium contributes to the complexity of the system due to its dynamic nature that could distort, delay or lose samples of the signal exchanged between the operator and the teleoperation system. It may quickly result in significant damage or destruction of the UAV.

1.1.1 Objectives

This thesis is devoted to the development of a teleoperation system of aerial vehicles, more specifically quadrotors, with the goal of helping users to control a quadrotor. To achieve this goal and to the deal with the challenges involved, three objectives are raised.

The first one is to develop control algorithms to ensure stability of the vehicle. Maintaining the stability of the aircraft against the behaviors of an operator without flight experience is an important characteristic when integrating the vehicle into a teleoperation system.

The second objective is to propose a novel solution in the design of operator interface. This must provide feedback to the pilot about the information of the system states during the flight, to make decisions, or change the mission. An important feature is that when the user is piloting the vehicle it is not in his LoS.

Finally, the third objective is to design a controller that mitigates the effects of time delay. This ensures stability of drone flight irrespective of the delay in data transmission.

1.1.2 Methodology

The vehicle's stability is ensured using a quaternion-based approach. Unlike conventional approaches, this describes vehicle rotation using unit quaternions; that is, a single entity describes the vehicle's attitude information. This approach provides a simple calculation of the controller, but one of its most essential features is avoiding discontinuities and gimbal lock problems. This controller can help to give robustness to the drone flight.

The controller of the aerial robot has as references the position and orientation of the virtual vehicle. Considering that an inexperienced user will use this interface, an interface is built by introducing a virtual telepresence approach. The control input device used is a standard joystick. The virtual environment is built with the remote environment's measurements and characteristics. Visual feedback from the remote vehicle is displayed in the virtual environment using a virtual object.

The delay problem due to the system's latency and the communication channel is modeled as a delayed control input where the delay is known. A predictor-based approach is applied using states and past inputs. This approach is applied to two models of a quadrotor for the analysis of its performance.

1.2 State of the art

In this subsection, the description of a teleoperation system will be introduced. Two characteristics that will be studied in this subsection are the interface design and the time delay. The background of these and some related works will be presented.

1.2.1 Teleoperation

Teleoperation could be defined as a "remote operation" or manipulation of a system in order to accomplish some task. "Remote" means that the controlled system is separated from the operator by a distance [4, 5]. This implies that there is no direct or visual contact from the operator with the controlled system. Then, a teleoperated system is composed by two main components: a local site (master/operator) and a remote site (slave) connected through a communication channel, see Figure 1.1.

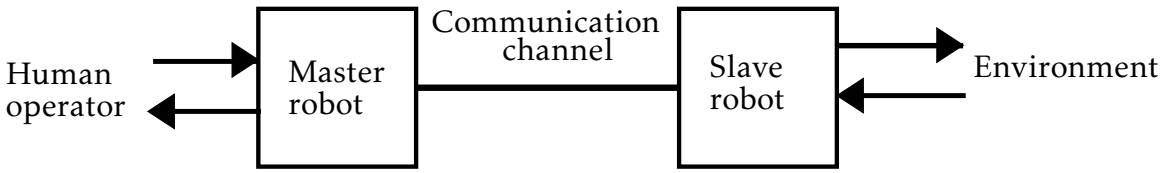


Figure 1.1 – Teleoperation framework.

Teleoperation extends the human capability to manipulate objects remotely by providing the operator with similar conditions as those at the remote location [6]. One of the first remote control systems was developed by Nikolas Tesla in 1898 in New York City, this system consisted of controlling a robotic boat using radio waves [7]. However, the first master-slave telerobotic system was developed by Ray Goertz in 1948 at the Argonne National Laboratory, where the first nuclear reactor was developed [8]. The system designed by Goertz consisted of an operator using mechanical pantographs at a local site and nuclear radioactive material placed in a "hot cell" at a remote site. These sites were separated by a protective window and/or mirrors so that the operator maintained visual contact with the radioctive material, see Figure 1.2.

The mechanical manipulators were replaced by electromechanical servos and in 1954 Goertz's team developed the first electromechanical manipulator with feedback servo control. After the decline of the nuclear industries, where teleoperation was essential to manipulate radioactive materials [9, 10], the advantages of teleoperation techniques began to extended to new areas.

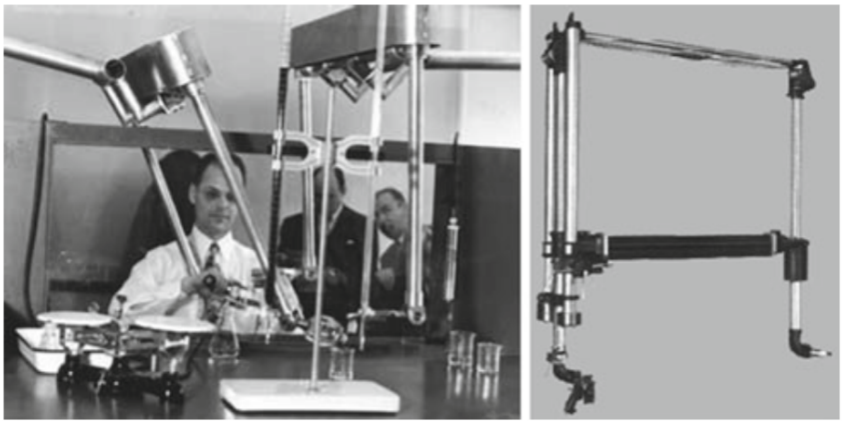


Figure 1.2 – Master-slave manipulator developed in 1948 by Ray Goertz.

The benefits of teleoperation were extended to deep-sea exploration in the 1960s because it represents a hostile environment for human operators. Then, deep-sea operations began with teleoperated submarines, frequently equipped with telemanipulators to perform underwater work tasks. These submarines are commonly called Remote Operated Vehicles (ROV's) although this expression can also be used for ground or flying vehicles. The development and implementation of teleoperation systems to operate unmanned underwater vehicles [11–16] at great distances in deep-sea led to the first teleoperation in outer-space. Particularly, space explorations require many resources to physically transfer human operators, therefore teleoperating space robots [17–24] is the most efficient solution. Teleoperation has also helped reduce medical invasions in surgeries, commonly called telesurgery [25, 26].

Teleoperation has recently become important in the military and security fields. Commonly tasks in these fields such as: recognition, information gathering, land-mine detection, route clearing, etc., represent a high risk for human operators. In order to avoid the loss of human lives teleoperated vehicles have been developed. Many kinds of land and air vehicles have been built for different types of environments. Military Unmmanned Ground Vehicles (UGV's), as SARGE model, are developed and equipped with vehicle localization (Global Positioning System (GPS), inertial navigation) and supervisory control to improve the performance. Frequently these vehicles are equipped with last generation teleoperation devices (such as stereo-vision or telepresence) to provide the best response in fast and dangerous missions.

UAVs have also a wide application field in military operations for example RQ1 predator in Figure 1.3, and X47B of US, Sharp Sword and Dark Sword of China.

Figure 1.3 – The RQ-1 Predator unmanned aircraft.

Figure 1.4 – RQ-1 Predator ground control station.

In most cases, the UAVs are remotely piloted by radio control or by satellite links. However, one of the advantages of this kind of aircraft is its ability to fly autonomously with the help of GPS and inertial navigation.

In general, the main objective of the application of teleoperated systems is to help eliminate or reduce the negative effects in hazardous environments such as: hostile environments, with adverse effects on human operators or inaccessible to them.

Despite the different telerobotics applications, the scientific community is currently investigating how to design teleoperation systems to make the execution of tasks more flexible and comfortable for the human operators with the best cost-effective.

Some issues about the design of teleoperation systems [1, 27, 28] are related to:

- **Robot:** a machine that receives the control signals using communication hardware. This robot will be integrated with actuators and sensors, depending on its applications and operating environments.
- **Communication channel:** a communication link is used to transmit sensors and control signals. Many communication channels can be used in teleoperation; some are wireless, radio waves, network lines, and the Internet. These can be chosen according to the environment and the application. However, the criteria to select them are bandwidth and latency.
- **Operator interface:** The human operator, at the local site, controls the robot from a workstation. The interface is usually multi-modal, consisting of at least a display visualizing (Graphical User Interface (GUI)) for the video from the robot's on-board camera(s) and other sensor or status information. Depending on the level of telepresence, currently other devices are helping the user to see or perceive information from the remote environment, for example: haptic joysticks, cyber-gloves, head-mounted displays, etc. Besides, the interface will require input devices to allow the operator to enter commands (via keyboard), or execute manual control of the robot (via joystick) These devices provide information from the remote site and sensor to encode the human control actions.
- **Time-delay problems:** In any electromechanical system the time delay is present. In most cases, it is imperceptible, but in others, it can make the system unstable. When an electromechanical system is part of a teleoperation, the time-delay problem increases due to bandwidth and latency by the communication channel or delays of the devices in the workstation. Therefore, a problem in the design of a teleoperation system can be the time-delay on the system.

From a control-theoretic point of view the main goal of teleoperation can be presented in two aspects [4–6]:

- **Stability:** maintain stability of the closed-loop system irrespectively of the behavior of the human operator or the environmental perturbations. Stability in teleoperation is not only related to the robot but also to the inherent delays of the system due to the delay in data transmission and the response of the human operator.

- **Transparency:** the human operator should feel as if she/he was present at the remote site. This means that the dynamic of the master and slave are canceled out or the equality of velocities and forces.

Satisfying these requirements, in a teleoperation system, extends human capabilities (manipulating objects or performing delicate tasks) by projecting their expertise to distant locations that are life-threatening.

Teleoperation is divided into two branches.

- **Unilateral teleoperation.** The human operator is independent of the whole system, and the operator impedance can not affect the system performance. This means that, the operator transmits the master motion and/or force to the slave site without feedback information from slave to master.

- **Bilateral teleoperation.** Includes the motion and/or force information transmissions from the slave site to the master site.

Unilateral teleoperation is easier and reliable to implement than the bilateral one. The idea in unilateral teleoperation is to make the slave track the human movements.

The main factors to consider in the design of a teleoperated system have been previously presented. They depend on the environment and the type of robot that will be used. One of the objectives of this research is the design of a teleoperation system for a UAV. Therefore, work related to teleoperation systems of drones and challenges are described below.

1.2.2 Teleoperation of unmanned aerial vehicles

The first developments of UAVs were for missions during the First World War. These vehicles were mainly conducted in military projects aiming to perform missions without risking human lives [2, 3]. The availability of fast, small computers, light-weight constructions and materials in the past ten years has lead to an enormous increase in the interest in developing UAVs for the commercial market. Currently, they are used with a wide variety of applications: border monitoring [29], plant assents inspection [30], forest fire monitoring [31], sensing of agricultural products [32], and SaR [33–35].

Drone teleoperation, commonly employed interfaces, can represent a non-intuitive and sophisticated control device for beginner users.

Several solutions have been proposed to reduce de cognitive effort and high concentration of the drone users [36]. Mainly teleoperation, based on haptic joysticks, has been employed to increase operational performance. These haptic devices are used to provide the human pilot with a sense of surrounding obstacles to avoid them [37–44]. For example, in [41] an artificial force field is proposed and evaluated. This field is configurable and assigns the constraints of the environment to haptic repulsive forces thereby avoiding collisions. In [38] the authors proposed a control strategy for changing the reference velocity on-line in a neighborhood of obstacles in order to avoid them. In [39], a UAV teleoperation scheme based on haptic feedback force was presented. This force-feedback gives the user the sensation of touching a rigid surface as if it were positioned above the reference position of the vehicle. In [37], a framework of haptic teleoperation algorithms for aerial vehicles was proposed. This framework was based on energy and concepts of network theory and port-Hamiltonian systems. Under this scheme, the user was provided with a force-feedback that corresponds to the speed of the aerial robot given by a local vision loop or a gust of wind. Haptic interfaces are also used to help operators control swarms of quadrotors, in [43] a method based on the notions of virtual structure and potential fields of multiple layers was applied allowing to avoid collisions within the swarm and with fixed obstacles, see Figure 1.5.

In most cases, the interfaces using a haptic joystick are also provided with one or more displays showing the flight of the drone from a camera located in the environment, or with images from the on-board camera. Thus, the user has tactile and visual feedback.

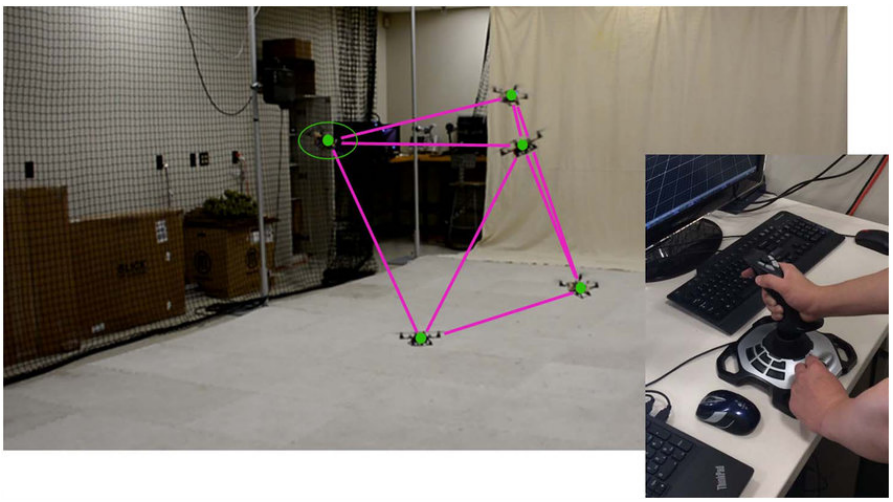


Figure 1.5 – Haptic joystick used in the mini-UAV swarm.

Recent technological advances have led to the development of interfaces with a UCD applied to Natural User Interfaces (NUIs). Different works have attempted to use innate human characteristics, such as voice, gestures, and vision, to interact with robots, particularly with aerial robots [45]. Some of the technologies that have allowed to leave the traditional devices (mouse, keyboard, remote controllers) are the devices of gesture recognition, eye tracking [46], voice recognition [47, 48], among others. Some devices used for the identification of gestures are the Microsoft Kinect and the Leap Motion Sensor, when the gestures are identified they are related to discrete control commands that are sent to the drone to teleoperate it, see Figure 1.6. For speech recognition a microphone is frequently used, in [45] an interface detects voice commands using Robot Operating System (ROS) software with the Pocketsphinx library and maps them to discrete control inputs, see Figure 1.7. Although these interfaces may be more intuitive, in most cases, these types of controls require the aircraft to be in the pilot's LoS to recognize user gestures or for the user to observe the drone's flight to perform a task. A restriction of the interactions that map the inputs of the control devices (such as gestures, voice, visual markers, etc.) to a set of control inputs of a quadrotor aircraft (take-off, land, rotate, go right, etc.) could be the lack of precise control of the tracking of its trajectory.

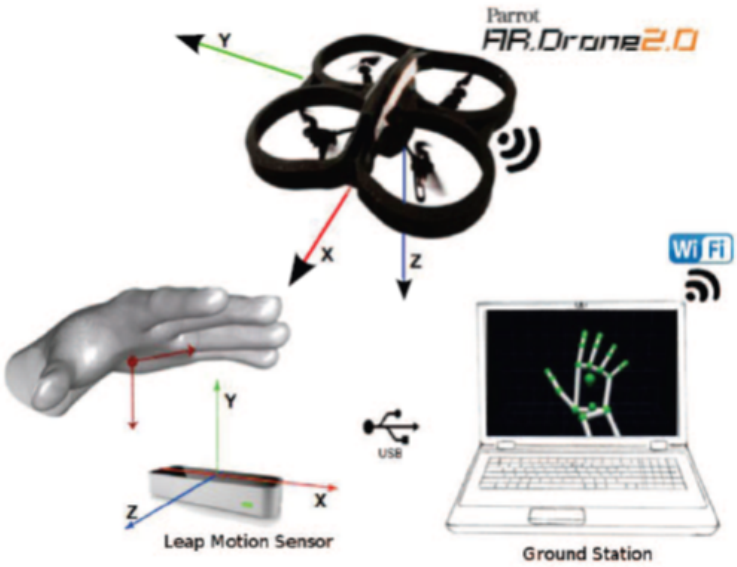


Figure 1.6 – A user controls a drone using hand gestures which are identified by a Leap Motion device [45].

Figure 1.7 – A human operator gives voice commands which are recorded by a microphone and mapped to control commands to a quadrotor [45].

Recent advances in the field of Virtual Reality (VR) have helped to generate sophisticated telepresence systems to pilot drones. These interfaces use VR devices to increase the transparency of the system. Generally, these systems involve visual feedback of First Person View (FPV); this means that, the human pilot observes the images of the on-board camera within its Field of View (FOV) range. Then, the user can have the feeling of being inside the UAV. The devices used are Head-Mounted Display (HMD) such as Oculus Rift and HTC Vive. These interfaces are commonly based on the positions of the pilot (hands, body, and head) [49–51], see Figure 1.8. A disadvantage of this kind of interface is that they can degrade situational awareness of the environment for lack of a third-person perspective. It means that the user could have difficulty to identify obstacles and other objects around the limited FOV of the on-board camera. Despite the possibility that the user can explore their environment to avoid obstacles, depending on the way of interaction, this could affect the mission. In long-term tasks, these position-based interfaces can cause operator fatigue from the use of their body and loss of awareness of real-world [52–54].

In order to reduce the impact of FPV issues in a VR environment, some interfaces have used AR and Mixed Reality (MR). AR technology involves overlays computer graphics onto real-world environments in real-time [53, 55, 56].

The three main characteristics of these interfaces are [57]:

1. In a combined scene, the user can see real and virtual objects.
2. The user can perceive how virtual objects are directly embedded in the real-world.
3. The virtual objects can be interacted with it in real-time.

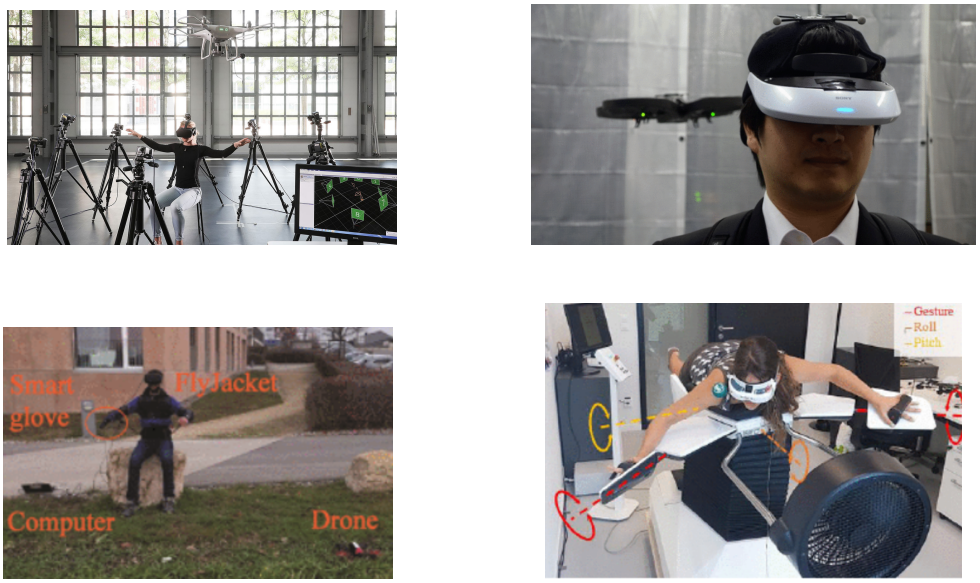
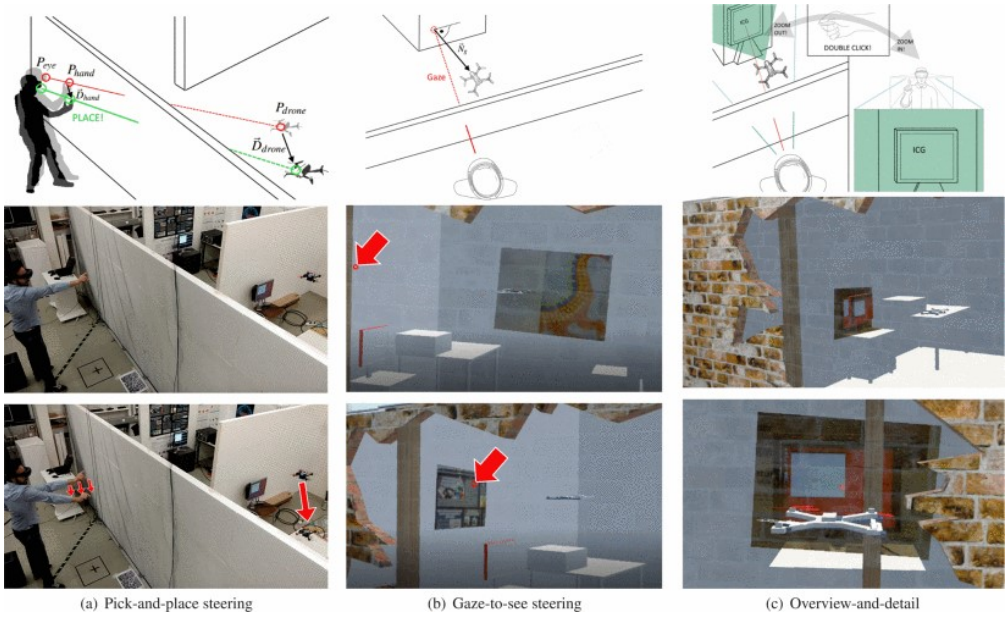


Figure 1.8 – Body position based immersion interfaces using VR.

Due to these characteristics, interfaces based on AR have been considered in various works for drone teleoperation [58–61]. For example, the authors in [53] present three interface designs to remotely operate a quadrotor using AR. The first design consists of increasing the real environment by displaying real-world objects within the FOV of the vehicle's camera, the second is to augment the robot by adding a panel for live video transmission on the quadrotor, the third design shows a fixed video streaming window to the user's periphery. With the AR, the user remains aware of the real world and can have added visual feedback in real-time to successfully perform their task as camera transmission, battery level, object recognition, etc., see Figure 1.9.

MR is the merging of real and virtual worlds to produce new environments and visualizations, where physical and digital objects coexist and interact in real time. MR can also be used for drone piloting, for example, in [62] the authors presented an interface based in a mixed reality environment and natural language to command a UAV, see Figure 1.10. One of the advantages of interfaces based on AR and MR is that it allows us to implement NUIs such as gaze and gestures, similar to our interaction with computers or touch screens. However, the problem in both methodologies is that the vehicle must remain in the user's LoS.

Despite the recent scientific interest in developing teleoperation interfaces for aerial robots using immersion devices, a study of the usability of these systems in real-world cases is still lacking.



(a) Pick-and-place steering (b) Gaze-to-see steering (c) Overview-and-detail

Figure 1.9 – Interaction techniques using an AR interface [58].

Figure 1.10 – A user gives references to objects and positions using a push-to-talk language in a MR environment [62].

Some alternatives to the issue of keeping the drone in the LoS of the human pilot have been developed in virtual environments. Recent research considers virtual interfaces with a third-person view for teleoperation of air vehicles. For example, in [63] a ground control station for drone based in an immersive virtual environment was developed, hence the user is informed of the position and status of the VR, see Figure 1.11. This figure shows as a user uses a joystick to pilot an aircraft with different views of this within a virtual environment.

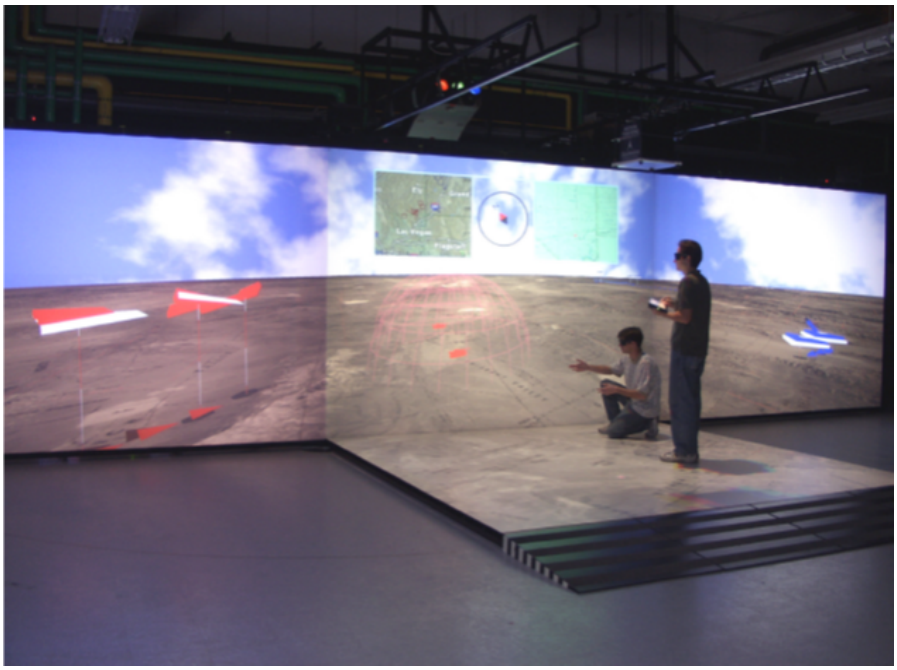


Figure 1.11 – Interface based on virtual environment [63].

Virtual environments are used in different applications to teleoperate cranes, submarines or mobile vehicles, robotic arms, among others. For example, in [64] the authors developed a system that integrates two frameworks: ROS software to control a mobile robot and Unity 3D software to visualize and interact with a virtual environment, which communicates using a yaml-based communication protocol. The user drives the robot through a virtual reality-based teleoperation interface on Unity 3D. In [65], a teleoperation architecture of a hydraulic crane is presented, it is composed of two subsystems: a sensor-equipped crane in a remote environment and a Crane VE software (virtual environment) developed in C ++, using the OpeneScene-Graph library, which are connected through an IP network, see Figure 1.12. In [66], a scheme of remote operation of a farm vehicle was presented. This system operates as follows: in a remote site in Japan a mobile vehicle is composed of actuators and sensors and in a local site located in the United States a user drives the vehicle from a three-dimensional stereoscopic reality system, both sites are communicated through the Internet in real-time. Virtual environments are also used to mitigate the delay of some systems depending on the precision of the virtual model of the robot. For example, in [67], a dynamic and geometric model of the remote environment was proposed in order to correct the errors between the geometric structure of a virtual model of a robotic arm and a real one.

Figure 1.12 – Interface based on virtual environment of a hydraulic crane [65].

The advantages of using virtual environments in some teleoperation systems is evident when the work environment is known. That is, it is not necessary to take video from the camera on board the robot to be able to teleoperate it. This is accomplished by creating a virtual model of the work environment and decreases the possible delay due to image processing when video is taken from the remote environment. Also, the flexibility of virtual environments helps synthesize information for the operator (in the form of views, charts, or other virtual objects). Thus, the user can make decisions quickly.

Several complications arise when studying teleoperated systems since the communication medium (wired or wireless) contributes substantially to the complexity of the overall system and introduces distortion, delays, and losses that impact stability and performance. These issues have motivated control-theoretic research in teleoperation over the past decades [6].

Many works devoted to developing teleoperation systems do not include strategies to mitigate the time delay. However, it is an essential issue in aerial vehicles because it could affect the vehicle's flight performance or, in the worst case, fall or crash due to instability. It is needed to mention that aerial vehicles have fast dynamics, unlike other robots, such as robotic arms or mobile robots. This characteristic makes them highly sensitive to time delay. Accordingly, it may be fundamental to address the time delay problem in these vehicles. Some theoretical control strategies that have been studied to address the delay problem will be briefly described below.

1.2.3 Time delay systems

Time-delay systems are often described by Functional Differential Equations (FDEs). These systems are also known as systems with aftereffects or dead-time, hereditary systems, or equations with deviating argument, or systems with time lag. In the 18th century, the first FDEs were studied by Bernoulli, Poisson, Laplace, Lagrange, among others, to solve several geometric problems [68].

At the beginning of the 20th century, a large number of practical problems were modeled by FDEs: viscoelasticity problem, predator-prey model in population dynamics, mathematical biology problems, and ship stabilization problem. Nowadays, it is known that the delay phenomenon is within the internal dynamics of many processes. An example is the internal combustion engine problem. The Mean Torque Production Model describes an internal combustion engine and is used for designing of controllers. In this model, the crankshaft rotation can be modeled from Newton's second law as following [69–71],

$$T_i(t-h_i) - T_f(t) - T_{load}(t) = I\dot{\omega}(t), \tag{1.1}$$

where T_i represents the indicated torque generated by the engine which is delayed by h_i seconds due to engine cycle delays such as fuel-air mixing, ignition delay, cylinder pressure force propagation, etc., and then added to the friction torque T_f and load torque T_{load}. The inertia matrix of crankshaft denoted by I and ω define the engine speed.

There exist different fields where the processes are modeled with delays such as chemical, economic, biological, physical, physiological processes, as well as in engineering sciences. Commonly sensors, actuators, and field networks that are involved in feedback loops introduce also delays. For this reason, they are strongly involved in challenging areas of information technologies and communication: high-speed communication networks [72–78], parallel computation [79–82], stability of networked controlled systems [83–86], computing times in robotics [87–89], teleoperated systems [90–96]. From the control point of view, many control strategies have been applied to teleoperation systems to mitigate the inherent time delay problem. Some control schemes will be described below, including predictive approaches.

The two-port network is a method that was discovered in the late 1980s. The interest in this method is due to its ability to study a mechanical system from the point of view of an electrical system where passivity results have already been established. The importance of the passivity of a system is that it requires minimal knowledge about human and environmental dynamics. The passivity of these systems guarantees the passivity of the entire teleoperation system, since one property is that the interconnection of passive systems is passive. For example, consider the two-port network illustrated in Figure 1.13 where e_i and f_i represents the efforts and flows which correspond to voltages and currents in electric circuits or forces and velocities in mechanical systems, respectively.

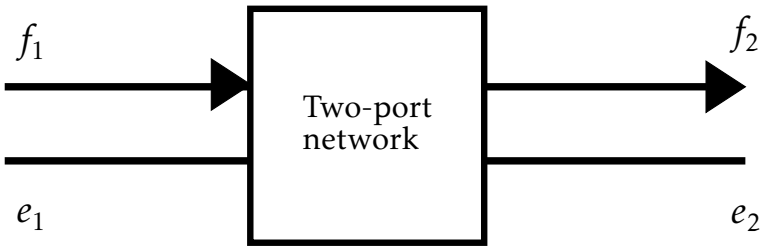


Figure 1.13 – Two-port scheme.

Depending on the signals available to the controller, different matrix representations can describe the behavior of this network, for example, the impedance matrix $Z(s)$ or hybrid $H(s)$. If the flows are considered to be the inputs on both sides of the two-port network, then an impedance representation of the master-slave system can be used to relate speeds to forces by

$$\begin{bmatrix} F_m \\ F_s \end{bmatrix} = Z(s) \begin{bmatrix} \dot{x}_m \\ \dot{x}_s \end{bmatrix} \tag{1.2}$$

where $Z(s)$ is the impedance matrix

$$Z(s) = \begin{bmatrix} z_{11}(s) & z_{12}(s) \\ z_{21}(s) & z_{22}(s) \end{bmatrix}. \tag{1.3}$$

However, if the force variable is available in the slave robot, then a hybrid representation can be obtained by

$$\begin{bmatrix} F_m \\ -\dot{x}_s \end{bmatrix} = P(s) \begin{bmatrix} \dot{x}_m \\ F_s \end{bmatrix} \tag{1.4}$$

where $P(s)$ is the hybrid matrix

$$P(s) = \begin{bmatrix} p_{11}(s) & p_{12}(s) \\ p_{21}(s) & p_{22}(s) \end{bmatrix}. \tag{1.5}$$

The elements of this matrix can be interpreted as: $p_{11}(s) = Z_{in}$, $p_{12}(s)$ =Force scaling, $p_{21}(s)$ =Velocity scaling and $p_{22}(s) = Z_{out}^{-1}$. This representation as a hybrid matrix is very useful when the time-delays are present in the communication channel.

Passivity-based control is a common nonlinear control architecture in the design of a teleoperation system. This has its origin in network theory and refers to the exchange of energy between interconnected systems. A passive system absorbs more energy than it produces. The passivity property followed by the absolute stability condition will be presented as follow: suppose a one-port network with input flow $v(t)$ and output effort $f(t)$, the total input to the network can be written as $f(t)v(t)$. Therefore, given zero energy storage at $t = 0$, a network is passive if and only if it satisfies the following inequality [4]:

$$\int_0^\infty f(s)v(s)ds \geq 0. \tag{1.6}$$

In general, for the teleoperation system, the human operator and the environment are said to be passive, meaning that the maps from velocity to force satisfy that there exist positive constants $a_m, a_s \geq 0$ such that

$$-\int_0^t \dot{p}_m^T(s) f_h(s) ds + a_m \geq 0, \tag{1.7}$$

and

$$-\int_0^t \dot{p}_s^T(s) f_e(s) ds + a_s \geq 0, \tag{1.8}$$

where $\dot{p}_m$ and $\dot{p}_s$ denote the joint velocity of the master and slave robot, respectively; and the force applied by human and environment are represented by f_h and f_e, respectively.

Many control strategies have been presented based on the passivity concept. For example, scattering theory [96, 97] is applied to pasify the system with constant time delays. The scattering approach is considered a theoretical approach to the delay problem. This consists of transforming a teleoperation problem into a transmission line problem. Then, the transformation allows to obtain a feasible solution to the delay problem.

The scattering approach the system can be seen as an n-port network with an effort-flow pair on each port. The relationship between the forces and velocities at all n ports can be represented by an impedance relationship in the frequency domain as $F(s) = Z(s)V(s)$, where $Z(s)$ is an $n \times n$ impedance matrix and $V(s)$ is the velocity vector. However, a disadvantage of this technique is that the impedance relationship is not unique. Considering the case of a two-port network where stress is force and flow is speed, the scattering operator can be defined as:

Definition 1
The scattering operator S is defined in terms of an incident wave $O_i(t) = f(t) + v(t)$ and a reflected wave $O_r(t) = f(t) - v(t)$ as

$$O_r(t) = S(O_i(t)) \tag{1.9}$$

For LTI systems, in a two-port network, the scattering matrix in the domain can be represented in terms of the hybrid matrix as follows

$$S(s) = \begin{bmatrix} (P(s)-I)(P(s)+I)^{-1} & 0 \\ 0 & -(P(s)-I)(P(s)+I)^{-1} \end{bmatrix}. \tag{1.10}$$

To guarantee passivity, the incident wave must have a lower energy content than the scattered wave. Therefore, the following passivity result can be established with respect to the scatter operator S.

Theorem 1
An n-port system is passive if and only if $\|S\|_\infty \leq 1$, where $\|S\|_\infty$ is the infinity norm of the corresponding scattering matrix [96].

Applying this result, assuming incoming and outgoing delay times are equal and constant, the transmission line model in the n-dimensional case that makes the channel passive is given by

$$F_m(t) = Z_0(\dot{x}_m(t) - \dot{x}_s(t-h)) + F_s(t-h) \tag{1.11}$$

$$\dot{x}_s = Z_0^{-1}(F_m(t-h) - F_s(t)) + \dot{x}_m(t-h) \tag{1.12}$$

where h represents the time delay on the communication line and Z_0 described the impedance matrix.

A variant of the scattering technique is the wave variables formulation [92, 97–99] also used to impose passivity on the delayed bilateral teleoperation system.

The difference with the scattering method is that instead of transmitting the power variables F_s and $\dot{x}_m$ as reference signals, the wave variables u_s and u_m are transmitted (see Figure 1.14), which can be expressed as

$$u_s = \sqrt{\frac{(F_s(t) - b\dot{x}_s(t))^2}{2b}} \tag{1.13}$$

$$u_m = \sqrt{\frac{(F_m(t) - b\dot{x}_m(t))^2}{2b}} \tag{1.14}$$

where b represents the characteristic impedance of the transmission line.

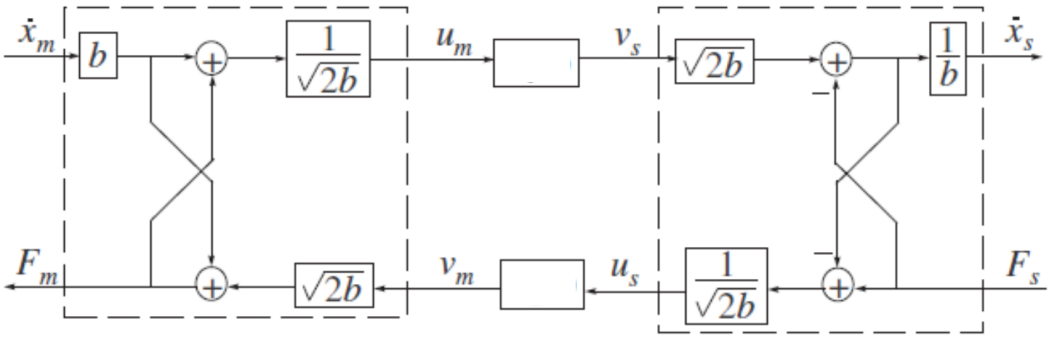


Figure 1.14 – Wave variables

If the channel is composed of constant time delays h, then the wave formulation results with the same control law given for the scattering approach, and the passivity analysis can be performed in time domain as

$$E(t) = \int_{t-h}^{t} \frac{(u_m^h(l)u_m(l) + u_s^h(l)u_s(l))}{2} dl \geq 0 \tag{1.15}$$

and therefore, the channel is passive.

Several control strategies in the wave domain have been possible thanks to the intrinsic passivity of the wave formulation. Otherwise, in the domain of the power variables, passivity would be lost.

Position drifting problem derived from the scattering method [100] can be overcome using damping intervention method. For example, in [101, 102] a large damping was injected into the master and slave systems in order to achieve asymptotic stability, also the human operator and the environment were modeled by a damping-like scheme plus a finite-energy perturbation.

An approach widely applied to teleoperation systems is the sliding mode control. Some advantages of this controller are its robustness against non-linearities and parametric uncertainties. However, in the case of teleoperation systems, it can help deal with the time delay. This scheme introduces a sliding surface $s(t)$ which is a function of the error between the positions and the velocities of the master and the slave:

$$s = \dot{e} + \lambda e, \tag{1.16}$$

where $\lambda > 0$, $e = \mathbf{x}_s - k_{\mathbf{x}}\mathbf{x}_m$, with $\mathbf{x}_m, \mathbf{x}_s$ are the generalized position coordinates of the robots, and $k_{\mathbf{x}}$ denotes the position scaling factor. Then, a controller based on the sliding mode scheme is designed to reach the desired sliding surface in finite time.

Many studies have been reported in the sliding mode control [103–107]. For example, in [108], the sliding control theory was applied to the 1-degrees of freedom (DoF) telemanipulation systems. These results were extended to teleoperation with time-varying delays in [109–111].

Model Predictive Control (MPC) is another technique applied in teleoperation systems. In general, the model of the plant is used to predict the future evolution of the system. In this prediction architecture, prediction can compensate for delays in communication channels. Forward and feedback channel delays can also be canceled by generating predictions with a longer horizon. For example, in [112] a pre-compensated slave robot is considered and an MPC controller on the master robot provides references in order for the remote system to track. However, the authors in [113] consider the case of the bilateral problem where a force feedback is applied and the controllers in the slave and master robots are designed as the solution to an optimization problem. In [114], the dynamics of the environment is mapped and simulated at the master robot using two-neural networks.

Most of the methods for compensating delay assume a perfect knowledge of the impedances of the master and slave robots, the environment and the operator in order to compensate them. A solution to eliminate this hypothesis is the adaptive controllers which mitigate the effects of the uncertainties of the parameters in master-slave model and user-environment model or both. In [115], an adaptive admittance control approach based on an intuitive relationship between the force data and human velocity was proposed. In this case, the damping parameter in the robot's admittance controller is decreased or increased during accelerating or decelerating motion of the operator. In [116], an internal disturbance rejection controller was proposed to improve the robustness of robot model uncertaines under the assumption of measurement of acceleration or boundedness of the inverse of the estimated inertia. In [117–119], the passive-based adaptive controllers are proposed for robustness to master and slave model uncertainties and time delay. In [119, 120], virtual internal model-based adaptive controllers are proposed for transparency and robustness to the delay.

