UNIVERSITY OF BLIDA 1

Faculty of Technologies

Automatic and Electrotechnical Department

THESIS OF DOCTORAT LMD

Specialty : Automatic

EVALUATION AND IMPLEMENTATION OF NONLINEAR CONTROL TECHNIQUES ON UNMANNED AERIAL VEHICLES

By

Dounia MERADI

Defended in public ahead of a jury composed of

M'hamed Bounekhla	Professor	University of Blida 1	President
Mohamed Tadjine	Professor	ENP Algiers	Examiner
Djilali Kouchih	MCA	University of Blida 1	Examiner
Zoubir Abdeslem Benselama	Professor	University of Blida 1	Supervisor
Ramdane Hedjar	Professor	University of King Saud	Co-Supervisor

Blida, 2023

ABSTRACT

This thesis focuses on modeling, designing, and controlling a micro unmanned aerial vehicle controlled by four rotors (quadrotors). It presents a contribution to the development and implementation of nonlinear control strategies to solve the trajectory tracking problem of quadrotors. Consequently, two nonlinear control schemes are proposed in this work. The first one developed is predictive sliding mode control (PSMC). This strategy aims to combine the advantages of sliding mode control (SMC) and nonlinear model predictive control (NMPC) to improve the tracking control performance for quadrotors in terms of optimality, input/state constraints satisfaction, and strong robustness against disturbances. Simulation results show that under wind turbulence, and time-variant uncertainties, the PSMC outperforms the other controllers by simultaneously providing disturbance rejection and guaranteeing that the control inputs are within bounded constraints. Currently, most quadrotor studies use classical approaches for an attitude like Euler-angles. These angles are intuitively understandable but raise problems such as gimbal locks, singularities, discontinuities, and highly nonlinear equations. The second control scheme developed in this work is based on NMPC with quaternion attitude parametrization delivered for tracking trajectory. The quaternion approach is used instead of Euler angles to overcome the problems of this latter. This proposed work is illustrated through a numerical simulation with obstacle avoidance. Finally, we will make our quadrotor. We begin by presenting its different components, then we identify the necessary parameters for the control methods implementation.

Keywords: *Unmanned Aerial Vehicle, Quadrotor, Nonlinear Control, Robust Control, Optimal Control, Predictive Sliding Mode Control, Quaternion.*

RÉSUMÉ

Cette thèse s'articule autour de la modélisation, la conception et le contrôle d'un micro-véhicule aérien sans pilote, contrôlé par quatre rotors (quadrirotors). Elle présente une contribution au développement et à l'implémentation des stratégies de commande non linéaires pour résoudre le problème de suivi de trajectoire des quadrirotors. Pour cela, deux stratégies de commandes non-linéaires sont proposés dans ce travail. La première étant la commande prédictive par modes glissants (PSMC). Cette stratégie vise à combiner les avantages de la commande par mode glissant (SMC) et la commande prédictive par modèle non-linéaire (NMPC) pour améliorer les performances de suivi de trajectoire des quadrirotors en termes d'optimalité, de satisfaction des contraintes d'entrées/états et de forte robustesse contre les perturbations. Les résultats de la simulation montrent que sous les rafales du vent et les incertitudes variant dans le temps, le PSMC surpasse les autres commandes en fournissant simultanément un rejet des perturbations et en garantissant que les entrées de commande sont dans des limites prédéfinis. Actuellement, la plupart des études sur les quadrirotors utilisent des approches classiques pour décrire l'attitude comme les angles d'Euler. Ces angles sont intuitivement compréhensibles mais posent des problèmes tels que les blocages de cardan, des singularités, des discontinuités et des équations hautement non linéaires. La deuxième stratégie de commande qui est développée dans ce travail est basée sur NMPC avec paramétrisation d'attitude quaternions délivrée pour le suivi de trajectoire. L'approche basée sur les quaternions est utilisée à la place des angles d'Euler pour pallier aux problèmes de ces derniers. Ce travail proposé est illustré par une simulation numérique avec évitement d'obstacles. Enfin, nous fabriquerons notre propre quadrirotor en commençant par présenter ses différents composants, puis nous identifions les paramètres nécessaires à sa mise en place pour l'implémentation des méthodes de commande.

Mots clés: *Drone, Quadrirotor, Commande Non-linéaire, Commande robuste, Commande Optimale, Commande Prédictive par Modes Glissants, Quaternious.*

الملخص

تركز هذه الأطروحة على نمذجة وتصميم والتحكم في طائرة دون طيار صغيرة يتم التحكم فيها بواسطة أربعة محركات (كوادكوبتر). زيادة على ذلك، فإنها تقدم مساهمة في تطوير وتنفيذ استراتيجيات التحكم غير الخطية لحل مشكلة تتبع المسار لطائرة رباعية المراوح. وبالتالي، في هذا العمل تم اقتراح استراتيجيتين غير خطيتين للمشكلة تتبع المسار. الاستراتيجية الأولى تتمثل في التحكم الاستشرافي مع نمط الانزلاق (PSMC). تهدف هذه الاستراتيجية إلى الجمع بين مزايا التحكم في نمط الانزلاق (SMC) والتحكم الاستشرافي غير الخطي (NMPC) لتحسين أداء التحكم في التتبع للرباعية من حيث الأمثلية ، ورضا قيود المدخلات / الحالات، والمتانة القوية ضد الاضطرابات. تظهر نتائج المحاكاة أنه في ظل كل من اضطراب الرياح، وعدم الدقة المتغيرة بمرور الزمن، يتفوق PSMC على وحدات التحكم الأخرى من خلال توفير في نفس الوقت رفض الاضطراب وضمان أن تكون مدخلات التحكم ضمن قيود محدودة. حاليًا، تستخدم معظم نمذجة المروحيات الرباعية الأساليب الكلاسيكية لتحديد معالم الموقف، مثل زوايا أويلر. هذه الزوايا مفهومة بشكل حدسي ولكنها تنشأ مشاكل مثل حَبْسُ اَلْمِحْوَرَانِيَة، والتفردات، والانقطاعات، والمعادلات غير الخطية للغاية. تعتمد استراتيجية التحكم الثانية التي تم تطويرها في هذا العمل على NMPC لتتبع المسار مع تحديد معلمات الموقف عن طريق كواتيرنيون. يتم استخدام الكواتيرنيون بدلاً من زوايا أويلر للتغلب على مشاكل هذا الأخير. تم توضيح هذا العمل المقترح من خلال محاكاة عددية مع تجنب العوائق. وفي الأخير اشتمل عملنا على تحقيق نموذج وتطبيق قانون تحكم للطائرة، وهذا بعد شرح مختلف المركبات، وكشف عن مختلف المعدات اللازمة.

الكلمات المفتاحية: طائرة دون طيار، رباعي المحركات، تحكم غير خطي، تحكم متين، تحكم أمثل، تحكم استشرافي مع نمط الانزلاق، الكواتيرنيون.

ACKNOWLEDGMENTS

First and foremost, I would like to thank God for the whole thing. This thesis is a result of six long years of hard work, and with the help of people who did not give up. During these years I learned a lot in my field and that is what leads me to make contributions, to continue research so as to improve my knowledge.

I am very grateful for the support of my supervisors, who have helped me through the PhD. I would first of all like to thank my main supervisor Pr. *Benselama Zoubir Abdlslam*. Without your early encouragement I would probably not have even considered doing a PhD. I would also like to thank my co-supervisor Pr. *Ramdane Hedjar*. I have enjoyed our many interesting discussions regarding this work, and I have really appreciated your help in wrapping my head around some of the more technical aspects.

My sincere thanks also go to the members of the jury: Pr. *M'hamed Bounekhla*, Pr. *Mohamed Tadjine* and Dr. *Djilali Kouchih* for agreeing to examine and evaluate my work.

Through out this work, I have been fortunate enough to collaborate with a number of people. I would particularly like to thank *Selmane Saoud* and *Soukhal Hanane*, I have really enjoyed working together, and appreciated our many discussions. Without your help and determination, much of the experimental work on quadrotor design would not have been possible. I would also like to thank Pr. *Roland Siegwart*, head of Autonomous System Lab ASL at ETH Zurich, to host me in their Lab as an academic guest. I'm grateful to Dr. Mohamed W.Mehrez for his videos on MPC and MHE which are very useful. Also, thanks to Y. Gharbi who helped me to make a box for components.

Above ground, I am indebted to my friends, whose value to me only grows with age: *Nour El Houda Gabour*, *Ajnague Saida*, *Laiddoudi Aicha* and *Benyoucef Rayane*.

At last but not least, I would like that my dear parents, my siblings, find here the expression of my most sincere and deep thanks in recognition of their sacrifices, encouragement, their prayers, helpers and support to ensure this work in the best conditions.

Dounia Meradi

Oct,2022

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Nomenclature

Acronyms

APM	ArduPilot Mega
ARMSE	Attitude Rout Mean Square Error
BLDC	Brushless Direct Current
CEE	Control Effort Energy
CSE	Control Signal Energy
DSMC	Discrete-time Sliding Mode Control
ESC	Electronic Speed Controllers
GCS	Ground Control Station
GPIO	General Purpose Input Output
HALE	High-Altitude Long-Endurance
HTOL	Horizontal Take-Off and Landing
I2C	Inter-Integrated Circuits
IBC	Integral Backstepping Control
IMU	Inertial Measurement Unit
IPOPT	Interior Point Optimizer
LIPO	Lithium Polymer
LQG	Linear Quadratic Gaussian
LQR	Linear Quadratic Regulator
MALE	Medium-Altitude Long-Endurance
NLP	Nonlinear Programming Problem
OCP	Optimization Control Problem
PID	Proportional Integral Derivative
PRMSE	Position Rout Mean Square Error

PSMC	Predictive Sliding Mode Control	
PWM	Pulse Width Modulation	
SMC	Sliding Mode Control	
SO(3)	Special Orthogonal group	
UAVs	Unmanned aerial vehicles	
VTOL	Vertical Take-Off and Landing	

Variables

Γ_e	The motor's supplied torque	[N · m]
Γ_r	The torque due to viscous friction	[N · m]
$\mathcal{U} \subseteq \mathbb{R}^4$	the set of feasible control inputs	
$\mathcal{X}$	the set of feasible states	
ϕ, θ, ψ	roll, pitch, yaw angles (Euler angles)	
Φ_i	The multiple of the angle between the arms of the quadrotor	[rad]
τ_{gyro}	Gyroscopic torque from propeller	[N · m]
υ	linear velocity of quadrotor	[m/s]
ξ	the quadrotor's position	[m]
b	Thrust coefficient	[kg · m]
d	Drag coefficient	[kg · m^2]
F_e	The back electromotive force	[V]
f_r	Damping of the DC motor and accessories	[N · s / m]
g	the acceleration of gravity	[m/s^2]
$h_v \in \mathbb{R}^+$	The vertical distance from propeller to the ground	[m]
$i(t)$	The armature's current	[A]
J	moment of inertia matrix	[kg/m^2]
J_x	rolling moment of inertia	[kg/m^2]
J_y	pitching moment of inertia	[kg/m^2]
J_z	yawing moment of inertia	[kg/m^2]
J_r	Inertia matrix of the rotor	[kg · m^2]
K_e	DC motor constant	[Nm/A]

l	distance from center of gravity to rotors	[m]
L_m	The armature's inductance	[H]
m	masse of quadrotor	[kg]
q	quaternion	
$r \in \mathbb{R}^+$	The radius of the propeller	[m]
R_m	The motor's armature resistance	$[\Omega]$
$T_0 \in \mathbb{R}^+$	The thrust generated by the propeller without ground effect	$[-]$
$T_{GE} \in \mathbb{R}^+$	The thrust generated by the propeller with ground effect	$[-]$
u	control input	

INTRODUCTION

Background

In recent years, the military and civil fields are increasingly interested in unmanned aerial vehicles (UAVs) that have gained this attention for a variety of reasons, involving their use in precision agriculture [1], maintenance and inspection services [2, 3], and data collection and exploration operations [4, 5]. Furthermore, in more and more situations, aerial robots are involved in activities that require both physical interactions and autonomous flight capabilities, such as payload transportation [6], object grasping [7, 8], and assembly of structures [9] or building walls [10]. The most common UAVs have a rotary-wing or fixed-wing configuration. Due to their unique characteristics including maneuverability, hovering in a stationary position, and vertical take-off and landing (VTOL) ability, rotary-wing UAVs can be used for detailed inspections or surveillance of pipelines and other hard-to-reach areas.