The advancement of networks and technologies led to, in the mid-90's, teleoperation over the Internet. Network robot technology has provided different remote services with applications in many areas: distance learning, telemedicine, entertainment, services, security applications, among others. However, Internet-based teleoperation brings different challenges. On the one hand, the bandwidth of communication channels, the communicating information across a packet-switched network results in random, time-varying delays. On the other hand, due to the limitations of the communication capability, it also leads to the loss of data packets from the slave to the master robot or vice-versa. Furthermore, the need to deal with discrete-time stability arises. As a result the performance of the teleoperation system deteriorates drastically and possibly becomes unstable.

In teleoperation over the Internet, the slave and master robots must transport their information through the software layers to the physical layer in discrete time. During the transport the data packets can suffer variable aleatory delays in the incoming delay time $h_{in}(t)$ and outgoing delay $h_{out}(t)$ which distort the transmitted signals. Two types of communication that reside in the transport layer in the ISO 7-layer reference model are Transmission Control Protocol (TCP) and UDP. On one hand, TCP provides reliable two-way communication and guarantees data delivery at the cost of retransmissions and long timeouts that are detrimental in real-time applications such as teleoperation. On the other hand, UDP does not require reception acknowledgments eliminating unnecessary waiting time, which makes it appealing for real-time applications such as teleoperation.

The packet loss inherent in this type of teleoperation is solved by applying reconstruction algorithms which address the problem from a passivity point of view. This means that the lost samples are reconstructed preserving the passivity of the system. Some policies used in packet loss are to use: packet replacement, previous packets or passive interpolation.

A problem in the transport of information is the transformation of the signals in continuous-time to the signals to discrete-time, as this could generate an excess of energy in the interconnection of ports. Under suitable conditions between the continuous and discrete-time ports, it is possible to obtain an equality of effort in each time step as

$$e_c(t) = e_d(k), \quad \forall t \in [kT, (k+1)T]. \tag{1.17}$$

Then, it is possible to set the following result: If we define for the discrete interconnection port of

$$f_d(k) = \frac{x((k+1)T) - x(kT)}{T}, \tag{1.18}$$

where $x(\cdot)$ represents the integral of the continuous flow, namely a position measurement, we obtain an equivalence between the continuous time and discrete time energy flow in the sense that each n;

$$E_d(n) = \sum_{k=0}^{n-1} e_d^T(k) f_d(k) T = \int_0^{nT} e_c^T(s) f_c(s) ds = E_c(nT) \tag{1.19}$$

which guarantees lossless connection between the two systems independent of the sampling period.

Other analysis strategies have been proposed for teleoperation such as: H_∞ design, shared compliant control, virtual internal model, among others. These solutions to tackle the time-delay problem from the control engineering perspective. The application of these approaches depends on how the delay is modeled and where it is located within the system.

The delay can be present in a system in three ways: in the system's states, in the inputs of the system, or both. In particular, systems where the delay is present on the inputs, can be considered challenging due to requiring infinite feedback for long delays. Moreover, many systems can be subject to input delays to represent actual physical transport delay in hydraulically actuated systems, chemical process systems, slow biological response, or problems where computational delay can be expressed as equivalent to input delay [121–134]. For example, communication latency in teleoperated robots can lead to poor performance and potential instability. It is a barrier to achieve high fidelity of the real robot movements [135]. The problems about performance and stability of these systems have been addressed by applying predictors-based feedback such as: Finite Spectrum Assignment (FSA) [128], the reduction approach [122], and continuous pole placement [136]. One of the classic predictor structures is the Smith predictor that originated the predictive control [137]. These techniques require a plant model for output prediction and has been widely studied and modified for control purposes [138–144].

Different works have addressed the problem of Linear Time-Invariant (LTI) plants with various kinds of input delay. Recently, input delay has been modeled by hyperbolic Partial Differential Equations (PDE), with this approach a backstepping design procedure has been developed to provided a stability proof based on Lyapunov-Krasovskii Functionals (LKFs) for predictor-based control of LTI systems with constant delays [145]. These results were extended to the case of LTI systems with unknown input delay [146]. Other recent predictor-based structure consists in implemented a so-called truncated predictor feedback using a static finite-dimensional time-varying state feedback control law, for LTI systems with known time-varying input delay [147].

A challenging opportunity to validate the performance of predictor-based methods is to use them in UAV dynamics. These vehicles are appropriate to validate because they require fast, accurate, and robust positioning performance. For example, when an aerial vehicle flies indoors, it requires the states of position and linear velocity, which are calculated using data from the sensors. These states are often delayed due to data transmission. The time delay can be increased when the vehicle is part of a teleoperation system due to the delay in the local master system. One approach to mitigate the time delay problem is to develop predictor-based controllers. Clearly, a predictor-based control should be robust to UAV model parameter and implementable in a way that does not make the teleoperation system more complex.

Neverthless, only few works on predictor-based controllers have been applied to UAVs. In [148], a discrete-time framework was adopted for LTI systems, the proposed technique is robust for small bounded uncertainties in the system parameters, delay, and sampling instants. Experimental results on a quadrotor for yaw control are provided. The authors in [149] proposed a Taylor series-based approach and experimental results on position control in a quadrotor with delayed linear velocity and position measurements are provided. Recently, a delayed attitude and height controller using prediction for simplified UAV dynamics was developed locally without controlling the lateral position of the vehicle [150]. A delayed force input to the outer-loop for a UAV visual servoing problem was considered in [151]. However, the problem is simplified assuming that the yaw angle is known for image coordinate reprojection. In [152], the authors represent the set of time delay as a disturbance that satisfies a Lipschitz condition. Inner-outer loop control mitigates the effect by bounded tracking error. In [153], the robustness of a nested saturation control for quadrotors with respect to small time delay was analyzed. These predictor-based schemes are commonly applied to drone models based on Euler angles without exploring the robustness that other models can give.

2 Quadrotor modeling

The first step towards designing a control strategy that ensures the stability of a quadrotor in the presence of delays is to establish a mathematical model of the system. Unlike other robotic systems, quadrotors are challenging to control due to their fast dynamics and nature underactuated (6 DoF with only 4 actuators), consequently we will adopt a robust model of the system is related. In this chapter, different modeling methodologies are analyzed.

This chapter is organized as follows: Section 2.1 introduces the elemental forces that affect the flight of a quadrotor robot. In Section 2.2 the Euler-Lagrange formalism is presented. An approach based on the Newton-Lagrange equations is studied in Section 2.3. In Section 2.4 a quaternion-based approach is described. Finally, some conclusions about the models are presented in Section 2.6.

2.1 Generalities

Throughout this thesis, the notation will be simplified whenever no confusion can arise from the context. Vectors in $\mathbb{R}^3$ will be denoted with an arrow. Quaternions will be denoted by bold letters, except the generalized vector $\mathbf{x}$. We will denote by $\mathcal{I}$ and $\mathcal{B}$ the inertial and body frames, respectively.

Quadrotors consist of four coplanar motors that are subject to the main force (thrust) and three moments (torques). These forces are provided by rotation of the propellers. Consequently given a rotor i with angular velocity ω_i, the force and moment on a propeller are yielded by

$$f_i = C_{f_i} \rho A r^2 \omega_i^2, \tag{2.1}$$

$$\tau_i = C_{\tau_i} \rho A r^2 \omega_i^2, \tag{2.2}$$

where f_i denotes the total thrust generated by rotor $i = 1, 2, 3, 4$, τ_i represents the rotor torque, r describes the rotor radius, ρ symbolizes the air density and $A = \pi r^2$ defines the front propeller disk area. C_f and C_τ are thrust and rotor torque coefficients.

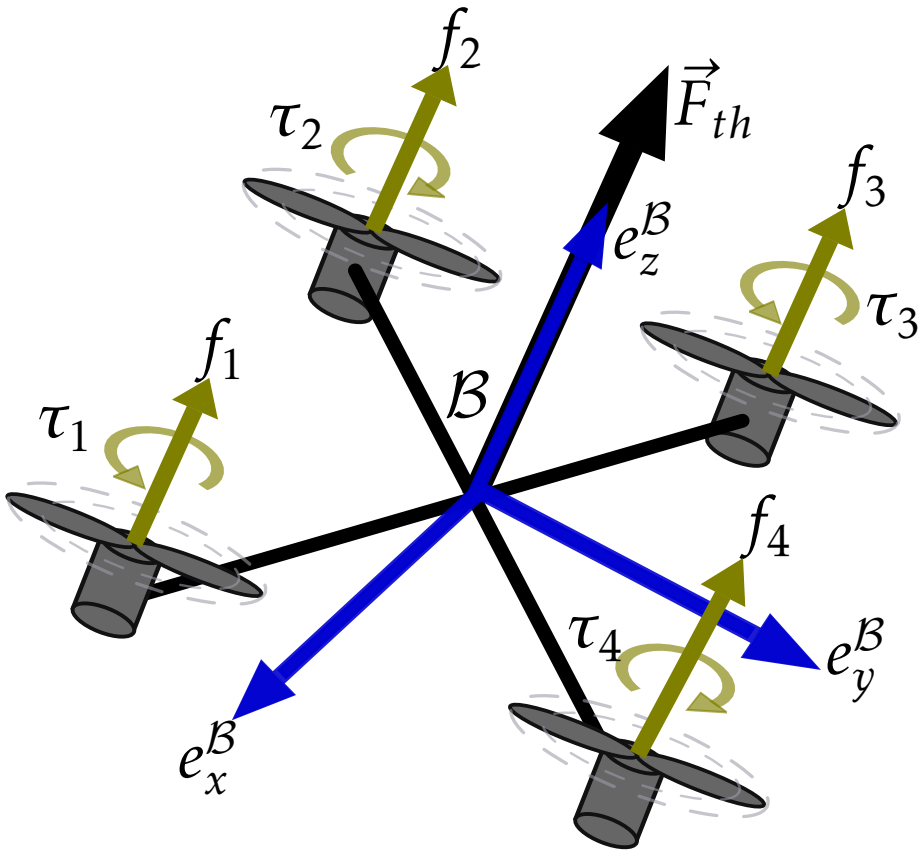


Figure 2.1 – Propeller forces and torques acting on a quadrotor.

It is often assumed that the propeller is not subject to high pressures, the area of the propeller is small and it does not have a high velocity. Therefore, aerodynamic parameters in equations (2.1) and (2.2) are considered constants $f_i \approx k_{f_i}\omega_i^2$ and $\tau_i \approx k_{\tau_i}\omega_i^2$ where k_{f_i} and k_{τ_i} represent aerodynamics coefficients of the propeller.

Given the symmetric configuration of the quadrotor the total torque on the drone is computed as

$$\vec{\tau} = \begin{bmatrix} \tau_\psi \\ \tau_\theta \\ \tau_\phi \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{4} \tau_{M_i} \\ l(f_1 + f_2 - f_3 - f_4) \\ l(f_1 - f_2 + f_3 - f_4) \end{bmatrix} \tag{2.3}$$

and the thrust force is given by

$$\vec{F}_{th} = \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} k_T \omega_i^2 \end{bmatrix} \tag{2.4}$$

where τ_θ, τ_ψ and τ_ψ represent the total torque components on the body frame, and $\vec{F}_{th}$ denotes the total thrust force in the vertical axis of the quadrotor.

2.2 Euler-Lagrange model

A quadcopter is considered as a solid body evolving in three dimensions and subject to a main force and three moments. Let us also consider a fixed coordinate system at a point on the ground called $\mathcal{I}$ and a mobile reference system to attached the vehicle $\mathcal{B}$, where the center of mass and the origin of $\mathcal{B}$ coincide.

The generalized coordinates of an air vehicle can be written as

$$\mathbf{x} = (\vec{\xi}, \vec{\eta})^T \tag{2.5}$$

where $\vec{\xi} = (x, y, z)^T \in \mathbb{R}^3$ denotes the position of the center of mass of the vehicle relative to the reference frame $\mathcal{I}$ and $\vec{\eta} = (\psi, \theta, \phi)^T \in \mathbb{S}^3$ define the Euler's angles (yaw, pitch and roll, respectively) which represent the orientation of the vehicle.

The Lagrangian equation $\mathcal{L}$ is defined as

$$\mathcal{L}(\mathbf{x}, \dot{\mathbf{x}}) = T_{trans} + T_{rot} - U, \tag{2.6}$$

where $T_{trans} = \frac{m}{2}\dot{\vec{\xi}}^T\dot{\vec{\xi}}$ describes the translational kinetic energy, $T_{rot} = \frac{1}{2}\vec{\Omega}^T J\vec{\Omega}$ denotes the rotational kinetic energy, $U = mgz$ describes its potential energy, $\vec{\Omega} = (\omega_x, \omega_y, \omega_z)^T$ symbolizes the angular velocity, J defines the inertia matrix, m represent its mass, g is the acceleration due to gravity and z is the altitude. The angular velocity vector $\vec{\Omega}$ in the body frame $\mathcal{B}$ is related to the generalized velocities $\dot{\vec{\eta}}$ (in the region where the Euler angles are valid) by means of the standard kinematic relationship

$$\vec{\Omega} = W_{\vec{\eta}}\dot{\vec{\eta}}, \tag{2.7}$$

where

$$W_{\vec{\eta}} = \begin{bmatrix} -\sin\theta & 0 & 1 \\ \cos\theta\sin\phi & \cos\phi & 0 \\ \cos\theta\cos\phi & -\sin\phi & 0 \end{bmatrix}, \tag{2.8}$$

note that $W_{\vec{\eta}}$ is invertible if $\theta \neq \frac{(2k-1)\phi}{2}$ where $k \in \mathbb{Z}^+$.

Applying the transformation (2.7) on the rotational kinetic energy, we obtain

$$T_{rot} = \frac{1}{2}\dot{\vec{\eta}}^T\mathbb{J}\dot{\vec{\eta}} \tag{2.9}$$

with $\mathbb{J} = \mathbb{J}(\vec{\eta}) = W_{\vec{\eta}}^T J W_{\vec{\eta}}$ and

$$J = \begin{bmatrix} J_{xx} & 0 & 0 \\ 0 & J_{yy} & 0 \\ 0 & 0 & J_{zz} \end{bmatrix}, \tag{2.10}$$

where J_{ii} denotes the moment of inertia with respect to the $i = x, y, z$ axes.

Note that the matrix $\mathbb{J}(\vec{\eta})$ acts as the inertia matrix for the full rotational kinetic energy of the vehicle expressed in terms of $\vec{\eta}$. Then the mathematical equations that represent the dynamics of the aerial vehicle are obtained using the following equation:

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{x}}} - \frac{\partial \mathcal{L}}{\partial \mathbf{x}} = \begin{bmatrix} \vec{F}_{th}^{\mathcal{I}} \\ \vec{\tau} \end{bmatrix}, \tag{2.11}$$

where $\vec{F}_{th}^{\mathcal{I}}$ denotes the translational external forces acting on the vehicle due to the control inputs, and $\vec{\tau}$ represents the external torques. We ignore the small body forces because they are generally of a much smaller magnitude that the principal control.

Considering from Figure 2.2 that the thrust force acts only in the z-axis, it can be written as $\vec{F}_{th} = \vec{n}_z u$, where $\vec{n}_z = (0,0,1)^T$ represents the thrust directed out of the top of the vehicle and $u = \|\vec{F}_{th}\|$. If this vector force is represented in the inertial frame using a rotation matrix R derived from the Euler angles as $\vec{F}_{th}^{\mathcal{I}} = R\vec{F}_{th}$, where

$$R = R(\psi,\theta,\phi) = \begin{bmatrix} C_\psi C_\theta & -S_\psi C_\theta & S_\theta \\ S_\psi C_\phi + C_\psi S_\theta S_\phi & C_\psi C_\phi - S_\psi S_\theta S_\phi & -C_\theta S_\phi \\ S_\psi S_\phi - C_\psi S_\theta C_\phi & C_\psi S_\phi + S_\psi S_\theta C_\phi & C_\theta C_\phi \end{bmatrix}, \tag{2.12}$$

where S_β and C_β stand for $\sin(\beta)$ and $\cos(\beta)$ with $\beta \in \{\psi,\theta,\phi\}$ respectively.

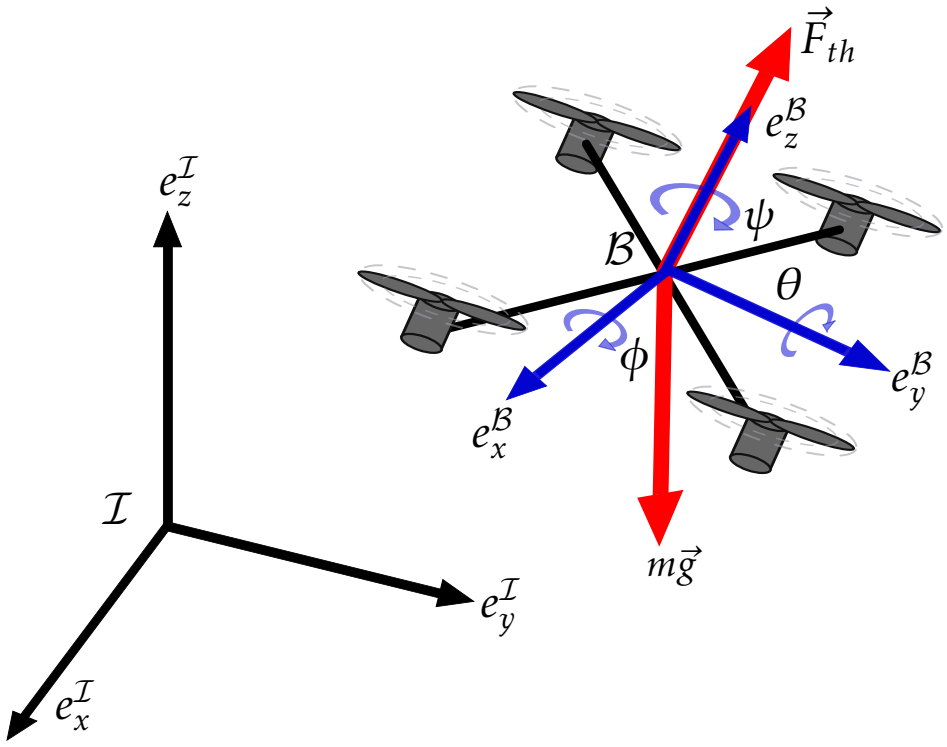


Figure 2.2 – Quadcopter scheme in an inertial frame.

Since the Lagrangian equation (2.6) does not contain crossed terms of $\dot{\vec{\xi}}$ and $\dot{\vec{\eta}}$ is kinetic energy then the Euler-Lagrange equation (2.11) can be divided into translation motion

$$m(\ddot{\vec{\xi}} - \vec{g}) = \vec{F}^{I}_{th}, \tag{2.13}$$

where $\vec{g} = (0, 0, -g)^T$ denotes the acceleration vector due to gravity, and the rotational motion

$$\mathbb{J}\ddot{\vec{\eta}} + \dot{\mathbb{J}}\dot{\vec{\eta}} - \frac{1}{2}\frac{\partial}{\partial \vec{\eta}}\left(\dot{\vec{\eta}}^T \mathbb{J}\right)\dot{\vec{\eta}} = \vec{\tau}. \tag{2.14}$$

Therefore, equations (2.13) and (2.14) can be rewriting as

$$m\ddot{\vec{\xi}} = \vec{F}^{\mathcal{I}}_{th} - mg\vec{n}_z \tag{2.15}$$

$$\mathbb{J}\ddot{\vec{\eta}} = \vec{\tau} - C(\vec{\eta}, \dot{\vec{\eta}})\dot{\vec{\eta}} \tag{2.16}$$

where $C(\vec{\eta}, \dot{\vec{\eta}}) = \dot{\mathbb{J}} - \frac{1}{2}\frac{\partial}{\partial \vec{\eta}}\left(\dot{\vec{\eta}}^T \mathbb{J}\right)$ represent to the Coriolis term and contains the gyroscopic and centrifugal terms. Computing equations (2.15)-(2.16) is an arduous task and, in several works, the full inertia matrix $\mathbb{J}$ is considered diagonal and the Coriolis matrix is, in general, neglected.

The inertia and Coriolis matrix can be obtained from (2.11) for the $\vec{\eta}$ dynamic. Therefore, the attitude dynamics yields

$$\frac{d}{dt}\left(\vec{\Omega}^T J \frac{\partial \vec{\Omega}}{\partial \dot{\mathbf{x}}}\right) - \vec{\Omega}^T J \frac{\partial \vec{\Omega}}{\partial \mathbf{x}} = \vec{\tau} \tag{2.17}$$

Then $\dfrac{\partial \vec{\Omega}}{\partial \dot{\mathbf{x}}} = W_{\vec{\eta}}$ and $\vec{\Omega}^T J \dfrac{\partial \vec{\Omega}}{\partial \dot{\mathbf{x}}} = (a_1, a_2, a_3)^T$ with

$$a_1 = -J_{xx}(\dot{\phi}S_\theta - \dot{\psi}S^2_\theta) + J_{yy}(\dot{\theta}C_\theta S_\phi C_\phi + \dot{\psi}C^2_\theta S^2_\phi) + J_{zz}(\dot{\psi}C^2_\theta C^2_\phi - \dot{\theta}C_\theta S_\phi C_\phi), \tag{2.18}$$

$$a_2 = J_{yy}(\dot{\theta}C^2_\phi + \dot{\psi}C_\theta S_\phi C_\phi) - J_{zz}(\dot{\psi}C_\theta S_\phi C_\phi - \dot{\theta}S^2_\phi), \tag{2.19}$$

$$a_3 = J_{xx}(\dot{\phi} - \dot{\psi}S_\theta), \tag{2.20}$$

where S_β and C_β stand for $\sin(\beta)$ and $\cos(\beta)$ with $\beta \in \{\psi, \theta, \phi\}$ respectively.

Differentiating $\vec{\Omega}^T J \frac{\partial \vec{\Omega}}{\partial \dot{\mathbf{x}}}$ with respect to time, we get

$$\begin{aligned}
\dot{a}_1 &= -J_{xx}(\ddot{\phi}S_\theta + \dot{\phi}\dot{\theta}C_\theta - \ddot{\psi}S_\theta^2 - 2\dot{\psi}\dot{\theta}S_\theta C_\theta) + J_{yy}(\ddot{\theta}C_\theta S_\phi C_\phi \\
&\quad -\dot{\theta}^2 S_\theta S_\phi C_\phi - \dot{\theta}\dot{\phi}C_\theta S_\phi^2 + \dot{\theta}\dot{\phi}C_\theta C_\phi^2 + \ddot{\psi}C_\theta^2 S_\phi^2 - 2\dot{\psi}\dot{\theta}S_\theta C_\theta S_\phi^2 \\
&\quad +2\dot{\psi}\dot{\phi}C_\theta^2 S_\phi C_\phi) + J_{zz}(\ddot{\psi}C_\theta^2 C_\phi^2 - 2\dot{\psi}\dot{\theta}S_\theta C_\theta C_\phi^2 - 2\dot{\psi}\dot{\phi}C_\theta^2 S_\phi C_\phi \\
&\quad -\ddot{\theta}C_\theta S_\phi C_\phi + \dot{\theta}^2 S_\theta S_\phi C_\phi + \dot{\theta}\dot{\phi}C_\theta C_\phi^2), \\
\dot{a}_2 &= J_{yy}(\ddot{\theta}C_\phi^2 - 2\dot{\theta}\dot{\phi}S_\phi C_\phi + \ddot{\psi}C_\theta S_\phi C_\phi - \dot{\psi}\dot{\theta}S_\theta S_\phi C_\phi + \dot{\psi}\dot{\phi}C_\theta C_\phi^2 \\
&\quad -\dot{\psi}\dot{\phi}C_\theta S_\phi^2) - J_{zz}(\ddot{\psi}C_\theta S_\phi C_\phi - \dot{\psi}\dot{\theta}S_\theta S_\phi C_\phi - \dot{\psi}\dot{\phi}C_\theta S_\phi^2 \\
&\quad +\dot{\psi}\dot{\phi}C_\theta C_\phi^2 - \ddot{\theta}S_\phi^2 - 2\dot{\theta}\dot{\phi}S_\phi C_\phi), \\
\dot{a}_3 &= J_{xx}(\ddot{\phi} - \ddot{\psi}S_\theta - \dot{\psi}\dot{\theta}C_\theta).
\end{aligned}$$

Analogously, we have

$$\frac{\partial \vec{\Omega}}{\partial \vec{\eta}} = \begin{bmatrix} 0 & -\dot{\psi}C_\theta & 0 \\ 0 & -\dot{\psi}S_\theta S_\phi & -\dot{\theta}S_\phi + \dot{\psi}C_\theta C_\phi \\ 0 & -\dot{\psi}S_\theta C_\phi & -\dot{\psi}C_\theta S_\phi - \dot{\theta}C_\phi \end{bmatrix}, \tag{2.21}$$

then $\vec{\Omega}^T J \frac{\partial \vec{\Omega}}{\partial \vec{\eta}} = (h_1, h_2, h_3)^T$ where

$$\begin{aligned}
h_1 &= 0, \\
h_2 &= -J_{xx}(\dot{\psi}\dot{\phi}C_\theta - \dot{\psi}^2 S_\theta C_\theta) - J_{yy}(\dot{\psi}\dot{\theta}S_\theta S_\phi C_\phi + \dot{\psi}^2 S_\theta C_\theta S_\phi^2) \\
&\quad -J_{zz}(\dot{\psi}^2 S_\theta C_\theta C_\phi^2 - \dot{\psi}\dot{\theta}S_\theta S_\phi C_\phi), \\
h_3 &= J_{yy}(-\dot{\theta}^2 S_\phi C_\phi - \dot{\psi}\dot{\theta}C_\theta S_\phi^2 + \dot{\psi}\dot{\theta}C_\theta C_\phi^2 + \dot{\psi}^2 C_\theta^2 S_\phi C_\phi) \\
&\quad +J_{zz}(-\dot{\psi}^2 C_\theta^2 S_\phi C_\phi + \dot{\psi}\dot{\theta}C_\theta S_\phi^2 - \dot{\psi}\dot{\theta}C_\theta C_\phi^2 + \dot{\theta}^2 S_\phi C_\phi).
\end{aligned}$$

Notice that

$$\tau = \begin{bmatrix} \tau_\psi \\ \tau_\theta \\ \tau_\phi \end{bmatrix} = \begin{bmatrix} \dot{a}_1 - h_1 \\ \dot{a}_2 - h_2 \\ \dot{a}_3 - h_3 \end{bmatrix}. \tag{2.22}$$

Thus, grouping terms and using (2.15), it follows that

$$\mathbb{J}(\vec{\eta}) = \begin{bmatrix} J_{xx}S_\theta^2 + J_{yy}C_\theta^2 S_\phi^2 + J_{zz}C_\theta^2 C_\phi^2 & C_\theta C_\phi S_\phi(J_{yy} - J_{zz}) & -J_{xx}S_\theta \\ C_\theta C_\phi S_\phi(J_{yy} - J_{zz}) & J_{yy}C_\theta^2 + J_{zz}S_\phi^2 & 0 \\ -J_{xx}S_\theta & 0 & J_{xx} \end{bmatrix} \tag{2.23}$$

and

$$C(\vec{\eta},\dot{\vec{\eta}}) = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}, \tag{2.24}$$

where

$$\begin{aligned}
c_{11} &= J_{xx}\dot{\theta}S_\theta C_\theta + J_{yy}(-\dot{\theta}S_\theta C_\theta S_\phi^2 + \dot{\phi}C_\theta^2 S_\phi C_\phi) \\
&\quad -J_{zz}(\dot{\theta}S_\theta C_\theta C_\phi^2 + \dot{\phi}C_\theta^2 S_\phi C_\phi), \\
c_{12} &= J_{xx}\dot{\psi}S_\theta C_\theta - J_{yy}(\dot{\theta}S_\theta S_\phi C_\phi + \dot{\phi}C_\theta C_\phi^2 + \dot{\psi}S_\theta S_\phi S_\phi^2) \\
&\quad +J_{zz}(\dot{\phi}C_\theta S_\phi^2 - \dot{\phi}C_\theta C_\phi^2 - \dot{\psi}S_\theta C_\theta C_\phi^2 + \dot{\theta}S_\theta S_\phi C_\phi), \\
c_{13} &= -J_{xx}\dot{\theta}C_\theta + J_{yy}\dot{\psi}C_\theta^2 S_\phi C_\phi - J_{zz}\dot{\psi}C_\theta^2 S_\theta C_\theta, \\
c_{21} &= -J_{xx}\dot{\psi}S_\theta C_\theta + J_{yy}\dot{\psi}S_\theta C_\theta S_\phi^2 + J_{zz}\dot{\psi}S_\theta C_\theta C_\phi^2, \\
c_{22} &= -J_{yy}\dot{\phi}S_\phi C_\phi + J_{zz}\dot{\phi}S_\phi C_\phi, \\
c_{23} &= J_{xx}\dot{\psi}C_\theta + J_{yy}(-\dot{\theta}S_\phi C_\phi + \dot{\psi}C_\theta C_\phi^2 - \dot{\psi}C_\theta S_\phi^2) \\
&\quad +J_{zz}(\dot{\psi}C_\theta S_\phi^2 - \dot{\psi}C_\theta C_\phi^2 + \dot{\theta}S_\phi C_\phi), \\
c_{31} &= -J_{yy}\dot{\psi}C_\theta^2 S_\phi C_\phi + J_{zz}\dot{\psi}C_\theta^2 S_\phi C_\phi, \\
c_{32} &= -J_{xx}\dot{\psi}C_\theta + J_{yy}(\dot{\theta}S_\phi C_\phi + \dot{\psi}C_\theta S_\phi^2 - \dot{\psi}C_\theta C_\phi^2) \\
&\quad -J_{zz}(\dot{\psi}C_\theta S_\phi^2 - \dot{\psi}C_\theta C_\phi^2 + \dot{\theta}S_\phi C_\phi), \\
c_{33} &= 0.
\end{aligned}$$

Note that equations (2.15) and (2.16) represent the mathematical model of an aerial vehicle having four rotors. These equations are also valid for other aerial configurations as long as the external forces and torques are rewritten. Observe that in this approach no aerodynamic effects are taken into account. In the following subsection these effects will be included.

2.3 Newton-Euler approach

The general mathematical model describing the dynamics of an UAV evolving in a three-dimensional space is obtained by representing the aircraft as a solid body, which is subject to non-conservative forces $\vec{F}_{\mathcal{I}} \in \mathbb{R}^3$ expressed in a inertial frame $\mathcal{I}$, and torques $\vec{\tau} \in \mathbb{R}^3$ applied to its center of mass and specified with respect to the body frame $\mathcal{B}$, and by using the Newton-Euler approach

$$m\ddot{\vec{\xi}} = \vec{F}_{\mathcal{I}}, \tag{2.25}$$

$$\dot{R} = R\hat{\Omega}, \tag{2.26}$$

$$J\dot{\vec{\Omega}} = -\vec{\Omega} \times J\vec{\Omega} + \vec{\tau} \tag{2.27}$$

where $\hat{\Omega}$ describes the anti-symmetric matrix of $\vec{\Omega}$ and J represents the constant inertia matrix around the center of mass.

In this approach, external perturbations (e.g. wind) and uncertainties in the model (e.g. blade flapping) are considered. This model contains aerodynamic effects and could be used for research purposes.

Consider the aerial vehicle in the presence of lateral wind. Then from Figure 2.2 it can be concluded that

$$\vec{F}_{\mathcal{I}} = R\vec{F}_{th} + \vec{g} \tag{2.28}$$

where $\vec{g} = (0, 0, -g)^T$ denotes the gravity acceleration vector. The main vector force produced by the rotors is considered as $\vec{F}_{th} = (0, 0, u)^T$.

Considering the total forces and torques from (2.3) and (2.4), and introducing them into (2.25) and (2.27), it follows that the translational dynamics of the aerial vehicle is represented by

$$m\ddot{x} = -u\sin\theta, \tag{2.29}$$

$$m\ddot{y} = u\cos\theta\sin\phi, \tag{2.30}$$

$$m\ddot{z} = u\cos\theta\cos\phi - mg \tag{2.31}$$

and its rotational dynamics is given by

$$\begin{bmatrix} \ddot{\psi} \\ \ddot{\theta} \\ \ddot{\phi} \end{bmatrix} = J^{-1}\left(\begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} - C(\vec{\eta}, \dot{\vec{\eta}})\begin{bmatrix} \dot{\psi} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix}\right), \tag{2.32}$$

where $C(\vec{\eta}, \dot{\vec{\eta}})$ is referred to as the Coriolis terms and contains the gyroscopic and centrifugal terms associated with the η dependence of J.

2.4 Quaternion-based model

An existing model in the literature is based upon the use of the quaternion representation to describe the attitude of the vehicle. In this subsection the model will be developed. First, some operations and properties of the quaternion space will be enunciated. Later, the model will be developed.

2.4.1 Quaternion algebra

Let $\mathbb{H}$ the quaternion space defined by

$$\mathbb{H} = \{\mathbf{q} = q_0 + q_1\hat{i} + q_2\hat{j} + q_3\hat{k} | q_0, q_1, q_2, q_3 \in \mathbb{R}, \hat{i}^2 = \hat{j}^2 = \hat{k}^2 = \hat{i}\hat{j}\hat{k} = -1\} \subset \mathbb{C}^2. \tag{2.33}$$

There are different ways of expressing the elements of space $\mathbb{H}$. One of them is differentiating the real part q_0 from the vector part $\vec{q} = (q_1, q_2, q_3)^T$.

Therefore, a quaternion can be written as

$$\mathbf{q} = \begin{bmatrix} q_0 \\ \vec{q} \end{bmatrix}. \tag{2.34}$$

The addition and product operations are defined as follows.

Addition: Let $\mathbf{q}, \mathbf{r} \in \mathbb{H}$, such that $\mathbf{q} = \begin{bmatrix} q_0 \\ \vec{q} \end{bmatrix}$ y $\mathbf{r} = \begin{bmatrix} r_0 \\ \vec{r} \end{bmatrix}$ then the sum is defined term-to-term

$$\mathbf{q} + \mathbf{r} = \begin{bmatrix} q_0 + r_0 \\ \vec{q} + \vec{r} \end{bmatrix} \tag{2.35}$$

Product: Let $\mathbf{q}, \mathbf{r} \in \mathbb{H}$, such that $\mathbf{q} = \begin{bmatrix} q_0 \\ \vec{q} \end{bmatrix}$ y $\mathbf{r} = \begin{bmatrix} r_0 \\ \vec{r} \end{bmatrix}$ then product is

$$\mathbf{q} \otimes \mathbf{r} = \begin{bmatrix} q_0 r_0 - \vec{q} \cdot \vec{r} \\ r_0 \vec{q} + q_0 \vec{r} + \vec{q} \times \vec{r} \end{bmatrix} \tag{2.36}$$

Theorem 2 (Quaternion exponential)
The exponential of a quaternion $\mathbf{q} = q_0 + \vec{q}$ *is given by*

$$\exp(\mathbf{q}) = \exp(q_0)\left(\cos(|\vec{q}|) + \frac{\vec{q}}{|\vec{q}|}\sin(|\vec{q}|)\right) \tag{2.37}$$

Theorem 3 (Quaternion logarithm)
The logarithm of a quaternion $\mathbf{q} = q_0 + \vec{q}$ *is given by*

$$\ln(\mathbf{q}) = \ln(|\mathbf{q}|) + \frac{\vec{q}}{|\vec{q}|}\arccos\left(\frac{q_0}{|\mathbf{q}|}\right) \tag{2.38}$$

Theorem 4 (Quaternion power)
Quaternion power can be defined as

$$\mathbf{q}^{\mathbf{p}} = \exp(\ln(\mathbf{q}) \otimes \mathbf{p}) \tag{2.39}$$

Note that if the exponent is a scalar value p, then the power is

$$\mathbf{q}^p = \exp(p \ln(\mathbf{q})) \tag{2.40}$$

Some important properties are listed below.

- The inverse element of the product operation is called conjugate quaternion and is defined as: given a quaternion $\mathbf{q} = q_0 + \vec{q}$ its conjugate quaternion is $\mathbf{q}^* = q_0 - \vec{q}$.

- The norm of a quaternion is defined as

$$\|\mathbf{q}\| = \sqrt{\mathbf{q} \otimes \mathbf{q}^*} = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}. \tag{2.41}$$

- Every quaternion except the zero quaternion with real and vector part 0 has an inverse quaternion defined as

$$\mathbf{q}^{-1} = \frac{\mathbf{q}^*}{\|\mathbf{q}\|^2}. \tag{2.42}$$

- If $\mathbf{q} \in \mathbb{H}$ and $\|\mathbf{q}\| = 1$ then $\mathbf{q}$ is called unit quaternion. Therefore, a unit quaternion fulfills that $\mathbf{q}^{-1} = \mathbf{q}^*$. To model the attitude of the drone we will use the unit quaternions.

Recall that the polar form of a unit complex number can be written as

$$\exp(\hat{i}\varphi) = \cos\varphi + \hat{i}\sin\varphi \tag{2.43}$$

where $\varphi \in [0, 2\pi)$ denote the angle. A quaternion has a similar representation, consider the rotation of the vector $\vec{\gamma} = (\gamma_x, \gamma_y, \gamma_z)^T$ and denote its magnitude $\gamma = \|\vec{\gamma}\|$ in radians, the unit vector is $\vec{\alpha} = \vec{\gamma}/\gamma$ then its axis-angle representation ca be expressed as

$$\vec{\gamma} = \gamma\vec{\alpha}, \tag{2.44}$$

that means that any vector can be decomposed as the product between its norm by its normalized representation. This property can be extended to quaternions and is known as Euler-Rodrigues formula, its axis-angle representation is defined as

$$\mathbf{q} = \exp\left(\frac{\vec{\gamma}}{2}\right) = \exp\left(\frac{\gamma\vec{\alpha}}{2}\right) = \cos\left(\frac{\gamma}{2}\right) + \vec{\alpha}\sin\left(\frac{\gamma}{2}\right) \tag{2.45}$$

Using the unit quaternion $\mathbf{q} = q_0 + \vec{q}$, $\|\mathbf{q}\| = 1$, we define an operator on vector $\vec{v} \in \mathbf{R}^3$,

$$\begin{aligned} L_{\mathbf{q}}(\vec{v}) &= \mathbf{q} \otimes \mathbf{v} \otimes \mathbf{q}^* && (2.46)\\ &= (q_0 - \|\mathbf{q}\|^2)\vec{v} + 2(\vec{q}\cdot\vec{v})\vec{q} + 2q_0(\vec{q}\times\vec{v}) && (2.47) \end{aligned}$$

where $\mathbf{v} = \begin{bmatrix} 0 \\ \vec{v} \end{bmatrix}$ is a pure quaternion built from $\vec{v}$ by adding a zero real part. Observations:

- The quaternion operator does not change the length of the vector $\vec{v}$,

$$\|L_{\mathbf{q}}(\vec{v})\| = \|\mathbf{q} \otimes \mathbf{v} \otimes \mathbf{q}^*\| = \|\mathbf{q}\| \otimes \|\mathbf{v}\| \otimes \|\mathbf{q}^*\| = \|\vec{v}\|. \tag{2.48}$$

- The direction of $\vec{v}$, if along $\mathbf{q}$, is left unchange by the operator $L_{\mathbf{q}}$. To verify this, we let $\vec{v} = b\vec{s}$ where b denote a constant, then

$$L_{\mathbf{q}}(\vec{v}) = \mathbf{q} \otimes \mathbf{v} \otimes \mathbf{q}^* = b\mathbf{q}. \tag{2.49}$$

  Essentially, any vector along $\mathbf{q}$ does not change under $L_{\mathbf{q}}$.

Note that the operator $L_{\mathbf{q}}$ acts like a rotation about $\mathbf{q}$, which can be established in the next theorem.

Theorem 5
For any unit quaternion

$$\mathbf{q} = q_0 + \vec{q} = \cos\frac{\gamma}{2} + \vec{\alpha}\sin\frac{\gamma}{2}, \tag{2.50}$$

and for any vector $\vec{v} \in \mathbb{R}^3$ *the action of the operator*

$$L_{\mathbf{q}}(\vec{v}) = \mathbf{q} \otimes \mathbf{v} \otimes \mathbf{q}^* \tag{2.51}$$

on $\vec{v}$ *is equivalent to a rotation of the vector through an angle* γ *around* $\vec{\alpha}$ *as the axis of rotation.*

Another interpretation of this theorem is that if $\vec{v}$ is in a frame of reference $\mathcal{F}_1$ and the resultant vector $\vec{v}' = L_{\mathbf{q}}(\vec{v})$ relative to another reference frame $\mathcal{F}_2$ then $\mathbf{q}$ represents a rotation of $\mathcal{F}_2$ relative to $\mathcal{F}_1$.

From the equation (2.45) we know that if $\mathbf{q}$ is a unit quaternion then $\mathbf{q} = \exp\left(\frac{\vec{\gamma}}{2}\right)$, from here we see that given an axis-angle representation we can transform it into a quaternion by applying the exponential function. Conversely, there is a logarithm function that maps a quaternion to its axis-angle representation through mapping

$$\mathbf{q} \mapsto \vec{\gamma} = 2\ln\mathbf{q}, \tag{2.52}$$

where the logarithm is defined as

$$\ln\mathbf{q} = \begin{cases} \dfrac{\vec{q}}{\|\vec{q}\|}\arccos(q_0), & if \quad \|\vec{q}\| \neq 0 \\ (0,0,0)^T, & if \quad \|\vec{q}\| = 0 \end{cases} \tag{2.53}$$

An important equation known in the literature is the kinematic kinematic equation of a rigid body. This relates the attitude and the angular velocity of the vehicle by

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{q} \otimes \mathbf{\Omega}, \tag{2.54}$$

where $\mathbf{\Omega} = \begin{bmatrix} 0 \\ \vec{\Omega} \end{bmatrix}$. From the kinematic equation it is possible to deduce a relationship between the angular velocity of the vehicle and the axis-angle representation of the vehicle as

$$\dot{\vec{\gamma}} = \vec{\Omega}. \tag{2.55}$$

2.4.2 Rigid body dynamic modeling

Recall the assumptions we have used to establish the models. Denote by $\mathcal{I}$ the inertial frame $\mathcal{B}$ the body frame attached to the vehicle. Let $\vec{\xi} \in \mathbb{R}^3$ be the drone position in $\mathcal{I}$, $\dot{\vec{\xi}}$ denotes its linear velocity, $\mathbf{q} = q_0 + (q_1, q_2, q_3)^T$ defines a unit quaternion representing the orientation of the vehicle in $\mathcal{I}$ and $\vec{\Omega} = (\omega_x, \omega_y, \omega_z)^T$ describes the rotational speed in $\mathcal{B}$ on the center of mass of the drone.

Using the Newton-Lagrange equations, the dynamics of any rigid body using unit quaternions to represent the rotational dynamics is given by

$$m\ddot{\vec{\xi}} = \vec{F}_I \tag{2.56}$$

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{q} \otimes \mathbf{\Omega} \tag{2.57}$$

$$J\dot{\vec{\Omega}} = \vec{\tau} - \vec{\Omega} \times J\vec{\Omega}. \tag{2.58}$$

where J is the inertia matrix, $\mathbf{\Omega} = \begin{bmatrix} 0 \\ \vec{\Omega} \end{bmatrix}$ denotes the pure quaternion representation of rotational speed, $\vec{\tau}$ represents the total torque, both with respect to the body frame, and $\vec{F}_I$ defines the total force applied to the body in the inertial frame.

The quadrotor consists on four parallel rotors with blades. The direction of the rotation of each blade is selected such that all torques on the rotor cancel out in stationary flight.

We can assume that some effects as blade flapping and the misalignment on the motors axes could be considered small enough. Thus we can consider the total thrust force generated by the rotating blades $\vec{F}_{th} = (0, 0, \sum_{i=1}^{4} f_i)^T$, and the total torque can be expressed by

$$\vec{\tau} = \begin{bmatrix} l(f_1 + f_4 - f_2 - f_3) \\ l(f_1 + f_2 - f_3 - f_4) \\ \sum_{i=1}^{4}(-1)^{i+1}\tau_i \end{bmatrix} \tag{2.59}$$

where l symbolize the distance of the mass center to the motor. We observe that $\vec{F}_{th}$ and $\vec{\tau}$ act on $\mathcal{B}$, then applying the operator $L_{\mathbf{q}}$ in (2.51), it is easy to change their reference frame to $\mathcal{I}$. Thus, the dynamical model can be written as

$$m\ddot{\vec{\xi}} = \vec{F}_{th}^{I} \tag{2.60}$$

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{q} \otimes \mathbf{\Omega} \tag{2.61}$$

$$J\dot{\vec{\Omega}} = \vec{\tau} - \vec{\Omega} \times J\vec{\Omega}. \tag{2.62}$$

where $\vec{g} = (0, 0, -g)^T$ is the gravity vector and $\vec{F}_{th}^{I} = L_{\mathbf{q}}(\vec{F}_{th}) + m\vec{g}$. We observe the dynamics of the translational and rotational model are coupled, due to the orientation of $\vec{F}_{th}^{I}$ depending on the vehicle's attitude $\mathbf{q}$. Nevertheless, using an appropriate approach and some properties of unit quaternions, the quadrotor can be easily stabilized despite its underactuated nature.

2.4.3 Decoupling the vehicle dynamics

Since the attitude sub-system of the quadrotor is completely actuated, we address it in this subsection.

From (2.60)-(2.62), the rotational dynamics can be expressed as

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{q} \otimes \mathbf{\Omega} \tag{2.63}$$

$$J\dot{\vec{\Omega}} = \vec{\tau} - \vec{\Omega} \times J\vec{\Omega}. \tag{2.64}$$

Applying the logarithmic mapping (2.53) to the attitude, the rotational model in (2.63)-(2.64) of the aerial vehicle could be expressed as

$$\dot{\vec{\gamma}} = \vec{\Omega} \tag{2.65}$$

$$J\dot{\vec{\Omega}} = \vec{\tau} - \vec{\Omega} \times J\vec{\Omega}. \tag{2.66}$$

where $\vec{\gamma} = 2\ln \mathbf{q}$.

The objective is to force the system to have a linear behavior, thus the terms J^{-1} and $\vec{\Omega} \times J\vec{\Omega}$ can be compensated using an appropriate $\vec{\tau} = J\vec{\tau}_u + \vec{\Omega} \times J\vec{\Omega}$ with $\vec{\tau}_u$ as the new control input. Then, (2.65)-(2.66) yields

$$\dot{\vec{\gamma}} = \vec{\Omega} \tag{2.67}$$
$$\dot{\vec{\Omega}} = \vec{\tau}_u, \tag{2.68}$$

or in matrix form,

$$\begin{bmatrix} \dot{\vec{\gamma}} \\ \dot{\vec{\Omega}} \end{bmatrix} = \begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix} \begin{bmatrix} \vec{\gamma} \\ \vec{\Omega} \end{bmatrix} + \begin{bmatrix} 0_{3\times3} \\ I_{3\times3} \end{bmatrix} \vec{\tau}_u, \tag{2.69}$$

where $0_{3\times3}$ and $I_{3\times3}$ denotes the zero and identity matrices.

If (2.67)-(2.68) is stabilized using any appropriate controller $\vec{\tau}_u$, then the axis-angle orientation $\vec{\gamma}$ and its angular velocity $\vec{\Omega}$ will converge to zero, which means the quaternion attitude will converge to $\mathbf{q}_1 = 1 + (0,0,0)^T$.

Given a desired attitude trajectory defined by a quaternion $\mathbf{q}_d$ and its angular velocity $\vec{\Omega}_d$, then (2.63)-(2.64) can be defined in terms of the error quaternion $\mathbf{q}_e := \mathbf{q}_d^* \otimes \mathbf{q}$ and its angular velocity $\vec{\Omega}_e$ as

$$\dot{\mathbf{q}}_e = \frac{1}{2}\boldsymbol{q}_e \otimes \vec{\Omega}_e \tag{2.70}$$
$$J\dot{\vec{\Omega}}_e = \vec{\tau} - \vec{\Omega}_e \times J\vec{\Omega}_e. \tag{2.71}$$

if $\vec{\tau}$ is correctly designed in terms of the attitude error. Then, $\mathbf{q}_e \to \mathbf{q}_1$ implying $\mathbf{q} \to \mathbf{q}_d$.

From (2.60)-(2.62), the translational dynamics are given by

$$m\ddot{\vec{\xi}} = \vec{F}_{th}^I - \vec{g}. \tag{2.72}$$

From (2.72), a desired force $\vec{F}_{th}^I$ can easily be designed such that $\mathbf{x}_{pos}$ and $\dot{\mathbf{x}}_{pos}$ converge to zero. If a position error is defined as $\vec{\xi}_e = \vec{\xi} - \vec{\xi}_d$, where $\vec{\xi}_d$ symbolize a desired position for the quadrotor, then the translational error dynamics can be written as

$$m\ddot{\vec{\xi}}_e = \vec{F}_{th}^I - \vec{g}. \tag{2.73}$$

Consequently, if an adequate controller is designed for $\vec{F}_{th}^I$ the position error will converge to zero, meaning the quadrotor can be stabilized in any desired position.

Analyzing (2.73), it yields that the translational model can be seen as a fully actuated system, in which $\vec{F}_{th}^{I}$ can be designed to point at any direction. But considering the complete model of the quadrotor, the force that the propellers really exert depends on the attitude subsystem as seen in (2.60)-(2.62).

Define a desired force $\vec{F}_{th}^{\mathcal{I}} \in \mathbb{R}^3$ with respect to $\mathcal{I}$ which stabilizes system (2.73) at the desired position, given the direction and magnitude of such force, the attitude can be controlled using $\vec{\tau}$ such that the quadrotor thrust vector $\vec{F}_{th}$ is aligned with $\vec{F}_{th}^{\mathcal{I}}$, thus orientating the quadrotor thrust in the direction required to control the translational dynamics.

This quaternion is derived from the shortest rotation between both vectors, and represented by $\mathbf{q}_t$.

Recalling the Euler-Rodrigues formula, $\mathbf{q}_t$ is defined as

$$\mathbf{q}_t = \exp\left(\frac{1}{2}\gamma_d \vec{\alpha}_d\right) = \cos\frac{\gamma_d}{2} + \vec{\alpha}_d \sin\frac{\varphi_d}{2}, \tag{2.74}$$

where $\vec{\alpha}_d$ and γ_d denote respectively the axis and the angle of the shortest rotation between $\vec{F}_{th}$ and $\vec{F}_{th}^{\mathcal{I}}$. Defining $\vec{n}_z$ and $\vec{n}_u$ as the normalized vectors of the $\vec{F}_{th}$ and $\vec{F}_{th}^{\mathcal{I}}$ respectively (note that $\vec{n}_z = (0,0,1)^T$ is constant), the cross product between these vectors is defined as

$$\vec{n}_z \times \vec{n}_u = \vec{\alpha}_d \sin\gamma_d, \tag{2.75}$$

where $\vec{\alpha}_d$ is a unit vector perpendicular to the plane containing $\vec{n}_u$ and $\vec{n}_z$, while the scalar product is given by

$$\vec{n}_z \cdot \vec{n}_u = \cos\gamma_d, \tag{2.76}$$

From the definition of the quaternion product, and treating $\vec{n}_u$ and $\vec{n}_z$ as quaternions with zero-value scalar parts, then applying the quaternion product (2.36) we obtain

$$\vec{n}_u \otimes \vec{n^*}_z = -\vec{n}_u \cdot \vec{n^*}_z + \vec{n}_u \times \vec{n^*}_z. \tag{2.77}$$

By the anti-commutative property of the cross product we known that

$$\vec{n}_u \times \vec{n^*}_z = -(\vec{n^*}_z \times \vec{n}_u) \tag{2.78}$$

and since $\vec{n}_z$ have zero-value scalar part then $\vec{n_z^*} = -\vec{n}_z$. Therefore we can rewrite (2.77) as

$$\vec{n}_u \otimes \vec{n^*}_z = \vec{n}_z \cdot \vec{n}_u + \vec{n}_z \times \vec{n}_u. \tag{2.79}$$

Replacing the cross (2.75) and inner (2.76) product definitions into (2.79) we conclude that

$$\vec{n}_u \otimes \vec{n^*}_z = \cos\gamma_d + \vec{u}_d \sin\gamma_d. \tag{2.80}$$

Note that (2.80) represents twice the desired rotation needed in (2.74). In order to express (2.80) as a quaternion we applying logarithmic function in (2.74) get the axis-angle representation

$$\gamma_d \vec{\alpha}_d = 2\ln \mathbf{q}_t \tag{2.81}$$

Using the exponetial function in (2.81) and the equation (2.80) we have

$$\exp(2\ln \mathbf{q}_t) = \exp(\gamma_d \vec{\alpha}_d) = \cos\gamma_d + \vec{\alpha}_d \sin\gamma_d = \vec{n}_u \otimes \vec{n^*}_z \tag{2.82}$$

Finally, solving the equation (2.82) for $\mathbf{q}_t$ we obtain the expression

$$\mathbf{q}_t = \exp\left(\frac{\ln(\vec{n}_u \otimes \vec{n^*}_z)}{2}\right) \tag{2.83}$$

Note since $\vec{n}_z$ only acts in the vertical axis of the quadrotor, then $\mathbf{q}_t$ will only compute rotations around the xy-plane of the inertial frame. Considering $\mathbf{q}_z$ as a desired rotation around the z-axis of the vehicle's body frame, the desired quaternion can be enhanced as

$$\mathbf{q}_d = \mathbf{q}_t \otimes \mathbf{q}_z \tag{2.84}$$

Introducing (2.84) into the rotational error dynamic model from (2.70), and if τ is designed such that $\mathbf{q} \to \mathbf{q}_d$, then it implies that $\vec{F}^{\mathcal{I}}_{th} \to \vec{F}_u$ such that system (2.73) can be stabilized if $\vec{F}_u$ is correctly designed.

2.5 Quaternion state feedback controller

In this section, a quaternion-based feedback controller is designed as an example using Lyapunov theory based on the proposed model from Section 3.2.1. This controller, consist on a force in $\mathbb{R}^3$ defined as

$$\vec{F}_u = \vec{F}_{PD} - m\vec{g}, \tag{2.85}$$

where

$$\vec{F}_{PD} = -K_{pt}(\vec{\xi} - \vec{\xi}_{ref}) - K_{dt}(\dot{\vec{\xi}} - \dot{\vec{\xi}}_{ref}), \tag{2.86}$$

$K_{pt}, K_{dt} \in \mathbb{R}^{3\times3}$ denote positive control gains, and $\vec{\xi}_{ref}$ symbolizes the desired position for the quadrotor.

Then, the force computed by (2.85) is used to determine the attitude control action by introducing

$$\vec{\tau}_u = \vec{\tau}_{PD} + \vec{\Omega} \times J\vec{\Omega}, \tag{2.87}$$

where

$$\vec{\tau}_{PD} = -2K_{pa} \ln(\mathbf{q} \otimes \mathbf{q}^*_{ref}) - K_{da}(\vec{\Omega} - \vec{\Omega}_{ref}) \tag{2.88}$$

and $\mathbf{q}_{ref}$ is given by (2.84), and the control gains are denoted by positive matrices $K_{pa}, K_{da} \in \mathbb{R}^{3\times3}$. The development and stability proof of this algorithm will be explained in the following subsections.

2.5.1 Translational controller

First, from (2.73), the linear translational subsystem can be written as

$$\begin{bmatrix} \dot{\vec{\xi}}_e \\ \ddot{\vec{\xi}}_e \end{bmatrix} = \begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix} \begin{bmatrix} \vec{\xi}_e \\ \dot{\vec{\xi}}_e \end{bmatrix} + \frac{1}{m} \begin{bmatrix} 0_{3\times3} \\ I_{3\times3} \end{bmatrix} \left(\vec{F}_u + m\vec{g} \right), \tag{2.89}$$

where $0_{3\times3}$ and $I_{3\times3}$ denotes the zero and identity matrices.

Propose the following positive definite function and its derivative as

$$V_t = \frac{1}{2} \begin{bmatrix} \vec{\xi}_e & \dot{\vec{\xi}}_e \end{bmatrix} \begin{bmatrix} \vec{\xi}_e \\ \dot{\vec{\xi}}_e \end{bmatrix} \tag{2.90}$$

$$\dot{V}_t = \begin{bmatrix} \vec{\xi}_e & \dot{\vec{\xi}}_e \end{bmatrix} \left[\begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix} \begin{bmatrix} \vec{\xi}_e \\ \dot{\vec{\xi}}_e \end{bmatrix} + \frac{1}{m} \begin{bmatrix} 0_{3\times3} \\ I_{3\times3} \end{bmatrix} \left(\vec{F}_u + m\vec{g} \right) \right]. \tag{2.91}$$

Proposing

$$\vec{F}_u = -\begin{bmatrix} K_{pt} & K_{dt} \end{bmatrix} \begin{bmatrix} \vec{\xi}_e \\ \dot{\vec{\xi}}_e \end{bmatrix} - m\vec{g}, \tag{2.92}$$

where $K_{pt}, K_{dt} \in \mathbb{R}^3$ contain control gains and are defined as

$$K_{pt} = \begin{bmatrix} k_{ptx} & 0 & 0 \\ 0 & k_{pty} & 0 \\ 0 & 0 & k_{ptz} \end{bmatrix}, \quad K_{dt} = \begin{bmatrix} k_{dtx} & 0 & 0 \\ 0 & k_{dty} & 0 \\ 0 & 0 & k_{dtz} \end{bmatrix}, \tag{2.93}$$

then (2.92) can be rewritten as

$$\dot{V}_t = \begin{bmatrix} \vec{\xi}_e & \dot{\vec{\xi}}_e \end{bmatrix} \left[\begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix} + \frac{1}{m} \begin{bmatrix} 0_{3\times3} \\ I_{3\times3} \end{bmatrix} \begin{bmatrix} K_{pt} & K_{dt} \end{bmatrix} \right] \begin{bmatrix} \vec{\xi}_e \\ \dot{\vec{\xi}}_e \end{bmatrix}. \tag{2.94}$$

In order to asymptotically stabilize the subsystem, K_{pt} and K_{dt} must be chosen such that the real parts of

$$eig\left(\begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix} + \frac{1}{m}\begin{bmatrix} 0_{3\times3} \\ I_{3\times3} \end{bmatrix}\begin{bmatrix} K_{pt} & K_{dt} \end{bmatrix}\right) \tag{2.95}$$

are negative definite.

If the condition satisfied, then asymptotic stability is ensured for system (2.89) since

$$V_t > 0, \quad \dot{V}_t < 0, \text{ for all } \vec{\xi}_e, \dot{\vec{\xi}}_e \neq \vec{0}. \tag{2.96}$$

2.5.2 Rotational controller

Considering $\mathbf{q}_e := \mathbf{q}^*_{ref} \otimes \mathbf{q}$ and $\vec{\gamma}_e = 2\ln(\mathbf{q}_e)$, $\dot{\vec{\gamma}}_e = \vec{\Omega} - 2\frac{d}{dt}\ln(\mathbf{q}_{ref})$, the same methodology from Section 2.5.1 is now applied to the rotational error model in its axis-angle representation by introducing

$$\begin{bmatrix} \dot{\vec{\gamma}}_e \\ \ddot{\vec{\gamma}}_e \end{bmatrix} = \begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix}\begin{bmatrix} \vec{\gamma}_e \\ \dot{\vec{\gamma}}_e \end{bmatrix} + \begin{bmatrix} 0_{3\times3} \\ J^{-1} \end{bmatrix}\left(\vec{\tau} - \dot{\vec{\gamma}}_e \times J\dot{\vec{\gamma}}_e\right), \tag{2.97}$$

Proposing a positive-definite function with its derivative as

$$V_a = \frac{1}{2}\begin{bmatrix} \vec{\gamma}_e & \dot{\vec{\gamma}}_e \end{bmatrix}\begin{bmatrix} \vec{\gamma}_e \\ \dot{\vec{\gamma}}_e \end{bmatrix} \tag{2.98}$$

$$\dot{V}_a = \begin{bmatrix} \vec{\gamma}_e & \dot{\vec{\gamma}}_e \end{bmatrix}\left[\begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix}\begin{bmatrix} \vec{\gamma}_e \\ \dot{\vec{\gamma}}_e \end{bmatrix} + \frac{1}{m}\begin{bmatrix} 0_{3\times3} \\ I_{3\times3} \end{bmatrix}\left(\vec{\tau} - \dot{\vec{\gamma}}_e \times J\dot{\vec{\gamma}}_e\right)\right]. \tag{2.99}$$

The controller can be defined as

$$\vec{\tau} = -\begin{bmatrix} K_{pa} & K_{da} \end{bmatrix}\begin{bmatrix} \vec{\gamma}_e \\ \dot{\vec{\gamma}}_e \end{bmatrix} + \dot{\vec{\gamma}}_e \times J\dot{\vec{\gamma}}_e \tag{2.100}$$

with control gains given by

$$K_{pa} = \begin{bmatrix} k_{pax} & 0 & 0 \\ 0 & k_{pay} & 0 \\ 0 & 0 & k_{paz} \end{bmatrix}, \quad K_{da} = \begin{bmatrix} k_{dax} & 0 & 0 \\ 0 & k_{day} & 0 \\ 0 & 0 & k_{daz} \end{bmatrix}. \tag{2.101}$$

Therefore, asymptotic stability for the rotational subsystem is ensured as long the real parts of

$$eig\left(\begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix} + \frac{1}{m}\begin{bmatrix} 0_{3\times3} \\ J^{-1} \end{bmatrix}\begin{bmatrix} K_{pa} & K_{da} \end{bmatrix}\right) \tag{2.102}$$

are negative such that

$$V_a > 0, \quad \dot{V}_a < 0, \text{ for all } \vec{\gamma}_e, \dot{\vec{\gamma}}_e \neq \vec{0}. \tag{2.103}$$

Introducing (2.92) into (2.83), a desired attitude is defined to compute $\mathbf{q}_e$ and $\vec{\gamma}_e$ such that the final controller expression yields (2.85) and (2.87).

2.6 Vehicle modeling conclusions

In this chapter the mathematical modeling of a quadrotor aircraft has been developed considering three different approaches: the Euler-Lagrange formalism, the Newton-Lagrange equations and the quaternion-based method (variant of the Newton-Lagrange equations). In the first two approaches the attitude of the quadcopter is described through Euler's angles while in the third method it is represented as a unit quaternion. The main advantage of this representation is to avoid the singularities inherent in the matrix of direct cosines obtained from Euler rotations.

Even though UAVs are inherently underactuated, the quaternion-based approach was introduced such that its dynamic equation can be analyzed and treated as a fully actuated system. Although its representation can appear to be less intuitive and difficult to conceptualize, the application of quaternions can really simplify dynamic models, and help in the design of better controllers.

3 Teleoperation system of a quadrotor

The numerous applications of UAVs and their wide use in the market have led the scientific community to investigate new ways to keep the human operator in the control loop. One solution is the use of drone teleoperation systems. However, these systems imply several challenges in terms of graphical user interaction, control, time delay, system ergonomics, mental overload, among others.

In this Chapter, a new teleoperation scheme for a drone is proposed. In this system, the master robot is a virtual drone on a local computer that is controlled through a joystick, the slave robot is a quadrotor vehicle in a remote environment, both environments are linked using UDP communication channel. The information about the states of the virtual drone are sent to the real drone to imitate the movements in the remote environment. The states of the real quadrotor are sent to the local environment where they are represented by a virtual object.

The proposed graphical user interface is based on a virtual telepresence approach. That is, the remote environment is virtually recreated in order for the user to feel immersed in the remote environment. This approach allows reducing computational resources since it is not necessary to take images from the drone camera or other devices.

The outline of this Chapter is the following: in Section 3.1 the main problems in common piloting of a drone are described. In Section 3.2 the teleoperation system modules are presented. In Subsection 3.2.1 the dynamics of the drone is introduced. The virtual and real vehicle controls are presented in subsections 3.2.2 and 3.2.3 respectively. In Section 3.3 the validation of the teleoperation scheme is introduced. In Section 3.4 the results of the implementation of the scheme in real time are described. Finally, in Section 3.5 presents some conclusions of the chapter.

3.1 Problem statement

Piloting a drone in direct view is a difficult task because it has six DoF, unlike a car that moves in a plane, UAVs move in a 3-dimensional space. Also, controlling a drone is not an intuitive task when the user loses the spatial orientation. For example, when the drone is directed towards the pilot as the command is in the opposite direction, see Figure 3.1. Maneuvering a drone is a highly complex task as can be seen in drones racing competitions. Then, performing a task with a drone for a beginner pilot can generate a high mental overload. To address this problem, some authors have implemented ways of piloting based on markers, gesture recognition and voice, among others.

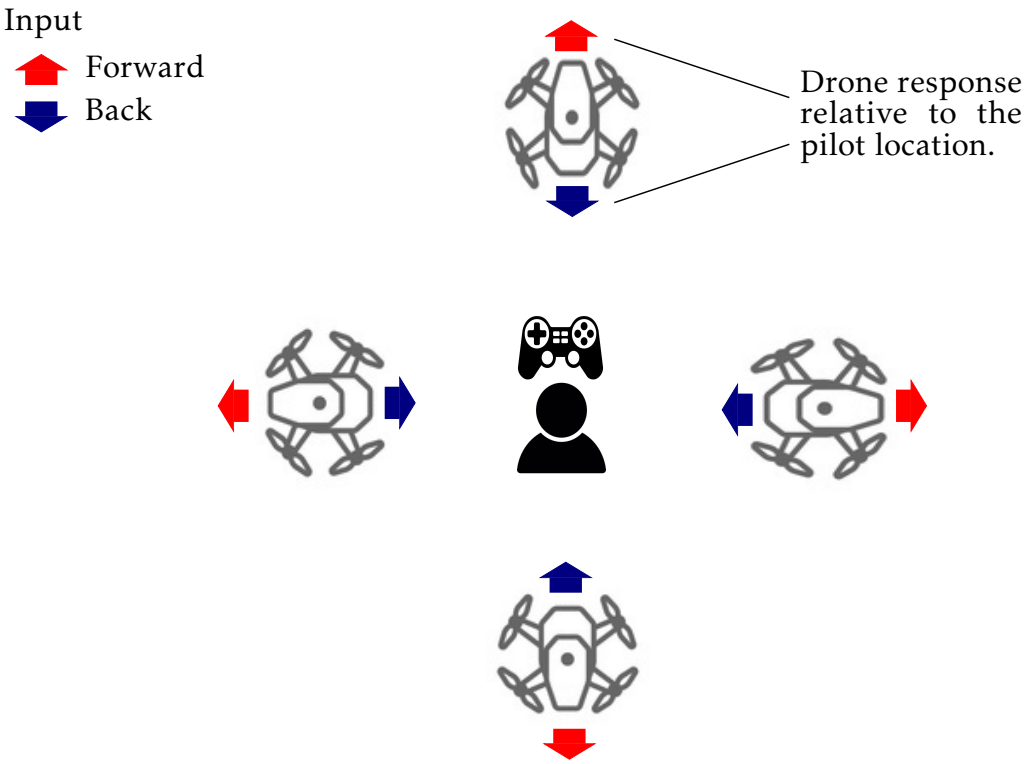


Figure 3.1 – A pilot with a fixed position loses the spatial orientation respect to the drone response.

In this work, an easy-to-use and intuitive teleoperation scheme is proposed, see Figure 3.2. In this scheme the master system is a virtual environment where a virtual drone is commanded manually via joystick. Two elements of this interface can help reduce the user's mental overload. First, this interface consists of different virtual views of the remote environment, in particular a first-person view close to the drone that can help the pilot not lose the orientation of the vehicle. Second, the information on the real states of the drone is concentrated and displayed to the user in the virtual environment. Then, the virtual environment interface is developed with different views of the virtual vehicle and is updated with the position and orientation coordinates of the real drone. The challenges in this teleoperation system are the design of a user-friendly graphical interface and to create a virtual and real space mapping to implement the controllers.

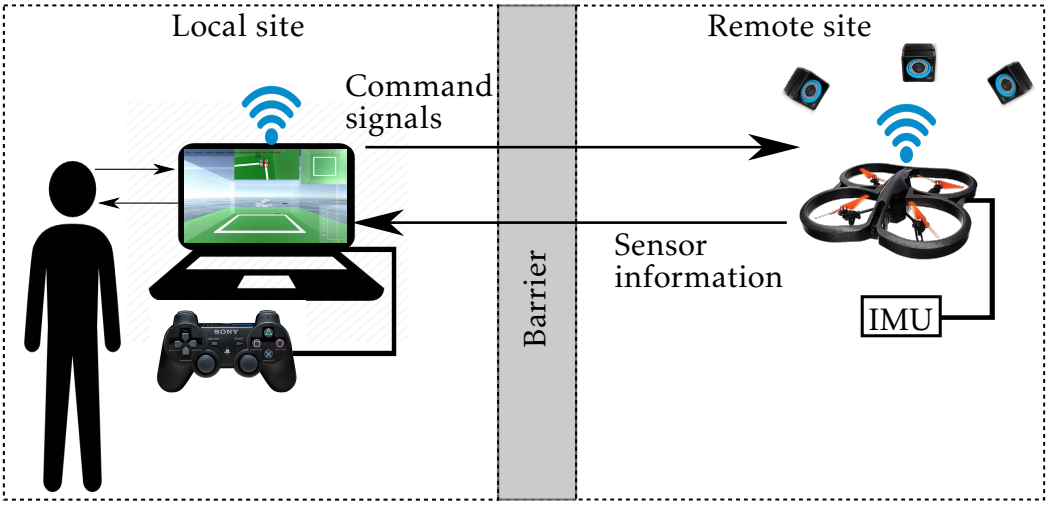


Figure 3.2 – Teleoperation system: a user pilots remotely a real drone from a virtual environment.

3.2 Teleoperation architecture

Our teleoperation system has a modular structure where the local environment simulation and the real drone are independent. This means, the user can teleoperate a drone in real time through low-level actions, or use only the simulation to generate some trajectories (high-level) that can then be sent to the real drone to execute it. In the following each module of the system will be described.

3.2.1 Quadrotor dynamic model

Different mathematical models about the dynamics of a quadrotor were described in Chapter 2. In this subsection, a brief summary of the Section 2.4 about of the quaternion-based model is presented in order to apply it to the development of the teleoperation system.

Under the hypotheses stated in the Section 2.4 we know that using the formalism of the Newton-Euler the dynamics of the UAV can be described by the equations (2.60)-(2.62).

3.2.2 Virtual representation

In this subsection, we replicate the dynamics of a real drone to simulate a virtual quadrotor. The objective is to generate realistic movements in the virtual drone to provide transparency in the system.

The variables that we will use to describe the dynamics of the virtual vehicle will be indicated with the subscript v. The virtual environment is developed in Unity 3D software. A rigid body similar to a quadrotor with the dynamics of the model (2.60)-(2.62) represents the virtual drone, see Figure 3.3. In the object, four motors are represented and the vehicle is controlled as described below.

In this environment, the inertial frame $\mathcal{I}_{ve} = \{e^{\mathcal{I}}_{ve_x}, e^{\mathcal{I}}_{ve_z}, e^{\mathcal{I}}_{ve_y}\}$ and a body frame of the vehicle $\mathcal{B}_{ve} = \{e^{\mathcal{B}}_{ve_x}, e^{\mathcal{B}}_{ve_z}, e^{\mathcal{B}}_{ve_y}\}$ is given by default (using left hand) as seen in the Figure 3.3.

To simulate the dynamics of the virtual quadrotor we consider that given a rotation in terms of the Euler angles ($\psi, \theta, \phi \in SO(3)$), it can be expressed as a quaternion using the following transformations

$$\mathbf{q}_{v_\psi} = \cos(\psi/2) + \sin(\psi/2)\mathbf{j} \tag{3.1}$$

$$\mathbf{q}_{v_\phi} = \cos(\phi/2) + \sin(\phi/2)\mathbf{i} \tag{3.2}$$

$$\mathbf{q}_{v_\theta} = \cos(\theta/2) + \sin(\theta/2)\mathbf{k}. \tag{3.3}$$

Then, to move in the virtual horizontal plane, the orientations θ and ϕ are controlled, which can be expressed as a combined quaternion $\mathbf{q}_{v_{\phi\theta}} = \mathbf{q}_{v_\phi} \otimes \mathbf{q}_{v_\theta}$ as

$$\begin{aligned}\mathbf{q}_{v_{\phi\theta}} = &\cos\left(\frac{\phi_v}{2}\right)\cos\left(\frac{\theta_v}{2}\right) + \sin\left(\frac{\phi_v}{2}\right)\cos\left(\frac{\theta_v}{2}\right)\mathbf{i} \\ &+\sin\left(\frac{\theta_v}{2}\right)\sin\left(\frac{\phi_v}{2}\right)\mathbf{j} + \cos\left(\frac{\phi_v}{2}\right)\sin\left(\frac{\theta_v}{2}\right)\mathbf{k}.\end{aligned} \tag{3.4}$$

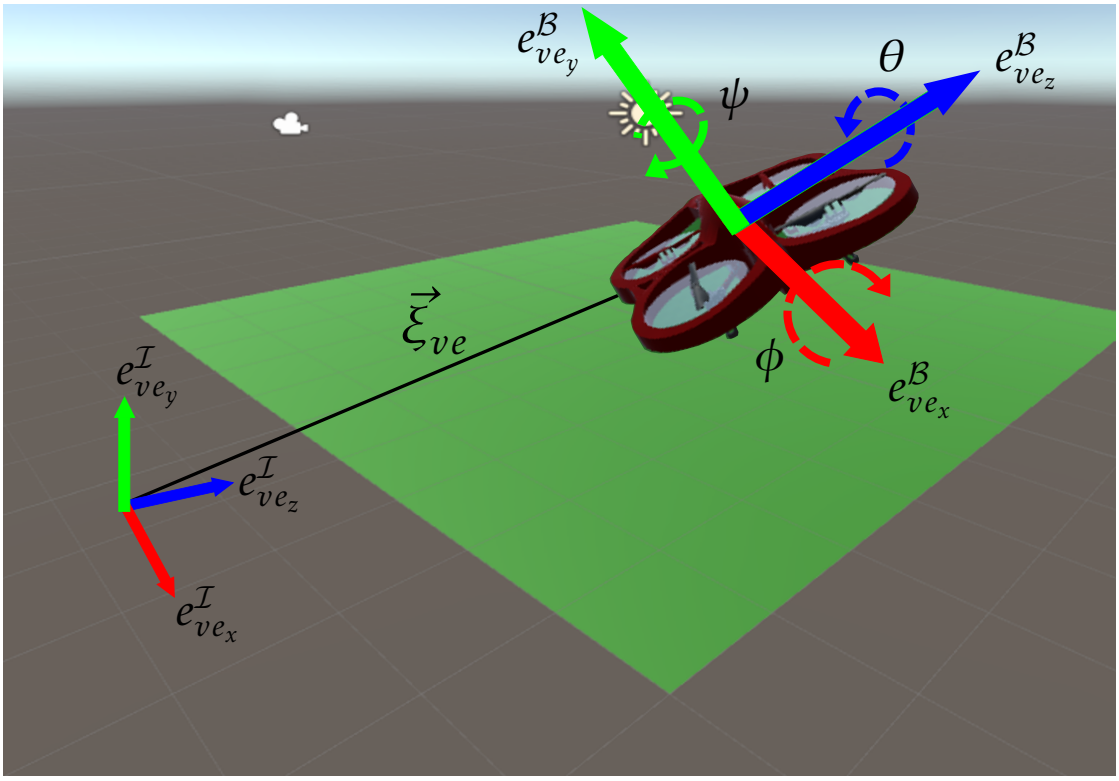


Figure 3.3 – System of coordinates in the inertial and body frames in Unity 3D.

A Proportional Integral Derivative (PID) controller for the pitch and roll angles will be used on the virtual model. The control diagram for the displacement is shown in Figure 3.4. In the scheme, we consider that a user gives orientation $\mathbf{q}_{user}$ and angular speed ω_{user} references to move the virtual vehicle, a PID control is used to reach the setpoints and generate the movement of the vehicle, finally the virtual environment will provide the orientation $\mathbf{q}_v$ and position $\vec{\xi}_v$ of the quadrotor in the virtual world.

The orientation error in pitch and roll is defined as

$$\mathbf{q}^{e}_{v_{\phi\theta}}(t) = \mathbf{q}^{d*}_{v_{\phi\theta}}(t) \otimes \mathbf{q}_{v_{\phi\theta}}(t) \tag{3.5}$$

where $\mathbf{q}_{v_{\phi\theta}}(t)$ is the measure of the pitch and roll angles given by a virtual gyroscope in Unity 3D and $\mathbf{q}^{d*}_{v_{\phi\theta}}(t)$ is the desired orientation given manually using a joystick device. The signal $\tau_{v\theta}$ that will be applied to the virtual vehicle.

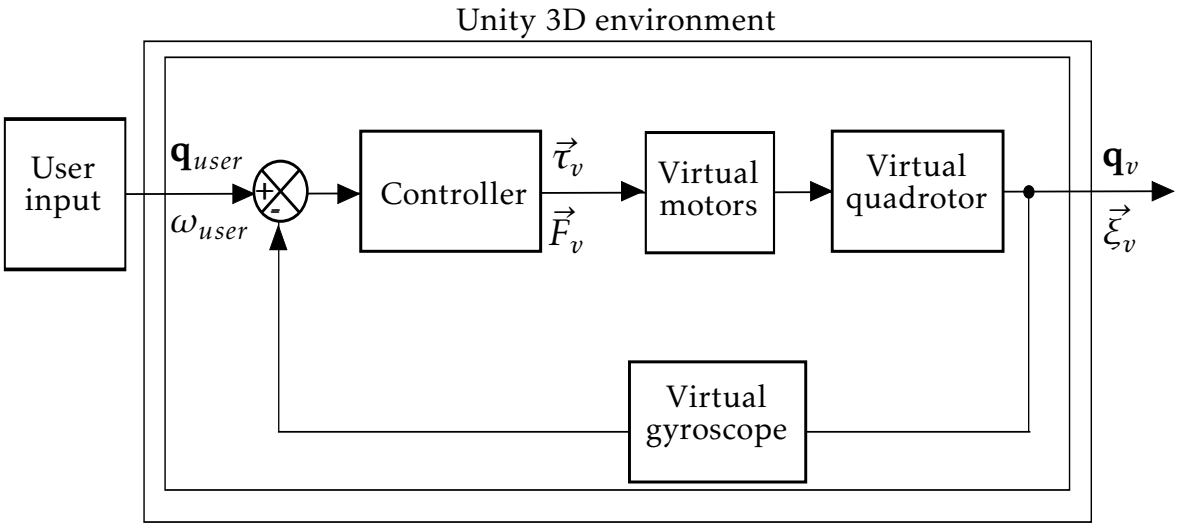


Figure 3.4 – Control scheme of the virtual drone.

The PID controller for the attitude (roll, pitch) is given by

$$\tau_{v\theta}(t) = k_{\theta p}\mathbf{q}^e_{v_{\phi\theta 3}}(t) + k_{\theta d}\left(\frac{d}{dt}\mathbf{q}^e_{v_{\phi\theta 3}}(t)\right) + k_{\theta i}\int_0^t \mathbf{q}^e_{v_{\phi\theta 3}}(\tau)d\tau, \tag{3.6}$$

$$\tau_{v\phi}(t) = k_{\phi p}\mathbf{q}^e_{v_{\phi\theta 1}}(t) + k_{\phi d}\left(\frac{d}{dt}\mathbf{q}^e_{v_{\phi\theta 1}}(t)\right) + k_{\phi i}\int_0^t \mathbf{q}^e_{v_{\phi\theta 1}}(\tau)d\tau, \tag{3.7}$$

where k_{jp}, k_{jd}, k_{ji} with $j \in \{\theta, \phi\}$ are negative constants. The yaw angle will be controlled by

$$\tau_{v\psi}(t) = k_{\omega p}\omega^e_{v_y}(t) + k_{\omega i}\int_0^t \omega^e_{v_y}(\tau)d\tau, \tag{3.8}$$

where $\omega^e_{v_y} = \omega^d_{v_y} - \omega_{v_y}$ where $\omega^d_{v_y}$ and ω_{v_y} symbolize the desired and actual angular velocity respectively, and $k_{\omega p}, k_{\omega i}$ are negative constants.