The quadrotor has aroused particular interest in VTOL vehicles and has become a popular research platform for testing numerous control techniques. Unlike conventional helicopters and due to their small size, payload capability, simple mechanical structure, and smooth maneuverability, the quadrotor is allowed to fly in indoor or outdoor environments, as well as near obstacles. Dr. Johann Borenstein invented the first quadrotor UAV in 1992, called HoverBot [11]. It is built from four helicopters tied at their tails. It could lift itself into the air but never fly off its sensor test. Currently, several universities and companies have developed and built their quadcopter. There have been academic successes, such as the: X4-flyer (Australian National University [12]), OS4 (EPFL [13, 14]), Starmac (Standford University [15]), and Pixhawk [12]. The Dragan flyer [16], Asctec Hummingbird [17], CrazyFlie [18], Parrot ARDrone, and DJI Wookong have been introduced to the commercial market. However, commercial quadrotors are typically associated with their hard-coded software and pre-programmed plant model. Therefore, to perform complex flight controls or modify their mathematical model, it is required to modify the quadrotor's autopilot embedded code.

Akin to most highly nonlinear systems, the quadrotor UAV has several characteristics that make tracking the trajectory or stabilizing the quadrotor more challenging. These characteristics include unknown nonlinearities, strong coupling between subsystems, under-actuation, measurement noise, parametric and non-parametric uncertainty in the model, output disturbances, and system failure. Consequently, the exact modeling of the quadrotor is required in order to design a suitable flight controller. There are various ways of expressing the motion dynamics, which mainly depend on the rotation parametrization. The most common attitude parametrizations are Euler angles, axis-angle, rotation matrix [19], Rodrigues parameters [20], and unit quaternion [21]. Euler angles are widely used to present the quadrotor's orientation, it is simple, unique, and can be easily understood. It suffers however from gimbal lock phenomena.

Quadrotors are an example of underactuated systems (i.e. they have more degrees of freedom (DOF) to be controlled than the number of independently controlled actuators exerting force or torque onto the system) that bring more complexity and challenge to the designed controller. As a result, this under actuation limits the number of system configurations that can directly be controlled. Particularly, the system cannot follow unrestricted flight in full vector space due to the lack of adequate control actions in their configuration space. Hence, the dynamics model of the quadrotors is not fully linearizable. In addition, the dynamics of VTOL vehicle changes dramatically for example when it performs near-ground maneuvers due to the ground effect. This adds extra challenges to the control design and requires more sophisticated control algorithms. Thus, Techniques developed for fully actuated robots cannot be directly applied to these types of mechanical systems. Therefore, nonlinear modeling techniques and modern nonlinear control theory are generally used to achieve high-performance autonomous flight under specific flight conditions, such as hovering and landing/takeoff.

The PID [22], PD [23], LQR [24], H_∞[25], and gain scheduling [26] are the common linear controllers used to command the quadrotors due to their simplicity. However, they can guarantee closed-loop stability only around an equilibrium point. Besides, they usually fail to track aggressive maneuvers. Several non-linear controls have been developed to conquer some of the shortcomings and limitations of linear control. Among them, Fuzzy logic controller [27, 28] , adaptive SMC [29, 30], Neural Networks (NN) [31, 32], and predictive control [33].

Motivation and Objectives

This thesis presents a contribution to the development and implementation of nonlinear control strategies to solve the trajectory-tracking problem of autonomous aerial vehicles. The vehicle used is a mini-drone of the quadrotor type characterized by an underactuated mechanical system.

In addition, this work proposes and investigates the use of two nonlinear control schemes, to be used for quadrotor tracking trajectory. The first control scheme developed in this work is the predictive sliding mode control (PSMC), a hybrid of the sliding mode control (SMC) and multivariable nonlinear model predictive control (NMPC). The idea was to examine how two different control methodologies can be combined to obtain at the same time a robust and optimal control method that would give the desired performance to the closed-loop system in the presence of disturbances, uncertainties, and states/inputs constraints. Additionally, the nonlinear model predictive control based on quaternion is developed to overcome the nonlinearity and singularity of Euler-angles, with several performance enhancements.

Outline

The present thesis comprises five chapters. The contents of each chapter are briefly described as follows:

Chapter 1 introduces a brief introduction to Unmanned Aerial Vehicles (UAVs), their history, types, and uses will then be presented. We will then move to the quadrotor-type UAVs and discuss their concept, applications, and advantages. Eventually, we will highlight different quadrotor control strategies established in the literature.

Chapter 2 presents the mathematical modeling of a quadrotor UAV based on

Newton-Euler formalism in full detail containing the rotor dynamics and aerodynamic effects acting on the quadrotor body.

Chapter 3 discusses classical nonlinear cascaded control strategies to perform quadrotor trajectory tracking. In this chapter three control techniques are developed; integral backstepping controller, sliding mode controller, and nonlinear model predictive controller. Ultimately, a robustness test is performed.

Chapter 4 describes the proposed hybrid nonlinear controllers divided into two parts. The first part concerns the PSMC which is a combination of discrete-time sliding mode control and the NMPC. Different tracking objectives are defined to compare the performance of PSMC with DSMC and NMPC. The results are presented and the findings are discussed. The second part concerns the quaternion-based NMPC for quadrotor tracking trajectory and avoiding obstacles. This method consists of a proportional derivative controller to calculate the desired quaternion and NMPC for the tracking trajectory.

Chapter 5 is devoted to the realization of our quadrotor; A description of the hardware, software, libraries, and software programming language used is presented. In the end, a flight test is performed.

In the end, a summary of the thesis is given, followed by conclusions that were made based on the research findings. Finally, recommendations are given which could serve as a starting point for future work.

Chapter 1 GENERALITIES ON UAVs

1.1 Introduction

This chapter aims to provide a brief introduction to the UAV systems' architecture, classification, and applications. The development of *Unmanned Aerial Vehicles* (UAVs) or un-crewed aerial vehicles, also known as drones, has generated a great interest in automatic control over the past decades. Many areas of control and robotics have been exploited to improve this type of system's performance. These UAVs have been used in both military and civilian fields, focusing on tasks such as search and rescue, road exploration [34], security, inspection [35, 36], and aerial cinematography [37], as well as maneuvers acrobatics. UAVs are most useful for these tasks when they are performed in hazardous and inaccessible environments.

Several categories of UAVs are in various stages of research, development, and use (rotary-wing, fixed-wing, flapping-wing, blimp, hybrid UAV). Depending on their categories, endurance, and purpose, the UAVs are designed from the size of a fighter aircraft (unmanned combat aerial vehicle, UCAV), down to micro aerial vehicles (MAVs). Rotary-wing UAVs can perform vertical take-off and landing (VTOL), stationary flight, and moderate-speed flight, making them suitable for performing tasks without a runway or launchpad.

1.2 UAVs Classification

Recent years have seen a substantial effort toward developing and deploying UAVs tailored to particular purposes. This effort has led to the development and design of different UAV types with distinct performance characteristics. Therefore, the classification of UAVs is necessary for presenting how diverse UAV systems are

Table 1.1: UAV classification based on size [38].

Class	Maximum Dimension
Ultra-Small (NAV)	< 7.5 cm
Very Small (MAV)	7.5–15 cm
Small	15–200 cm
Medium	2–10 m
Large	> 10 m

and demonstrating their abilities. This section discusses the different UAV classifications that have been proposed according to various parameters, including their size, weight, range and endurance, maximum altitude, engine type, and configuration, to assist in selecting the appropriate parameters for UAVs.

1.2.1 Size-Based Classification

In terms of categorizing UAVs, their size is one of the efficient metrics. UAVs come in different sizes, each designed to serve a particular purpose, ranging from small insect-sized devices to large aircraft. Table 1.1 summarizes the different classes of UAVs based on size.

1.2.2 Endurance, Altitude, and Range

In analyzing UAV performance, endurance is considered one of the preeminent attributes, which describes the total time taken during the flight before recharging/refueling. It is directly related to the battery capacity and the amount of current produced by the motor in an electric fixed-wing airplane or quadrotor to keep the aircraft in the air. The endurance of UAVs can range from 1 to more than 40 hours, depending on the mission. UAV efficiency and endurance are affected by the mass and volume of fuel or batteries loaded. An important factor determining endurance is the type and the design of the UAVs. Due to longer ranges and higher loiter times of operations, military applications often require greater endurance than civilian ones.

Another important metric for classification is the range of the UAV, which can be calculated easily using different parameters. It determines the distance a UAV can fly away from its ground control station. It is dependent on other UAV parameters, especially its payload weight. According to their maximum endurance and altitude, UAVs can be categorized into 7 different categories

1. *High-Altitude Long-Endurance (HALE):* They can fly at altitudes over 20000 m with an endurance of several days. They are mainly used for long-range surveillance missions or reconnaissance. Today, the only Hale UAV available is well-known, the American Global Hawk, capable of flying for 33.1 hours.

2. *Medium-Altitude Long-Endurance (MALE):* They can fly between 5000-15000 m of altitude for a maximum of 24 hours. MALE UAVs are also used for surveillance missions.

3. *Medium-Range or Tactical UAV (TUAV):* They can fly between 100 and 300 km of altitude. They are smaller and operated with simpler systems than their HALE and MALE counterparts.

4. *Close Range UAVs:* They have an operation range of 100 km. They are mainly used in civil applications such as powerline inspection, crop-spraying, traffic monitoring, homeland security, etc.

5. *Mini UAV (MUAV):* They weigh about 20 kg and have an operating range of about 30 km.

6. *Micro UAV (MAV):* They have a maximum wingspan of 150 mm. They are mainly used indoors where they are required to slowly and hover.

7. *Nano Air Vehicles (NAV):* They have a small size of about 10 mm. they are mainly used in swarms for applications such as radar confusion. They are also used for short-range surveillance if equipped with an equally small camera.

1.2.3 Configuration-Based Classification

Depending on their missions, UAVs have a wide variety of configurations. However, based on their configuration, they can be categorized into four classes: fixed-wing also called horizontal take-off and landing (HTOL) UAVs, rotary-wing also called VTOL UAVs, hybrid UAVs, and unconventional UAVs.

1.2.3.1 Fixed-wing UAVs

As the name indicates, fixed-wing UAVs have fixed wings at a specific place that produce lift force when they accelerate forward. Typically, to accelerate horizontally these UAVs require a runway to take off and land. In comparison to VTOL vehicles, HTOL UAVs can carry much more payload while using less power, and their wings typically provide flight times and longer range than VTOLs; As a result, they are well-suited for missions requiring long endurance such as surveillance, mapping, and defense. Fixed-wing UAVs may not be suitable for fixed inspection applications where an aircraft may be required to hold a very precise position in order to capture still images such as the serial number of a pylon or minute damage to structures. In addition, they are more prone to damage in comparison to VTOL UAVs during landing. HTOL UAVs can be classified into four main types: tailplane-aft, tailplane forward, tail-aft on booms, and tailless.

1.2.3.2 Rotary-wing UAVs

Rotary-wing UAVs have been widely used due to their unmatched features and unique capabilities, such as VTOL ability, being stationary in the air (hovering), simplicity in design, and performing movements at a moderate speed in a three-dimensional (3D) space. VTOL UAVs configurations vary depending on their rotor number or position. Helicopters with one main rotor and a tail rotor are the most common example. There are four main categories of rotary-wing: single rotor, coaxial rotor, tandem rotor, and multi-rotors. Most common (conventional) multirotor configurations, Figure 1.1, consist of an even number of rotors arranged symmetrically in one (or more) parallel planes, such as quadrotor [39], hexarotor [40], or octorotor [41]. Conventional configurations are underactuated and strongly coupled systems due to the coupling between the horizontal translational and rotational dynamics, therefore they cannot move arbitrarily in 3D space. With proper selection of the multirotor configuration geometric arrangement, it is possible to get a fully-actuated aircraft system with decoupled position and orientation. Such configurations represent omnidirectional aerial robots [42] that can perform complex and precise movements.

While multi-rotor UAVs offer numerous benefits, they have several drawbacks.