In the virtual environment the quadrotor will be controlled by the operator via joystick, by defining a desired pitch and roll angles, a yaw angular rate, and a total thrust. For safety and to avoid gimbal lock problems, the desired Euler angles are bounded at $0 < \theta_v, \phi_v \leq \pi/3$ radians.

The virtual environment can be adapted to fit any space, this means that, it can be inside a complex structure such as a building or simple as a cube or even a sphere. In addition, it is possible that the virtual drone could be bounded inside a simulated cubic scene using the following equations

$$\begin{aligned} x_1(t) \leq x_v(t) \leq x_2(t) \\ y_1(t) \leq y_v(t) \leq y_2(t) \\ z_1(t) \leq z_v(t) \leq z_2(t) \end{aligned} \tag{3.9}$$

where x_i, y_i, z_i with $i = 1, 2$ can be functions of time or constants measured in meters.

In Figure 3.5, the graphical user interface is illustrated. It is composed by (A) a frontal view where it is possible to visualize the vehicle moving in a virtual environment, (B) a top view that could help beginner pilots, (C) a fixed top view on the virtual environment to observe the location of the virtual quadrotor. Finally, (D) displays the position errors between the virtual and real drone.

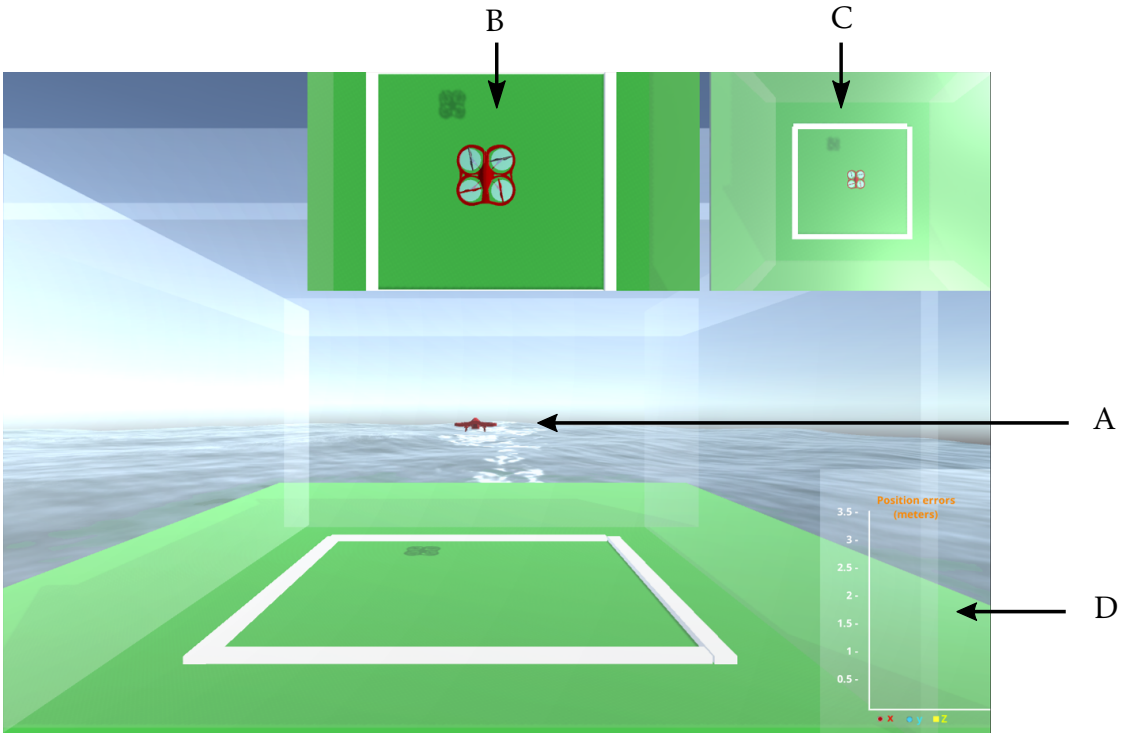


Figure 3.5 – Graphical interface of the virtual environment. (A) Virtual drone, (B) top view on the vehicle, (C) top view on the workspace, (D) position errors.

The outputs in the simulation environment are the vehicle position ξ_v in the virtual space and its attitude quaternion $\mathbf{q}_v$ that will be sent to the real drone as references.

From Figure 3.6 observe that the virtual workspace is fed back with information coming from the real drone. An additional virtual object will be introduced (called *phantom drone*), which represents the real states of the real drone.

The virtual and real environments use different coordinate systems, this means that the x, y, and z axes point respectively towards south, up, and east directions in the virtual scenario, while in real-world coordinates follow the east, north, down convention.

To match the information of both environments a mapping is defined as

$$\mathbf{q}_{r \leftarrow v} = \frac{1}{2} + \frac{1}{2}\mathbf{i} + \frac{1}{2}\mathbf{j} + \frac{1}{2}\mathbf{k}. \tag{3.10}$$

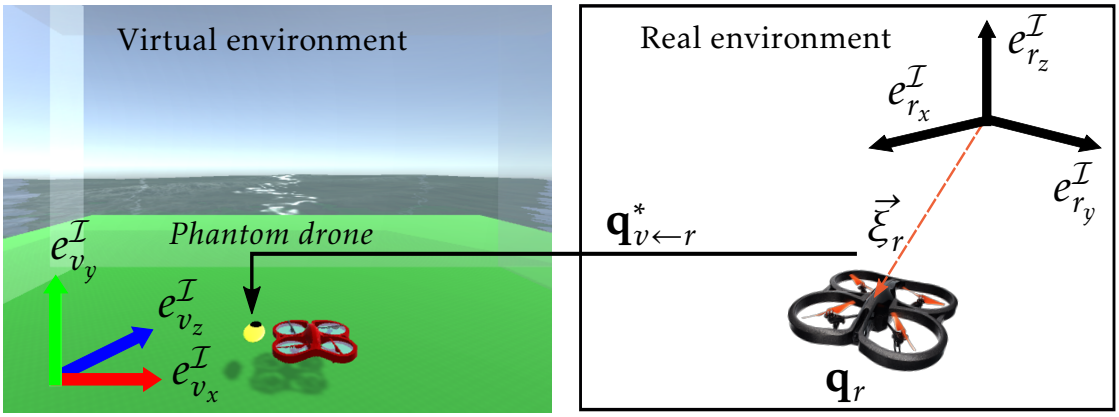


Figure 3.6 – Visual feedback information of the real drone under the transformations $\mathbf{q}^*_{v\leftarrow r}$.

Remember that a vector can be directly treated as a quaternion with zero scalar part, and viceversa, then, the position of the virtual drone represented in the real environment is given by

$$\xi_r = \mathbf{q}_{r\leftarrow v} \otimes \xi_v \otimes \mathbf{q}^*_{r\leftarrow v} \tag{3.11}$$

and the attitude is transformed as

$$\mathbf{q}_r = \mathbf{q}_{r\leftarrow v} \otimes \mathbf{q}_v \otimes \mathbf{q}^*_{r\leftarrow v}. \tag{3.12}$$

Following quaternion operations, the inverse transformation returns the remote site information, see Figure 3.2, to the virtual environment to feedback simple virtual avatar for the user.

The advantages of displaying an avatar instead of a video from the real drone camera is that it requires less communication resources, and the mental overload imposed on the operator is lighter compared to a full animated rendering of the real drone.

3.2.3 Real quadrotor control algorithm

Analogously to model (2.60)-(2.62) the real quadrotor dynamic equations are described by using the suffix r. A quaternion-based controlled was adopted from [154], to give robustness to the real-world vehicle, this algorithm is given by firstly defining a desired force $F_u \in \mathbb{R}^3$ to track the position signals from the virtual environment.

$$\begin{aligned} \vec{F}_u = \; & m\vec{g} - K_{pt}(\vec{\xi}_r - \mathbf{q}_{r\leftarrow v} \otimes \vec{\xi}_v \otimes \mathbf{q}^*_{r\leftarrow v}) \\ & -K_{dt}(\dot{\vec{\xi}}_r - \mathbf{q}_{r\leftarrow v} \otimes \dot{\vec{\xi}}_v \otimes \mathbf{q}^*_{r\leftarrow v}), \end{aligned} \tag{3.13}$$

where $K_{pt} = \text{diag}([k_{ptx} > 0, k_{pty} > 0, k_{ptz} > 0])$ and $K_{dt} = \text{diag}([k_{dtx} > 0, k_{dty} > 0, k_{dtz} > 0])$ represent the proportional and derivative control gains respectively. Then, a desired quaternion rotation is computed to align the real vehicle's vertical thrust vector $\vec{F}^{\mathcal{I}}_{th_r}$ with the desired control force, this rotation is computed by introducing dot and cross products into the Euler-Rodrigues equations as

$$\mathbf{q}_t = \frac{\left(\begin{bmatrix}0\\0\\1\end{bmatrix} \cdot \vec{F}_u + \|\vec{F}_u\|\right) + \begin{bmatrix}0\\0\\1\end{bmatrix} \times \vec{F}_u}{\left\|\left(\begin{bmatrix}0\\0\\1\end{bmatrix} \cdot \vec{F}_u + \|\vec{F}_u\|\right) + \begin{bmatrix}0\\0\\1\end{bmatrix} \times \vec{F}_u\right\|}, \tag{3.14}$$

The desired quaternion rotation is then tracked by a control torque $\vec{\tau}_u \in \mathbb{R}^3$ given by

$$\begin{aligned}\vec{\tau}_u =& J_r(-2K_{pa}\ln(\mathbf{q}_{r\leftarrow v} \otimes \mathbf{q}^*_{v\psi} \otimes \mathbf{q}^*_{r\leftarrow v} \otimes \mathbf{q}^*_t \otimes \mathbf{q}_r) \\ & - K_{da}(\Omega_r)) + \Omega_r \times J_r\Omega_r,\end{aligned} \tag{3.15}$$

where $\mathbf{q}_{v\psi}$ denotes the virtual vehicle's yaw orientation, expressed as a quaternion, J_r defines the inertia matrix, $K_{pa} = \text{diag}\{[k_{pax} > 0, k_{pay} > 0, k_{paz} > 0]\}$ and $K_{da} = \text{diag}\{[k_{dax} > 0, k_{day} > 0, k_{daz} > 0]\}$.

The vehicle total thrust is driven by the absolute value of the control force, following

$$\vec{F}_{th_r} = \begin{bmatrix}0\\0\\\|\vec{F}_u\|\end{bmatrix}. \tag{3.16}$$

Remark from Figure 3.7 that the input of this controller is the position $\vec{\xi}_v$ and orientation of the virtual drone q_v that generate the output signals given by the torque $\vec{\tau}_r$ and the thrust force $\vec{F}_{th_r}$ to follow the actions of the virtual drone, a motion capture system measures the position of the vehicle and the IMU gives the orientation that will feedback the virtual environment.

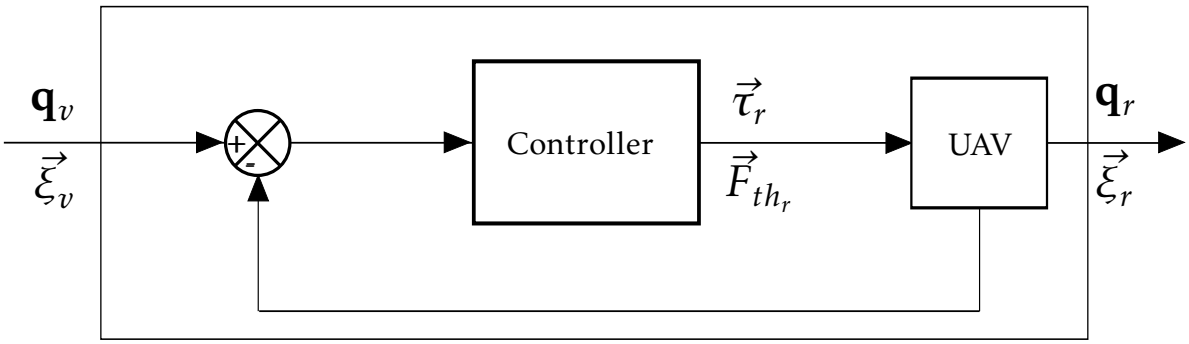


Figure 3.7 – Control scheme of the real drone.

3.3 Numerical results

The control algorithms and the teleoperation scheme were first tested using the simulation in Unity 3D and an emulator of a real drone included in the Framework Libre Air (FL-AIR) environment.

The virtual environment was designed using Unity 3D version 5.6.4 on a PC running Windows 10 operating system. The Unity 3D system is compatible with most current virtual reality and augmented reality devices. A scene was developed on this framework, including a simulated master drone.

The quadrotor vehicle (slave system) was coded using the FL-AIR framework, developed at the Heudiasyc laboratory, which includes libraries to program controllers and commands real drones from a ground station running on Linux OS (Ubuntu 18.04). FL-AIR includes an emulator of a real drone [155], which is compatible with the software that will be uploaded to the real drones.

UDP communication was used to link the computer where the virtual drone was coded with the slave UAV. Figure 3.8 illustrates both virtual environments, on the left side, the master system is shown and on the right side the slave system.

The information of the real drone feeds back the master system and is illustrated as a phantom drone.

In Figure 3.9, the position performance in $\mathbb{R}^3$ of real quadrotor is shown. Observe that the user performs some aggressive movements and the emulated drone can follow them. The controller ensures that the real drone follows a trajectory and avoids possible aggressive movement when trying to reach it. Remark in Figure 3.10 that the slave drone accurately tracks the attitude trajectory of the virtual drone.

The position error given by the performance of both drones is displayed to the user in the graphical interface, this signal is presented in Figure 3.11.

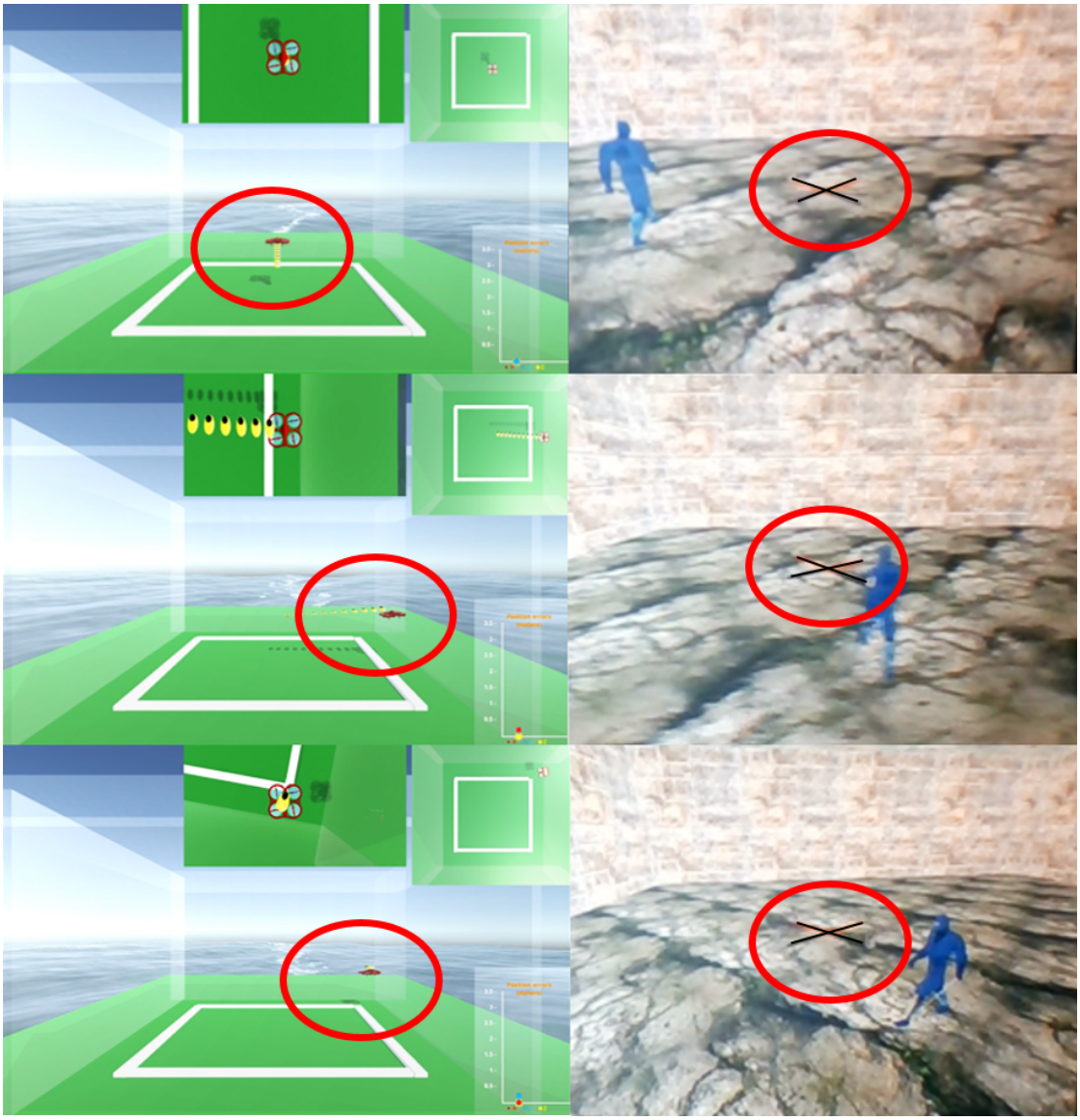


Figure 3.8 – Validation of the system using the FL-AIR simulator and the virtual environment.

The practical goal of this work is to implement the proposed scheme in a real drone which follows the dynamic of a virtual vehicle. Control algorithms and teleoperation communication were validated with simulations. Then, these scheme were migrated to a real configuration, which is described in the following section.

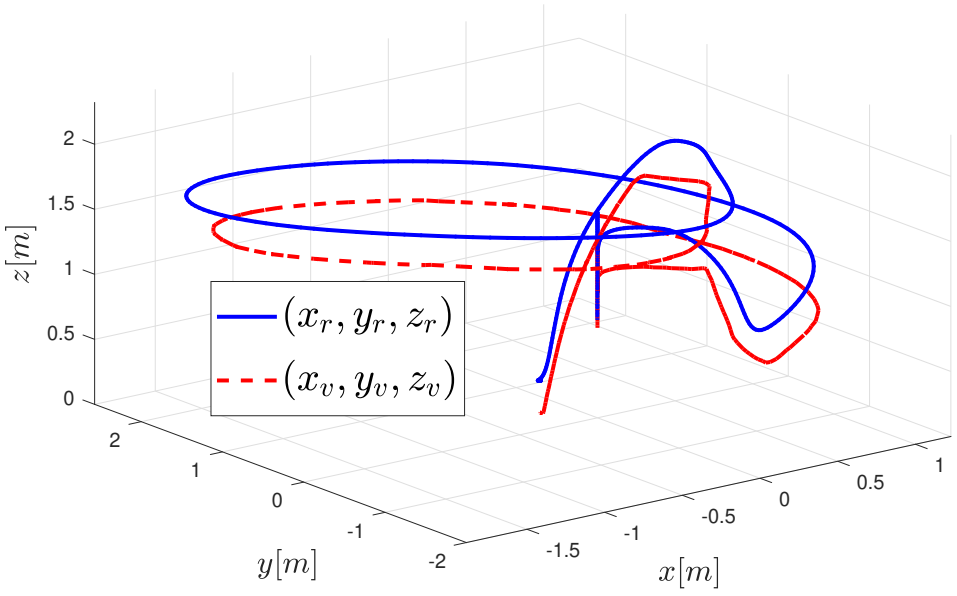


Figure 3.9 – Quadcopter performance in the virtual (x_v, y_v, z_v) and the real (x_r, y_r, z_r) scenarios.

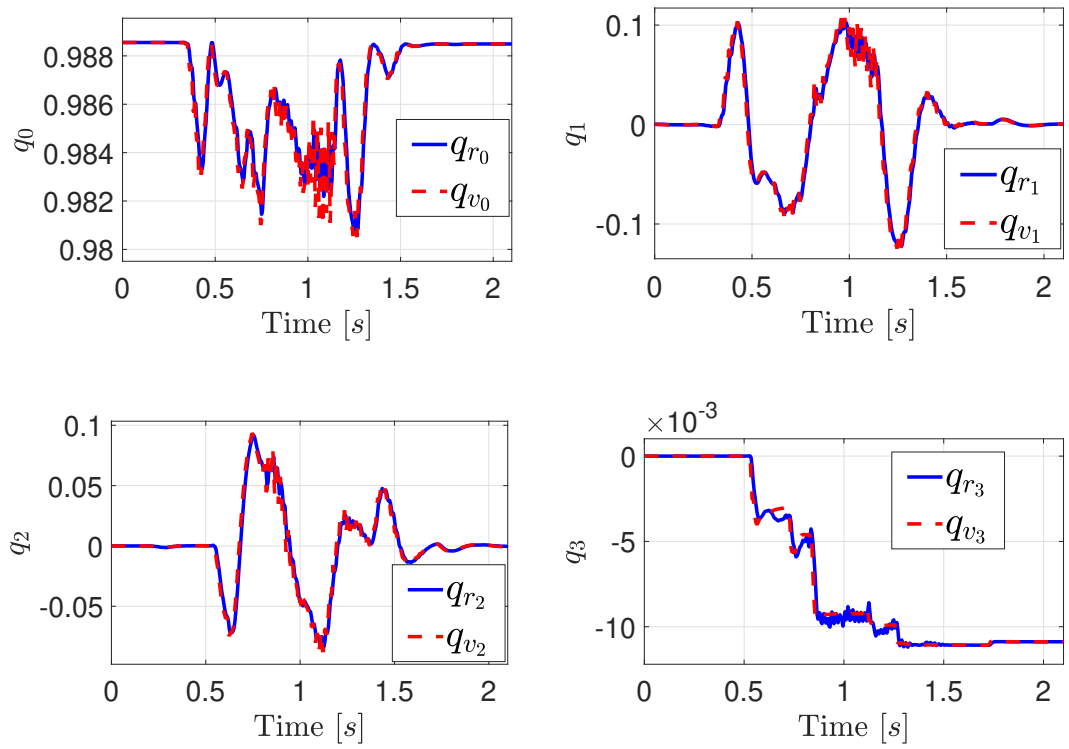


Figure 3.10 – Orientation components q_r of the slave drone and its trajectory.

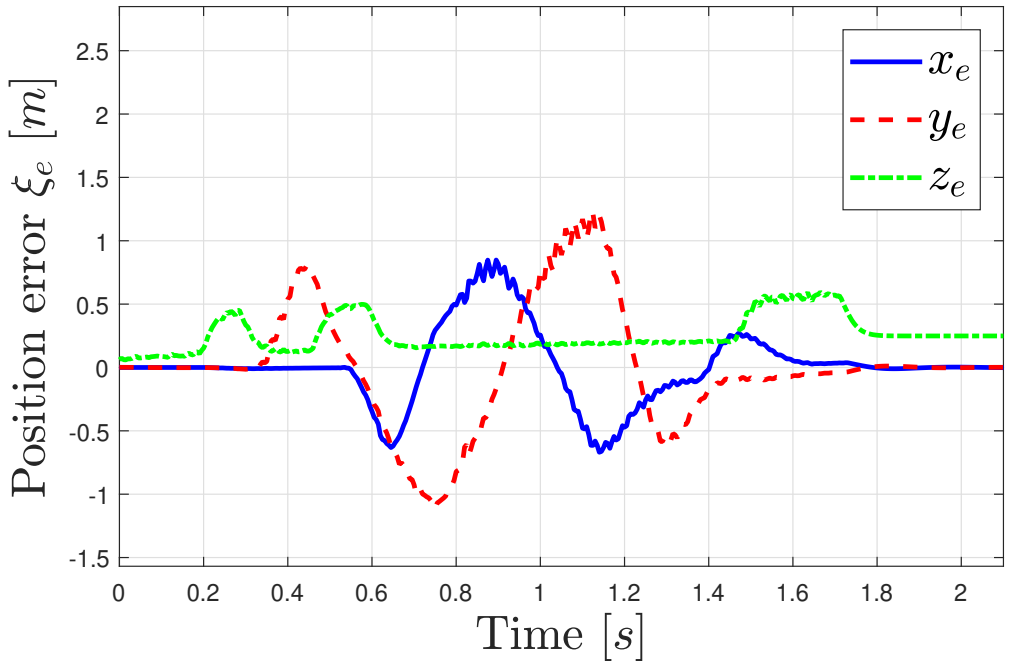


Figure 3.11 – Position errors computed using states from the virtual and real drone (emulated from FL-AIR).

3.4 Experimental results

The real time validation of our teleoperation system is performed in an indoor flight arena. This area is equipped with an Optitrack motion capture system that provides accurate information about the real position of the drone. An AR.Drone Parrot 2.0 quadrotor was used, whose factory firmware was erased and reprogrammed with our framework FL-AIR, the attitude information was measured with an Inertial Measurement Unit (IMU). The same UDP communication protocol was used, from the emulated tests only switching the virtual slave and real drone's Internet Protocol (IP) addresses. In the following, the task performed by a user to analyze the performance of the teleoperation system is detailed.

3.4.1 Real-time application

To verify the robustness of the teleoperation system, we consider the following scenario: a beginner pilot must perform the monitoring of a zone following a square trajectory in the virtual scenario. A physical barrier is imposed to prevent accidents with the real drone such that the pilot must operate only with the information feedbacked to the virtual environment, see Figure 3.12.

The translational performance between the real and virtual drones is illustrated in Figure 3.13. Notice here that the user does not perfectly perform the flight task but the proposed scheme allows the real drone to successfully perform the mission, the upper view of this mission is shown in Figure 3.14.

The proposed scheme involves a quaternion rotation for the quadrotor thrust vector to move the real vehicle, the computed attitude trajectory, and its accurate tracking is illustrated in Figure 3.15.

This test can be seen in the following video link: `https://youtu.be/DgHF7M_zcTY`

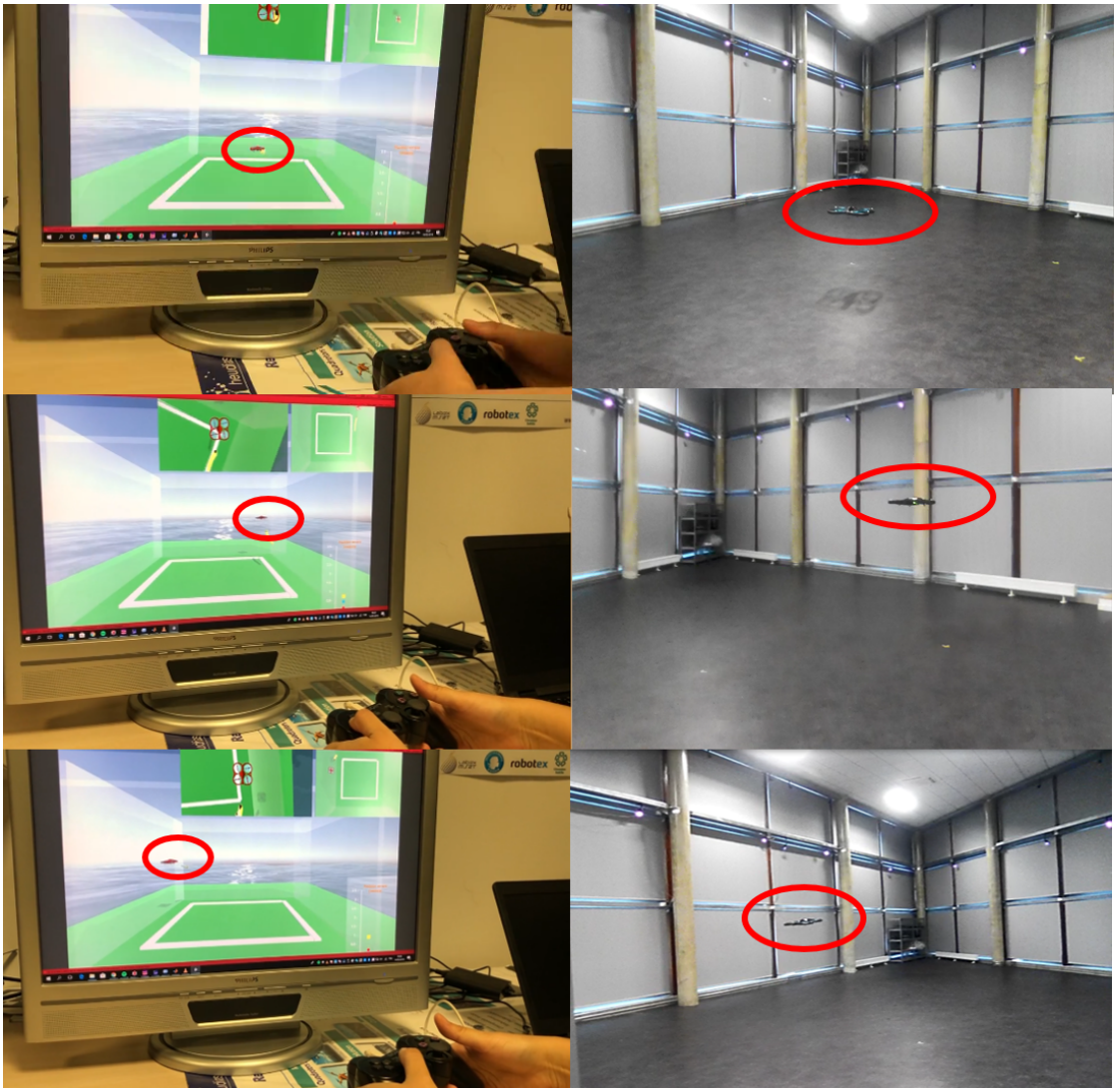


Figure 3.12 – On the left side a user pilots the virtual drone through a joystick. On the right side, a real drone imitates the behavior of the virtual vehicle.

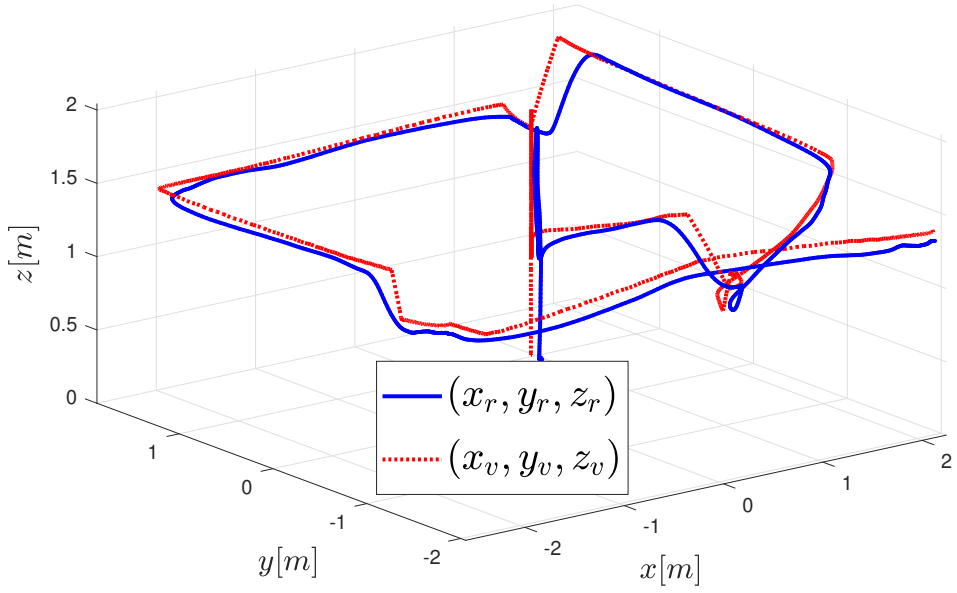


Figure 3.13 – 3D-position performance trajectory given by the user and imitated by the real drone.

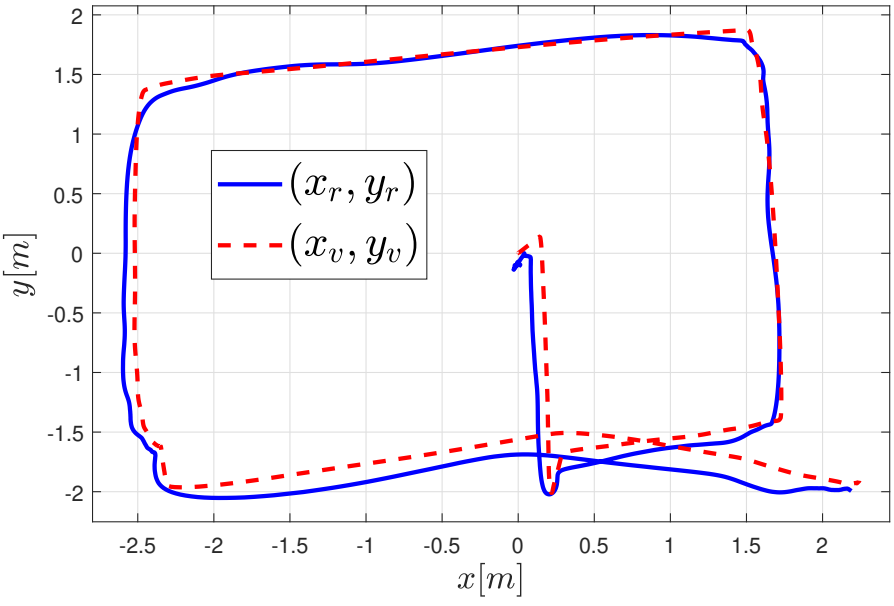


Figure 3.14 – x-y behavior of the experimental test.

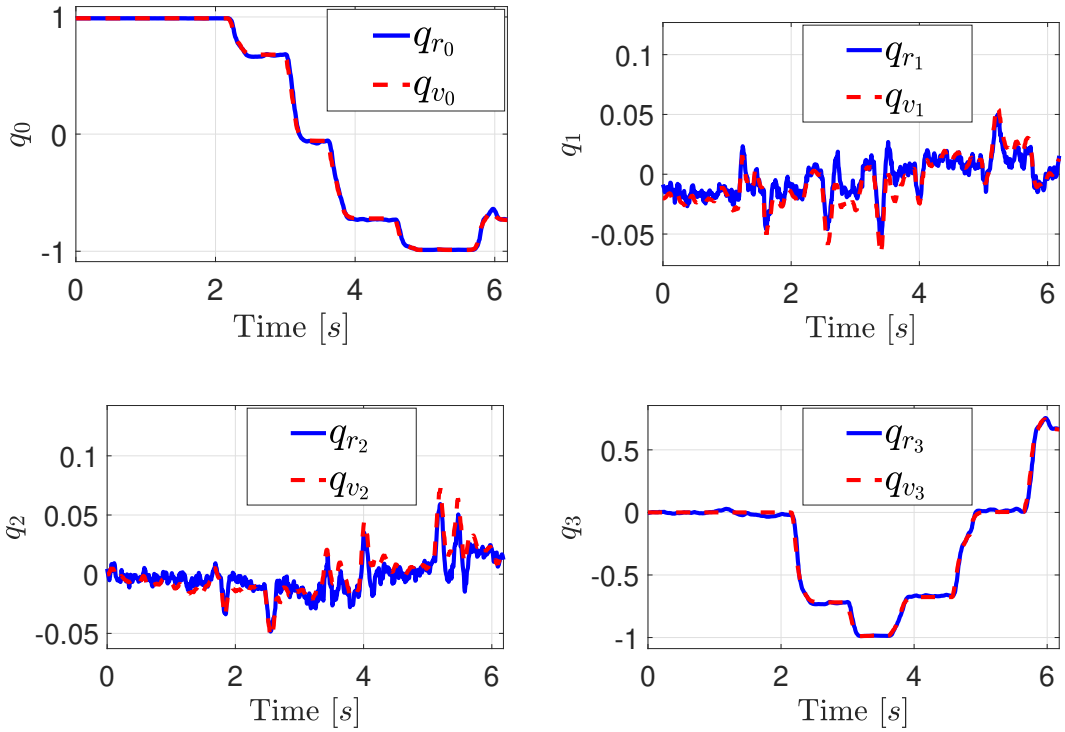


Figure 3.15 – Attitude performance of the real drone vs desired values from the virtual drone.

3.5 Conclusion

In this chapter, a new teleoperation system based on a virtual telepresence approach was presented. This system is composed of an interface that makes the task of piloting a drone simple and intuitive. This means that, the interface allows the user to be more focused on performing the task than on understanding the dynamics of the drone. The control schemes implemented allow the real drone to follow the movements of the virtual vehicle even when the pilot performs aggressive maneuvers. As well, the visual information of an avatar and the control based on quaternions generates a low computational cost. Our teleoperation scheme can be extended to an immersion system by using virtual reality glasses. This is due to the compatibility of the virtual environment application in unity 3D with the majority of virtual reality devices.

4 System with input delay

Time-delay often appears in many control systems, either in the state, the control input, or the measurements. There can be transport, communication, or measurements delays. Control systems often operate in the presence of delays, primarily due to the time it takes to acquire the information needed for desicion-making, to create control decisions, and to execute these decisions. Actuators, sensors, and field networks that are involved in feedback loops usually introduce delays. Delays are strongly involved in challenging areas of communication and information technologies deteriorating the performance of systems. Therefore, it is important to study techniques to mitigate the time delay.

This chapter is divided as follows: Section 4.1 introduces the main problems that make the study of delayed systems challenging. Section 4.2 presents one of the first approaches for solving systems with input delay: the Smith predictor, which introduced the concept of predictive technique. Section 4.3 proposes a predictor-based controller on the Fundamental Theorem of Calculus (FTC). In Sections 4.4 and 4.5 the predictor-based controller proposed is applied in the classical Euler-Lagrange model and the quaternions-based model, respectively, showing its trajectory tracking performance in simulations. Finally, Section 4.6 presents some conclusions of the chapter.

4.1 Features of TDSs

Time-Delay Systems (TDSs) are also called systems with aftereffect or dead-time hereditary systems, differential-difference equations or equations with deviating argument. They belong to the class of FDEs which are infinite-dimensional, as opposed to Ordinary Differential Equations (ODEs). This feature can be observed with the following example.

4.1.1 TDSs are infinite-dimensional

Consider the linear equation

$$\dot{x}(t) = x(t). \tag{4.1}$$

It is well known that $x(t) = e^t$ is a solution and that all solutions are of the form $x(t) = Ce^t$ where C is an arbitrary constant. We can assume that $C \in \mathbb{C}$ and then the formula obtained covers all the complex solutions of the equation. This idea is transformed into a general method to obtain solutions of linear equations with constant coefficients: propose solutions of the form

$$x(t) = e^{\lambda t}, \tag{4.2}$$

and the values λ that cancel out the characteristic polynomial associated are computed. Then, replacing (4.2) into (4.1) we have $\lambda x(t) = x(t)$ and the characteristic polynomial of (4.1) is $P(\lambda) = \lambda - 1$, with $\lambda = 1$ as the only root.

However, the situation is difficult when introducing a delay in equation (4.1). Consider the system with delay

$$\dot{x}(t) = x(t-h), \tag{4.3}$$

where h is the instant of delay time. Indeed, replacing (4.2) into (4.3) gives

$$\lambda x(t) = e^{-\lambda h} x(t), \tag{4.4}$$

then λ must be a solution of the characteristic equation $P(\lambda) = 0$, where P is not a polynomial but a transcendent function

$$P(\lambda) = \lambda - e^{-\lambda h}. \tag{4.5}$$

Since $\lambda = 0$ is not a root, the change of variable

$$z = \frac{1}{\lambda}, \tag{4.6}$$

can be applied in the equation (4.5) and gives

$$ze^{-h/z} = 1. \tag{4.7}$$

Note that the function $f(z) = ze^{-h/z}$ has an essential singularity at $z = 0$ and it does not vanish in $\mathbb{C}\backslash\{0\}$. Hence, from Great Picard Theorem it follows that $f^{-1}(1)$ has infinite elements, this means that, the solutions of the characteristic

equation form the set

$$\{\lambda_k \quad | \quad k \in \mathbb{N}, \quad |\lambda_k| \longrightarrow \infty\}. \tag{4.8}$$

Therefore, the differential equation (4.3) has complex infinite solutions [156]. With this example, one of the main features of studying TDSs has been observed.

The formulation of the initial value problem and the definition of a solution in a delayed system are different concepts from those we known in ODEs as we will see below.

4.1.2 Initial value problem

It is well known that a particular solution of a delay-free system, $\dot{x} = F(t, x)$, is defined by its initial conditions, which include an initial instant t_0 and an initial state $x_0 \in \mathbb{R}^n$. This is not case when dealing with a solution for a time-delay system. Here the knowledge of t_0 and x_0 is not sufficient even to define the value of the time derivative of $x(t)$ at the initial time instant t_0. To define a solution of a system with delay, one needs to select an initial time instant $t_0 \geq 0$ and an initial function $\varphi : [-h, 0] \longrightarrow \mathbb{R}^n$. The initial value problem is formulated as follows. Given an initial time instant $t_0 \geq 0$ and an initial function φ, find a solution of the system that satisfies the condition

$$x(t_0 + s) = \varphi(s), \quad s \in [-h, 0]. \tag{4.9}$$

The initial function φ belongs to a certain functional space [157].

4.1.3 Solution concept

Consider the simple delay equation:

$$\dot{x}(t) = -x(t - h), \quad x(t) \in \mathbb{R}, \quad h > 0, \quad t \geq 0. \tag{4.10}$$

In order to define its solution for $t \in [0, h]$, we have to define the right-hand side $x(t - h)$ for $t \in [0, h]$, which results in the initial value function

$$x(s) = \varphi(s), \quad s \in [-h, 0], \tag{4.11}$$

instead of initial value $x(0)$ for ODEs with $h = 0$. In order to find a solution to this problem, we shall use the step method initiated by Bellman [158]. First, we find a solution on $t \in [0, h]$ by solving

$$t \in [0, h], \quad \dot{x}(t) = -\varphi(t - h), \quad x(0) = \varphi(0). \tag{4.12}$$

Then we continue this procedure for $t \in [h, 2h]$, $t \in [2h, 3h]$, ... For the constant $\varphi \equiv \varphi_0$ the step method gives polynomial in t solution. Figure 4.1 shows the solutions for $h = 1$ and for the initial functions $\varphi \equiv 1$ and $\varphi = 0.5t$. In this figure we can see that several solutions that achieve the same value $x(t^*)$ at some instants t^*.

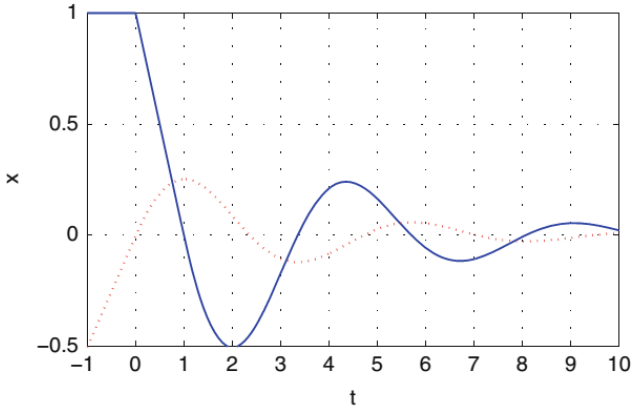


Figure 4.1 – Solutions with $h = 1$ and $\varphi \equiv 1$ (plain blue) or $\varphi = 0.5t$ (dotted red).

This is different from ODEs, i.e. from $\dot{x}(t) = -x(t)$, where through each $x(t^*)$ only one solutions passes. Therefore, in TDSs, a proper state is a function

$$x_t : [-h, 0] \longrightarrow \mathbb{R} : \quad x_t(s) = x(t + s), \quad s \in [-h, 0], \tag{4.13}$$

corresponding to the past time-interval $[t - h, t]$ [158].

These concepts and properties give an idea of the complexity of studying systems with delay. After understanding these concepts, I began the study of the Smith predictor, which is one of the most used approaches in TDSs.

4.2 Smith predictor

The Smith Predictor [137] is one of the broadest strategies for controlling linear systems with delays. The idea of this technique is to use a prediction of the state or output from the system model to compensate the delay.

Linear systems, or systems that can be linearized, with delay can be modeled as:

- Input delay

  Suposse that the delay h is known, then the system can be expressed by

$$\dot{x}(t) = Ax(t) + Bu(t - h) \tag{4.14}$$
$$y(t) = Cx(t). \tag{4.15}$$

- Output delay

Suposse that the delay h is known, then the system can be expressed by

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{4.16}$$
$$y(t) = Cx(t-h) \tag{4.17}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times 1}$, $C \in \mathbb{R}^{1\times n}$, $h \in \mathbb{Z}^+$.
The transfer function for both cases is

$$G_p(s) = \frac{y(s)}{u(s)} = G_r(s)e^{-hs}, \tag{4.18}$$

where

$$G_r(s) = C(sI - A)^{-1}B. \tag{4.19}$$

The representation in block diagram is presented in Figure 4.2.

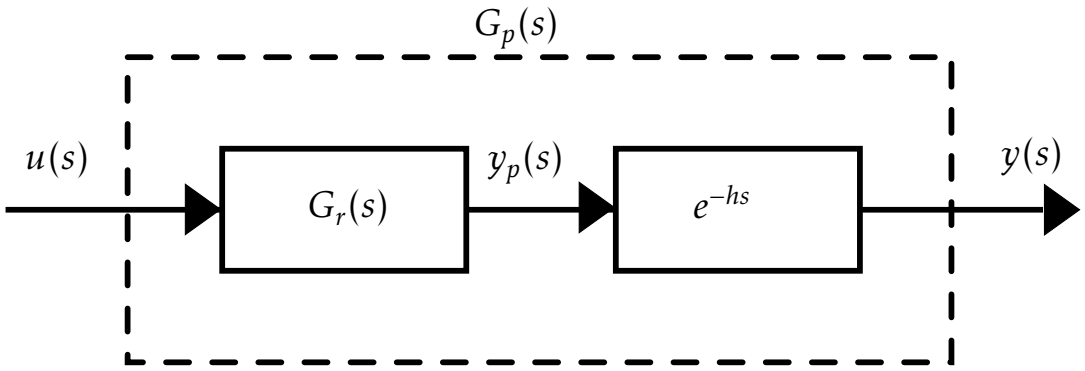


Figure 4.2 – The output is given by $y(s) = y_p(s)e^{-hs}$ where y_p is the output of the process.

If the process output $y_p(s)$ was known and it will feed-back the process, then we could design the controller without delay. But we do not know $y_p(s)$ as illustrate in Figure 4.3.

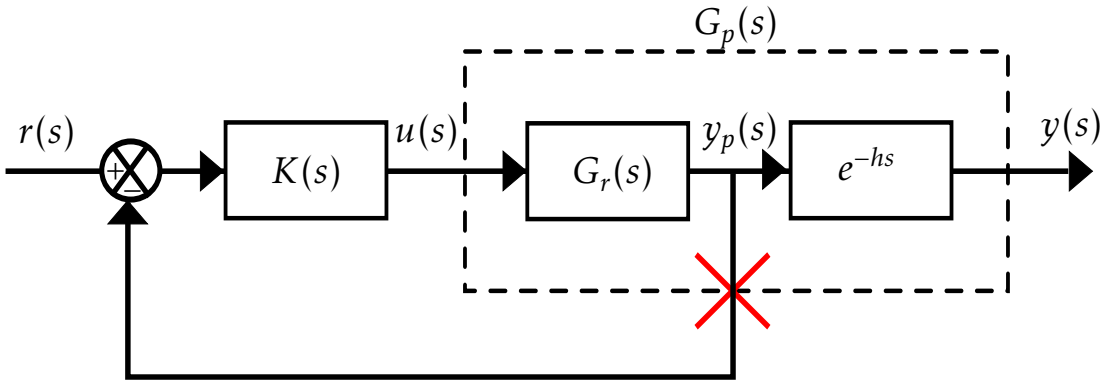


Figure 4.3 – The output y_p of the process is unknown and $K(s)$ is a controller.

Then, to solve this problem Smith proposed that given a known model of the

process

$$G(s)e^{-hs}. \tag{4.20}$$

it is possible to obtain the values $y_p(t)$ by means of a prediction $\hat{y}_p(t)$. For this, we must first know the process model without delay $G(s)$.

After applying the control action we would obtain the prediction of the output without delay

$$\hat{y}_p(s) = G(s)u(s) \tag{4.21}$$

This output prediction would be obtained h instants of time (because the output of the real process is delayed) with respect to the output of the real process, see Figure 4.4.

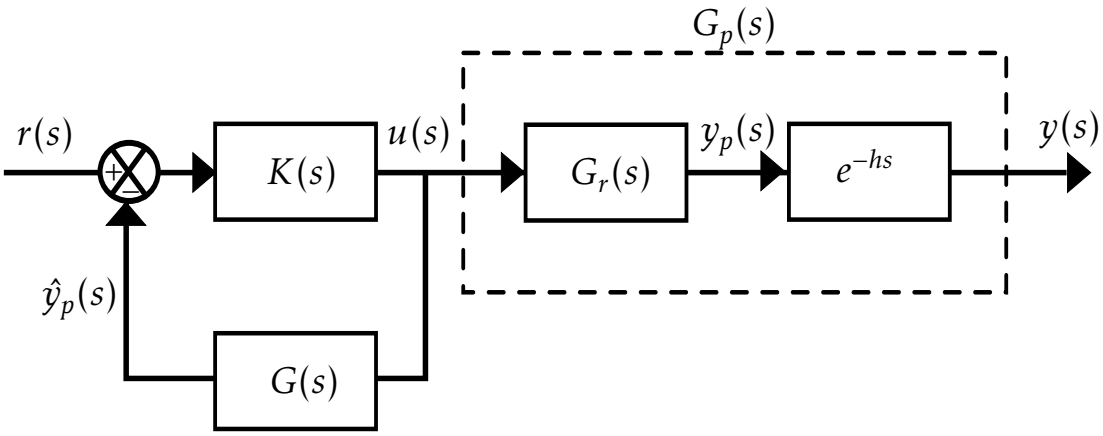


Figure 4.4 – Diagram of the Smith solution.

To obtain a feedback system, we will compare the controlled variable $y(t)$ with the delayed prediction h instants of time $\hat{y}_p(t-h)$, see Figure 4.5. The perturbations or differences between the model and the real process will feed the control system as a corrective factor on the differences in the prediction.

Consider the following example to illustrate these ideas.

4.2.1 Academic example

Let us suppose a stable process given by

$$G_r(s) = \frac{5.6e^{-93.9s}}{40.2s+1}. \tag{4.22}$$

Also suppose that the model is a perfect model of the system, this means that,

$$G(s)e^{-hs} = G_r(s)e^{-h_r s}.$$

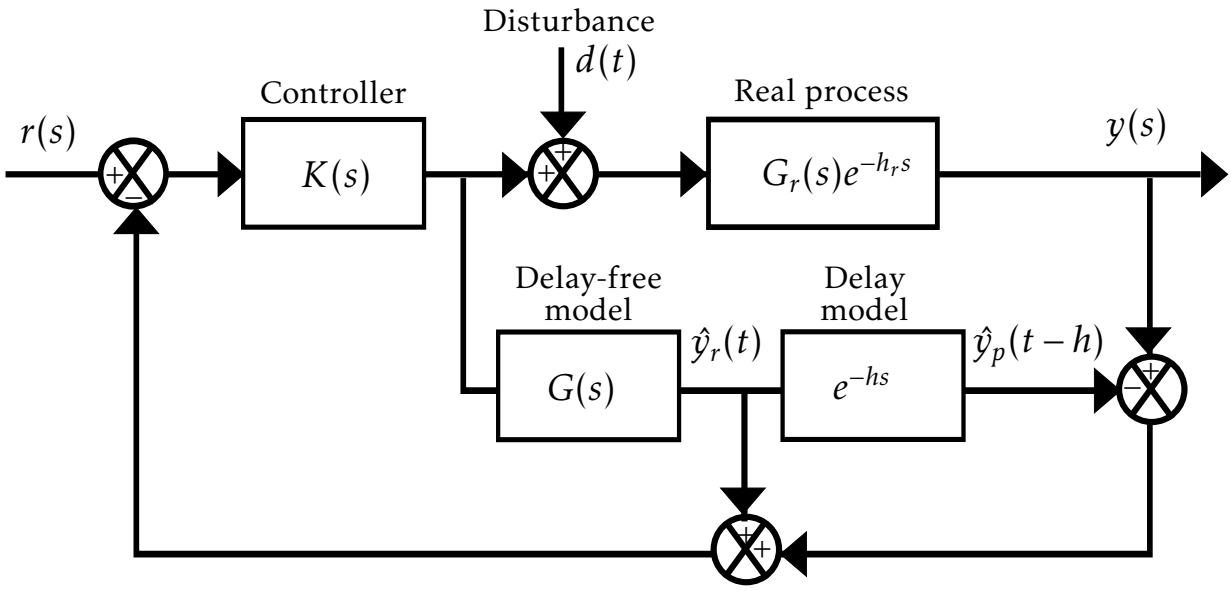


Figure 4.5 – Smith predictor scheme.

Applying a PI controller

$$k_p\left(1+\frac{1}{T_i s}\right), \tag{4.23}$$

where $k_p = 0.0501$ and $T_i = 47.3$ seconds, we obtain the response depicted in Figure 4.6. We observe that this controller is able to compensate the delay in the system.

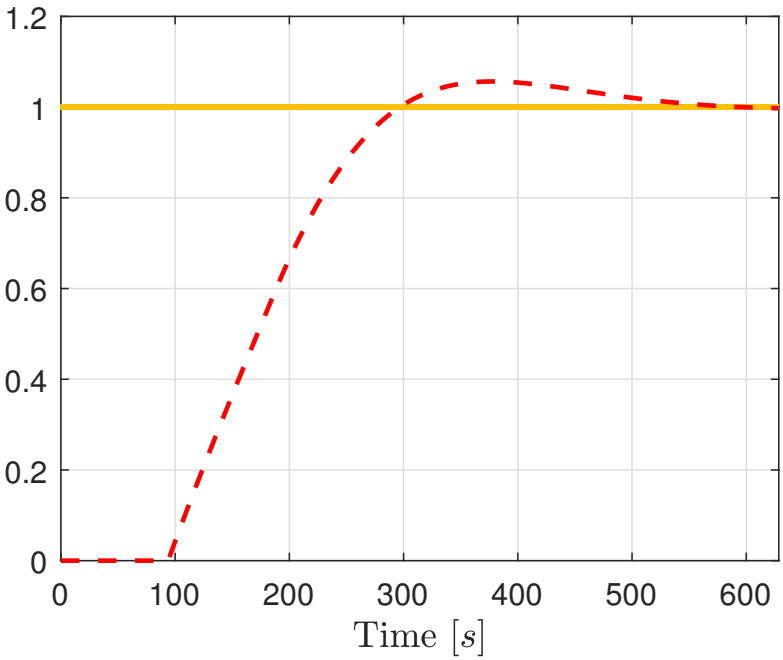


Figure 4.6 – Response of the PI controller (dotted red line).

Figure 4.7 shows the response when applying the Smith predictor to the system compared to PI controller. It is clear that the Smith predictor has a better performance with respect to this controller.

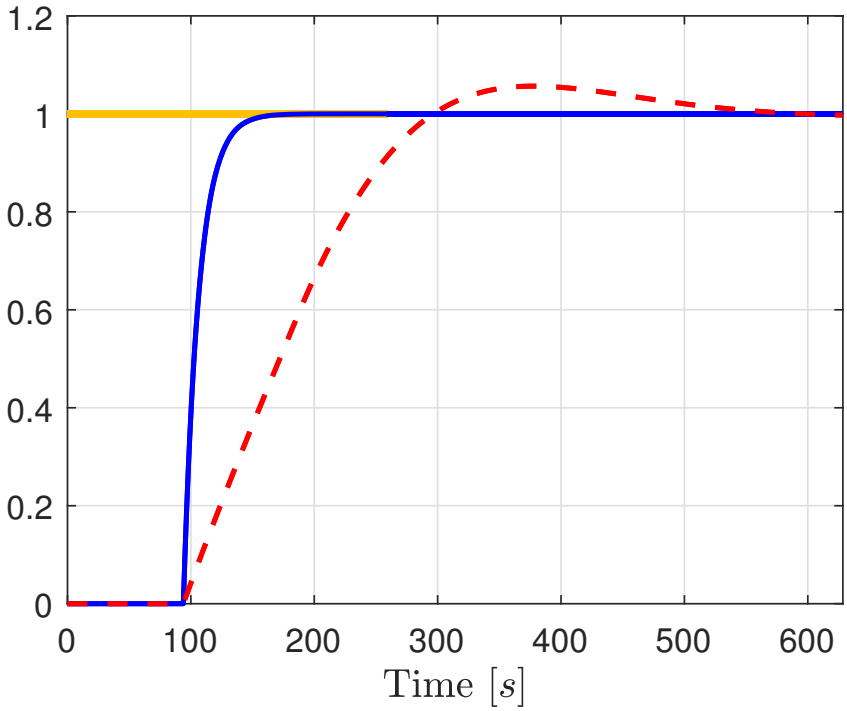


Figure 4.7 – PI controller response (dotted red line) and Smith predictor response (blue line).

It has been observed that some assumptions of the Smith predictor are that the time delay is known and that the system model is known and identical to the real one. The ideas of the Smith predictor design can be extended to problems that can be modeled as a cascade of systems. This gives rise to other types of predictors as will be seen in Section 4.3.

4.3 Predictor-based control

Many predictor-based techniques have been developed. However, they commonly have the disadvantage that the algorithms developed are complex and their implementation is difficult. In this section, a predictor-like technique is presented. This approach allows controlling a delay-free system with a regular controller neglecting the delay. The main result is the following and later results of its computational implementation will be introduced.

Lemma 1
Consider the delayed chain of integrators

$$\dot{x}_1 = x_2 \tag{4.24}$$
$$\dot{x}_2 = x_3 \tag{4.25}$$
$$\vdots = \vdots \tag{4.26}$$
$$\dot{x}_{n-1} = x_n \tag{4.27}$$
$$\dot{x}_n = u(t-h) \tag{4.28}$$

where $u(t-h)$ is the control input delayed by h units of time. The controller

$$u(t) = \vec{K}^T \vec{x}_{pre}(t) \tag{4.29}$$

stabilize the system (4.24)-(4.28). Here K is a vector gain stabilizing the delay-free system and $\vec{x}_{pre}(t)$ is the predicted state defined as

$$\vec{x}_{pre}(t) = e^{hA}\vec{x}(t-h) + \varphi(u(t)) \tag{4.30}$$

where

$$e^{hA} = \begin{pmatrix} 1 & h & \frac{h^2}{2} & \cdots & \frac{h^{n-1}}{(n-1)!} \\ 0 & 1 & h & \cdots & \frac{h^{n-2}}{(n-2)!} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & h \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \tag{4.31}$$

and

$$\varphi(u(t)) = \begin{pmatrix} \int_{t-h}^{t}\int_{t-h}^{t_1}\cdots\int_{t-h}^{t_{n-1}} u(t_n)dt_n dt_{n-1}\cdots dt_1 \\ \int_{t-h}^{t}\int_{t-h}^{t_1}\cdots\int_{t-h}^{t_{n-2}} u(t_{n-1})dt_{n-1}dt_{n-2}\cdots dt_1 \\ \vdots \\ \int_{t-h}^{t}\int_{t-h}^{t_1} u(t_2)dt_2 dt_1 \\ \int_{t-h}^{t} u(t_1)dt_1 \end{pmatrix} \tag{4.32}$$

This result is based in the FTC. This result was applied in the following cases. First, it was applied to a simple case, a system given by a double integrator described in Subsection 4.3.1. It was then implemented in the longitudinal dynamics of a quadrotor presented in Subsection 4.4.1. Finally, in Subsection 4.4.2 the results of applying the predictor to the system with delayed altitude are shown.

4.3.1 Double integrator with input delay

Consider a second-order system with input delay u

$$\ddot{x}(t) = u(t-h) \tag{4.33}$$
$$y(t) = x(t), \tag{4.34}$$

and the initial conditions

$$x(0) = 0, \tag{4.35}$$
$$u(t) = 0, \quad t \in [-h, 0], \quad h > 0 \tag{4.36}$$

where h is a constant delay.

Rewriting $x_1(t) = x(t)$ and $x_2(t) = \dot{x}(t)$, we obtain the following system of first-order cascade equations

$$\dot{x}_1(t) = x_2(t) \tag{4.37}$$
$$\dot{x}_2(t) = u(t-h), \tag{4.38}$$

or in matrix form

$$\dot{X}(t) = AX(t) + Bu(t-h), \tag{4.39}$$

where

$$X(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{4.40}$$

We assume the output as

$$Y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} X(t). \tag{4.41}$$

States can be predicted as described below. First, to reconstruct the state $\dot{x}$ the FTC is applied to $\dot{x}_2$ on $[t-h, t]$ giving

$$\int_{t-h}^{t} \dot{x}_2(s)ds = x_2(t) - x_2(t-h) \tag{4.42}$$

or

$$x_{2,pre}(t) = x_2(t-h) + \int_{t-h}^{t} \dot{x}_2(s)ds. \tag{4.43}$$

Substituting (4.38) into (4.43) yields

$$x_{2,pre}(t) = x_2(t-h) + \int_{t-h}^{t} u(s)ds. \tag{4.44}$$

Rewriting in terms of $\dot{x}$, it can be concluded that

$$\dot{x}_{pre}(t) = \dot{x}(t-h) + \int_{t-h}^{t} u(s)ds. \tag{4.45}$$

Second, similarly to $\dot{x}_2$, the FTC is applied to $\dot{x}_1$ on $[t-h, t]$ obtaining

$$x_{1,pre}(t) = x_1(t-h) + \int_{t-h}^{t} \dot{x}_1(s)ds. \tag{4.46}$$

Notice that the equation (4.46) is equivalent to

$$\begin{aligned} x_{1,pre}(t) &= x_1(t-h) + \int_{t-h}^{t} [\dot{x}_1(t-h) + \dot{x}_1(s) - \dot{x}_1(t-h)]ds & (4.47)\\ &= x_1(t-h) + \int_{t-h}^{t} \dot{x}_1(t-h)ds + \int_{t-h}^{t} [\dot{x}_1(s) - \dot{x}_1(t-h)]ds & (4.48)\\ &= x_1(t-h) + h\dot{x}_1(t-h) + \int_{t-h}^{t} [\dot{x}_1(s) - \dot{x}_1(t-h)]ds. & (4.49) \end{aligned}$$

Integrating into the equation (4.49) can be expressed by

$$\dot{x}_1(s) - \dot{x}_1(t-h) = \int_{t-h}^{s} \ddot{x}_1(l)dl = \int_{t-h}^{s} u(l)dl \tag{4.50}$$

Therefore, substituting (4.50) into (4.49) concludes that

$$x_{1,pre}(t) = x_1(t-h) + h\dot{x}_1(t-h) + \int_{t-h}^{t}\int_{t-h}^{s} u(l)dlds. \tag{4.51}$$

4.3.2 Two integrator: simulations results

We assume a constant reference $x_{1,ref}(t) = 1, \quad \forall t \geq 0$ and the sampling time $T_s = 0.01$. From (4.29) the controller becomes

$$u(t) = -k_p(x_{1,pre}(t) - x_{1,ref}(t)) - k_d x_{2,pre}(t), \tag{4.52}$$

where $k_p = 0.209$ and $k_d = 0.975$ was applied in (4.37)-(4.38) with $h = 0$. The results are shown in Figure 4.8.

To analyze the performance of the predictor-based controller, a delay of $h = 1.5$ seconds is introduced into the system (4.37)-(4.38). System responses without delay, with delay and with delay compensation are shown in Figure 4.9 and 4.10.

Having analyzed the good performance of the controller for a simple system such as the double integrator the next step was to analyze its performance in a classic model of a quadrotor.

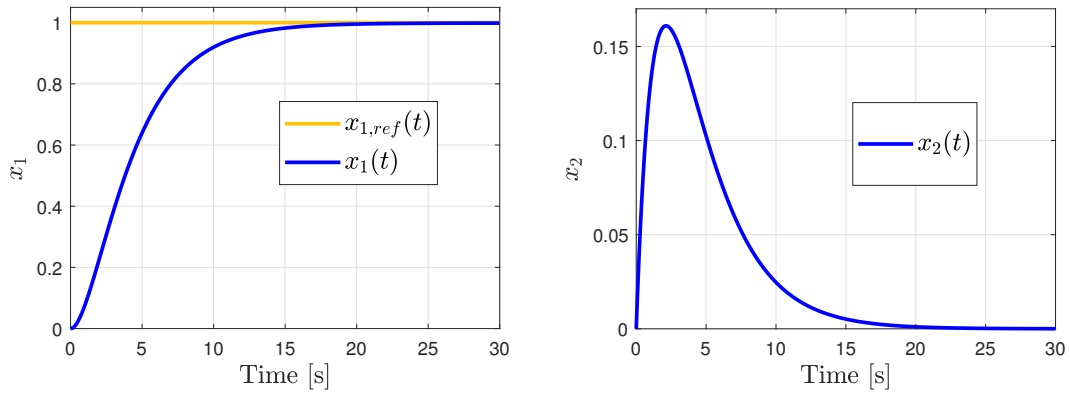


Figure 4.8 – States reponses without delay, $h = 0$ seconds.

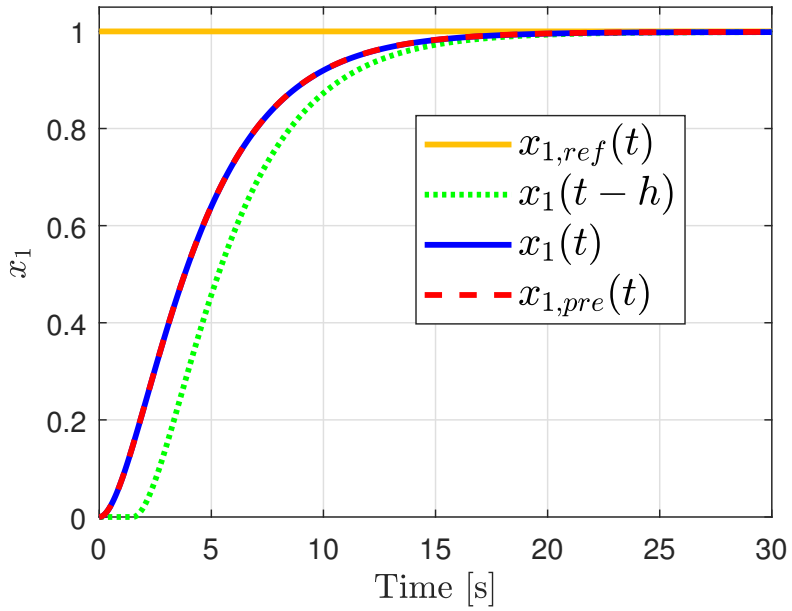


Figure 4.9 – State $x_1(t)$ with constant reference $x_{1_{ref}}(t) = 1$, delayed measure $x_1(t-h)$ with delay $h = 1.5$ seconds and the estimated state $x_{1,pre}(t)$.

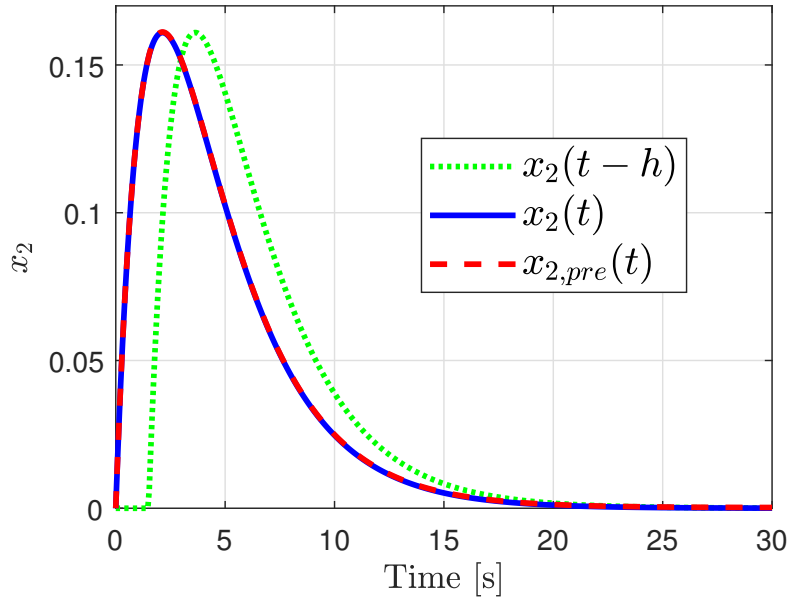


Figure 4.10 – State x_2 that represent velocity, $h = 1.5$ seconds.

4.4 Euler-Lagrange model with input delays

4.4.1 Longitudinal dynamic with input delay

It is well known that the longitudinal dynamics of a quadrotor can be expressed by

$$\ddot{x}(t) = -g\tan(\theta(t)) \tag{4.53}$$
$$\ddot{\theta}(t) = u_\theta(t). \tag{4.54}$$

A controller based on nested saturations is proposed in [159],

$$u_\theta(t) := -\sigma_{\theta_1}(\dot{\theta} + \sigma_{\theta_2}(\xi_1 + \sigma_{\theta_3}(\eta_1 + \sigma_{\theta_4}(\rho_1)))) \tag{4.55}$$

where

$$\xi_1 := \theta + \dot{\theta} \tag{4.56}$$
$$\eta_1 := 2\theta + \dot{\theta} + \frac{\dot{x}}{g} \tag{4.57}$$
$$\rho_1 := \dot{\theta} + 3\theta + \frac{3\dot{x}}{g} + \frac{x}{g} \tag{4.58}$$

in order to stabilize the quadrotor vehicle at hover without delay in the control input. However, this control law can be also used to stabilize the quadrotor even in presence of delay [160].

Then, introducing a delay $h = 0.35$ seconds in the control input (4.55) and applying the result of Lemma 1 we get the responses shown in Figures 4.11-4.14. Here, we observe that the states of the system with delay (green line) have an oscillatory behavior. After applying the predictor (red line) such a behavior decreases considerably, having a behavior similar to the responses of the delay-free system (blue line). Predictor performance is best seen in Figures 4.12-4.14.

Analogously, the predictor was also applied to quadrotor model with altitude delay as described in Section 4.4.2.

4.4.2 Quadrotor model with delayed altitude control

The following control algorithm was proposed in [159] to control the trusth of a quadrotor

$$u(t) := (r_1(t) + mg)\left(\frac{1}{\cos(\theta_1(t))\cos(\phi_1(t))}\right) \tag{4.59}$$

where

$$r_1(t) := -k_{p_z}(z_1(t) - z_1^{ref}(t)) - k_{d_z}z_2(t) \tag{4.60}$$

and k_{p_z}, k_{d_z} are positive constants.

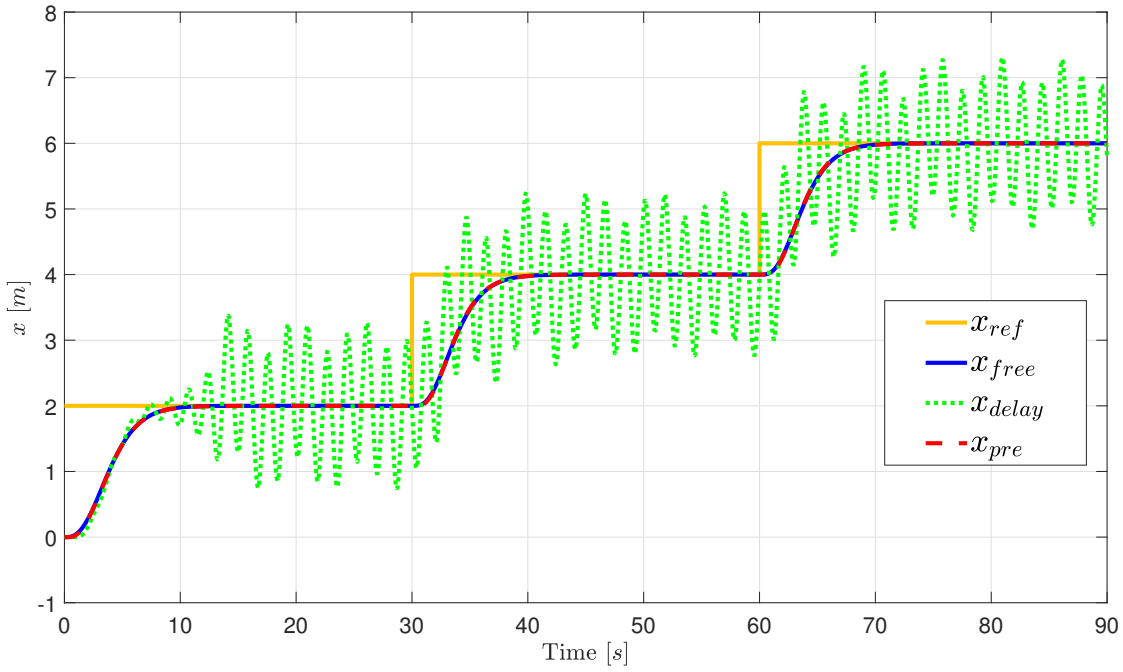


Figure 4.11 – State x with steped reference and delay $h = 0.35$ seconds.

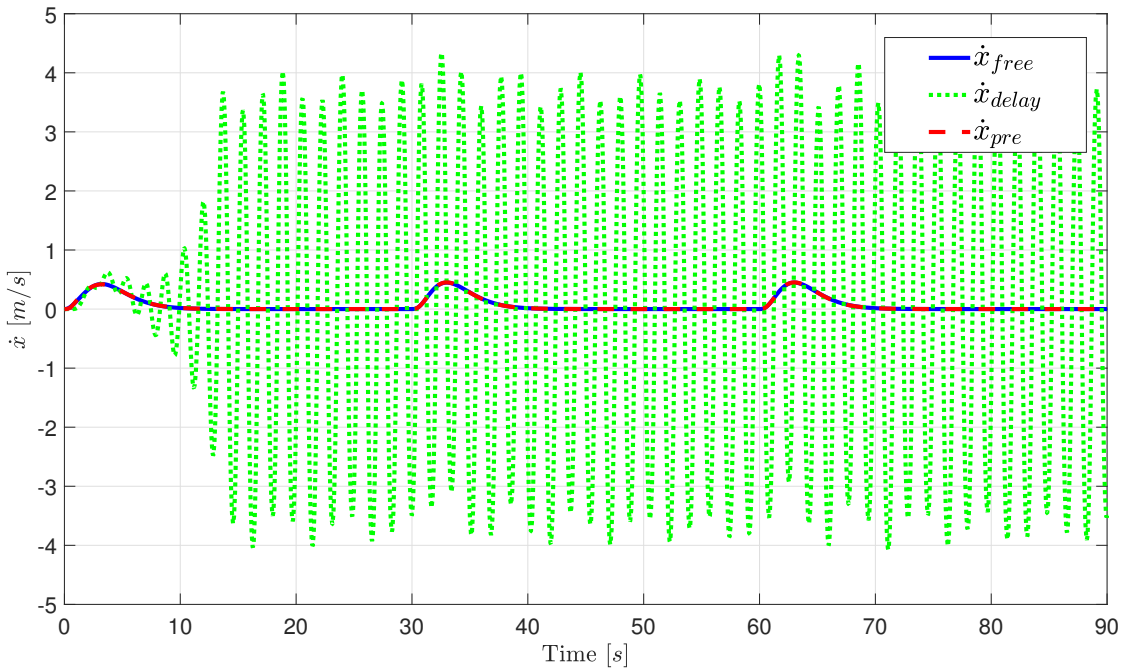


Figure 4.12 – Linear velocity $\dot{x}$ performance, $h = 0.35$ seconds.

The delayed thrust control is defined by adding a virtual delay h_1 in altitude control (4.60) then

$$u(t-h_1) := (r_1(t-h_1)+mg)\left(\frac{1}{\cos(\theta_1(t))\cos(\phi_1(t))}\right) \tag{4.61}$$

where

$$r_1(t-h_1) := -k_{p_z}(z_1(t-h_1) - z_1^{ref}(t-h_1)) - k_{d_z}z_2(t-h_1). \tag{4.62}$$

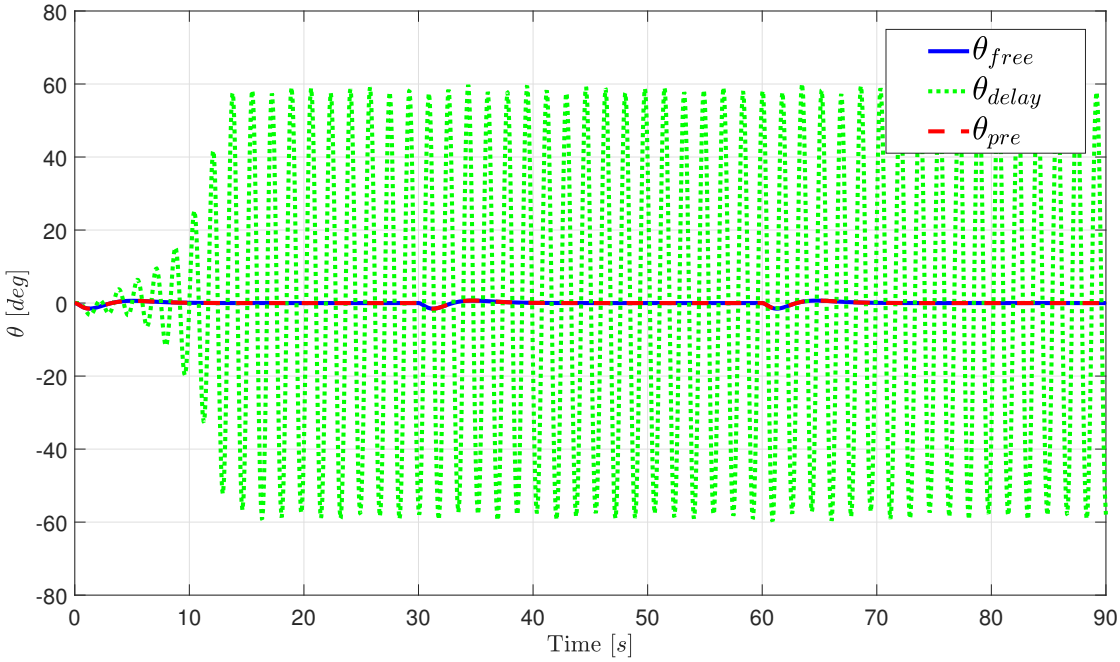


Figure 4.13 – Roll state with delay $h = 0.35$ seconds.

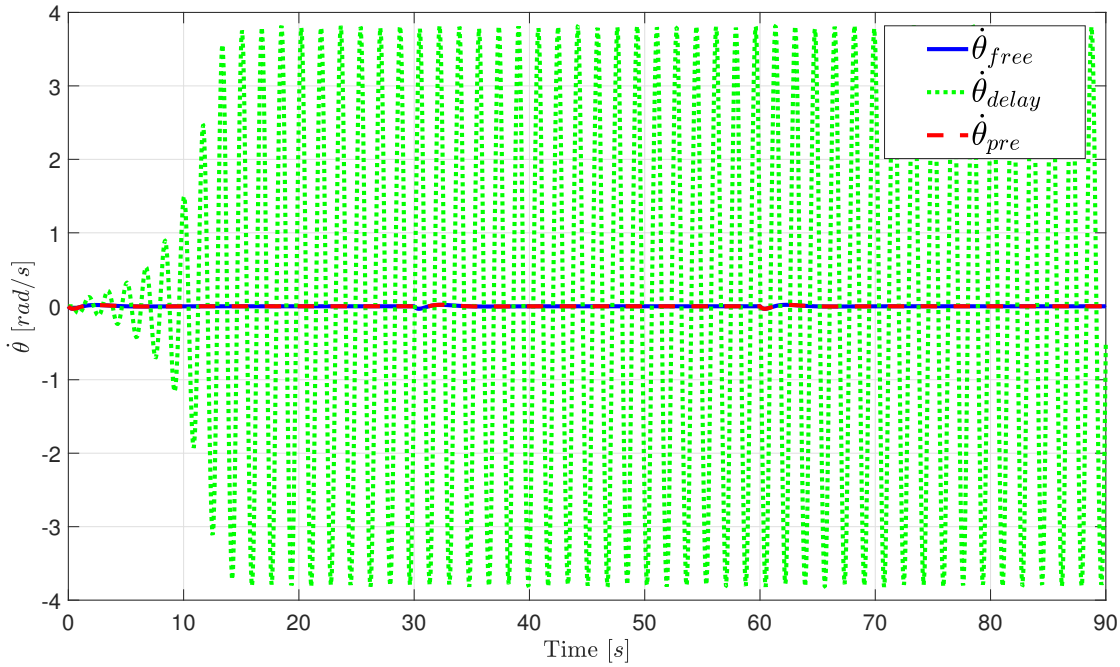


Figure 4.14 – Angular speed with delay $h = 0.35$ seconds.