Most notable is their short flight time; multi-rotors require a lot of power to maintain the lift. As a result, a professional multi-rotor UAV typically has a flight time between twenty and forty minutes [43]. The complexity of multi-rotor aircraft is another major disadvantage. For instance, quadcopters fall from the sky when one of the motors fails, resulting in the loss of expensive equipment. Multi-rotors are prone to self-destruction during a crash. The last point is that multi-rotor drones are noisy and more dangerous than fixed-wings. Multiple exposed props spinning at extremely high speeds can cause serious injury, while loud buzzing could disrupt sensitive environments or attract unwanted attention.

UAVs have six degrees of freedom, three for position and three for orientation. Quadrotors are always underactuated since the number of controls necessary for four propellers is less than the number of degrees of freedom. As well, other multi-rotor such as hexacopter, tricopter, and octocopter are underactuated because they have collinear propellers that generate forces aligned to one direction in the body frame.

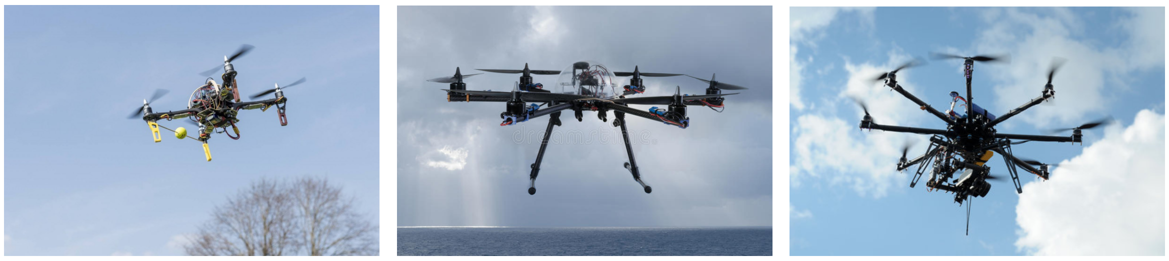
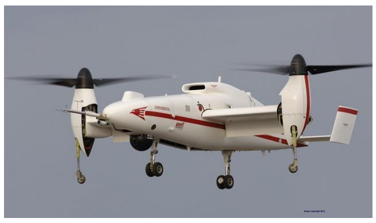

(a) Quadrotor (b) Hexarotor (c) Octarotor

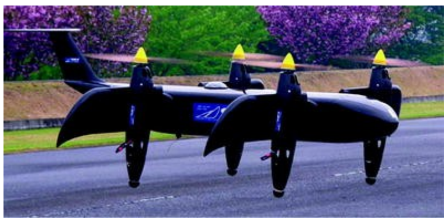
Figure 1.1: Conventional multi-rotor.

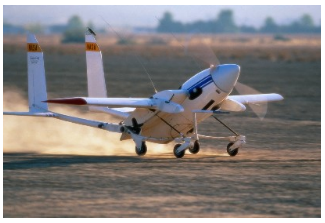
1.2.3.3 Hybrid UAVs

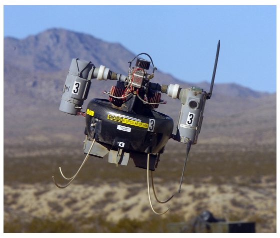
Recent research has been focusing on combining the advantages of HTOL and VTOL by developing UAVs capable of VTOL and longer endurance flights by changing their configuration into a fixed wing during the mission, which increases speed and reduces power consumption. Although hybrid designs provide salutary properties, their tilting mechanisms make them extremely complex mechanically and aerodynamically. While convertible designs have the advantages of both rotary-wing and fixed-wing UAVs [44]. Therefore, they have very high design and maintenance costs.

As shown in Figure 1.2, hybrid UAVs can be divided into five categories: tilt-rotor, tilt-wing, tilt-body, ducted fan, and tail-sitter UAVs.


(a) Tilt-rotor [45]


(b) Tilt-wing [46]


(c) Tilt-body [47]


(d) Ducted-fan [48]

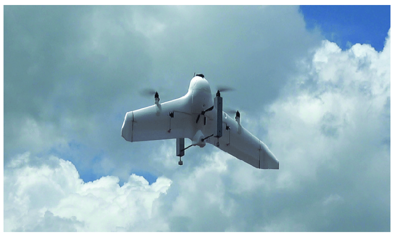

(e) Tail-sitter [49]

Figure 1.2: Hybrid UAVs.

1.2.3.4 Unconventional UAVs

In the context of UAVs classification, unconventional UAVs can be defined as those that cannot be placed in any previously mentioned categories. Usually, flying robot based on bio-inspiration belongs to this category. In Figure 1.3, several unconventional UAV types are shown. The FESTO AirJelly [50] as shown in Figure 1.3a, was inspired by jellyfish and is deemed an unconventional UAV; A central electric drive unit and an intelligent adaptive mechanism allow this drone to glide through the air. A helium-filled ballonet enables this drone to perform this task and provides the necessary buoyancy. AirJelly is the first drone with a peristaltic drive system. As a result of this new concept, which is based on the recoil principle, the jellyfish is moved gently through the air. Controlling the flight of insects and birds is possible with live bio UAVs; For instance, Chinese researchers succeeded in implanting a chip into a pigeon's brain (Figure 1.3b), that allowed them to control its movement remotely [51]. Bio-inspired UAVs, like flapping wing UAVs, take inspiration from birds and insects. Due to their aerodynamic complexity because of the birds' and insects' biological structure, they have more complicated mechanisms than fixed-wing or rotary-wing UAVs. Using novel methods, researchers from Harvard University constructed a bee-shaped flapping wing UAV that weighs 380 mg [52].

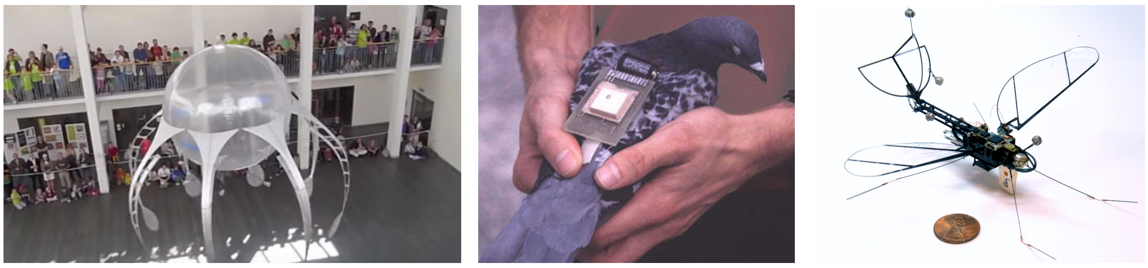

(a) FESTO AirJelly (b) Live bio UAV (c) 3.2 g flapping-wing

Figure 1.3: Unconventional UAVs.

1.3 Quadrotors

The Quadcopter is a UAV with four identical rotors. The quadcopter changes the rotational speed of these rotors to control the thrust and torque which achieve the desired motion. The main advantage of quadcopters over conventional helicopters is that quadcopter propellers are relatively smaller than helicopter propellers. This enables the quadcopter to fly with less risk and more stability in challenging environments.

1.3.1 Applications

In civilian or military fields, quadrotors have many uses and applications due to their maneuverability, small size, and simplicity of control. By integrating sensors and cameras, quadrotors can now perform difficult and risky missions without human intervention. These uses include, but are not limited to:

1.3.1.1 Inspection and monitoring

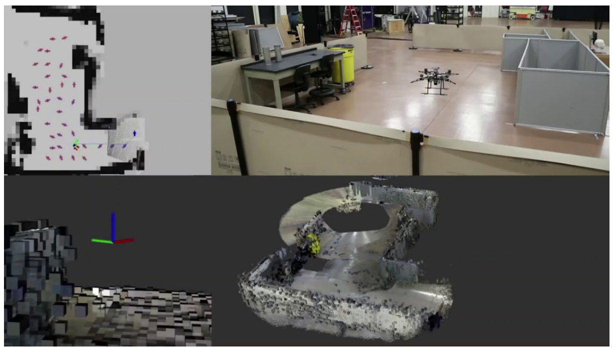
For the time being, UAVs play a critical role in traffic monitoring as they continuously gather data about traffic and road conditions and send it to a monitoring center in real-time. UAVs offer a several of benefits over traditional road monitoring (radar sensors, fixed surveillance video cameras), including flexibility, a wide range of coverage instead of a fixed one, fast time response, and a high level of accuracy to detect incidents. Abdullah et *al.* used a quadrotor integrated with a camera in their project to provide real-time aerial [53] video for traffic and emergency management in Malaysia. In this study, the quadrotor provided useful information for ground staff to determine congestion levels and monitor drivers for violations.

1.3.1.2 Farming

For agriculture and forestry, a quadrotor is a fascinating platform since it can provide accurate geographical information about the condition of the soil and drought monitoring. In addition, it can use for fertilizer and pesticide spraying, Figure 1.4b. In [54], the quadrotor equipped with a camera has been used to estimate the *Normalized Difference Vegetation Index* (NDVI) used to assess the health of crops. An automated survey of agriculture fields was performed according to this metric, allowing an estimate of the crop's conditions instead of actual observations and care by local farmers.

1.3.1.3 Search and Rescue (SAR) Missions

As the effects of climate change become more severe, natural disasters are becoming more common. Consequently, search and rescue operations will become more crucial for societies worldwide. Often, rescue services are involved in missions in rural areas, treating injured victims or searching for missing persons. Inspecting a destroyed building after an earthquake or during a fire is exactly the kind of job that human rescuers would like drones to do for them. A flying robot could look for people trapped inside and guide the rescue team toward them. But the drone would often have to enter the building through a crack in a wall, a partially open window, or through bars, something the typical size of a drone does not allow. Researchers from the Robotics and Perception Group at the University of Zurich and the Laboratory of Intelligent Systems at EPFL created a new morphing quadrotor [55] inspired by birds that fold their wings in mid-air to cross narrow passages. This new drone can squeeze itself to pass through gaps and then go back to its previous shape while continuing to fly Figure 1.4c. And it can even hold and transport objects along the way.


(a) Autonomous Quadrotor 3D Mapping and Exploration

(b) Pesticide spraying by quadrotor.

(c) An example of foldable quadrotors that could aid search and rescue in wrecked building.

Figure 1.4: Examples of applications completed by a quadrotor UAV.

1.3.2 Advantages

The simplicity and robustness of quadrotors' mechanical design make them more popular than other types of small aerial vehicles. Typically, quadrotors are constructed from a rigid cross-shaped frame that can be designed as lightweight while exhibiting a high impact resistance. Compared to other multi-rotors, where the rotors are parallel, quadrotors use a minimal number of motors to control the degrees of freedom. As a result, they are the most uncomplicated configuration of these multi-rotors.

1.3.3 Survey of Quadrotor Control Algorithms

Similarly to most highly nonlinear systems, quadrotor dynamics have several characteristics that make designing trajectory tracking or stabilization difficult. Those characteristics include unknown nonlinearities, under actuation, strong coupling between subsystems, parametric and non-parametric uncertainties in the model, measurement noise, output disturbances, and system failure. Consequently, such issues have become more attractive and challenging for researchers to solve by designing new controller approaches and methods.

The quadrotor is an example of an underactuated system with six degrees of freedom (DoF) to be controlled and only four independent control inputs; Therefore, this under actuation limits the number of directly controllable system configurations. In particular, the system cannot follow the unconstrained flight in full vector space due to a lack of sufficient control actions in their configuration space. Besides that, a quadrotor can only generate a thrust along its vertical body-fixed axis, so for horizontal motion within the plane, a specific orientation is required. This is an essential property of most control strategies proposed for tracking. As a result, the existing control frameworks are usually based on a cascade feedback strategy that involves the following steps. First, the system is composed of two subsystems, namely the inner loop and outer loop, which determine the attitude and position dynamics of UAVs, respectively. Then, feedback control is designed for each loop separately. An inner loop is typically controlled at a higher frequency than an outer loop, around five to ten times faster [56]. As a final step, the loops are joined by using the outer loop feedback signal to provide a reference for the inner loop. A block diagram of the cascade feedback strategy is given in Figure 1.5.