This delayed input is applied to the drone model in order to recover the state with the predictor. Using Lemma 1, the $z_1(t)$ and $z_2(t)$ states can be estimated as

$$\begin{aligned}\hat{z}_1(t) &= z_1(t-h_1) + h_1 z_2(t-h_1) + \int_{t-h_1}^{t}\int_{t-h_1}^{s} u(l)dlds \\ \hat{z}_2(t) &= z_2(t-h_1) + \int_{t-h_1}^{t} u(s)ds.\end{aligned}$$

Let us assume the initial conditions $x_1(0) = 0$, $y_1(0) = 0$, $z_1(0) = 13$ and the constant stepped reference $z_1^{ref}(t)$ and $x_1^{ref}(t) = y_1^{ref}(t) = 0$. The parameters values are given in the Table 4.1. The other states are controlled with the algorithms in the equations (4.63)-(4.65).

Parameter	Value
m	1
g	9.8
k_{p_z}	2.6
k_{d_z}	3
k_{p_ψ}	0.1
k_{d_ψ}	0.8
M_1, N_1	22
M_2, N_2	11
M_3, N_3	5.5
M_4, N_4	2.7

Table 4.1 – Parameters of quadrotor simulation

$$u_\psi(t) := -k_{p_\psi}(\psi_1(t) - \psi_1^{ref}(t)) - k_{d_\psi}\psi_2(t) \tag{4.63}$$
$$u_\theta(t) := -\sigma_{M_1}(\theta_2(t) + \sigma_{M_2}(\xi_1(t) + \sigma_{M_3}(\eta_1(t) + \sigma_{M_4}(\rho_1(t))))) \tag{4.64}$$
$$u_\phi(t) := -\sigma_{N_1}(\phi_2(t) + \sigma_{N_2}(\xi_2(t) + \sigma_{N_3}(\eta_2(t) + \sigma_{N_4}(\rho_2(t))))) \tag{4.65}$$

where

$$\xi_1(t) := \theta_1(t) + \theta_2(t) \tag{4.66}$$
$$\eta_1(t) := 2\theta_1(t) + \theta_2(t) - \frac{x_2(t)}{g} \tag{4.67}$$
$$\rho_1(t) := \theta_2(t) + 3\theta_1(t) - \frac{3x_2(t)}{g} - \frac{x_1(t)}{g} \tag{4.68}$$
$$\xi_2(t) := \psi_1(t) + \psi_2(t) \tag{4.69}$$
$$\eta_2(t) := 2\psi_1(t) + \psi_2(t) + \frac{y_2(t)}{g} \tag{4.70}$$
$$\rho_2(t) := \psi_2(t) + 3\psi_1(t) + \frac{3y_2(t)}{g} + \frac{y_1(t)}{g} \tag{4.71}$$

and M_i, N_i with $i = 1, 2, 3, 4$ are the bounds of the saturation functions and k_{p_ψ}, k_{d_ψ} are positive constants.

In Figures 4.15 and 4.16 we observe the behavior of the states when the delay $h_1 = 0.4$ seconds is applied. Here, the response of the delayed system (green line) has an oscillatory behavior in the first instants while the response applying the predictor (red line) compensates the delay by eliminating oscillations and generating a behavior resembling to the response delay-free system (blue line).

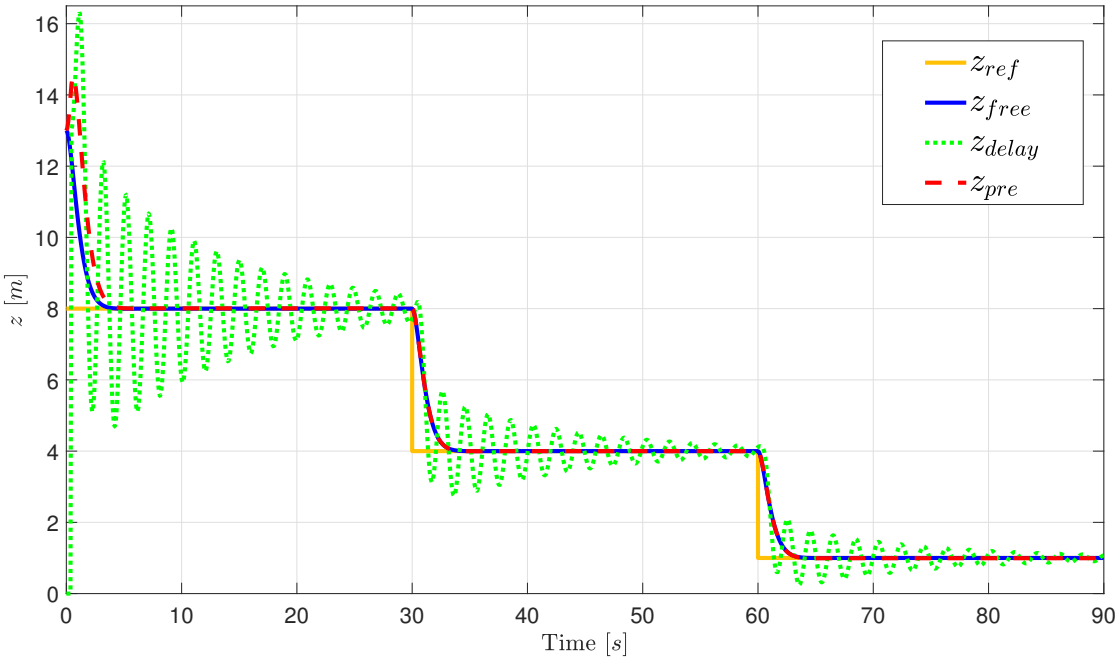


Figure 4.15 – Altitude response with delay $h_1 = 0.4$ seconds.

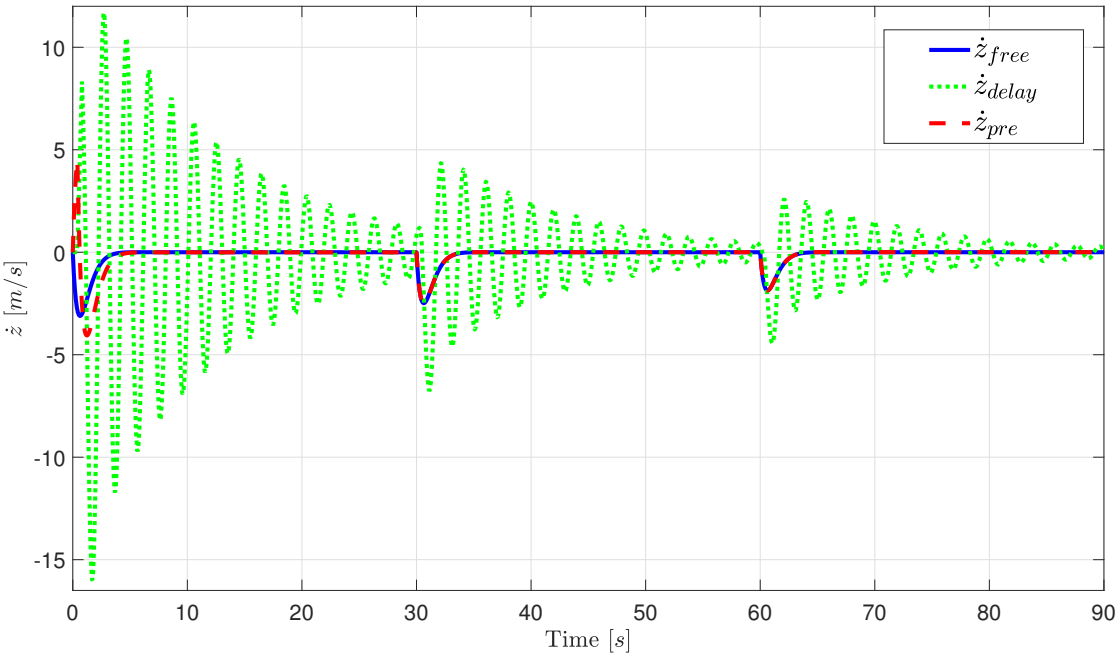


Figure 4.16 – Linear velocity around of the z-axis, delay $h_1 = 0.4$ seconds.

4.4.3 Tracking trajectory with input delays

In this subsection, we will apply a virtual delay h_1 in the translational dynamics and h_2 in the rotational dynamics on the Euler-Lagrange model described in Section 2.2. Then, we will apply the proposed approach to reconstruct the states of the system.

Using the control laws (4.59), (4.63), (4.64) and (4.65), that stabilize the quadrotor in hover without the presence of delay, we can define the delayed control laws as

$$u(t,h_1,h_2) := u(z(t-h_1),\dot{z}(t-h_1),\theta(t-h_2),\phi(t-h_2)) \tag{4.72}$$

$$u_\psi(t,h_1,h_2) := u_\psi(\psi(t-h_2),\dot{\psi}(t-h_2),\psi_{ref}(t-h_2),\dot{\psi}_{ref}(t-h_2)) \tag{4.73}$$

$$u_\theta(t,h_1,h_2) := u_\theta(\theta(t-h_2),\dot{\theta}(t-h_2),x(t-h_1),\dot{x}(t-h_1)) \tag{4.74}$$

$$u_\phi(t,h_1,h_2) := u_\phi(\phi(t-h_2),\dot{\phi}(t-h_2),y(t-h_1),\dot{y}(t-h_1)). \tag{4.75}$$

Developing the approach described in Section 4.3, we can retrieve the states of the system under the delays in the control inputs applying the following expressions

$$x_{pre}(t) = x(t-h_1)+h_1\dot{x}(t-h_1)-g\int_{t-h_1}^{t}\int_{t-h_1}^{s}\theta_{pre}(l)dlds \tag{4.76}$$

$$\dot{x}_{pre}(t) = \dot{x}(t-h_1)-g\int_{t-h_1}^{t}\theta_{pre}(s)ds \tag{4.77}$$

$$\theta_{pre}(t) = \theta(t-h_2)+h_2\dot{\theta}(t-h_2)+\int_{t-h_2}^{t}\int_{t-h_2}^{s}u_\theta(l)dlds \tag{4.78}$$

$$\dot{\theta}_{pre}(t) = \dot{\theta}(t-h_2)+\int_{t-h_2}^{t}u_\theta(s)ds \tag{4.79}$$

$$y_{pre}(t) = y(t-h_1)+h_1\dot{y}(t-h_1)+g\int_{t-h_1}^{t}\int_{t-h_1}^{s}\phi_{pre}(l)dlds \tag{4.80}$$

$$\dot{y}_{pre}(t) = \dot{y}(t-h_1)+g\int_{t-h_1}^{t}\phi_{pre}(s)ds \tag{4.81}$$

$$\phi_{pre}(t) = \phi(t-h_2)+h_2\dot{\phi}(t-h_2)+\int_{t-h_2}^{t}\int_{t-h_2}^{s}u_\phi(l)dlds \tag{4.82}$$

$$\dot{\phi}_{pre}(t) = \dot{\phi}(t-h_2)+\int_{t-h_2}^{t}u_\phi(s)ds \tag{4.83}$$

$$z_{pre}(t) = z(t-h_1)+h_1\dot{z}(t-h_1)+\frac{1}{m}\int_{t-h_1}^{t}\int_{t-h_1}^{s}r_1(l)dlds \tag{4.84}$$

$$\dot{z}_{pre}(t) = \dot{z}(t-h_1)+\frac{1}{m}\int_{t-h_1}^{t}r_1(s)ds \tag{4.85}$$

$$\psi_{pre}(t) = \psi(t-h_2)+h_2\dot{\psi}(t-h_2)+\int_{t-h_2}^{t}\int_{t-h_2}^{s}u_\psi(l)dlds \tag{4.86}$$

$$\dot{\psi}_{pre}(t) = \dot{\psi}(t-h_2)+\int_{t-h_2}^{t}u_\psi(s)ds. \tag{4.87}$$

Next we will give an example of the implementation of the equations described above.

Numerical example: square path tracking

The following example was developed in Simulink with a sampling time of $T_s = 0.001$ and considering the Euler-Lagrange model (2.15)-(2.16). First, we will apply the control laws (4.59), (4.63), (4.64) and (4.65) for the delay-free system where the constants are the same as in the Table 4.1 and the reference position is given by

$$\xi_{ref} = \begin{cases} (t, 0, 11)^T, & \text{if } t < 10, \\ (10, t-10, 11)^T, & \text{if } 10 < t \leq 20 \\ (30-t, 10, 11)^T, & \text{if } 20 < t \leq 30 \\ (0, 40-t, 11)^T, & \text{if } t > 30, \end{cases} \tag{4.88}$$

and the attitude reference is $\psi_{ref} = \theta_{ref} = \phi_{ref} = 0$. Second, to apply the virtual delays to the control inputs, we will use the control laws defined in (4.72)-(4.75). Finally, we implement the equations (4.76)-(4.87) in order to recover the states of the system in presence of delay $h_1 = h_2 = 0.35$ seconds.

Figure 4.17 shows the oscillatory behavior in the 3D-position when the system has a delays in the control inputs (green line). Also, we can see the performance of the predicted states (red line) which is similar to the behavior of the delay-free system (blue line). Figure 4.18 shows each of the components of the position. Oscillatory behavior can be clearly seen in the linear and angular velocity states shown in the Figures 4.19 and 4.21 respectively. The response in attitude can be seen in Figure 4.20 where the orientation in the θ and ϕ angles are destabilized in the system with delay and are recovered by the predictor.

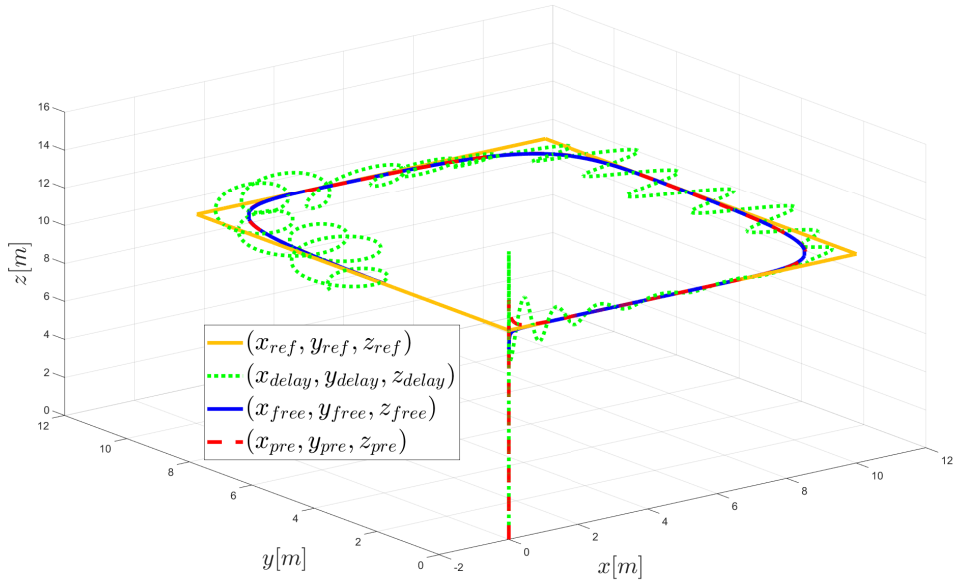


Figure 4.17 – 3D-position response of the Euler-Lagrange model with delayed inputs. Trajectory tracking (ref); position evolution with nested saturation controllers in absence of the delay ($free$); with nested saturation controllers in the presence of the delay ($delay$); with predictor-based controller in the presence of the delay $h = 0.35$ seconds (pre).

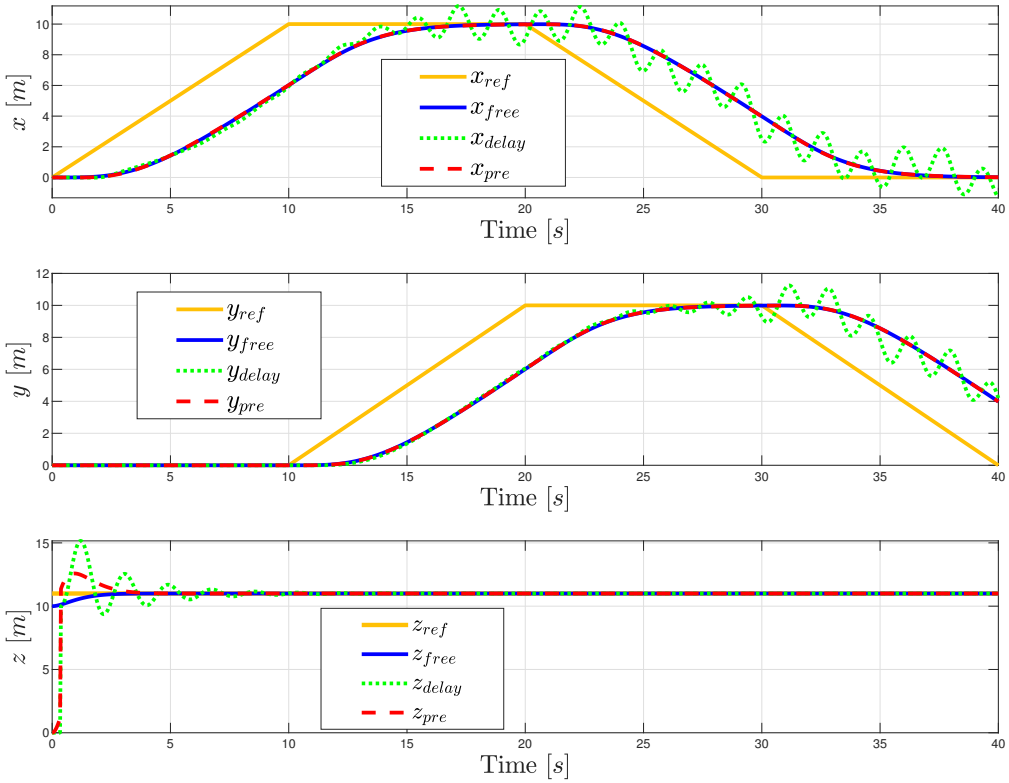


Figure 4.18 – States position of the Euler-Lagrange model with delayed inputs $h = 0.35$ seconds.

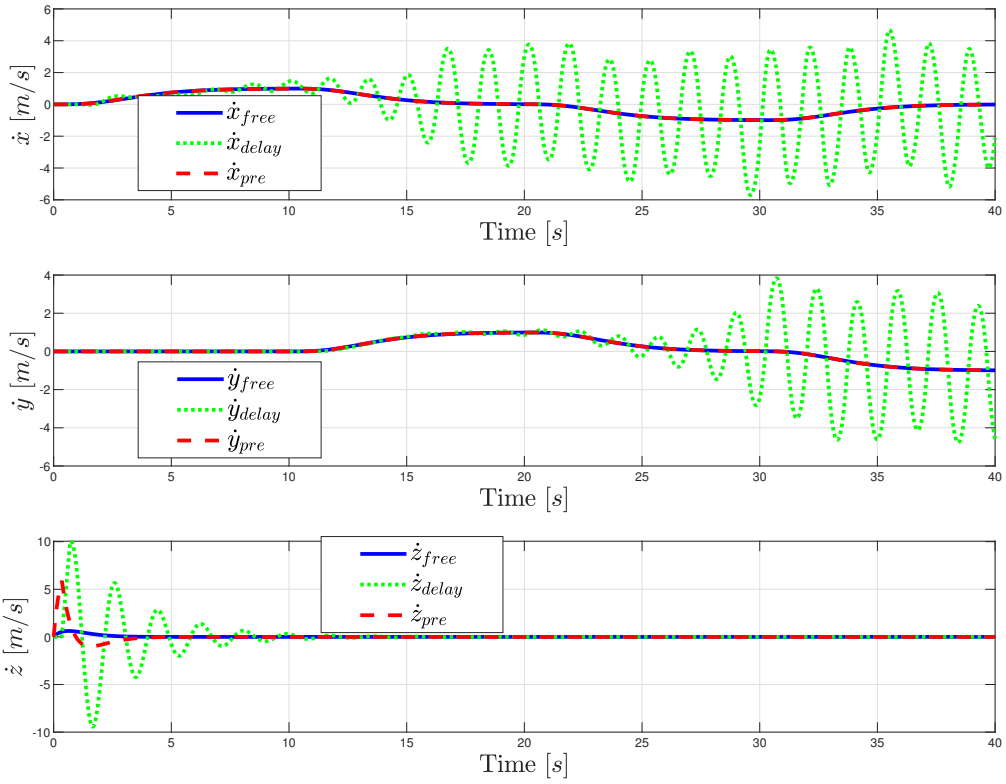


Figure 4.19 – Linear velocity response along the x, y and z axis, delay $h = 0.35$ seconds.

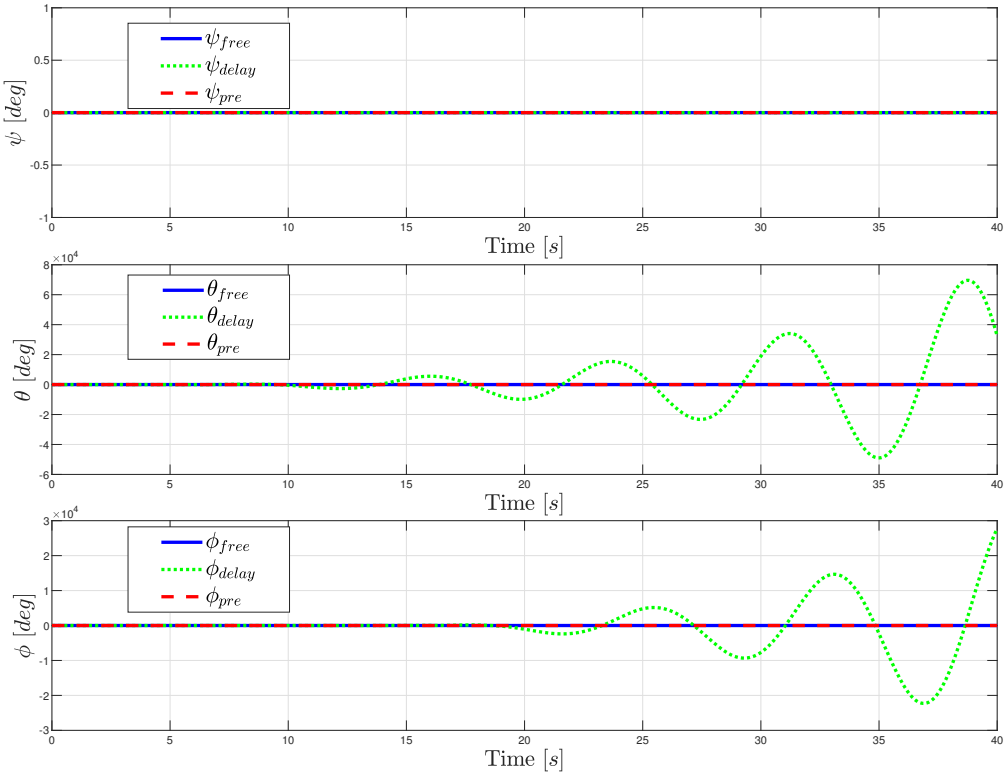


Figure 4.20 – Attitude response in yaw (ψ), roll (θ) and pitch (ϕ), delay $h = 0.35$ seconds.

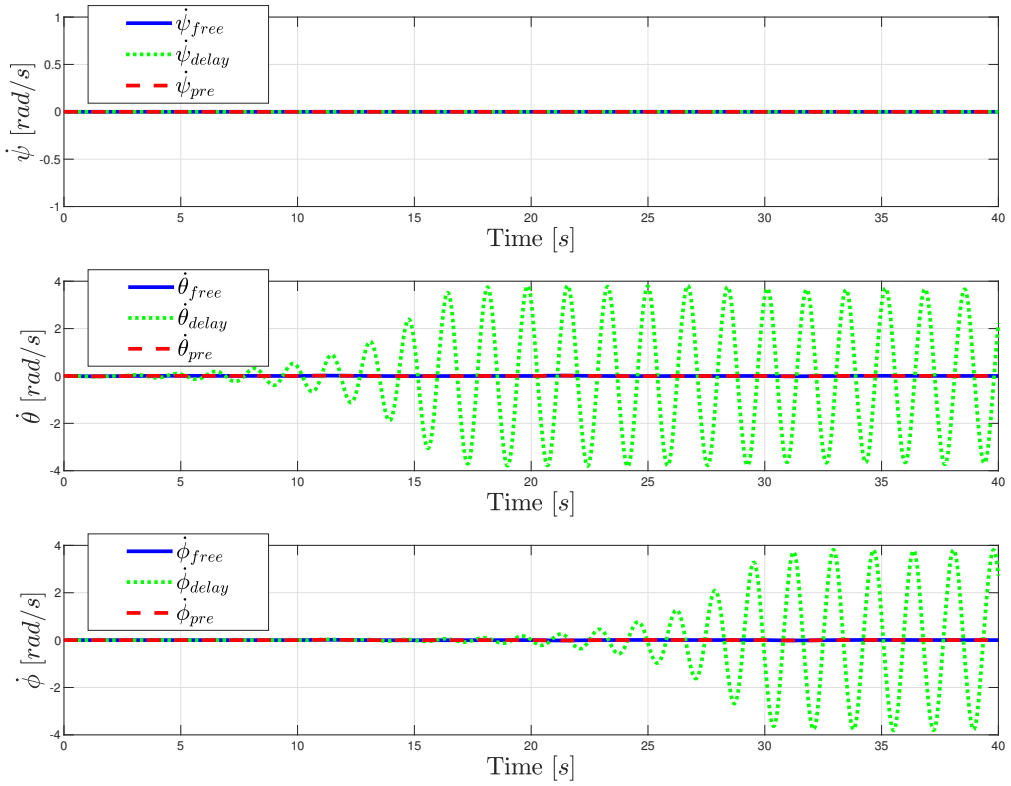


Figure 4.21 – Angular speed, delay $h = 0.35$ seconds.

4.5 Quaternion-based model with input delays

In this section, we apply our approach to recover the states of the modeled system with quaternions in the presence of input delay. First, in Subsection 4.5.1 we consider the model with delayed translational dynamics. Second, in Subsection 4.5.2 we consider rotational dynamics with delayed input. Finally, in Subsection 4.5.3, we consider trajectory tracking with delays in force control inputs and torques in trajectory tracking.

4.5.1 Quadrotor translational model with input delay

In this subsection, we introduce the quaternion-based model with the control force delayed. First, we consider a classical controller that uses linear feedback for this problem in the delay-free case. Second, we present our predictor-based controller approach in order to stabilize the states. For this, we consider the system (2.60)-(2.62) with attitude control given by (2.87) and using the control force law (2.85) we can define the input delay h_1 as

$$\vec{F}_u(t-h_1) := \vec{F}_{PD}(t-h_1) - m\vec{g}, \tag{4.89}$$

where

$$\vec{F}_{PD}(t-h_1) = -K_{pt}(\vec{\xi}(t-h_1) - \vec{\xi}_{ref}(t-h_1)) - K_{dt}(\dot{\vec{\xi}}(t-h_1) - \dot{\vec{\xi}}_{ref}(t-h_1)). \tag{4.90}$$

The references of position and linear velocity are given by $\vec{\xi}_{ref}(t) = (0.1, 0.5, 1)^T$ and $\dot{\vec{\xi}}_{ref}(t) = (0,0,0)^T$. We analyze the system with input delay in the translational model using a sampling time $T_s = 0.001$, it is destabilized after a delay of $h_1 = 0.5$ seconds. However, applying the predictor-based control described in the Subsection 4.3 we can retrieve the states as

$$\vec{\xi}_{pre}(t) = \vec{\xi}(t-h_1) + h_1\dot{\vec{\xi}}(t-h_1) + \frac{1}{m}\int_{t-h_1}^{t}\int_{t-h_1}^{s}\vec{F}_{PD}(l)dlds \tag{4.91}$$

$$\dot{\vec{\xi}}_{pre}(t) = \dot{\vec{\xi}}(t-h_1) + \frac{1}{m}\int_{t-h_1}^{t}\vec{F}_{PD}(s)ds. \tag{4.92}$$

In Figure 4.22, we can see in blue line the position responses of the system (2.60)-(2.62) without input delay using the force and torque controllers in (2.85) and (2.87), respectively. Also, we can see in green line the position responses in presence of delay $h_1 = 0.5$ seconds only in the force control input defined in (4.89) and how the oscillations increase in the x and y states. In red line, we can observe the performance of the predictor in the position given by (4.91) and its tendency to delay-free state.

Similarly, in Figure 4.23 we can observe the behavior in the linear velocity. The attitude and angular velocity states we can see in the Figures 4.24 and 4.25, respectively.

4.5.2 Attitude dynamics with delayed control input

We consider the system (2.63)-(2.64) with the control input delayed h_2 seconds

$$\dot{\vec{\gamma}}(t) = \vec{\Omega}(t) \tag{4.93}$$

$$J\dot{\vec{\Omega}}(t) = \vec{\tau}(t-h_2) - \vec{\Omega}(t)\times J\vec{\Omega}(t), \tag{4.94}$$

where $\vec{\gamma}(t) = 2\ln\mathbf{q}(t)$ and the attitude control delayed is defined as

$$\vec{\tau}(t-h_2) := -2K_{pa}\ln(\mathbf{q}(t-h_2)\otimes\mathbf{q}^*_{ref}(t-h_2)) - K_{da}(\vec{\Omega}(t-h_2) - \vec{\Omega}_{ref}(t-h_2)) \tag{4.95}$$

where $\vec{\tau} \in \mathbb{R}^3$ represents the torques caused by the combined action of the quadrotor's motors, and $K_{pa} = diag([k_{pax} > 0, k_{pay} > 0, k_{paz} > 0])$, and $K_{da} = diag([k_{dax} > 0, k_{day} > 0, k_{daz} > 0])$ denote the proportional and derivative control gains respectively.

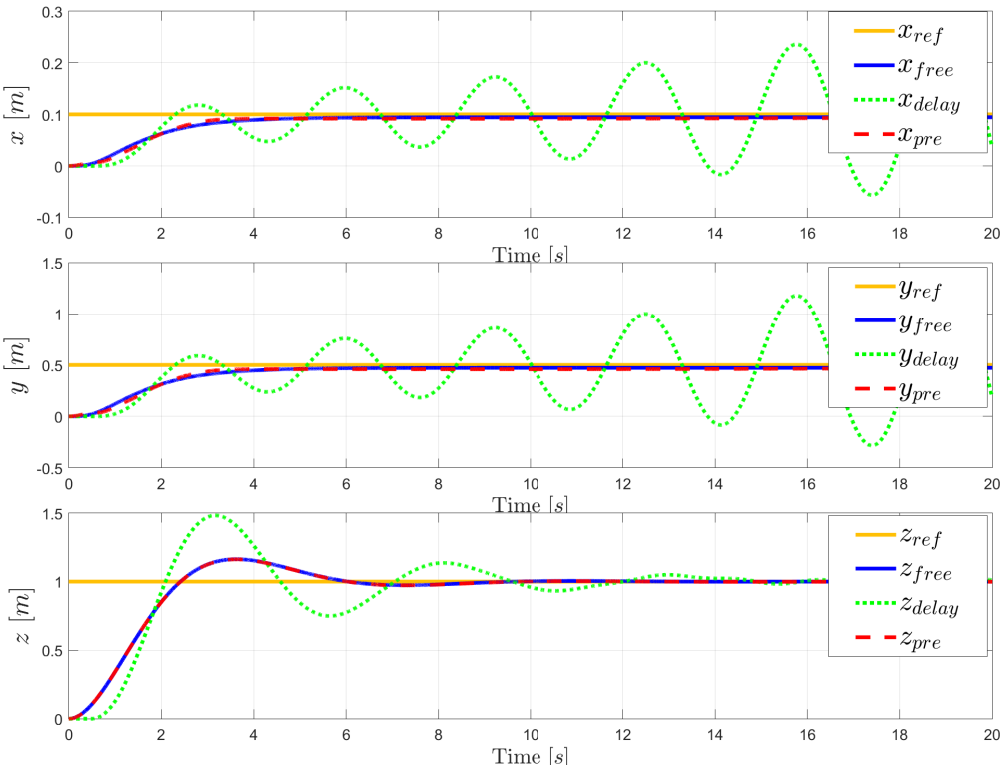


Figure 4.22 – Position state with input delay $h = 0.5$ seconds. The subindex $free$ denote the response of the system without delay, pre indicates the response of the system with delay and applying the predictor, and $delay$ the state of the system with delay.

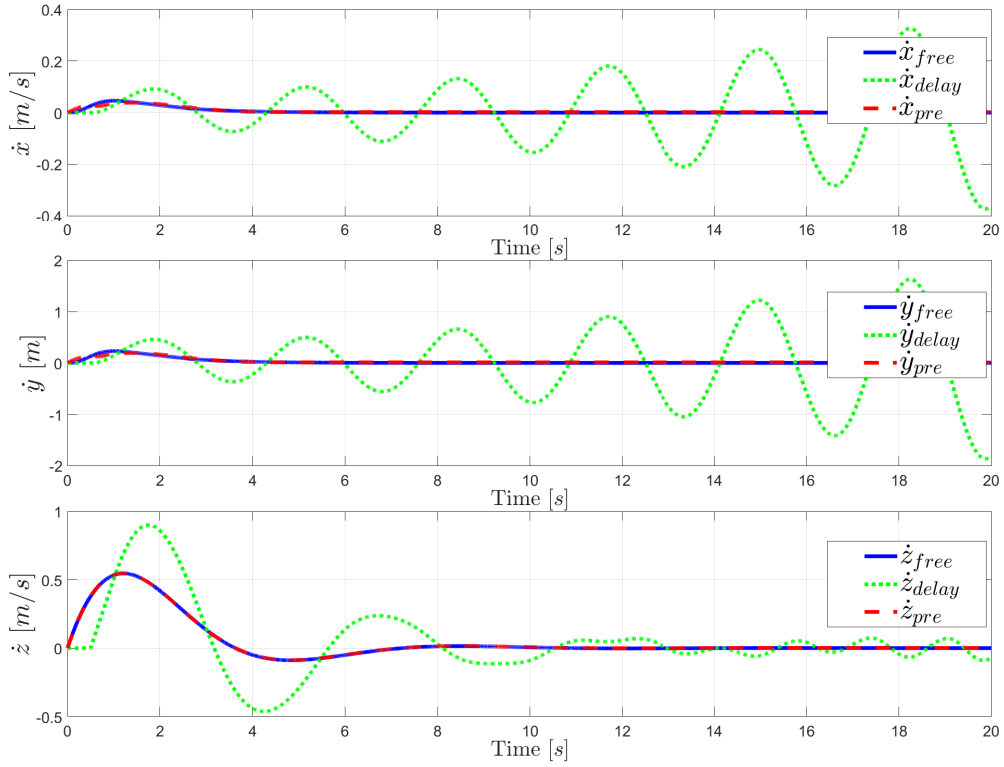


Figure 4.23 – Linear velocity state with input delay $h = 0.5$ seconds.

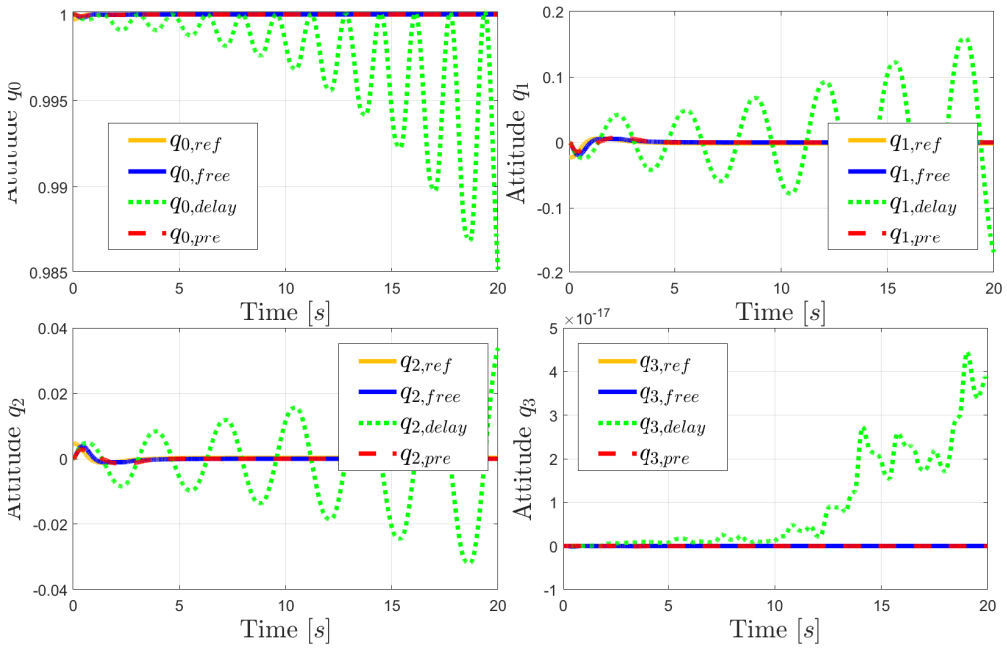


Figure 4.24 – Attitude quaternion state with input delay $h = 0.5$ seconds.

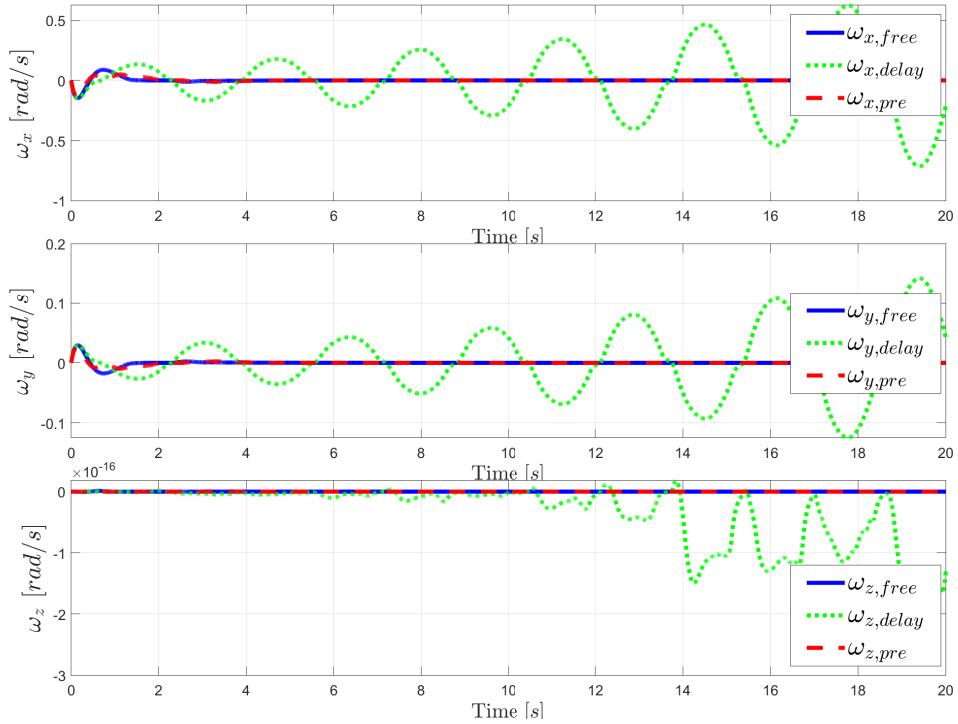


Figure 4.25 – Angular velocity state with input delay $h = 0.5$ seconds.