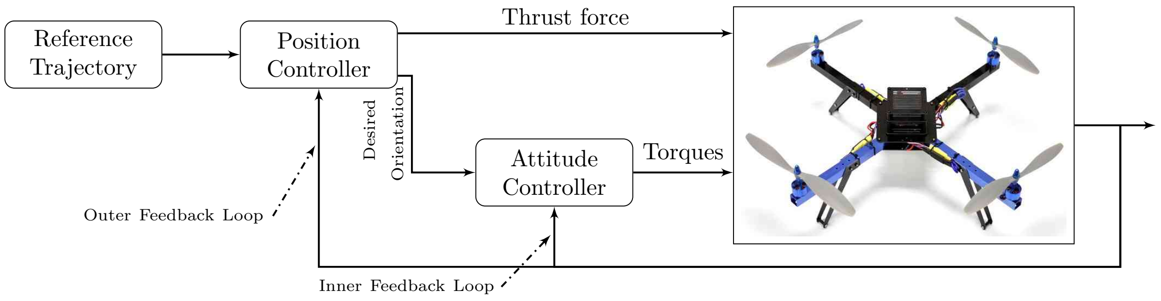


Figure 1.5: Cascade feedback strategy.

1.3.3.1 Linear Control Approaches

The development of quadrotors began with linear controllers, which proved sufficient for stabilizing flight. Our study examines several of these control techniques, including a *Proportional Integral Derivative* (PID) controller, Linear Quadratic controller, H_∞ controller, and gain scheduling controller.

1) Proportional Integral Derivative Controller: Due to its simplicity, the PID controller (Figure 1.6), is one of the most popular controllers. In control theory, the PID controller is considered as a classical and is frequently used for a wide range of electrical and mechanical systems. A PID controller has considerable advantages over more complicated formulations, which makes it well-suited to a wide variety of applications. Among the main advantages of PID controllers include: their ease of implementation and can be applied without knowing the dynamical model, the ease of tuning the gains (parameters), and the reliability and consistency of their algorithms. However, PID controllers cannot directly implement in quadrotors because they are nonlinear under-actuated systems; Nevertheless, many researchers have adopted the PID controller. As a result, the PID controller is now widely used for many commercial quadrotors.

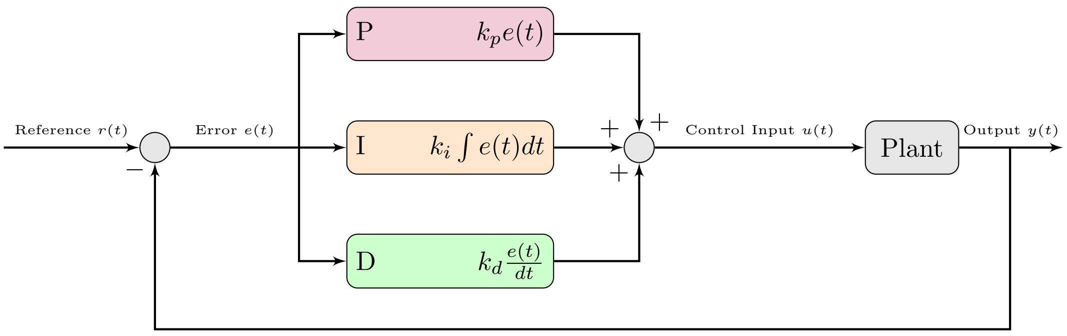


Figure 1.6: Block diagram of PID controller.

S. Bouabdallah et *al.* developed the PID controller for a quadrotor with full autonomy [57]; Using the Euler-Lagrange formulation, they derived the quadrotor's dynamic system model that includes the gyroscopic effects. The PID controller obtained a positive perspective in experimental results by successfully controlling orientation angles under minor perturbation. In [58], the authors addressed three separate aerodynamic effects arising when the flight regime deviates from hover; namely angle of attack, velocity, and airframe design. In their experimental test, they found that the PD controller was sufficient for pitch manoeuvers; However, blade flapping required additional measures as speed increased. Three different types of PID controllers updated are used for the quadrotor's attitude control [59]. The gyroscope directly measures the angular rates, which are stabilized by the inner loop. The outer loop stabilizes Euler angles that are derived by combining and filtering data from gyroscope and accelerometer measurements.

2) *Linear Quadratic Controller:* This section provides an overview of linear quadratic controllers for quadrotors, including *Linear Quadratic Regulator* (LQR) and *Linear Quadratic Gaussian* (LQG) controller. First, using weighting factors provided by the user, the system is optimized by LQR based on the cost function and minimum cost. In the second, the LQG controller is obtained by combining the optimal LQR feedback with the optimal Kalman filter, Figure 1.7. In what follows, we discuss several implementations of these strategies for quadrotors.

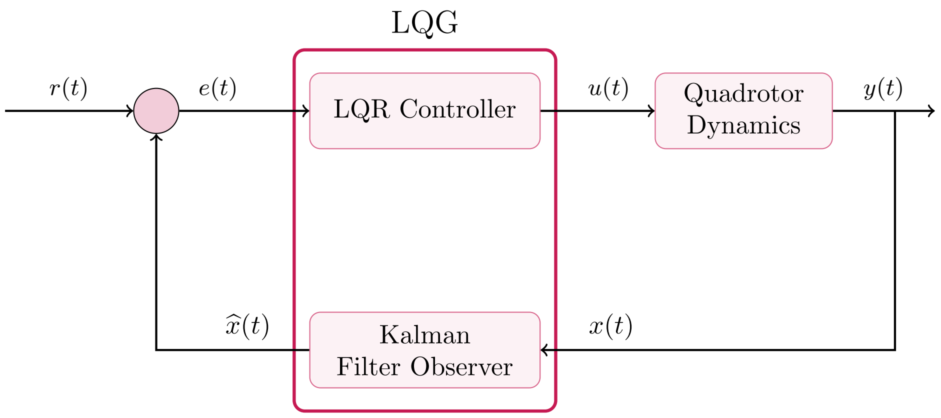


Figure 1.7: Block diagram of LQG controller.

The LQR technique is used in [57] compared to PID, but the performance of the experimentally tested controller was not satisfactory. The steady-state error remained and oscillations appeared. The authors explain it by the model imperfections due to the non-consideration of the motors' dynamics. The work cited in [60] then considered the complete dynamic model to make the drone follow a predefined vertical trajectory. The controller design is made based on the linearized model. The obtained controller is then implemented on the nonlinear model for simulations. The simulation results demonstrated the effectiveness of the controller in a perfect environment and the presence of disturbances due to a wind of 0.1 m/sec along the x-axis; this value remains low compared to a wind speed outdoors on an ordinary day. The same control technique is experimented with in [61], the objective of which is to make a Quanser Qball-X4 quadrotor follow a horizontal trajectory fixed beforehand; the LQR controller is compared to a PID controller, and the results of the simulation and real-time flight show better performance of the LQR controller. The proposed guidance law in [62] employs the LQR method to optimize the energy consumption, where the dynamic system equations are linearized at different operation points. Simulation results demonstrates that the proposed control effectively minimizes both the tracking error and energy consumption. Security of the closed-loop control quadrotor system from false data injection attacks has been addressed in [63]. As the first step, a framework was proposed to detect attacks and augment the information for controller design; An LQR controller was then designed based on the framework and the extended information and successfully implemented in a real platform.

3) *H_∞ Controller:* The H_∞ control algorithm is a linear robust method for developing a control system that can handle external disturbances and parametric uncertainties. Therefore, it is a good choice for quadrotors' linear control, where they often are influenced by model uncertainties and wind gusts. Two full linear controllers based on optimal control theory have been presented in [64]. The first technique is based on the L_2 norm which is a linear quadratic servo. The second one optimizes the L_∞ norm and is designed using the H_∞ control approach. The work cited in [65] compares the H_∞ method to a control method based on three different loops, one of which is responsible for compensating for parametric uncertainties and disturbances, simulations and experimentation that only concerned hovering showed that the $H_\infty-$based linear controller can reject disturbances, but remains limited in reducing nonlinear effects when implemented on the real system. The same observation is noted in [64], where the controller loses its altitude trajectory tracking capability when implemented on the complete nonlinear system.

1.3.3.2 NonLinear Control Approaches

To overcome some of the shortcomings of the linear controllers, such as guaranteeing stability only around an equilibrium point and the inability to handle the nonlinear part of the model, therefore, several nonlinear controllers have been derived based on the nonlinear dynamics of the system. An overview of some nonlinear controllers is presented in this subsection.

1) *Sliding Mode Controller:* The SMC has been significantly used to control the quad-rotor. Because of its attractive finite-time convergence characteristics and robustness to parametric uncertainties and perturbations. Since the SMC suffers from the chattering phenomena caused by the reaching law and has high control effort, many researchers have been working on those troubles. One of the proposed solutions is the integral sliding mode control [66]. The integral action added to the sliding manifold has the ability to eliminate the reaching phase and reduce the chattering on the control inputs. Irfan *et al.* [67] applied an *Improved Integral Power Rate Exponential Reaching Law* (IIPRERL) Sliding Mode Control strategy to deal with the unwanted chattering problem, stability, and the oscillations in the quadrotor responses in the presence of matched disturbances. The simulation results of IIPRERL-SMC have shown no chattering on the control inputs compared to SMC. In discrete-time, the authors in [68] have proposed the DSMC for quadrotor where the linear extrapolation method has been employed to obtain the discrete-time model of the quadrotor.

2) *Backstepping Controller:* The backstepping control technique was developed by Kanellakopoulos et al. [69]. This technique is among the nonlinear control methods based on the Lyapunov theory for the stability study. The application of this theory is often inhibited by difficulties in finding the appropriate Lyapunov function. However, the recursive look of the backstepping command provides a systematic algorithm that makes this task easier. It consists of organizing the main system into cascaded subsystems, for which virtual control laws are designed in several steps in descending order until the control law is obtained which stabilizes the overall system. Many researchers have developed backstepping controllers for quadrotors. In [70], this technique is developed only for attitude subsystem control. In [71], the control is split in two loops, where the altitude control is done in an external loop and the rotation control in an internal loop. While a full control model using this method is applied in [72] through three interconnected subsystems: an under-actuated subsystem connecting the positions x, y, roll, and pitch, a fully actuated subsystem for the z position, and the yaw angle, and the last subsystem representing the dynamics of the propeller forces. In [73], this command has been implemented with integral action to eliminate steady-state error while ensuring asymptotic stability [69] of the closed-loop system.

3) *Feedback Linearization:* A feedback linearization method is a common approach in nonlinear control. Using this method, the nonlinear system dynamics are algebraically transformed into an equivalent fully or partially linearized system; A nonsingular matrix was then generated using similarity transformations based on the linear system. As a result, the state variables of the nonlinear system can be converted into the linear system using this form of diffeomorphism. Then, a linear control theory can be used to design the controller, and afterward, the solution from the linearized system is transformed back into the nonlinear system [74]. This method requires that the model be perfectly known, which is not easily assured in practice. In contrast, if the system has a relative degree lower than the system order, it becomes necessary to study the stability of the internal dynamics, namely, the zeros dynamics; If this dynamic is unstable, the linearizing control will not be able to guarantee the overall system stability.

Since quadrotor dynamics have nonlinearities, this approach is appropriate to eliminate the nonlinearities and design a flight controller using linear control theory. In [75], the system is transformed into SISO subsystems to which the input-output linearization is applied; the simulations validated the stability and robustness of the control law in the presence of wind (wind force of 10 N) and d parametric uncertainties (20% uncertainties on the moments of inertia). A different approach is adopted in [76], where the assumption of small angles is assumed to simplify the model of the system. Subsequently, only the x, y, z outputs are considered to determine the control inputs by input-output linearization. As for the angle ψ, it is controlled by a PD controller. In [77], Holger Voos presented a control strategy including feedback linearization to cope with the nonlinear dynamic behavior of Micro-UAV quadrotor. Feedback linearization is proposed to be fault-tolerant for quadrotors [78]; According to it, the strategy to deal with a propeller failure for a quadrotor is to give up controlling its yaw angle and spin the remaining propellers horizontally instead.

4) Model Predictive Control: In the realm of optimal control for quadrotors, both the linear and non-linear model predictive control MPC has been widely used, showing a good tracking ability, handling to input/state constraints [79], and avoiding obstacles [80]. In [81], a nonlinear and linear MPC has been presented for a quadrotor to track different reference trajectories where the NMPC has been made by using a state-dependent coefficient representation. Moreover, stability analysis of Unconstrained/constrained for both controllers has been provided. The simulation results in the case of with or without disturbances showed that the NMPC outperformed the linear MPC. Since the MPC depends explicitly on accurate model-plant as well as that the quadrotor is a strongly constrained coupled non-linear system which is usually prone to parameters variation on mass and inertia due to payload. For that, any mismatched parameters or disturbances can decrease the stability of the system when using the conventional MPC approaches.

5) *Adaptive Control:* The adaptive controller provides a mechanism of parametric adjustability to control a system. The structure of this nonlinear controller typically has two loops, one for the normal feedback and the other for adjusting parameters [82]. Most of the time, adaptive control is widely used for quadrotors and is combined with other types of controllers like SMC [83, 84], backstepping [85] and fuzzy control [86].

1.3.3.3 Data-based Controller

Intelligent control algorithms aim to control a system by integrating various artificial intelligence techniques, some of which are biologically inspired. In contrast to other control strategies, intelligent controls have the advantage of covering a wide range of uncertainty. These controllers are discussed in the following subsections

1) *Fuzzy Logic-based Controller:* As a form of artificial intelligence, *Fuzzy Logic Control* (FLC) is generally used for developing and designing an intelligent control system. Essentially, fuzzy control is the process of using human thinking or linguistic control strategy based on expert knowledge that is converted into fuzzy rules. FLC inputs must undergo three basic steps before generating outputs namely fuzzification, decision-making stage, and defuzzification, Figure 1.8. There exist two major types of fuzzy systems, the Mamdami fuzzy system, and the Takagi-Sugeno (TS) fuzzy system. In Mamdani fuzzy system the output of the control rules bloc, Figure 1.8, is a linguistic variable however, we need a defuzzifier phase which is achieved by the defuzzification bloc. Contrary to the Mamdani fuzzy system the output of the control rules bloc is a numerical variable for the TS fuzzy system; so in this case, the control rules bloc is absent. A trajectory tracking controller based on

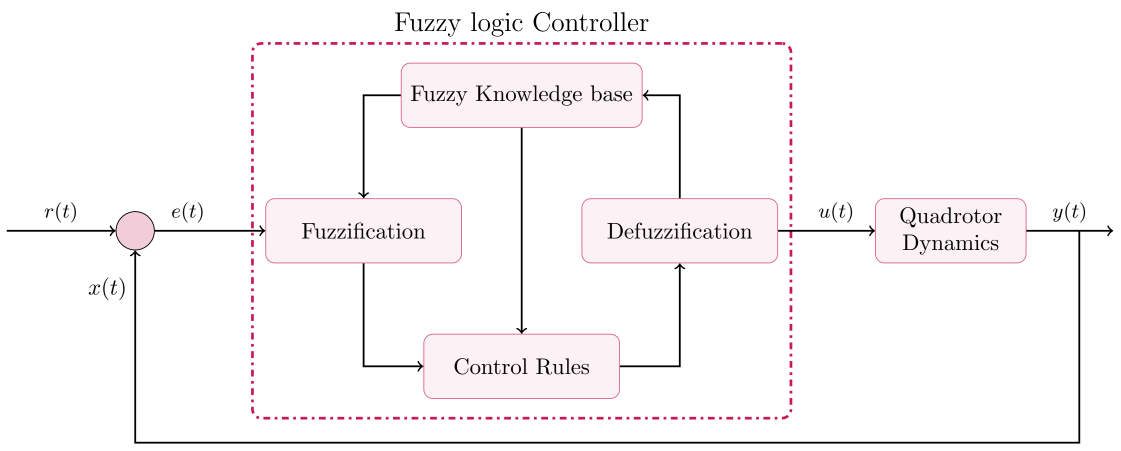


Figure 1.8: Block diagram of Fuzzy logic controller.

TS fuzzy modeling was proposed in [87]. The authors developed a TS fuzzy for a complex and higher-dimensional problem and reduced the number of rules required by the Mamdani model. As compared to the LQR controller, they demonstrated in simulation that the proposed TS fuzzy achieved a good tracking performance.

2) *Artificial Neural Network (ANN)-based Controller:* The use of artificial neural networks (ANNs) in control applications has increased considerably over the past few years. It has the ability to model non-linear systems that can be the most facilely exploited in the synthesis of non-linear controllers. ANN direct inverse control [88] has been implemented on a quadrotor for tracking trajectory to learn online the complete dynamics of the quadrotor including uncertain nonlinear terms, and to overcome the limitation of PID tuning. Better achievement in the altitude dynamics was attained by this approach compared to the PID control method.

3) *Reinforcement Learning (RL)-based Controller:* *Reinforcement learning* (RL), also known as neuro-dynamic programming or approximate dynamic programming, is a field of research developed by the *Artificial Intelligence* (AI) community for achieving optimal sequential decision-making under system and environment uncertainty. The roots of RL can be traced back to the 60's and a thorough overview of its evolution can be found in [89, 90]. Contrary to optimal control theory, RL is based on evaluative, rather than instructive, feedback and comes in different forms, which may or may not include partial knowledge of the environment or the system. The process typically involves hand-engineering a reward function, which assigns a reward, or penalty, to the actions that induce desired, or undesired, outcomes, respectively. An RL algorithm is then assigned to find a policy (or controller, in control engineering terminology) that solves the control objective optimally, given the problem constraints and uncertainties. To sum up, RL algorithms use the reward function as a guide, and through trial and error, learn to model the system and its environment, which then leads to a policy that provides an optimal solution to the assigned problem.

Hwangbo et al. [91] applied RL for quadcopter control, particularly navigation control. They developed a novel deterministic on-policy learning algorithm that outperformed trust region policy optimization. In experiments, they demonstrated that quadcopter could stabilize in the air even under very harsh initialization by using this policy.

1.3.3.4 Hybrid Control Algorithms

The best single linear or nonlinear controllers had some limitations in controlling quadrotors. Many researchers have tackled this problem by amalgamating the advantages of two or more algorithms. The integral backstepping combined with sliding mode control for quadrotors under external disturbances has been proposed in [92]; This hybrid method has improved the tracking accuracy, avoided the discontinuous control signal, and increased the robustness of the controller. In [93], the control structure conceives of a linear MPC for position tracking and a nonlinear H_∞ for attitude stabilization. Predictive control combined with fuzz logic control has been used in [94]. The control strategy is divided into 2-part; the first part consists of linear MPC to control Euler angles, and the NMPC to control the height of the quadrotor; the second part consists of utilizing a 2-level fuzzy logic controller to assist the first controller when the error exceeds a predefined value. In [95], a hybrid robust adaptive flight algorithm has been designed and developed for the quadrotor UAV to address the issues caused by the slow-varying mass. This control method involves the design of an observer to estimate the real-time mass. For the attitude controller, they added a hyperbolic tangent function to PI control to eliminate external disturbances. The backstepping and hybrid robust adaptive controllers have been combined to design the position controller.

1.4 Conclusion

The use of UAVs has become widespread in many fields, including military and civil fields. A brief state-of-the-art classifications of UAVs was presented in this chapter. The quadrotor is a micro unmanned aerial vehicle controlled by four rotors. This type of vehicle was chosen to extend the number of civilian applications of air vehicles such as crowd detection, fire detection, object transport in inaccessible areas, and extensive farmland exploration. A background of nonlinear controllers was presented in the final section of this chapter.

Chapter 2 QUADROTOR MODELING

2.1 Introduction

The dynamics of VTOL vehicles are particularly complex, as the nonlinearities and aerodynamic interactions are numerous. It then becomes arduous to build a complete mathematical model including all the undesirable effects; It would be necessary to characterize all these effects for each flight configuration, for this reason, a simplified dynamic model is generally favored. The main challenge in developing this model is that it must be sufficiently rich to simulate the drone's behavior with precision while also being simple to allow the analysis and synthesis of control laws.

In this chapter, the equations governing the quadrotor's dynamics are addressed. First, the different possible movements done by the quadrotor are described. Then, the rigid-body dynamics are derived to describe the forces and moments generated by actuators. Also, the discrete-time model using integration by the fourth order of the Runge-Kutta method is presented. In subsequent chapters, this representation is used to synthesize the controller laws.

2.2 General Description of the Quadrotor

The quadrotor is a VTOL vehicle able to perform quasi-stationary flights. It consists of four fixed-pitch blades coupled with DC brush-less motors fixed to the end of a rigid cross-shaped body as shown in Figure 2.1. Indeed, each propeller is rotating at a certain angular speed ω_i generates a force F_i and a moment M_i.

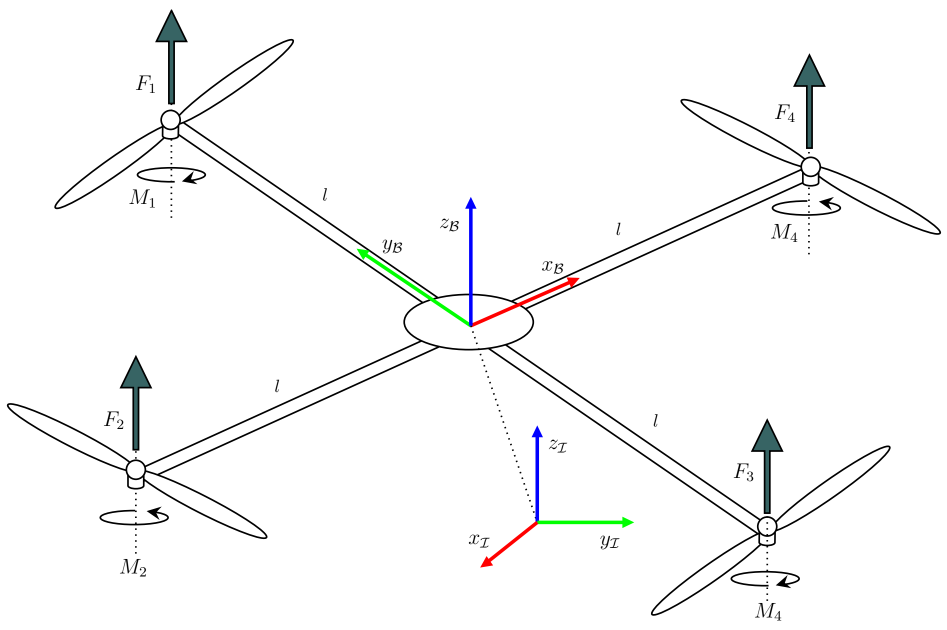


Figure 2.1: Propeller forces and torques acting on a quadrotor, and the coordinate frame system.

2.2.1 Forces and Moments Acting on Quadrotors

The quadrotor is affected by various forces, namely the lift, thrust, drag, and weight forces. This latter is due to the quadrotor's mass and always acts in the direction of gravity. The lift force occurs when pressure differences across the quadrotor (in the vertical direction). Thus, the blade size, shape, and speed determine how much lift force the propeller will generate. Next, drag is the force acting on the quadrotor in the opposite direction of motion due to air resistance. It can occur due to pressure differences and the viscosity of air. The quadrotor's aerodynamic shape is selected to reduce drag. Each propeller is rotating at a certain angular speed ω_i generates a force F_i directed upwards and the counteracting torque directed opposite to the direction of the particular rotation. Propellers with the angular speed ω_1 and ω_3 [rad/s] spin counterclockwise, while the other two spin clockwise. The thrust force is proportional to the square of the angular speed and is given by [56]

$$F_i := C_T \rho A_i r_i^2 \omega_i^2, \tag{2.1}$$

where, C_T is the non-dimensionalized thrust coefficient depending on the rotor profile and geometry, $A_i = \pi r_i^2$ is the disk area generated by the propeller, r_i is the radius, and ρ is the air density. The torque developed on the airframe is determined by the drag force. It is proportional to the angular velocity and is given by [56]

$$M_i := C_D \rho A_i r_i^3 \omega_i^2, \tag{2.2}$$

where, C_D is the non-dimensionalized drag coefficient

Typically, aerodynamic parameters derived from (2.1) and (2.2) are considered constants $b \approxeq C_T \rho A_i r^2$, $d \approxeq C_D \rho A_i r^3$ and can be determined by testbed calculations [96].