We apply the FTC to $\dot{\vec{\Omega}}$ in the interval $[t-h_2, t]$,

$$\int_{t-h_2}^{t} \dot{\vec{\Omega}}(s)ds = \vec{\Omega}(t) - \vec{\Omega}(t-h_2), \tag{4.96}$$

or

$$\vec{\Omega}(t) = \vec{\Omega}(t-h_2) + \int_{t-h_2}^{t} \dot{\vec{\Omega}}(s)ds, \tag{4.97}$$

replacing $\dot{\vec{\Omega}}(s) = \tau(s)$ we can to predict the angular velocity as

$$\vec{\Omega}_{pre}(t) = \vec{\Omega}(t-h_2) + \int_{t-h_2}^{t} \tau(s)ds. \tag{4.98}$$

Analogously, we apply the FTC to $\dot{\vec{\gamma}}$ in $[t-h_2, t]$,

$$\int_{t-h_2}^{t} \dot{\vec{\gamma}}(s)ds = \vec{\gamma}(t) - \vec{\gamma}(t-h_2), \tag{4.99}$$

or

$$\vec{\gamma}_{pre}(t) = \vec{\gamma}(t-h_2) + \int_{t-h_2}^{t} \dot{\vec{\gamma}}(s)ds. \tag{4.100}$$

Equivalently,

$$\begin{aligned}
\vec{\gamma}_{pre}(t) &= \vec{\gamma}(t-h_2) + \int_{t-h_2}^{t} [\dot{\vec{\gamma}}(t-h_2) + \dot{\vec{\gamma}}(s) - \dot{\vec{\gamma}}(t-h_2)]ds && (4.101)\\
&= \vec{\gamma}(t-h_2) + \int_{t-h_2}^{t} \dot{\vec{\gamma}}(t-h_2)ds + \int_{t-h_2}^{t} [\dot{\vec{\gamma}}(s) - \dot{\vec{\gamma}}(t-h_2)]ds && (4.102)\\
&= \vec{\gamma}(t-h_2) + h_2\dot{\vec{\gamma}}(t-h_2) + \int_{t-h_2}^{t} [\dot{\vec{\gamma}}(s) - \dot{\vec{\gamma}}(t-h_2)]ds. && (4.103)
\end{aligned}$$

Substituting $\dot{\vec{\gamma}}(t) = \vec{\Omega}(t)$,

$$\vec{\gamma}_{pre}(t) = \vec{\gamma}(t-h_2) + h_2\vec{\Omega}(t-h_2) + \int_{t-h_2}^{t} [\vec{\Omega}(s) - \vec{\Omega}(t-h_2)]ds \tag{4.104}$$

Using the FTC in the integrand of the equation (4.104)

$$\vec{\Omega}(s) - \vec{\Omega}(t-h_2) = \int_{t-h_2}^{s} \dot{\vec{\Omega}}(l)dl = \int_{t-h_2}^{s} \vec{\tau}(l)dl \tag{4.105}$$

Therefore, we can recover the axis-angle representation as

$$\vec{\gamma}_{pre}(t) = \vec{\gamma}(t-h_2) + h_2\vec{\Omega}(t-h_2) + \int_{t-h_2}^{t}\int_{t-h_2}^{s} \vec{\tau}(l)dlds \tag{4.106}$$

and the attitude state predicted can be computed by

$$\mathbf{q}_{pre}(t) = e^{\left[\ln \mathbf{q}(t-h_2) + \frac{h_2}{2}\vec{\Omega}(t-h_2) + \int_{t-h_2}^{t}\int_{t-h_2}^{s} \frac{\vec{\tau}(l)}{2}dlds\right]}. \tag{4.107}$$

Example: Rotational dynamic with attitude control delayed

Consider the system

$$\dot{\mathbf{q}}(t) = \frac{1}{2}\mathbf{q}(t)\otimes\boldsymbol{\Omega}(t) \tag{4.108}$$

$$J\dot{\vec{\Omega}}(t) = \vec{\tau}(t-h_2) - \vec{\Omega}(t)\times J\vec{\Omega}(t) \tag{4.109}$$

where

$$J = \begin{bmatrix} 12 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 9 \end{bmatrix} \quad kg\cdot m^2. \tag{4.110}$$

The initial conditions are $\mathbf{q}(0) = \frac{1}{2} + (-\frac{1}{2}, -\frac{1}{2}, \frac{1}{2})^T$ and $\vec{\Omega}(0) = (0,0,0)^T$. The desired attitude will be given by $\mathbf{q}_{ref}(t) = 0.9795 + (0.05791, 0.1691, 0.0933)^T$ that in Euler angles is equivalent to $(\psi_{ref}, \theta_{ref}, \phi_{ref})^T = (10, 20, 5)^T$. The input delayed is defined by the equation in (4.89) with initial conditions $\mathbf{q}(t-s) = 1 + (0,0,0)^T$ and $\vec{\Omega}(t-s) = (0,0,0)$ with $s \in [0, h_2]$, gain constants $k_{pax} = k_{pay} = k_{paz} = 1$ and $k_{dax} = k_{day} = k_{daz} = 2$.

In Figure 4.26, we observe the attitude response components $\mathbf{q} = q_0 + (q_1, q_2, q_3)^T$. The subscript ref denotes the components of the attitude reference (yellow line), $free$ represent the response of the free-delay system using a PD controller (blue line), $delay$ denotes the response of the system (4.108)-(4.109) with input delayed defined by the equation (4.95) with $h_2 = 0.5$ seconds (green line), pre refers to the predicted response obtained from the equation (4.107) (red line). We can observe that the response of the system with delay (green line) has an oscillatory behavior in the vector part between 2 and 10 seconds. However, applying the proposed predictor it is possible to recover the state (red line). In order to geometrically understand the performance of the predictor, Figure 4.27 shows the attitude expressed by Euler's angles.

Similarly, Figure 4.28 shows the components of angular velocity $\vec{\Omega} = (\omega_x, \omega_y, \omega_z)$. We can see how using the predictor (red line) in the equation (4.98) eliminates the oscillatory effect generated by the delay (green line).

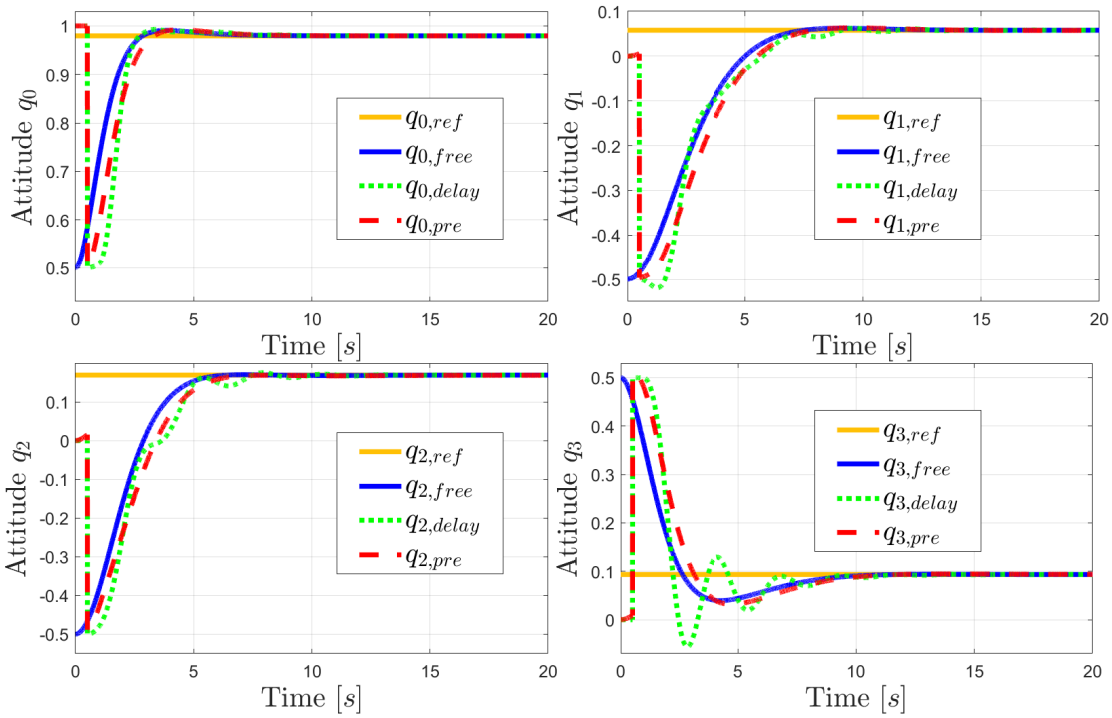


Figure 4.26 – Attitude quaternion response of the quaternion-based model with attitude control delayed. Attitude tracking (ref); orientation evolution with PD controller in absence of the delay ($free$); with PD controller in the presence of the delay ($delay$); with predictor-based controller in the presence of the delay $h_2 = 0.5$ seconds (pre).

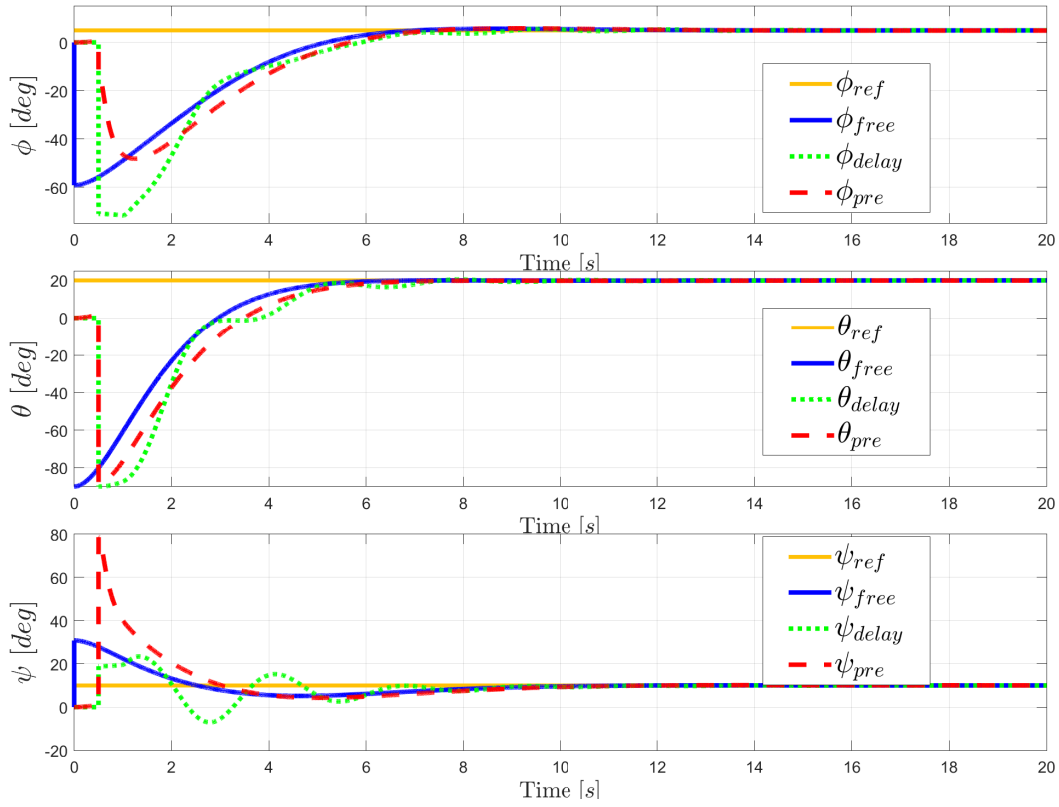


Figure 4.27 – Representation of attitude in Figure 4.26 as Euler's angles.

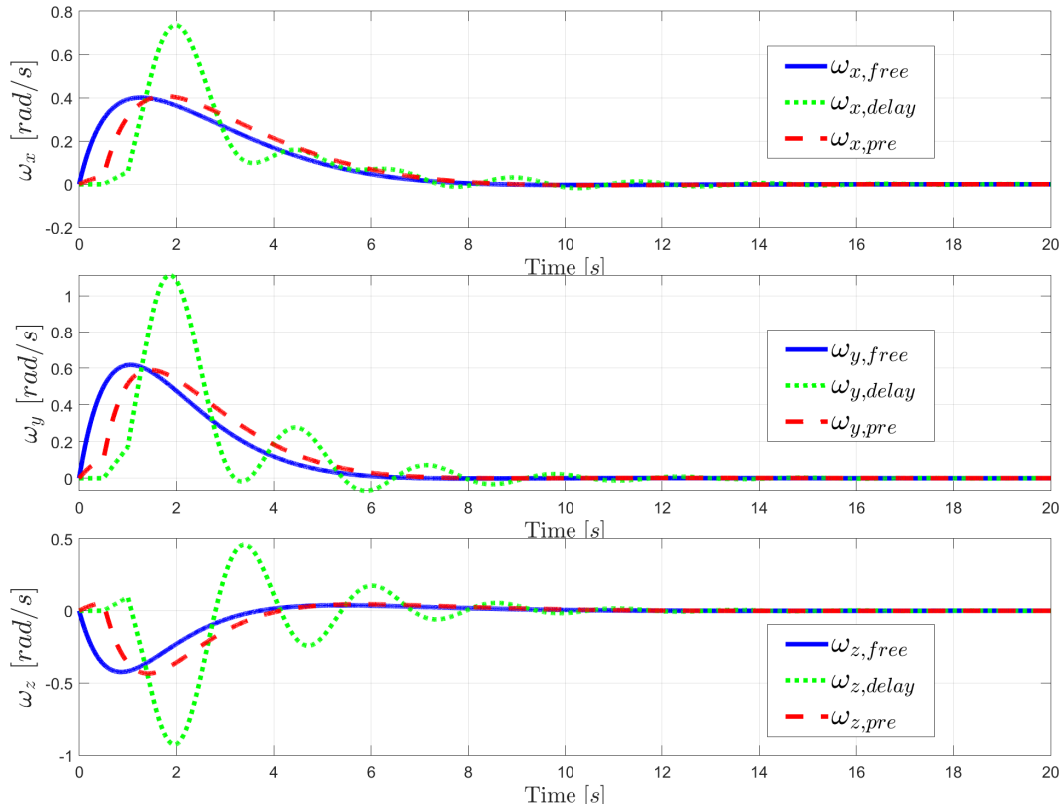


Figure 4.28 – Angular velocity components, delay $h_2 = 0.5$ seconds.

4.5.3 Tracking trajectory

In this subsection, we apply the predictor on the (2.60)-(2.62) model where

$$J = \begin{bmatrix} 0.177 & 0 & 0 \\ 0 & 0.177 & 0 \\ 0 & 0 & 0.354 \end{bmatrix} \quad kg \cdot m^2. \tag{4.111}$$

with delayed inputs defined as (4.89) and (4.95). A delay in the dynamics of $h_1 = h_2 = 0.35$ seconds will be applied during the tracking of the trajectory given by (4.88) and the yaw reference $\psi_{ref} = 0$. With initial conditions $\mathbf{q}(t - s_2) = 1 + (0,0,0)^T$ and $\vec{\xi}(t - s_1) = \dot{\vec{\xi}}(t - s_1) = \vec{\Omega}(t - s_2) = (0,0,0)^T$ with $s_1 \in [0, h_1]$ and $s_2 \in [0, h_2]$ and gain constants

$$K_{pt} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad K_{dt} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.112}$$

$$K_{pa} = \begin{bmatrix} 50 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 50 \end{bmatrix}, \quad K_{da} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}. \tag{4.113}$$

In Figure 4.29 we can see the effect of the delay in green line. These effects are more pronounced in quaternion-based model than in the Euler-Lagrange model in Figure 4.17. Despite of oscillatory behavior, the predictor (red line) recovers the states of the delay-free system. We can observe each component of position in Figure 4.30. The performance of the predictor can also be seen in the linear velocity in Figure 4.31. Responses in attitude can be seen in Figure 4.32. The angular speed is shown in Figure 4.35.

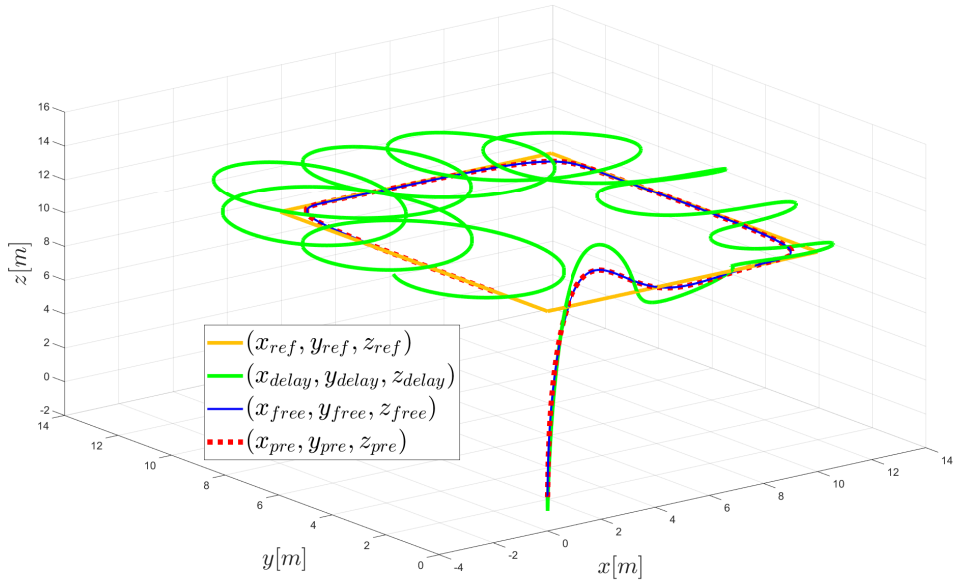


Figure 4.29 – 3D-position response of the quaternion-based model with delayed inputs. Trajectory tracking (ref); position evolution with PD controllers in absence of the delay ($free$); with PD controllers in the presence of the delay ($delay$); with predictor-based controller in the presence of the delay $h = 0.35$ seconds (pre).

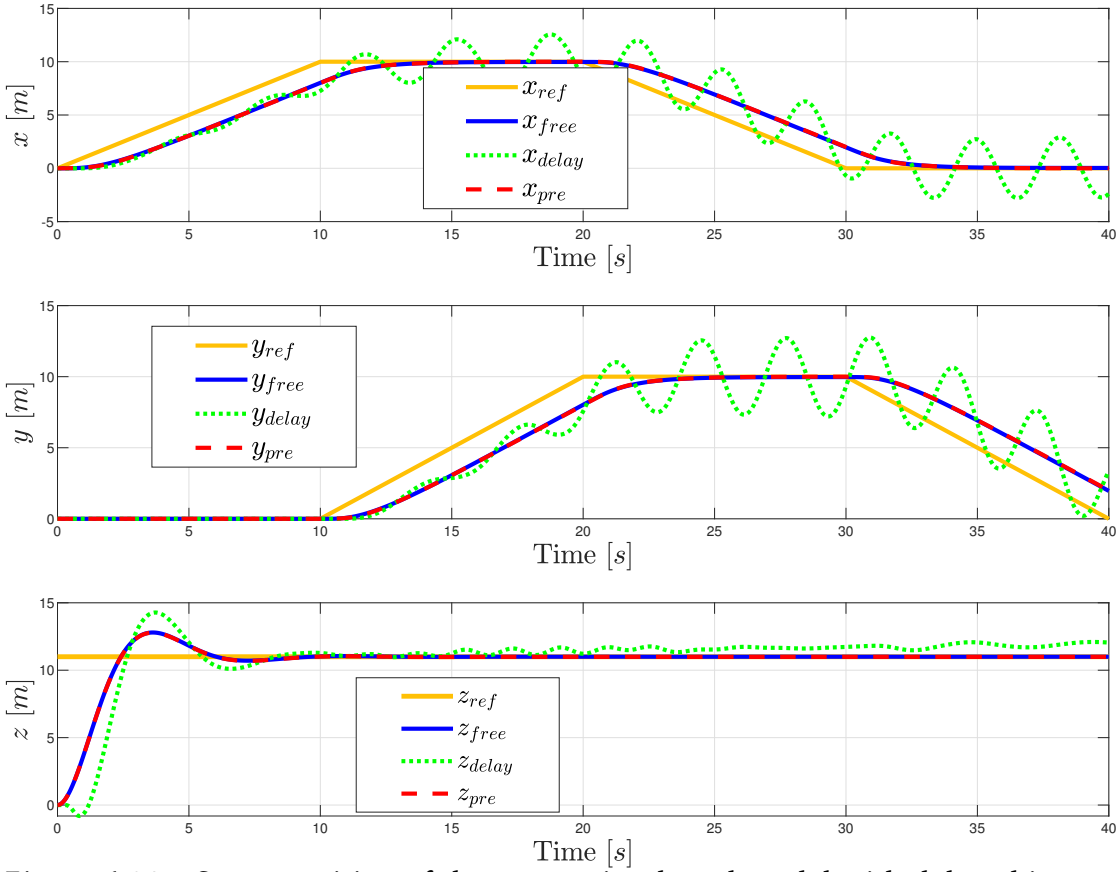


Figure 4.30 – States position of the quaternion-based model with delayed inputs $h = 0.35$ seconds.

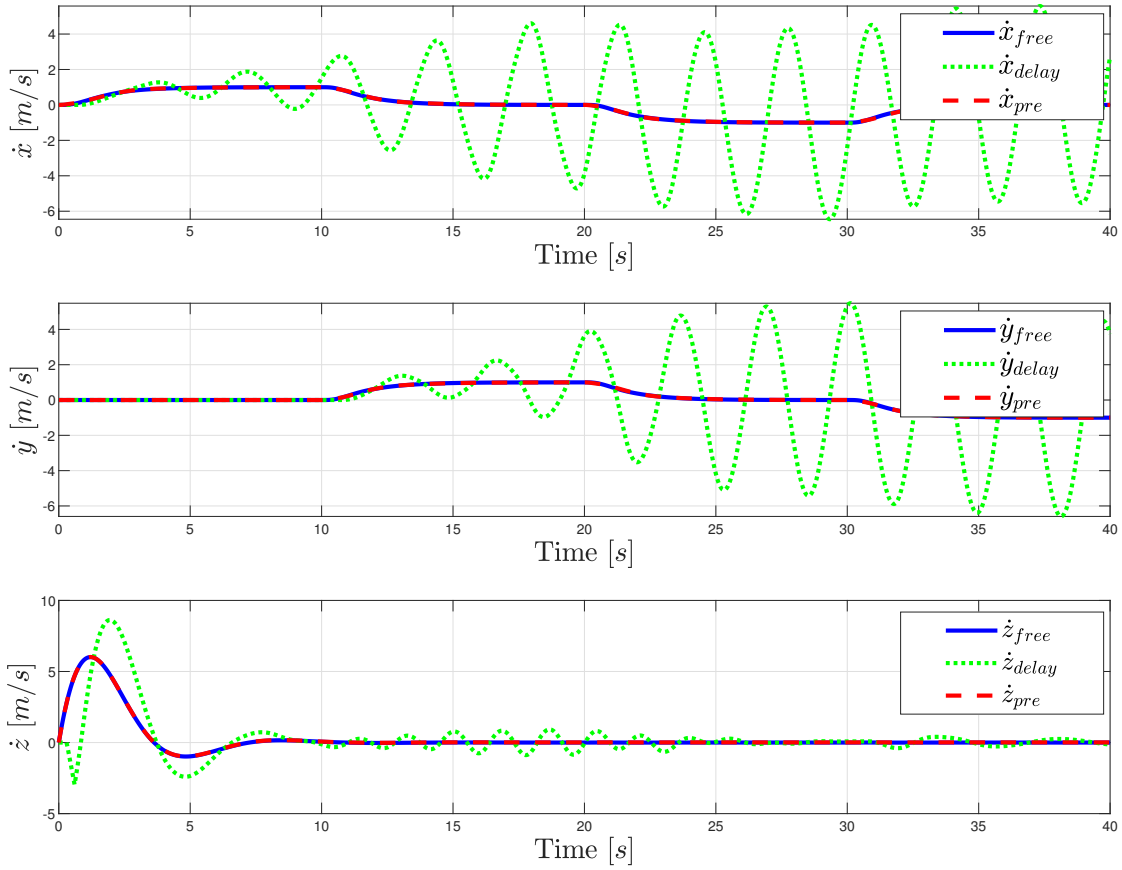


Figure 4.31 – Linear velocity response along the x, y and z axis, delay $h = 0.35$ seconds.

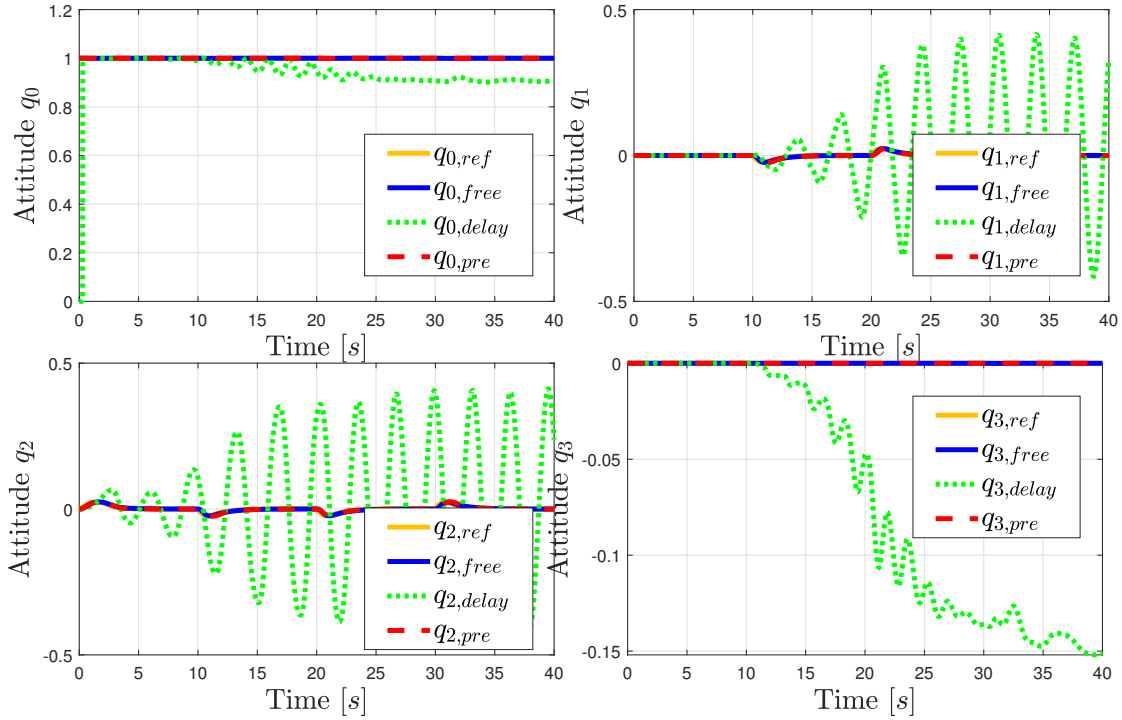


Figure 4.32 – Attitude response, delay $h = 0.35$ seconds.

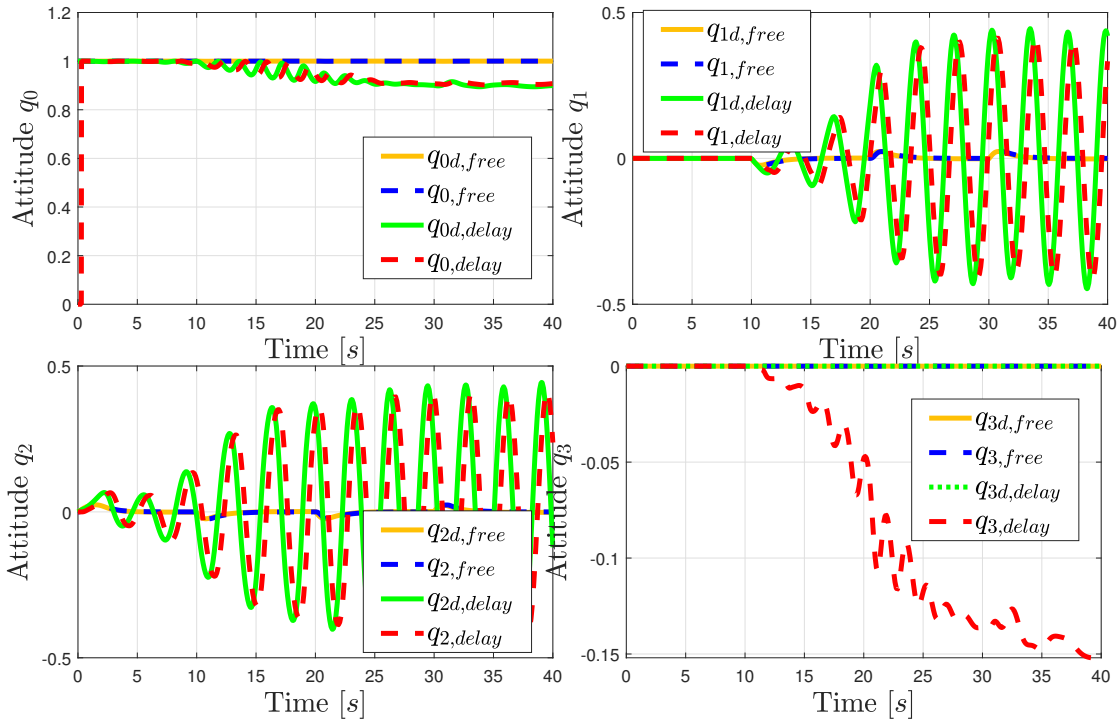


Figure 4.33 – Desired and measured attitude of the delay-free system and delayed system, delay $h = 0.35$ seconds.

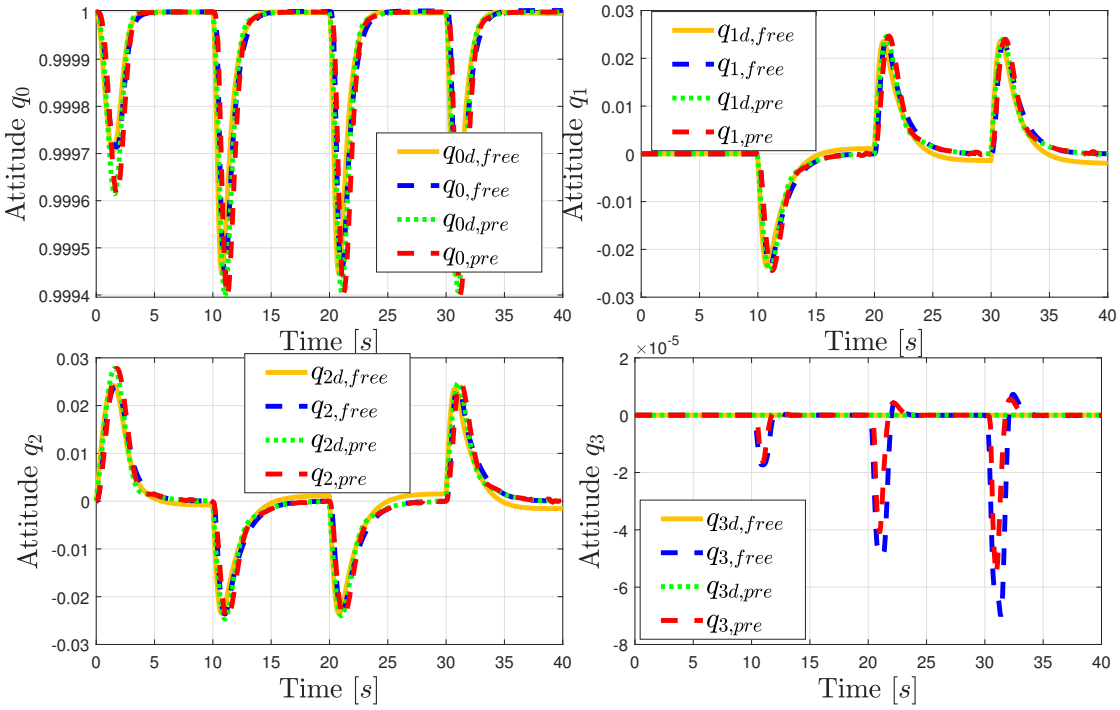


Figure 4.34 – Desired and measured attitude of the delay-free system and predicted-based control in the system with delay, delay $h = 0.35$ seconds.

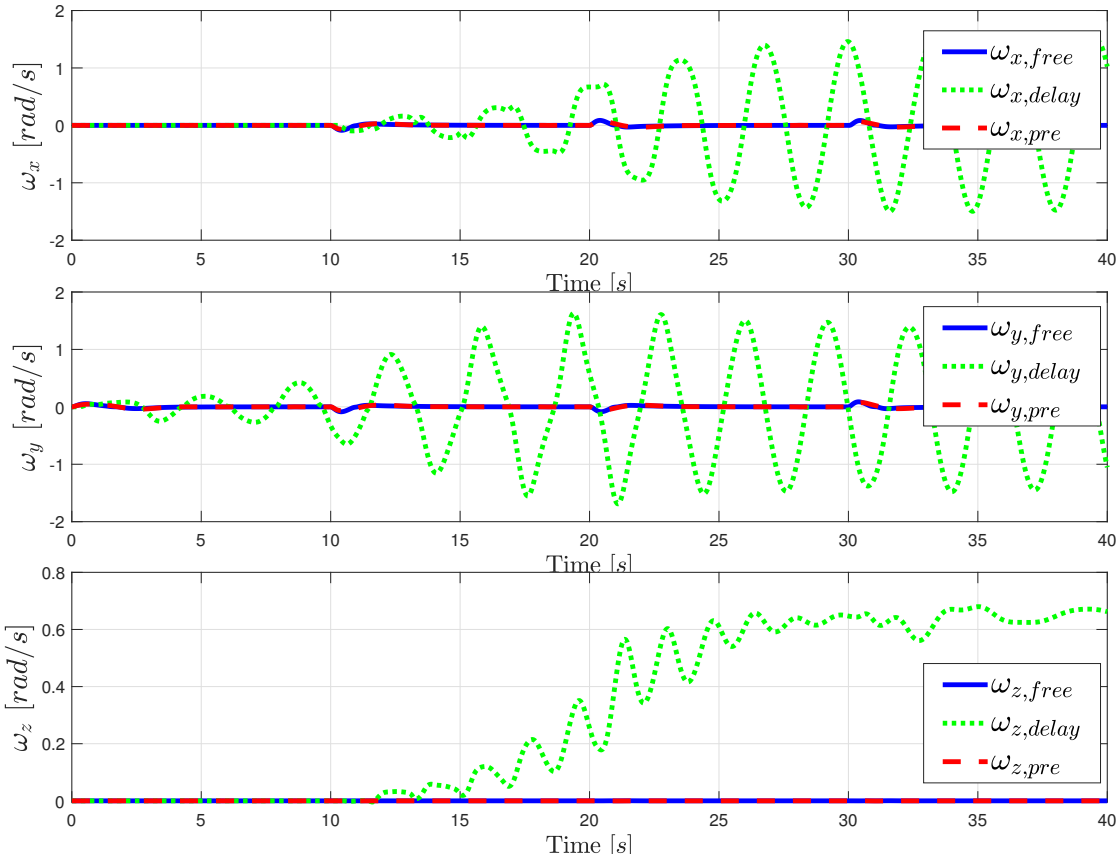


Figure 4.35 – Angular speed, delay $h = 0.35$ seconds.

4.6 Conclusion

On one hand, the analysis of the basic concepts of delay systems allowed us to understand the challenges of TDSs. The main concepts are the definition of the state through a function instead of a vector and the representation of the characteristic equation as a transcendent function implies infinite dimensional solutions. On the other hand, the research about the Smith predictor provided an idea about the predictor-based approaches, applying the Smith predictor to a delayed system a better response performance was observed (see Figure 4.7). Finally, the study of the works presented in [160] and [149] provided us with a methodology to predict the states of a quadrotor with input delays using the FTC. This approach allowed us to perform simulations in Simulink where we validated the performance of the predictor. An advantage of this method is its feasibility to apply in a real drone.

5

Conclusions and future work

UAVs have been in constant development since the First World War due to the benefits of their physical characteristics such as agility, autonomy, and size in different applications. Much research has been done about these vehicles in fields such as guidance, navigation, control, among others. The use of drones in search and rescue in natural disasters has been spreading since they can be operated remotely without endangering the human operator in dangerous environments. Even though drones are remotely operated vehicles, a drone teleoperation system could further extend their capabilities. Therefore, it is crucial and challenging to develop new drone teleoperation systems using emerging technologies.

In this thesis, a practical teleoperation system of a quadrotor has been developed. On the one hand, a virtual drone simulation commanded through a standard joystick was established in the local site as the master system. A local controller was implemented in the virtual quadrotor to conduct its flight by a pilot. This user interface was designed in Unity 3D; first and third-person views were inserted into the environment. These views help the user not to lose awareness of the environment when the vehicle rotates. Besides, this local system can be used as a training simulator since its operation does not depend on the real quadrotor.

On the other hand, in the remote system, a real quadrotor in a structured environment was equipped with another local controller to perform flight tasks. The controller was designed with a quaternion-based approach which gives flight robustness. A UDP communication links the quadrotor aircraft to the virtual interface. The virtual vehicle's position and orientation states are sent and followed as references by the quadrotor in the real environment. The states of the real vehicle are returned to the virtual environment. They are shown to the user as a virtual ellipsoid. This return of information from the drone's real states, as a virtual object, does not generate high bandwidth consumption compared to the visual feedback from the camera on board the drone.

Derived from different tests with our system, when the local interface is far from the real quadrotor a delay problem was observed. This problem was addressed within the framework of predictive type controllers. A controller based on the Fundamental Theorem of Calculus was designed. These types of controllers depends on the system model. Two models were studied: the Euler Lagrange model and a model based on quaternions. Numerical simulations were developed with both models. The simulations showed a good performance recovering the states of the system for a certain delay time. However, given the nature of the models used, a better performance was observed in the quaternion-based model.

5.1 Future work

During the development of this thesis, several points that can be improved have been detected, hence future work are numerous.

First of all, this work was mainly focused on developing a quadrotor teleoperation system for inexperienced users using a common joystick. However there are more sophisticated devices that provide tactile feedback such as haptic joysticks, bracelets or gloves that can be integrated into the virtual system to provide greater transparency.

Another important point regarding the teleoperation system is the improvement of the dynamics of the virtual vehicle by designing virtual sensors such as engines, propellers, and the physical structure. The ROS system could be used for this purpose.

In this work, the remote environment was assumed to be known, therefore the information from the drone camera was not taken into account. However, for unknown remote environments it might be possible to take information from the drone's camera and using image processing techniques to recreate the environment, or to install LIDAR devices on-board the vehicle.

In terms of theoretical control, an improvement to this system could be the design of a controller that regulates the states of the virtual and real vehicle in order to reduce errors and become a bilateral system. Implementation of a trajectory predictive technique such as model predictive control could help to determine the feasibility of trajectories.

In this project, the delay time was assumed to be constant. However, this assumption could be removed by modeling the delay as a bounded continuous variant time function, which is more realistic and challenging. In this sense, another line of research is the study of the estimation of the system delay when it is variable.

The controller proposed in this work could be improved by analytically determining the critical values that the system can support using the LMI method to determine the limits of the model parameters.

Experiments with the predictor-based controller will be tested in the future. Integration of the proposed predictor-based controller for delay compensation within the teleoperation system also remains future work.

Bibliography

[1] Werner Alexander Isop et al. "High-level teleoperation system for aerial exploration of indoor environments". In: *Frontiers in Robotics and AI* 6 (2019).

[2] Conrad Monson et al. "Addressing the human element in unmanned aerial vehicles". In: *36th AIAA Aerospace Sciences Meeting and Exhibit*. 1998, p. 1032.

[3] DR Oliver and AL Money. *Unmanned Aerial Vehicles Roadmap*. Tech. rep. Technical Report, Department of Defense, Washington DC, 2001.

[4] Zhijun Li, Yuanqing Xia, and Chun-Yi Su. "Intelligent networked teleoperation control". In: (2015).

[5] Róbinson Jiménez Moreno, Fabio Andrés Espinosa Valcárcel, and Darío Amaya Hurtado. "Teleoperated systems: a perspective on telesurgery applications". In: *Revista Ingeniería Biomédica* 7.14 (2013), pp. 30–41.

[6] Peter F Hokayem and Mark W Spong. "Bilateral teleoperation: An historical survey". In: *Automatica* 42.12 (2006), pp. 2035–2057.

[7] Tesla Nikola. *Method of and apparatus for controlling mechanism of moving vessels or vehicles*. US Patent 613,809. 1898.

[8] Raymond C Goertz, William M Thompson, and Robert A Olsen. *Electronic master slave manipulator*. US Patent 2,846,084. 1958.