According to the quadrotor symmetrical configuration, the total torque and thrust force produced on the vehicle by the propellers are as follows

$$F_{th} = \begin{pmatrix} 0 \\ 0 \\ \sum\limits_{i=1}^{4} b\omega_i^2 \end{pmatrix} \tag{2.3}$$

$$\tau = \begin{pmatrix} l \sum\limits_{i=1}^{4} F_i \sin(\Phi_i) F_i \\ -l \sum\limits_{i=1}^{4} F_i \cos(\Phi_i) F_i \\ \sum\limits_{i=1}^{4} (-1)^i M_i \end{pmatrix} = \begin{pmatrix} bl(\omega_4^2 - \omega_2^2) \\ bl(\omega_3^2 - \omega_1^2) \\ d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{pmatrix} \tag{2.4}$$

2.2.2 Unmodeled Forces and Moments

Quadrotor models are subject to several aerodynamic and gyroscopic effects. Despite being essential for the design of a complete system, most of these effects cause only minor disturbances without taking them into account. However, propeller flutter and induced drag are fundamental effects that contribute significantly to understanding the natural stability of quadrotors. These effects are particularly essential because they cause forces in the x-y plane of the quadrotor, and its under-actuated directions, which cannot be easily dominated by high gain control.

2.2.2.1 Air friction

The quadrotor frame and the propellers provide air resistance which generates a frictional force that opposes the linear and rotary motion of the quadrotor. This force is proportional to the square of the speed of the quadrotor and depends on the air conditions as well as the geometry of the quadrotor [97].

2.2.2.2 Blade-flapping

Propeller flapping occurs when a propeller moves horizontally. This displacement is the result of the difference in speed between the part of the propeller that attacks airflow and the part that withdraws from the airflow. Consequently, this difference in thrust causes the plane of the propeller to tilt, which changes the direction of the thrust vector.

2.2.2.3 Ground effect

A ground effect occurs when the airflow generated by the propeller is disrupted by a surface and thereby increases thrust. At low speeds, the ground effect can be modeled as follows [98]:

$$\frac{T_{GE}}{T_0} = \frac{1}{1 - \left(\frac{r}{4h_v}\right)^2} \tag{2.5}$$

Notice that the thrust increases due to the ground effect, but this effect decreases considerably even at low heights.

2.2.2.4 Gyroscopic torque

It is created in moving physical systems with rotating parts and tends to resist the movements of the quadrotor. The general expression of this torque is given by:

$$\tau_{gyro} = \sum_{i=1}^{4} \Omega \wedge J_r \begin{pmatrix} 0 \\ 0 \\ (-1)^{i+1}\omega_i \end{pmatrix} \tag{2.6}$$

where τ_{gyro} is gyroscopic torque from propeller, J_r is inertia matrix of the rotor, and $\wedge$ is the cross product.

2.2.3 Quadrotor's Movements

A quadrotor is an aerial mobile robot controlled only by four motors and defined by six degrees of freedom in space, three for rotational, and others for translational movements. Therefore, it is an underactuated system (the number of inputs is less than the number of outputs). The basic quadrotor movements are achieved by varying the speed of each rotor thereby changing the thrust produced. The quadrotor tilts toward the direction of the slower rotor, which then takes into account the translation along this axis. Consequently, the quadrotor is a coupled system, meaning that it cannot translate without doing roll or pitch rotation, so any change in speed results in movement with at least three degrees of freedom.

2.2.3.1 Vertical movement (Throttle)

This represents the ascent/descent movement of the quadrotor with all four motors running at the same speed, (Figures 2.2(a) and 2.2(b)). Therefore, we increase the latter speed for raising the quadrotor and decrease it for lowering it.

2.2.3.2 Left/right movement (Roll)

This movement is caused by (occurs) following a rotation around the X-axis of the quadrotor body frame by acting on the left and right motors, (Figures 2.2(c) and 2.2(d)). For example, to tilt the quadrotor to the right, the left motor speed is increased while decreasing that of the right motor.

2.2.3.3 Forward/backward movement (Pitch)

To obtain forward (backward) motion, the rear (front) motor speed is increased, and that of the front (rear) motor speed is reduced. The Quadcopter pitch angle will be affected by simultaneously reducing (increasing) rear (front) rotor speed, which allows a moment to be created around the Y-axis while keeping the same lift force and moments of yaw and roll remain zero (Figures 2.2(e) and 2.2(f)).

2.2.3.4 Yaw (rotation around the Z-axis)

This is a rotation around the Z-axis of the coordinate system linked to the quadrotor (Figures 2.2(g) and 2.2(h)). To rotate the quadcopter clockwise (anti-clockwise), the speed of the front and rear motors (left and right) with the clockwise (anti-clockwise) direction of rotation is reduced, and the speed of the left and right motors is increased (forward and reverse) whose direction of rotation is counterclockwise (clockwise) while keeping the sum of the speeds unchanged, which means that the yaw moment created by the latter is greater than that created by the other two without change lift, roll or pitch.

2.2.3.5 Horizontal movement

The horizontal movement cannot be obtained directly in the case of the quadrotor, that is to say, it cannot be made to move forward or sideways without action on the pitch angles or the roll, because it is an under-actuated system. However, horizontal displacement is a consequence of the same two angles. Indeed, if we maintain a given pitch angle, the normal lift force to the plane of the four rotors will have a non-zero component along the Y axis and it is this component that allows lateral movement in its direction. Likewise, for movement along the X-axis, it suffices to maintain a given roll angle. By combining these movements, the quadrotor can be moved in any desired direction.

2.3 Quadrotor's Model Dynamic

In this work, several assumptions are made when building the model:

Assumption 1 ***Rigid bodies*** *The rigidity of every component in the drone is assumed to be ideal, which means that a vector joining two points of a solid is invariant within any basis attached to this solid.*

Assumption 2 ***Motor dynamics*** *The motor dynamics are ignored. This will allow us to not consider, the equations between the motor's rotational speed and the feeding current and voltage.*

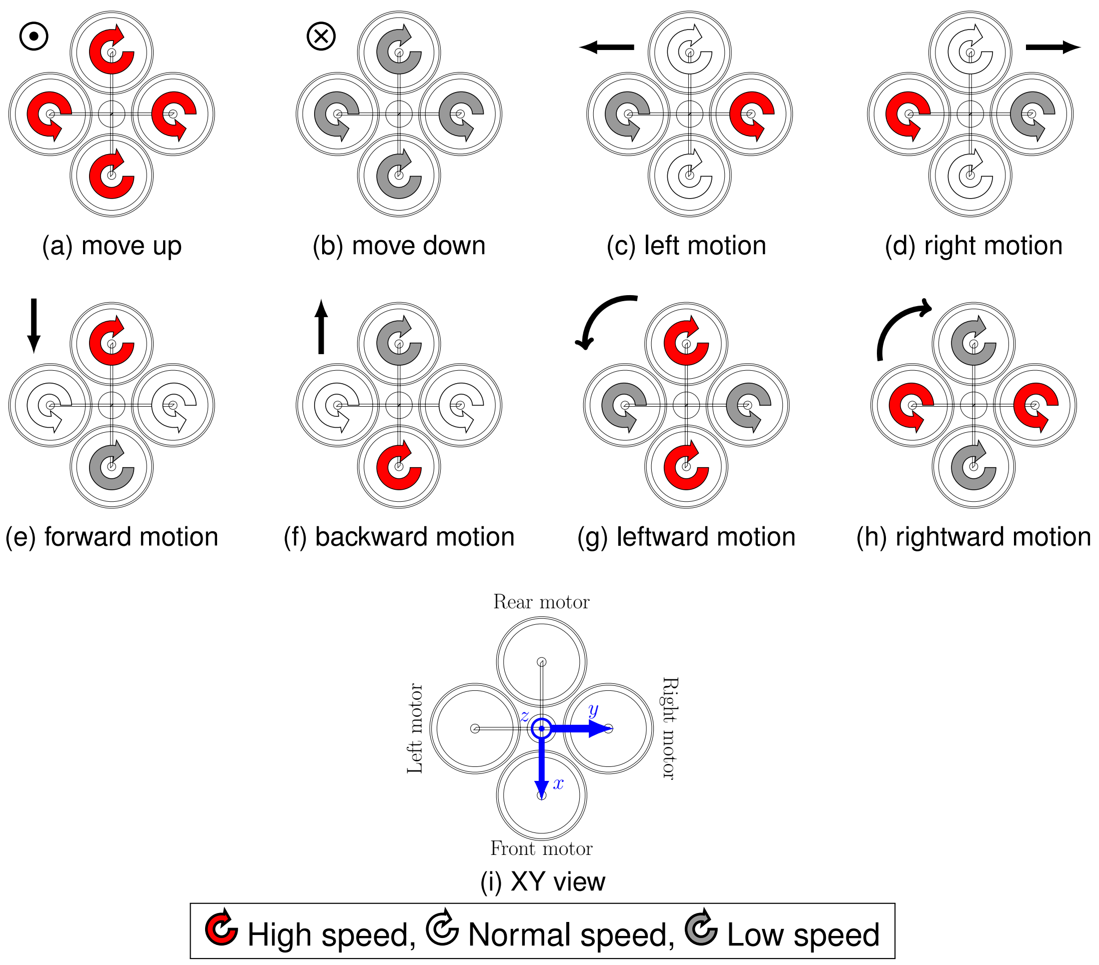


Figure 2.2: Illustration of the quadrotor's motions generated by varying its rotor angular speeds.

Assumption 3 *The center of gravity and the origin of the body-fixed frame are assumed to coincide. By doing so, the off-diagonal terms in the inertia matrix will be zero.*

Assumption 4 *The thrust and the drag are proportional to the square of the rotor's speed. This will be useful in the procedure of determining the motor's parameters.*

To describe the mathematical model of the quadrotor, we define two reference coordinate frames represented in Figure 2.1. The inertial frame $\{\mathcal{I}\}$ is defined by $\{O_I, e_x, e_y, e_z\}$, and the body-fixed frame $\{\mathcal{B}\}$ attached to the quadrotor's gravity center and is defined by $\{O_B, b_x, b_y, b_z\}$.

2.3.1 Attitude Parametrization

The most common attitude parametrizations are Euler angles, axis-angle [99], rotation matrix [100, 101], Rodrigues parameters [20] , and unit quaternion [102]. Euler angles are widely used to present the quadrotor's orientation, it is simple, unique, and can be easily understood. It suffers however from gimbal lock phenomena. There has been considerable attention paid to unit quaternion-based attitude control laws since the unit quaternion is the minimal globally singular-free description of a rigid body's attitude. Nevertheless, each attitude is made up of two distinct unit quaternions covering the rotational configuration space SO(3). In addition, even though Euler's angle representation only has three parameters, it requires the evaluation of trigonometric functions, thereby making it computationally more demanding. In [103], the trajectory linearization control TCL approach based on quaternion has been developed for a quadrotor. This approach has successfully achieved effective attitude tracking when the pitch angle is near the singularity compared to the controller based on Euler angles that failed. The uncertainty and disturbance estimator-based attitude controllers for a quadrotor have been developed using the unit quaternion in [104] to achieve global tracking of attitude while simultaneously combating coupling effects, unmodeled dynamics, and external disturbances.

2.3.1.1 Attitude parametrization: Euler angles

In this thesis, the convention Z-Y-X of the Euler angles $\{\psi, \theta, \phi\}$ has been used to describe the quadrotor's rotation. Therefore, the attitude of the quadrotor is represented by the matrix, R, in $SO(3) := \{R \in \mathbb{R}^{3\times3} | R^\top R = RR^\top = I, det(R) = 1\}$ which allows the passage from frame $\mathcal{B}$ to frame $\mathcal{I}$. This matrix is obtained by successive rotations Figure 2.3. In the beginning, the body-fixed frame coincides with the inertial frame. Then, the first rotation is a rotation around the $z_\mathcal{I}$-axis, of the yaw angle $\psi \in]\pi, \pi[$, that makes the $y_\mathcal{I}$-axis coincides with the $y^{'}$-axis and the $x_\mathcal{I}$ axis with the x' axis. The second rotation is a rotation around the y'-axis, of pitch angle $\theta \in] -\frac{\pi}{2}, \frac{\pi}{2}[$ makes the $z_\mathcal{I}$ axis coincides with the $z^{''}$-axis and the x'-axis with the x''-axis. The third rotation is a rotation around the x''-axis, of roll angle $\phi \in] -\frac{\pi}{2}, \frac{\pi}{2}[$, that makes the y''-axis coincide with the $y_\mathcal{B}$-axis and the z''-axis with the $z_\mathcal{B}$-axis.

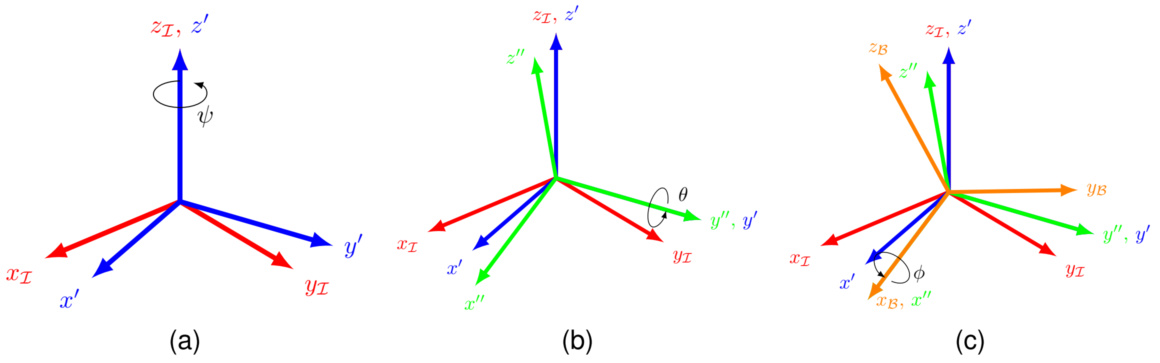


Figure 2.3: Z-Y-X Euler angles: (a) rotation around Z-axis, (b) rotation around Y-axis, (c) rotation around X-axis.

$$R_\psi = \begin{pmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{pmatrix}, R_\theta = \begin{pmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{pmatrix}, R_\phi = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi & c_\phi \end{pmatrix} \tag{2.7}$$

Finally, we obtain the total rotation matrix by multiplying the three preceding rotation matrices successively:

$$\begin{aligned} R &= R_\psi \times R_\theta \times R_\psi \\ &= \begin{pmatrix} c_\theta c_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ c_\theta s_\psi & c_\phi c_\psi + s_\phi s_\theta s_\psi & -s_\phi c_\psi + c_\phi s_\theta s_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{pmatrix} \end{aligned} \tag{2.8}$$

where c_x (resp. s_x) represents the simplified notation of $cos(x)$ (resp.$sin(x)$).

Decomposing the angular velocity $\Omega_{\mathcal{B}/\mathcal{I}}$ along the axes of each of the rotations R_ϕ, R_θ and R_ψ highlights the derivatives of the 3 parameters ϕ, θ and ψ

$$\begin{aligned} \boldsymbol{\Omega_{\mathcal{B}/\mathcal{I}}} &= \dot{\phi}\boldsymbol{x_{\mathcal{B}}} + \dot{\theta}\boldsymbol{y}” + \dot{\psi}\boldsymbol{z_{\mathcal{I}}} \\ &= \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix} + R_\phi \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} + R_\phi R_\theta \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix} \end{aligned} \tag{2.9}$$

leading to

$$\begin{pmatrix}\Omega_x\\ \Omega_y\\ \Omega_z\end{pmatrix} = \underbrace{\begin{pmatrix}1 & 0 & -s_\theta\\ 0 & c_\phi & s_\phi c_\theta\\ 0 & -s_\phi & c_\phi c_\theta\end{pmatrix}}_{\Theta}\begin{pmatrix}\dot\phi\\ \dot\theta\\ \dot\psi\end{pmatrix} \tag{2.10}$$

and

$$\begin{pmatrix}\dot\phi\\ \dot\theta\\ \dot\psi\end{pmatrix} = \begin{pmatrix}1 & s_\phi t_\theta & c_\phi t_\theta\\ 0 & c_\phi & -s_\phi\\ 0 & s_\phi/c_\theta & c_\phi/c_\theta\end{pmatrix}\begin{pmatrix}\Omega_x\\ \Omega_y\\ \Omega_z\end{pmatrix} \tag{2.11}$$

Remark 1 *Equation (2.11) highlights two singularities at $\theta = 0$ and $\theta = \pi$. They correspond to the situation of a gimbal lock, for which the yaw and the roll axes are equal, resulting in the loss of one degree of freedom of the representation.*

For low amplitude of roll and pitch angles values and the angular speeds are small, (2.11) can be linearized that corresponds to the cases studied in most literature work [105], this gives the

$$\begin{pmatrix}\dot\phi\\ \dot\theta\\ \dot\psi\end{pmatrix} \approx \begin{pmatrix}\Omega_x\\ \Omega_y\\ \Omega_z\end{pmatrix} \tag{2.12}$$

2.3.1.2 Attitude parametrization: Unit quaternion

Quaternions constitute another way to parametrize the attitude of a rigid body, by the mean of 4 parameters. Though it is not a unique representation, it has the advantage to be singularity-free.
A quaternion is defined by

$$\boldsymbol{q} \triangleq q_0 + q_1\boldsymbol{i} + q_2\boldsymbol{j} + q_3\boldsymbol{k} \in \mathbb{H}, \tag{2.13}$$

where $q_0, q_1, q_2, q_3 \in \mathbb{R}$; $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ correspond the imaginary units defined by equations $i^2 = j^2 = k^2 = ijk = -1$ as described by [106] ; and $\mathbb{H}$ is the quaternion space.

An alternative representation is in vector form

$$\boldsymbol{q} = [\ q_0 \quad \boldsymbol{q_v}\ ]^\top \tag{2.14a}$$

$$= [\ q_0 \quad q_1 \quad q_2 \quad q_3\ ]^\top \tag{2.14b}$$

$$= [\ \cos\left(\tfrac{\gamma}{2}\right) \quad \boldsymbol{\eta}\sin(\tfrac{\gamma}{2})\ ]^\top \tag{2.14c}$$

where, q_0 and $\boldsymbol{q_v} = [\ q_1 \quad q_2 \quad q_3\ ]^\top$ are respectively, the scalar and vector part of the quaternion. $\boldsymbol{\eta} \in \mathbb{R}^3$ is a unit vector corresponding to the axis of rotation and γ represents the rotation angle. The conjugate, the norm, and the inverse of the unit quaternion are defined as $\boldsymbol{q^*} = \begin{bmatrix} q_0 & -\boldsymbol{q_v} \end{bmatrix}^\top$, $\|q\| = \sqrt{q_0^2 + q_v^\top q_v}$, $\boldsymbol{q^{-1}} = \frac{\boldsymbol{q^*}}{\|q\|}$. From the components of $\boldsymbol{q}$, the expression of R is given by

$$\begin{aligned} R_q &= (q_0^2 - \boldsymbol{q}_v^\top q_v)\boldsymbol{I_3} + 2\boldsymbol{q_v}\boldsymbol{q_v}^\top - 2q_0\boldsymbol{q_v}^\times \\ &= \begin{pmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_3) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_3) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{pmatrix} \end{aligned} \tag{2.15}$$

with $(.)^\times$ represents the skew matrix.

The quaternion kinematics is described as follows

$$\dot{q} = \frac{1}{2}q \otimes \begin{pmatrix} 0 \\ \Omega \end{pmatrix} = \frac{1}{2}\Xi(q)\,\Omega = \frac{1}{2}\Gamma(\Omega)\,q \tag{2.16}$$

with

$$\Xi(q) = \begin{pmatrix} -q_v^\top \\ q_0 I_{3\times 3} + q_v^\times \end{pmatrix}, \Gamma(\Omega) = \begin{pmatrix} 0 & -\Omega^\top \\ \Omega & -\Omega^\times \end{pmatrix} \tag{2.17}$$

Remark 2 *a numerical integration of the relation (2.16) requires care since the norm of the quaternion should be maintained.*

2.3.1.3 Attitude parametrization: Rotation matrices

Finally, the attitude can directly be parameterized via the 9 components of the rotation matrix $R_{\mathcal{I}\to\mathcal{B}}$. This representation is both regular and unique, but requires 9 parameters and particular care regarding numerical errors. The matrix $R_{\mathcal{I}\to\mathcal{B}}$ must indeed be orthogonal with a determinant equal to 1, which constitutes a total of 6 constraints on the components of $R_{\mathcal{I}\to\mathcal{B}}$.

$$R_{\mathcal{I}\to\mathcal{B}} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \tag{2.18}$$

2.3.1.4 Attitude parameterization comparison

In a nutshell, the Euler angles representation has only three parameters and is widely adopted. This results in a nonlinear model due to the presence of trigonometric functions.

There has been considerable attention paid to unit quaternion-based attitude control laws since the unit quaternion is the minimal globally singular-free description of a rigid body's attitude. Nevertheless, each attitude is made up of two distinct unit quaternions covering the rotational configuration space SO(3). As a result of the unwinding phenomenon, rigid bodies undergo rapid rotation to find their equilibrium point, even if an equivalent position may exist that is closer to their original orientation. It is caused by the topological structure of S3, the unit sphere in R4, and by the fact that quaternions are double coverings of the special orthogonal group SO(3). This structure produces two quaternions for any given rotation.

There are two most popular modeling techniques for aerial vehicles namely the Euler-Lagrange [107, 108, 96, 109] and the Newton-Euler approaches [110]. This latter is used in this thesis to describe the quadrotor's dynamics.

Euler angles By applying the fundamental principle of dynamics, we obtain the equations representing the dynamic behavior of the quadrotor:

$$\begin{cases} \dot{\xi} = v \\ m\dot{v} = RTe_z + mge_z + K_{ft}v + F_{des} \\ \Omega = \begin{bmatrix} \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^\top \\ J\dot{\Omega} = -\Omega^\times J\Omega + \tau + \tau_{des} \end{cases} \tag{2.19}$$

where $\xi = [x, y, z]^\top \in \mathbb{R}^3$ and $v = [\dot{x}, \dot{y}, \dot{z}]^\top \in \mathbb{R}^3$ are respectively, the position and the linear velocity relative to the inertial frame. m is the total mass of the quadrotor. K_{f_t} denotes the translational drag coefficient. g is the acceleration due to gravity. $J \in \mathbb{R}^{3\times3}$ is the quadrotor's inertia matrix with the body-fixed frame. The terms F_{des},$\tau_{des} \in \mathbb{R}^3$ represent the external disturbances applied on the quadrotor. Finally, The terms $K_{ft} = diag(K_{ft_x}, K_{ft_y}, K_{ft_z})$ denote the translation drag coefficients, and $(x)^\times$ represents the skew-matrix of the vector x.

Unit quaternion Similar to Euler angles parametrization, the quadrotor's dynamics are as follows

$$\dot{x} = \begin{bmatrix} \dot{\xi} \\ \dot{v} \\ \dot{q} \\ \dot{\Omega} \end{bmatrix} := f(x, u) = \begin{bmatrix} v \\ \frac{1}{m}R_qT - ge_z + \frac{1}{m}K_{ft}v + F_{des} \\ \Gamma(\Omega)q \\ J^{-1}(-\Omega(t)^\times J\Omega(t) + \tau + \tau_{des}) \end{bmatrix} \tag{2.20}$$

The relation between the propeller's angular speeds and the generated aerodynamic forces and the moments due to the propellers is expressed as:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} b & b & b & b \\ 0 & -lb & 0 & lb \\ -lb & 0 & lb & 0 \\ d & -d & d & -d \end{bmatrix} \times \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \tag{2.21}$$

2.3.1 State Space Model

From the equation (2.19), the quadrotor is an under-actuated system with four inputs $\{T, \tau_x, \tau_y, \tau_z\}$ and six outputs $\{x, y, z, \phi, \theta, \psi\}$. To put the quadrotor equations in state-space form, the state vector of the system $x \in \mathbb{R}^{12}$ is chosen as:

$$\begin{aligned} x &= \begin{bmatrix} x, & \dot{x}, & y, & \dot{y}, & z, & \dot{z}, & \phi, & \dot{\phi}, & \theta, & \dot{\theta}, & \psi, & \dot{\psi} \end{bmatrix}^\top \\ &= \begin{bmatrix} x_1, & x_2, & x_3, & x_4, & x_5, & x_6, & x_7, & x_8, & x_9, & x_{10}, & x_{11}, & x_{12} \end{bmatrix}^\top \end{aligned} \tag{2.22}$$

The physical limitations of the quadrotor's motors speeds are bounded between minimum angular velocity $\underline{\omega}$ and maximum angular velocity $\bar{\omega}$. The maximum and minimum thrust force and torques values provided by these limitations are:

$$\begin{aligned} 4b\underline{\omega}^2 \leqslant u_1 \leqslant 4b\bar{\omega}^2 \\ bl(\underline{\omega}^2 - \bar{\omega}^2) \leqslant u_2 \leqslant bl(\bar{\omega}^2 - \underline{\omega}^2) \\ bl(\underline{\omega}^2 - \bar{\omega}^2) \leqslant u_3 \leqslant bl(\bar{\omega}^2 - \underline{\omega}^2) \\ 2d(\underline{\omega}^2 - \bar{\omega}^2) \leqslant u_4 \leqslant 2d(\bar{\omega}^2 - \underline{\omega}^2) \end{aligned} \tag{2.23}$$

where $[u_1, u_2, u_3, u_4]^\top = [T, \tau_x, \tau_y, \tau_z]^\top$ for simple notification.