[9] MW Thring. "Teleoperated Robotics in Hostile Environments, edited by H. Lee Martin and Daniel P. Kuban. Robotics International of SME (Distributed in the UK byAmerican Technical Publishers, Hitchin, Herts.), 1985 (£ 35)". In: *Robotica* 5.1 (1987), pp. 81–81.

[10] Gilles Clement et al. "An overview of CAT control in nuclear services". In: *Proceedings. 1985 IEEE International Conference on Robotics and Automation*. Vol. 2. IEEE. 1985, pp. 713–718.

[11] D Yoerger and J-J Slotine. "Supervisory control architecture for underwater teleoperation". In: *Proceedings. 1987 IEEE International Conference on Robotics and Automation*. Vol. 4. IEEE. 1987, pp. 2068–2073.

[12] Dana Yoerger, James Newman, and J-J Slotine. "Supervisory control system for the JASON ROV". In: *IEEE Journal of Oceanic Engineering* 11.3 (1986), pp. 392–400.

[13] A Madni, Yee-yeen Chu, and Amos Freedy. "Intelligent interface for remote supervision and control of underwater manipulation". In: *Proceedings OCEANS'83*. IEEE. 1983, pp. 106–110.

[14] Janez Funda and Richard P Paul. "A symbolic teleoperator interface for time-delayed underwater robot manipulation". In: *OCEANS 91 Proceedings*. IEEE. 1991, pp. 1526–1533.

[15] Dong-Soo Kwon et al. "Design of a teleoperation controller for an underwater manipulator". In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. Vol. 4. 2000, pp. 3114–3119.

[16] Qingping Lin and Chengi Kuo. "Virtual tele-operation of underwater robots". In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. Vol. 2. 1997, pp. 1022–1027.

[17] Woo-Keun Yoon et al. "Model-based space robot teleoperation of ETS-VII manipulator". In: *IEEE Transactions on Robotics and Automation* 20.3 (2004), pp. 602–612.

[18] Sukhan Lee, G Bekey, and AK Bejczy. "Computer control of space-borne teleoperators with sensory feedback". In: *Proceedings. 1985 IEEE International Conference on Robotics and Automation*. Vol. 2. IEEE. 1985, pp. 205–214.

[19] L Jenkins. "Telerobotic work system-space robotics application". In: *Proceedings. 1986 IEEE International Conference on Robotics and Automation*. Vol. 3. IEEE. 1986, pp. 804–806.

[20] Takashi Imaida et al. "Ground-space bilateral teleoperation of ETS-VII robot arm by direct bilateral coupling under 7-s time delay condition". In: *IEEE Transactions on Robotics and Automation* 20.3 (2004), pp. 499–511.

[21] Gerd Hirzinger, Johann Heindl, and Klaus Landzettel. "Predictive and knowledge-based telerobotic control concepts". In: *1989 IEEE International Conference on Robotics and Automation*. IEEE Computer Society. 1989, pp. 1768–1769.

[22] Gerd Hirzinger et al. "Sensor-based space robotics-ROTEX and its telerobotic features". In: *IEEE Transactions on robotics and automation* 9.5 (1993), pp. 649–663.

[23] Gerd Hirzinger. "The space and telerobotic concepts of the DFVLR ROTEX". In: *Proceedings. 1987 IEEE International Conference on Robotics and Automation*. Vol. 4. IEEE. 1987, pp. 443–449.

[24] ANTALK Bejczy and Zoltan Szakaly. "Universal computer control systems (uccs) for space telerobots". In: *Proceedings. 1987 IEEE International Conference on Robotics and Automation*. Vol. 4. IEEE. 1987, pp. 318–324.

[25] Akhil J Madhani, Günter Niemeyer, and J Kenneth Salisbury. "The black falcon: a teleoperated surgical instrument for minimally invasive surgery". In: *Proceedings. 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications (Cat. No. 98CH36190)*. Vol. 2. IEEE. 1998, pp. 936–944.

[26] Janez Funda et al. "Constrained Cartesian motion control for teleoperated surgical robots". In: *IEEE Transactions on Robotics and Automation* 12.3 (1996), pp. 453–465.

[27] Sebastian Gnatzig et al. "A System Design for Teleoperated Road Vehicles." In: *ICINCO (2)*. 2013, pp. 231–238.

[28] S Lichiardopol. "A survey on teleoperation". In: *Technische Universitat Eindhoven, DCT report* 20 (2007), pp. 40–60.

[29] Randal W Beard et al. "Decentralized cooperative aerial surveillance using fixed-wing miniature UAVs". In: *Proceedings of the IEEE* 94.7 (2006), pp. 1306–1324.

[30] Malcolm Thomas Connolly et al. "The Use of Multi Rotor Remotely Operated Aerial Vehicles (ROAVs) as a Method of Close Visually Inspecting (CVI) Live and Difficult to Access Assets on Offshore Platforms". In: *Abu Dhabi International Petroleum Exhibition and Conference*. Society of Petroleum Engineers. 2014.

[31] David W Casbeer et al. "Cooperative forest fire surveillance using a team of small unmanned air vehicles". In: *International Journal of Systems Science* 37.6 (2006), pp. 351–360.

[32] Fausto G Costa et al. "The use of unmanned aerial vehicles and wireless sensor network in agricultural applications". In: *2012 IEEE International Geoscience and Remote Sensing Symposium*. IEEE. 2012, pp. 5045–5048.

[33] Michael A Goodrich et al. "Using a mini-uav to support wilderness search and rescue: Practices for human-robot teaming". In: *2007 IEEE International Workshop on Safety, Security and Rescue Robotics*. IEEE. 2007, pp. 1–6.

[34] Haider AF Almurib, Premeela T Nathan, and T Nandha Kumar. "Control and path planning of quadrotor aerial vehicles for search and rescue". In: *SICE Annual Conference 2011*. IEEE. 2011, pp. 700–705.

[35] Tomonari Furukawa et al. "Recursive Bayesian search-and-tracking using coordinated UAVs for lost targets". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE. 2006, pp. 2521–2526.

[36] Matteo Macchini et al. "Hand-worn Haptic Interface for Drone Teleoperation". In: *arXiv preprint arXiv:2004.07111* (2020).

[37] Stefano Stramigioli, Robert Mahony, and Peter Corke. "A novel approach to haptic tele-operation of aerial robot vehicles". In: *2010 IEEE International Conference on Robotics and Automation*. IEEE. 2010, pp. 5302–5308.

[38] Minh-Duc Hua and Hala Rifaï. "Obstacle avoidance for teleoperated underactuated aerial vehicles using telemetric measurements". In: *49th IEEE Conference on Decision and Control (CDC)*. IEEE. 2010, pp. 262–267.

[39] S. Omari et al. "Bilateral haptic teleoperation of VTOL UAVs". In: *2013 IEEE International Conference on Robotics and Automation*. 2013, pp. 2393–2399.

[40] Robert Mahony et al. "A new framework for force feedback teleoperation of robotic vehicles based on optical flow". In: *2009 IEEE International Conference on Robotics and Automation*. IEEE. 2009, pp. 1079–1085.

[41] Thanh Mung Lam et al. "Artificial force field for haptic feedback in UAV teleoperation". In: *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 39.6 (2009), pp. 1316–1330.

[42] Hala Rifaï et al. "Haptic-based bilateral teleoperation of underactuated unmanned aerial vehicles". In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 13782–13788.

[43] Dingjiang Zhou, Zijian Wang, and Mac Schwager. "Agile coordination and assistive collision avoidance for quadrotor swarms using virtual structures". In: *IEEE Transactions on Robotics* 34.4 (2018), pp. 916–923.

[44] Paolo Stegagno et al. "Vision-based autonomous control of a quadrotor UAV using an onboard RGB-D camera and its application to haptic teleoperation". In: *IFAC Proc. Volumes* 46.30 (2013), pp. 87–92.

[45] Ramon A Suarez Fernandez et al. "Natural user interfaces for human-drone multi-modal interaction". In: *International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE. 2016, pp. 1013–1022.

[46] John Paulin Hansen et al. "The use of gaze to control drones". In: *Proc. Symposium on Eye Tracking Research and Applications*. ACM. 2014, pp. 27–34.

[47] Shuwei Zhang et al. "Research on Multi-modal Interactive Control for Quadrotor UAV". In: *2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC)*. IEEE. 2019, pp. 329–334.

[48] Abdur Razzaq Fayjie et al. "Voice enabled smart drone control". In: *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE. 2017, pp. 119–121.

[49] Carine Rognon et al. "Flyjacket: An upper body soft exoskeleton for immersive drone control". In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 2362–2369.

[50] Alexandre Cherpillod, Dario Floreano, and Stefano Mintchev. "Embodied flight with a drone". In: *2019 Third IEEE International Conference on Robotic Computing (IRC)*. IEEE. 2019, pp. 386–390.

[51] Keita Higuchi, Katsuya Fujii, and Jun Rekimoto. "Flying head: A head-synchronization mechanism for flying telepresence". In: *2013 23rd International Conference on Artificial Reality and Telexistence (ICAT)*. IEEE. 2013, pp. 28–34.

[52] Jenifer Miehlbradt et al. "Data-driven body–machine interface for the accurate control of drones". In: *Proceedings of the National Academy of Sciences* 115.31 (2018), pp. 7913–7918.

[53] Hooman Hedayati, Michael Walker, and Daniel Szafir. "Improving collocated robot teleoperation with augmented reality". In: *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*. 2018, pp. 78–86.

[54] Matteo Macchini, Fabrizio Schiano, and Dario Floreano. "Personalized Telerobotics by Fast Machine Learning of Body-Machine Interfaces". In: *IEEE Robotics and Automation Letters* 5.1 (2019), pp. 179–186.

[55] Ronald T Azuma. "A survey of augmented reality". In: *Presence: Teleoperators & Virtual Environments* 6.4 (1997), pp. 355–385.

[56] Scott A Green et al. "Human-robot collaboration: A literature review and augmented reality approach in design". In: *International journal of advanced robotic systems* 5.1 (2008), p. 1.

[57] Ronald Azuma et al. "Recent advances in augmented reality". In: *IEEE computer graphics and applications* 21.6 (2001), pp. 34–47.

[58] Okan Erat et al. "Drone-augmented human vision: Exocentric control for drones exploring hidden areas". In: *IEEE transactions on visualization and computer graphics* 24.4 (2018), pp. 1437–1446.

[59] Sébastien Thon et al. "Flying a drone in a museum: An augmented-reality cultural serious game in Provence". In: *2013 Digital Heritage International Congress (DigitalHeritage)*. Vol. 2. IEEE. 2013, pp. 669–676.

[60] Metehan Unal, Erkan Bostanci, and Evren Sertalp. "Distant augmented reality: Bringing a new dimension to user experience using drones". In: *Digital Applications in Archaeology and Cultural Heritage* (2020), e00140.

[61] Janna Huuskonen and Timo Oksanen. "Soil sampling with drones and augmented reality in precision agriculture". In: *Computers and electronics in agriculture* 154 (2018), pp. 25–35.

[62] Baichuan Huang et al. "Flight, Camera, Action! Using Natural Language and Mixed Reality to Control a Drone". In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. 2019.

[63] Bryan Walter et al. "Virtual UAV ground control station". In: *AIAA 3rd" Unmanned Unlimited" Technical Conference, Workshop and Exhibit*. 2004, p. 6320.

[64] Robert Codd-Downey et al. "From ROS to unity: Leveraging robot and virtual environment middleware for immersive teleoperation". In: *2014 IEEE International Conference on Information and Automation (ICIA)*. IEEE. 2014, pp. 932–936.

[65] Simon Westerberg et al. "Virtual environment teleoperation of a hydraulic forestry crane". In: *2008 IEEE International Conference on Robotics and Automation*. IEEE. 2008, pp. 4049–4054.

[66] MA Steffen, JD Will, and N Murakami. "Use of virtual reality for teleoperation of autonomous vehicles". In: *American Society of Agricultural and Biological Engineers Biological Sensorics Conference*. Citeseer. 2007.

[67] Li Huijun and Song Aiguo. "Virtual-environment modeling and correction for force-reflecting teleoperation with time delay". In: *IEEE Transactions on Industrial Electronics* 54.2 (2007), pp. 1227–1233.

[68] Keqin Gu and Silviu-Iulian Niculescu. "Survey on recent results in the stability and control of time-delay systems". In: *J. Dyn. Sys., Meas., Control* 125.2 (2003), pp. 158–165.

[69] Minghui Kao and John J Moskwa. "Turbocharged diesel engine modeling for nonlinear engine control and state estimation". In: (1995).

[70] John Harold Horlock and DE Winterbone. "The thermodynamics and gas dynamics of internal-combustion engines. Volume II". In: (1986).

[71] Jeffrey A Cook and Barry K Powell. "Modeling of an internal combustion engine for control analysis". In: *IEEE Control Systems Magazine* 8.4 (1988), pp. 20–26.

[72] H Mounier et al. "High Speed Network Congestion Control with a Simplified Time-Varying Delay Mode". In: *IFAC Proceedings Volumes* 31.18 (1998), pp. 43–47.

[73] Srikant Shakkottai, Rayadurgam Srikant, and Sean Meyn. "Boundedness of utility function based congestion controllers in the presence of delay". In: *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228)*. Vol. 1. IEEE. 2001, pp. 616–621.

[74] Chaouki Abdallah and Jolm Chiasson. "Stability of communications networks in the presence of delays". In: *IFAC Proceedings Volumes* 34.23 (2001), pp. 171–174.

[75] L Bushnell. "Editorial: networks and control". In: *IEEE Control System Magazine* 21.1 (2001), pp. 22–99.

[76] P-F Quet, Sthanunathan Ramakrishnan, and Hitay Ozbay. "On the H_∞ controller design for congestion control in communication networks with a capacity predictor". In: *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228)*. Vol. 1. IEEE. 2001, pp. 598–603.

[77] Enis Biberovic, Altug Iftar, and Hitay Ozbay. "A solution to the robust flow control problem for networks with multiple bottlenecks". In: *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228)*. Vol. 3. IEEE. 2001, pp. 2303–2308.

[78] Rauf Izmailov. "Analysis and optimization of feedback control algorithms for data transfers in high-speed networks". In: *SIAM journal on control and optimization* 34.5 (1996), pp. 1767–1780.

[79] Chaouki Abdallah et al. "Load balancing instabilities due to time delays in parallel computations". In: *IFAC Proceedings Volumes* 34.23 (2001), pp. 175–179.

[80] Chaouki T Abdallah et al. "Linear time delay model for studying load balancing instabilities in parallel computations". In: *International Journal of Systems Science* 34.10-11 (2003), pp. 563–573.

[81] John Chiasson et al. "The effect of time delays on the stability of load balancing algorithms for parallel computations". In: *IEEE Transactions on Control Systems Technology* 13.6 (2005), pp. 932–942.

[82] JD Birdwell et al. "The effect of time delays in the stability of load balancing algorithms for parallel computations". In: *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*. Vol. 1. IEEE. 2003, pp. 582–587.

[83] Johan Nilsson et al. "Real-time control systems with delays". In: (1998).

[84] Shan-Ben Chen, L Wu, and QL Wang. "Self-learning fuzzy neural networks for control of uncertain systems with time delays". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 27.1 (1997), pp. 142–148.

[85] R Rakkiyappan, Natarajan Sakthivel, and Jinde Cao. "Stochastic sampled-data control for synchronization of complex dynamical networks with control packet loss and additive time-varying delays". In: *Neural Networks* 66 (2015), pp. 46–63.

[86] M Kaiser. "Time-delay neural networks for control". In: *IFAC Proceedings Volumes* 27.14 (1994), pp. 967–972.

[87] Kang G Shin and Xianzhong Cui. "Computing time delay and its effects on real-time control systems". In: *IEEE Transactions on control systems technology* 3.2 (1995), pp. 218–224.

[88] Pradeep Khosla. "Choosing sampling rates for robot control". In: *Proceedings. 1987 IEEE International Conference on Robotics and Automation*. Vol. 4. IEEE. 1987, pp. 169–174.

[89] Amit Ailon and Michael I Gil. "Stability analysis of a rigid robot with output-based controller and time delay". In: *Systems & control letters* 40.1 (2000), pp. 31–35.

[90] Günter Dieter Niemeyer. "Using wave variables in time delayed force reflecting teleoperation". PhD thesis. Massachusetts Institute of Technology, 1996.

[91] Arnaud Lelevé, Philippe Fraisse, and Pierre Dauchez. "Telerobotics over IP networks: Towards a low-level real-time architecture". In: *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*. Vol. 2. IEEE. 2001, pp. 643–648.

[92] Günter Niemeyer and J-JE Slotine. "Towards force-reflecting teleoperation over the internet". In: *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*. Vol. 3. IEEE. 1998, pp. 1909–1915.

[93] Won S Kim, Blake Hannaford, and Antal K Bejczy. "Force-reflection and shared compliant control in operating telemanipulators with time delay". In: *Robotics and Automation, IEEE Transactions on* 8.2 (1992), pp. 176–185.

[94] Günter Niemeyer and J-JE Slotine. "Stable adaptive teleoperation". In: *IEEE Journal of oceanic engineering* 16.1 (1991), pp. 152–162.

[95] Günter Niemeyer and J-JE Slotine. "Designing force reflecting teleoperators with large time delays to appear as virtual tools". In: *Proceedings of International Conference on Robotics and Automation*. Vol. 3. IEEE. 1997, pp. 2212–2218.

[96] Robert J Anderson and Mark W Spong. "Bilateral control of teleoperators with time delay". In: *IEEE Transactions on Automatic control* 34.5 (1989), pp. 494–501.

[97] T Mung Lam, Max Mulder, and MM Van Paassen. "Collision avoidance in UAV tele-operation with time delay". In: *2007 IEEE International Conference on Systems, Man and Cybernetics*. IEEE. 2007, pp. 997–1002.

[98] Yasuyoshi Yokokohji, Takashi Imaida, and Tsuneo Yoshikawa. "Bilateral control with energy balance monitoring under time-varying communication delay". In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*. Vol. 3. IEEE. 2000, pp. 2684–2689.

[99] Huiyu Sun et al. "Bilateral teleoperation of an unmanned aerial vehicle for forest fire detection". In: *2017 IEEE International Conference on Information and Automation (ICIA)*. IEEE. 2017, pp. 586–591.

[100] Hui-yu Sun et al. "Energy-optimized consensus formation control for the time-delayed bilateral teleoperation system of UAVs". In: *International Journal of Aerospace Engineering* 2018 (2018).

[101] Emanuel Slawiñski et al. "Teleoperation of a mobile robot with time-varying delay and force feedback". In: *Robotica* 30.1 (2012), pp. 67–77.

[102] E Slawiñski, D Santiago, and V Mut. "Control for delayed bilateral teleoperation of a quadrotor". In: *ISA transactions* 71 (2017), pp. 415–425.

[103] Shahzad Khan, Asif Sabanovic, and Ahmet Ozcan Nergiz. "Scaled bilateral teleoperation using discrete-time sliding-mode controller". In: *IEEE Transactions on Industrial Electronics* 56.9 (2009), pp. 3609–3618.

[104] L-G García-Valdovinos, Vicente Parra-Vega, and Marco A Arteaga. "Observer-based sliding mode impedance control of bilateral teleoperation under constant unknown time delay". In: *Robotics and Autonomous Systems* 55.8 (2007), pp. 609–617.

[105] Hyun Chul Cho and Jong Hyeon Park. "Stable bilateral teleoperation under a time delay using a robust impedance control". In: *Mechatronics* 15.5 (2005), pp. 611–625.

[106] Aleš Hace and Karel Jezernik. "Bilateral teleoperation by sliding mode control and reaction force observer". In: *2010 IEEE International Symposium on Industrial Electronics*. IEEE. 2010, pp. 1809–1816.

[107] John M Daly and David WL Wang. "Bilateral teleoperation using unknown input observers for force estimation". In: *2009 American Control Conference*. IEEE. 2009, pp. 89–95.

[108] Pietro Buttolo, Petter Braathen, and Blake Hannaford. "Sliding control of force reflecting teleoperation: Preliminary studies". In: *Presence: Teleoperators & Virtual Environments* 3.2 (1994), pp. 158–172.

[109] Jong H Park and Hyun C Cho. "Sliding mode control of bilateral teleoperation systems with force-reflection on the internet". In: *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)(Cat. No. 00CH37113)*. Vol. 2. IEEE. 2000, pp. 1187–1192.

[110] Hyun Chul Cho et al. "Sliding-mode-based impedance controller for bilateral teleoperation under varying time-delay". In: *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*. Vol. 1. IEEE. 2001, pp. 1025–1030.

[111] Jong Hyeon Park and Hyun Chul Cho. "Sliding-mode controller for bilateral teleoperation with varying time delay". In: *1999 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (Cat. No. 99TH8399)*. IEEE. 1999, pp. 311–316.

[112] Alberto Bemporad. "Predictive control of teleoperated constrained systems with unbounded communication delays". In: *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No. 98CH36171)*. Vol. 2. IEEE. 1998, pp. 2133–2138.

[113] J Sheng and MW Spong. "Model predictive control for bilateral teleoperation systems with time delays". In: *Canadian Conference on Electrical and Computer Engineering 2004 (IEEE Cat. No. 04CH37513)*. Vol. 4. IEEE. 2004, pp. 1877–1880.

[114] Andrew C Smith and K Van Hashtrudi-Zaad. "Neural network-based teleoperation using Smith predictors". In: *IEEE International Conference Mechatronics and Automation, 2005*. Vol. 3. IEEE. 2005, pp. 1654–1659.

[115] Vincent Duchaine and Clement M Gosselin. "General model of human-robot cooperation using a novel velocity based variable impedance control". In: *Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC'07)*. IEEE. 2007, pp. 446–451.

[116] RH Middletone and GC Goodwin. "Adaptive computed torque control for rigid link manipulators". In: *1986 25th IEEE Conference on Decision and Control*. IEEE. 1986, pp. 68–73.

[117] Emmanuel Nuño, Romeo Ortega, and Luis Basañez. "An adaptive controller for nonlinear teleoperators". In: *Automatica* 46.1 (2010), pp. 155–159.

[118] Nikhil Chopra and Mark W Spong. "Adaptive coordination control of bilateral teleoperators with time delay". In: *2004 43rd IEEE Conference on Decision and Control (CDC)(IEEE Cat. No. 04CH37601)*. Vol. 5. IEEE. 2004, pp. 4540–4547.

[119] NVQ Hung, T Narikiyo, and HD Tuan. "Nonlinear adaptive control of master–slave system in teleoperation". In: *Control Engineering Practice* 11.1 (2003), pp. 1–10.

[120] Stella Clarke et al. "Prediction-based methods for teleoperation across delayed networks". In: *Multimedia Systems* 13.4 (2008), pp. 253–261.

[121] B Wayne Bequette. "Nonlinear control of chemical processes: A review". In: *Industrial & Engineering Chemistry Research* 30.7 (1991), pp. 1391–1413.

[122] Zvi Artstein. "Linear systems with delayed controls: a reduction". In: *IEEE Transactions on Automatic control* 27.4 (1982), pp. 869–879.

[123] YAPE Fiagbedzi and A Pearson. "Feedback stabilization of linear autonomous time lag systems". In: *IEEE Transactions on Automatic Control* 31.9 (1986), pp. 847–855.

[124] Mrdjan Jankovic. "Recursive predictor design for linear systems with time delay". In: *2008 American control conference*. IEEE. 2008, pp. 4904–4909.

[125] Keqin Gu, Jie Chen, and Vladimir L Kharitonov. *Stability of time-delay systems*. Springer Science & Business Media, 2003.

[126] Miroslav Krstic. "Lyapunov tools for predictor feedbacks for delay systems: Inverse optimality and robustness to delay mismatch". In: *Automatica* 44.11 (2008), pp. 2930–2935.

[127] Woosuk Kwon and A Pearson. "Feedback stabilization of linear systems with delayed control". In: *IEEE Transactions on Automatic control* 25.2 (1980), pp. 266–269.

[128] AWOAZ Manitius and A Olbrot. "Finite spectrum assignment problem for systems with delays". In: *IEEE transactions on Automatic Control* 24.4 (1979), pp. 541–552.

[129] Sabine Mondié and Wim Michiels. "Finite spectrum assignment of unstable time-delay systems with a safe implementation". In: *IEEE Transactions on Automatic Control* 48.12 (2003), pp. 2207–2212.

[130] Jean-Pierre Richard. "Time-delay systems: an overview of some recent advances and open problems". In: *automatica* 39.10 (2003), pp. 1667–1694.

[131] Young-Hoon Roh and Jun-Ho Oh. "Robust stabilization of uncertain input-delay systems by sliding mode control with delay compensation". In: *Automatica* 35.11 (1999), pp. 1861–1865.

[132] SNAPDS Evesque et al. "Adaptive control of a class of time-delay systems". In: *J. Dyn. Sys., Meas., Control* 125.2 (2003), pp. 186–193.

[133] Robert Riener and Thomas Fuhr. "Patient-driven control of FES-supported standing up: a simulation study". In: *IEEE Transactions on rehabilitation engineering* 6.2 (1998), pp. 113–124.

[134] Diana Yanakiev and Ioannis Kanellakopoulos. "Longitudinal control of automated CHVs with significant actuator delays". In: *Proceedings of the 36th IEEE Conference on Decision and Control*. Vol. 5. IEEE. 1997, pp. 4756–4763.

[135] John Chiasson and Jean Jacques Loiseau. *Applications of time delay systems*. Vol. 352. Springer, 2007.

[136] Wim Michiels et al. "Continuous pole placement for delay equations". In: *Automatica* 38.5 (2002), pp. 747–761.

[137] Otto JM Smith. "A controller to overcome dead time". In: *ISA J.* 6 (1959), pp. 28–33.

[138] Wei Dong Zhang and You Xian Sun. "Modified Smith predictor for controlling integrator/time delay processes". In: *Industrial & engineering chemistry research* 35.8 (1996), pp. 2769–2772.

[139] Anne Nortcliffe and Jonathan Love. "Varying time delay Smith predictor process controller". In: *ISA transactions* 43.1 (2004), pp. 61–71.

[140] MR Matausek and AD Micic. "A modified Smith predictor for controlling a process with an integrator and long dead-time". In: *IEEE transactions on automatic control* 41.8 (1996), pp. 1199–1203.

[141] S Majhi and DP Atherton. "Modified Smith predictor and controller for processes with time delay". In: *IEE Proceedings-Control Theory and Applications* 146.5 (1999), pp. 359–366.

[142] Somanath Majhi and Derek P Atherton. "Obtaining controller parameters for a new Smith predictor using autotuning". In: *Automatica* 36.11 (2000), pp. 1651–1658.

[143] Pedro García and Pedro Albertos. "A new dead-time compensator to control stable and integrating processes with long dead-time". In: *Automatica* 44.4 (2008), pp. 1062–1071.

[144] I-Lung Chien, Sheng Chun Peng, and Jun Hong Liu. "Simple control method for integrating processes with long deadtime". In: *Journal of Process Control* 12.3 (2002), pp. 391–404.

[145] Miroslav Krstic. *Delay compensation for nonlinear, adaptive, and PDE systems*. Springer, 2009.

[146] Delphine Bresch-Pietri and Miroslav Krstic. "Adaptive trajectory tracking despite unknown input delay and plant parameters". In: *Automatica* 45.9 (2009), pp. 2074–2081.

[147] Bin Zhou, Zongli Lin, and Guang-Ren Duan. "Truncated predictor feedback for linear systems with long time-varying input delays". In: *Automatica* 48.10 (2012), pp. 2387–2399.

[148] Rogelio Lozano et al. "Robust prediction-based control for unstable delay systems: Application to the yaw control of a mini-helicopter". In: *Automatica* 40.4 (2004), pp. 603–612.

[149] Jairo Ordaz et al. "Predictor-based position control of a quad-rotor with delays in GPS and vision measurements". In: *Journal of intelligent & robotic systems* 70.1-4 (2013), pp. 13–26.

[150] Ricardo Sanz et al. "Predictor-based control of a class of time-delay systems and its application to quadrotors". In: *IEEE Transactions on Industrial Electronics* 64.1 (2016), pp. 459–469.

[151] Hamed Jabbari Asl, Seyyed H Mahdioun, and Jungwon Yoon. "Vision-based control of an underactuated flying robot with input delay". In: *Transactions of the Institute of Measurement and Control* 40.2 (2018), pp. 446–455.

[152] Hao Liu et al. "Robust control for quadrotors with multiple time-varying uncertainties and delays". In: *IEEE Transactions on Industrial Electronics* 64.2 (2016), pp. 1303–1312.

[153] Ning Cao. "Control of quadrotor unmanned aerial vehicles with saturation and time delay". In: (2018).

[154] Hernan Abaunza et al. "Quaternion based control for circular UAV trajectory tracking, following a ground vehicle: Real-time validation". In: *IFAC-PapersOnLine* 50 (2017), pp. 11453–11458.

[155] Guillaume Sanahuja and et al. *Fl-AIR Framework libre AIR*. urlhttps://devel.hds.utc.fr/so Accessed 29-07-2019.

[156] Pablo Amster. "Ecuaciones diferenciales con retardo". In: *Cursos y seminarios de matemática Serie B* (2017).

[157] Vladimir Kharitonov. *Time-delay systems: Lyapunov functionals and matrices*. Springer Science & Business Media, 2012.

[158] Emilia Fridman. *Introduction to time-delay systems: Analysis and control*. Springer, 2014.

[159] Pedro Castillo Garcia, Rogelio Lozano, and Alejandro Enrique Dzul. *Modelling and control of mini-flying machines*. Springer Science & Business Media, 2006.

[160] A Alatorre, Pedro Castillo, and S Mondie. "Improving the performance of aerial vehicles with delayed measures via state predictor-control: Experimental data validation". In: *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE. 2014, pp. 910–919.

[161] Grigore C Burdea. "Invited review: the synergy between virtual reality and robotics". In: *IEEE Transactions on Robotics and Automation* 15.3 (1999), pp. 400–410.

[162] Dongjun Lee, Oscar Martinez-Palafox, and Mark W Spong. "Bilateral teleoperation of a wheeled mobile robot over delayed communication network". In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. 2006, pp. 3298–3303.

[163] Christopher Stanton, Anton Bogdanovych, and Edward Ratanasena. "Teleoperation of a humanoid robot using full-body motion capture, example movements, and machine learning". In: *Proc. Australasian Conference on Robotics and Automation (ACRA)*. 2012.

[164] Hirohisa Hirukawa et al. "Humanoid robotics platforms developed in HRP". In: *Robotics and Autonomous Systems* 48.4 (2004), pp. 165–175.

[165] Abeje Yenehun Mersha, Stefano Stramigioli, and Raffaella Carloni. "On bilateral teleoperation of aerial robots". In: *IEEE Transactions on Robotics* 30.1 (2014), pp. 258–274.

[166] Terrence Fong and Charles Thorpe. "Vehicle teleoperation interfaces". In: *Autonomous robots* 11.1 (2001), pp. 9–18.

[167] João Marcelo Teixeira et al. "Teleoperation using google glass and ar, drone for structural inspection". In: *XVI Symposium on Virtual and Augmented Reality*. IEEE. 2014, pp. 28–36.

[168] Jzau-Sheng Lin and Zi-Yang Jiang. "Implementing remote presence using quadcopter control by a non-invasive BCI device". In: *Computer Science and Information Technology* 3.4 (2015), pp. 122–126.

[169] Ding Cheng-jun et al. "Design of mobile robot teleoperation system based on virtual reality". In: *IEEE International Conference on Automation and Logistics*. 2009, pp. 2024–2029.

[170] Vladimir J Lumelsky and Edward Cheung. "Real-time collision avoidance in teleoperated whole-sensitive robot arm manipulators". In: *IEEE Transactions on Systems, Man, and Cybernetics* 23.1 (1993), pp. 194–203.

[171] Massimo Bergamasco et al. "An arm exoskeleton system for teleoperation and virtual environments applications". In: *Proc. IEEE International Conference on Robotics and Automation (ICRA)*. 1994, pp. 1449–1454.

[172] Ahmed Mashood et al. "A gesture based kinect for quadrotor control". In: *International Conference on Information and Communication Technology Research (ICTRC)*. IEEE. 2015, pp. 298–301.

[173] Zainal Rasyid Mahayuddin, Hairina Mohd Jais, and Haslina Arshad. "Comparison of human pilot (remote) control systems in multirotor unmanned aerial vehicle navigation". In: *International Journal on Advanced Science, Engineering and Information Technology* 7.1 (2017), pp. 132–138.

[174] Vincenzo Lippiello and Fabio Ruggiero. "Cartesian impedance control of a UAV with a robotic arm". In: *IFAC Proceedings Volumes* 45.22 (2012), pp. 704–709.

[175] Masmoudi Mostefa et al. "Design of mobile robot teleoperation system based on virtual reality". In: *2015 3rd International Conference on Control, Engineering & Information Technology (CEIT)*. IEEE. 2015, pp. 1–6.

[176] Pierre-Jean Bristeau et al. "The navigation and control technology inside the ar. drone micro uav". In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 1477–1484.

[177] Jesus Pestana et al. "Computer vision based general object following for gps-denied multirotor unmanned vehicles". In: *American Control Conference (ACC), 2014*. IEEE. 2014, pp. 1886–1891.

[178] Shokoofeh Pourmehr et al. ""You two! Take off!": Creating, modifying and commanding groups of robots using face engagement and indirect speech in voice commands". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2013, pp. 137–142.

[179] Luis Fernando Sanchez, Hernan Abaunza, and P Castillo. "Safe navigation control for a quadcopter using user's arm commands". In: *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE. 2017, pp. 981–988.

[180] Kevin Pfeil, Seng Lee Koh, and Joseph LaViola. "Exploring 3d gesture metaphors for interaction with unmanned aerial vehicles". In: *Proceedings of the 2013 international conference on Intelligent user interfaces*. ACM. 2013, pp. 257–266.

[181] Andrea Sanna et al. "A Kinect-based natural interface for quadrotor control". In: *Entertainment Computing* 4.3 (2013), pp. 179–186.

[182] Tiago Ogioni Costalonga et al. "Gesture-Based Controllers to Guide a Quadrotor Using Kinect Sensor". In: *2014 Joint Conference on Robotics: SBR-LARS Robotics Symposium and Robocontrol (SBR LARS Robocontrol)*. IEEE. 2014, pp. 109–112.

[183] Mark R Mine. "Virtual environment interaction techniques". In: *UNC Chapel Hill CS Dept* (1995).

[184] Gregory Dudek, Junaed Sattar, and Anqi Xu. "A visual language for robot control and programming: A human-interface study". In: *Robotics and Automation, 2007 IEEE International Conference on*. IEEE. 2007, pp. 2507–2513.

[185] Dale A Lawrence. "Stability and transparency in bilateral teleoperation". In: *IEEE transactions on robotics and automation* 9.5 (1993), pp. 624–637.

[186] Jonathan Steuer. "Defining virtual reality: Dimensions determining telepresence". In: *Journal of communication* 42.4 (1992), pp. 73–93.

[187] Jeff Craighead et al. "A survey of commercial & open source unmanned vehicle simulators". In: *Robotics and Automation, 2007 IEEE International Conference on*. IEEE. 2007, pp. 852–857.

[188] Vladimir I Pavlovic, Rajeev Sharma, and Thomas S. Huang. "Visual interpretation of hand gestures for human-computer interaction: A review". In: *IEEE Transactions on pattern analysis and machine intelligence* 19.7 (1997), pp. 677–695.

[189] Deborah Hix et al. "User-centered design and evaluation of a real-time battlefield visualization virtual environment". In: *Virtual Reality, 1999. Proceedings, IEEE*. IEEE. 1999, pp. 96–103.

[190] Jace Regenbrecht, Alireza Tavakkoli, and Donald Loffredo. "A Robust and intuitive 3D interface for teleoperation of autonomous robotic agents through immersive virtual reality environments". In: *2017 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE. 2017, pp. 199–200.

[191] Florian Jeanne et al. "EBAGG: Error-Based Assistance for Gesture Guidance in Virtual Environments". In: *2017 IEEE 17th International Conference on Advanced Learning Technologies (ICALT)*. IEEE. 2017, pp. 472–476.

[192] Patrícia RJA Alves et al. "Forward collision warning systems using heads-up displays: Testing usability of two new metaphors". In: *Intelligent Vehicles Symposium Workshops (IV Workshops), 2013 IEEE*. IEEE. 2013, pp. 1–6.

[193] Paul George et al. "DAARIA: Driver assistance by augmented reality for intelligent automobile". In: *arXiv preprint arXiv:1209.6140* (2012).

[194] J Knutzon et al. "Command and control in distributed mission training: An immersive approach". In: *NATO Symposium on Critical Design Issues for the Human-Machine Interface*. 2003.

[195] Glesio Garcia de Paiva et al. "Immersive Ground Control Station for Unmanned Aerial Vehicles". In: *International Conference on Computational Science and Its Applications*. Springer. 2017, pp. 595–604.

[196] M Masmoudi et al. "The teleoperation of a mobile robot in a network without a quality of service guaranteed". In: *2015 12th International Symposium on Programming and Systems (ISPS). IEEE*. 2015, pp. 1–6.

[197] Shafiqul Islam et al. "Haptics and virtual reality based bilateral telemanipulation of miniature aerial vehicle over open communication network". In: *Advanced Robotics (ICAR), 2017 18th International Conference on*. IEEE. 2017, pp. 334–339.

[198] TM Lam, M Mulder, and MM Van Paassen. "Haptic feedback in uninhabited aerial vehicle teleoperation with time delay". In: *Journal of guidance, control, and dynamics* 31.6 (2008), pp. 1728–1739.

[199] Antal K Bejczy, Won S Kim, and Steven C Venema. "The phantom robot: predictive displays for teleoperation with time delay". In: *Proceedings., IEEE International Conference on Robotics and Automation*. IEEE. 1990, pp. 546–551.

[200] Laboratory Heudyasic. *Translife platform*. `https://translife.hds.utc.fr/technical-description/translife/`. Accessed 11-09-2019.

[201] Wang Wei and Yuan Kui. "Teleoperated manipulator for leak detection of sealed radioactive sources". In: *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*. Vol. 2. IEEE. 2004, pp. 1682–1687.

[202] Amna ER Greaves. "State of the art in nuclear telerobotics: focus on the man/machine connection". In: *Telemanipulator and Telepresence Technologies*. Vol. 2351. International Society for Optics and Photonics. 1995, pp. 130–147.

[203] Thomas B Sheridan and William L Verplank. *Human and computer control of undersea teleoperators*. Tech. rep. Massachusetts Inst of Tech Cambridge Man-Machine Systems Lab, 1978.

[204] N Farr et al. "An integrated, underwater optical/acoustic communications system". In: *OCEANS'10 IEEE SYDNEY*. IEEE. 2010, pp. 1–6.

[205] Alan FT Winfield. "Wireless video tele-operation using internet protocols". In: *Proceedings of the Fourteenth International Conference on Unmanned Air Vehicle Systems, Bristol*. 1999.

[206] Jianhong Cui et al. "A review of teleoperation system control". In: *Proceedings of the Florida Conference on Recent Advances in Robotics*. Florida Atlantic University Boca Raton, FL. 2003, pp. 1–12.

[207] Alan Ft Winfield. "Future directions in tele-operated robotics". In: *Telerobotic applications* (2000), pp. 147–163.

[208] Abeje Y Mersha, Stefano Stramigioli, and Raffaella Carloni. "Bilateral teleoperation of underactuated unmanned aerial vehicles: The virtual slave concept". In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. 2012, pp. 4614–4620.

[209] Morgan Quigley, Michael A Goodrich, and Randal W Beard. "Semi-autonomous human-UAV interfaces for fixed-wing mini-UAVs". In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*. Vol. 3. IEEE. 2004, pp. 2457–2462.

[210] Ning Cao and Alan F Lynch. "Predictor-based controllers for UAVs with input delay". In: *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE. 2017, pp. 803–808.

[211] SA Tobias. "Machine tool vibration research". In: *International Journal of Machine Tool Design and Research* 1.1-2 (1961), pp. 1–14.

[212] Yana Yang, Changchun Hua, and Xinping Guan. "Adaptive fuzzy finite-time coordination control for networked nonlinear bilateral teleoperation system". In: *IEEE Transactions on Fuzzy Systems* 22.3 (2013), pp. 631–641.