According to equations (2.19) and (2.22) the dynamic model can be written in the following compact form:

$$\dot{x}(t) := \begin{cases} \dot{x}_{2i-1}(t) = x_{2i}(t), \quad i = 1, 2, ..., 6 \\ \dot{x}_{2i}(t) = f_i\left(x(t)\right) + \Delta f_i(x(t)) + \left(g_i\left(x(t)\right) + \Delta g_i(x(t))\right) u(t) + w_i(t) \end{cases} \tag{2.24}$$

where, $f_i(x(t))$ represents the dynamics of the plant and does not depend on the inputs, $g_i(x(t))$ is a non-zero input coefficient. Note that the subscripts (i) are provided to represent a single scalar value. $\Delta f_i(.)$ and $\Delta g_i(.)$ denote the bounded unknown parametric uncertainties, $w_i(t)$ is the bounded external disturbance, where: $|\Delta f_i(.)| \leqslant \Delta f_{i_{max}}$, $|\Delta g_i(.)| \leqslant \Delta g_{i_{max}}$, and $|w_i(t)| \leqslant w_{i_{max}}$.

The (2.24) can be simplified as:

$$\dot{x}(t) := \begin{cases} \dot{x}_{2i-1}(t) = x_{2i}(t), \qquad i = 1, 2, ..., 6 \\ \dot{x}_{2i}(t) = f_i\left(x(t)\right) + g_i\left(x(t)\right) u(t) + d_i(t) \end{cases} \tag{2.25}$$

where, $d_i(t) = \Delta f_i(x(t)) + \Delta g_i(x(t))u(t) + w_i(t)$, $f_i(x(t))$ and $g_i(x(t))$ are defined as follow:

$$\begin{aligned}
f_1(x(t)) &= \frac{K_{ft_x}}{m}x_1, & f_4(x(t)) &= \frac{(J_y - J_z)}{J_x}x_{10}x_{12},\\
f_2(x(t)) &= \frac{K_{ft_y}}{m}x_3, & f_5(x(t)) &= \frac{(J_z - J_x)}{I_y}x_8x_{12},\\
f_3(x(t)) &= \frac{K_{ft_z}}{m}x_5 - g, & f_6(x(t)) &= \frac{(J_x - J_y)}{J_z}x_8x_{10}.
\end{aligned} \tag{2.26}$$

$$\begin{aligned}
g_1(x(t)) &= \frac{1}{m}(\cos(x_1)\cos(x_5)\sin(x_3) + \sin(x_1)\sin(x_5))\\
g_2(x(t)) &= \frac{1}{m}(\cos(x_1)\sin(x_3)\sin(x_5) - \sin(x_1)\cos(x_5))\\
g_3(x(t)) &= \frac{1}{m}\cos(x_1)\cos(x_3)\\
g_4(x(t)) &= \frac{l}{J_x}\\
g_5(x(t)) &= \frac{l}{J_y}\\
g_6(x(t)) &= \frac{1}{J_z}
\end{aligned} \tag{2.27}$$

In this regard, it is important to note in the latter system that the orientation angles and their time derivatives(rotational dynamics) do not depend on translation components. On the other hand, translations depend on the quadrotor angles; This implies that the system described by (2.25) ideally consists of two subsystems, the translation subsystem and the rotation subsystem.

2.3.1.5 Discrete Time Quadrotor's Model

The continuous system model (2.24) should be transferred to a discrete model, in order to use later for designing a discrete control law. Hence, using forward approximation under the assumption of a sufficiently small sampling period, we obtain the discrete system as follows:

$$x[k+1] := \begin{cases} x_{2i-1}[k+1] = x_{2i-1}[k] + hx_{2i}[k] \; i = 1, 2, ..., 6 \\ x_{2i}[k+1] = x_{2i}[k] + h\left(f_i\left(x[k]\right) + \Delta f_i(x[k]) + \left(g_i\left(x[k]\right) + \Delta g_i(x[k])\right)u[k]\right) \\ \qquad + w_i[k] \end{cases} \tag{2.28}$$

where h is the sampling time, k represents the k^{th} sampling time, $f_i\left(x[k]\right)$ and

$g_i\left(x[k]\right)$ are given as follows

$$\begin{aligned}
f_1(x[k]) &= \frac{K_{ft_x}}{m}x_1[k] \\
f_2(x[k]) &= \frac{K_{ft_y}}{m}x_3[k] \\
f_3(x[k]) &= \frac{K_{ft_z}}{m}x_5[k] - g \\
f_4(x[k]) &= \frac{(J_y - J_z)}{J_x}x_{10}[k]x_{12}[k] \\
f_5(x[k]) &= \frac{(J_z - J_x)}{J_y}x_8[k]x_{12}[k] \\
f_6(x[k]) &= \frac{(J_x - J_y)}{J_z}x_8[k]x_{10}[k]
\end{aligned} \tag{2.29}$$

$$\begin{aligned}
g_1(x[k]) &= \frac{1}{m}(\cos\left(x_1[k]\right)\cos\left(x_5[k]\right)\sin(x_3[k]) + \sin(x_1[k])\sin(x_5[k])) \\
g_2(x[k]) &= \frac{1}{m}(\cos\left(x_1[k]\right)\sin\left(x_3[k]\right)\sin(x_5[k]) - \sin(x_1[k])\cos(x_5[k])) \\
g_3(x[k]) &= \frac{1}{m}\cos(x_1[k])\cos(x_3[k]) \\
g_4(x[k]) &= \frac{l}{J_x} \\
g_5(x[k]) &= \frac{l}{J_y} \\
g_6(x[k]) &= \frac{1}{J_z}
\end{aligned} \tag{2.30}$$

Remark 3 *The discretization model of the quadrotor in equation* (2.28) *is an approximation model for the original uncertain model in equation* (2.24)*. In order to have an approximately similar behavior between the discrete-time* (2.28) *and original continuous-time* (2.24) *systems, the sampling time should be sufficiently fast in comparison with system dynamics. Furthermore, this Euler discretization is used only in the controller synthesis because it is simpler to understand, in simulation, we will use the 4-th order of Runge-Kutta discretization.*

The discrete dynamic system equation (2.28) can be rewritten as :

$$\begin{cases} x_{2i-1}[k+1] = x_{2i-1}[k] + hx_{2i}[k], \quad i = 1, 2, ..., 6 \\ x_{2i}[k+1] = x_{2i}[k] + h\left(f_i\left(x[k]\right) + g_i\left(x[k]\right)u[k]\right) + d[k] \end{cases} \tag{2.31}$$

where $d[k] = h(\Delta f_i(x[k]) + \Delta g_i(x[k])u[k]) + w_i[k]$, $d[k] \in \mathbb{R}^6$ denotes the general uncertainties (including the modeling errors and external disturbances), and it is assumed bounded in a compact set $\mathcal{D}$ that contains the origin: $d[k] \in \mathcal{D}$, where the set $\mathcal{D}$ is bounded by ε in 2-norm, *i.e.*, for all $d[k] \in \mathcal{D}$, there is $\|d[k]\| \leq \varepsilon$.

Let $u[k] = [u_1[k], u_2[k], u_3[k], u_4[k]]^T$ being the control input. The main objective is to synthesize non-linear control laws for a quad-rotor in order to track the desired trajectory $\{x_d, y_d, z_d, \psi_d\}$. In the next chapter, we assume the case without disturbances, i.e $d[k] = 0$ and it's unknown to the controllers.

Table 2.1: Parameters of the used quadroter model.

Symbol	**Value**	**Unit**
m	0.486	kg
g	9.806	m/s^2
l	0.25	m
b	2.9842×10^{-5}	$N/rad/s$
d	3.232×10^{-7}	$N.m/rad/s$
J	$\begin{pmatrix} 3.8278 & 0 & 0 \\ 0 & 3.8288 & 0 \\ 0 & 0 & 7.6566 \end{pmatrix} \times 10^{-3}$	kg/m^2
K_{ft}	$\begin{pmatrix} 5.567 & 0 & 0 \\ 0 & 5.567 & 0 \\ 0 & 0 & 6.354 \end{pmatrix} \times 10^{-4}$	$N/m/s$
$\underline{\omega}$	0	rad/s
$\bar{\omega}$	280	rad/s

2.4 Rotors dynamics

The most common motors used in quadcopters are brushless Direct Current (BLDC) motors due to their high reliability, long life, minimal maintenance requirements, excellent controllability, and wide speed range. They require each phase to be powered by an external source during the correct period of rotation. To do this, power circuits called Electronic Speed Controllers (ESC) are used to perform the function of inverter and control circuits.

In the following, the BLDC motor and ESC are considered equivalent to the brushed DC motor, and $V_e(t)$ the effective voltage applied by the ESC to the terminals of the BLDC motor, Figure 2.4.

By applying Kirchhoff's Voltage Law for the electrical part, and according to Faraday's laws of electromagnetic induction, a DC motor can be represented by the fol-

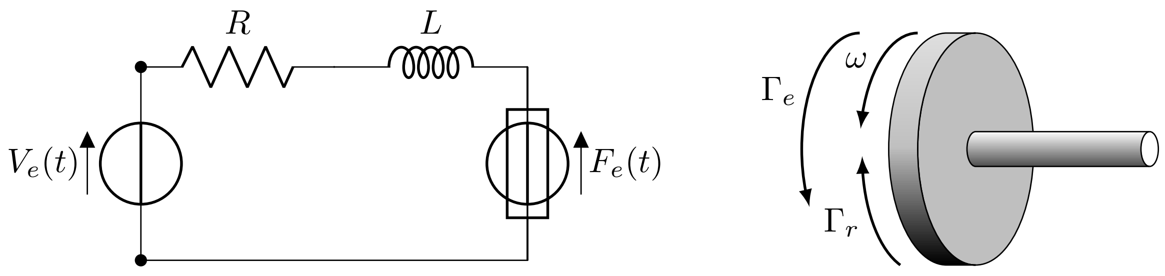


Figure 2.4: Schematic representation of DC motor.

lowing electro-mechanical equations:

$$V_e(t) = R_m i(t) + L_m \frac{di(t)}{dt} + F_e(t) \tag{2.32a}$$

$$J_r \dot{\omega} = \Gamma_e + \Gamma_r \tag{2.32b}$$

where, $i(t)$ is the armature's current, $F_e(t) = K_e \omega$, is the back electromotive force, K_e is DC motor constant, L_m is the armature's inductance, R_m is the motor's armature resistance, $\Gamma_e = K_e i(t)$ is the motor's supplied torque, $\Gamma_r = -f_r \omega$ is the torque due to viscous friction, f_r is the damping of the DC motor and accessories.

The motors used in quadcopters are usually small in size, so the inductance is very low, and therefore the electrical part is much faster than the mechanical part. Equation (2.32a) becomes:

$$V_e(t) = R_m i(t) + K_e \omega \tag{2.33}$$

Using the Laplace transform on equations (2.32b) and (2.33) and simplifying, we get:

$$\frac{\omega(s)}{V_e(s)} = \frac{1}{\frac{R_m J_r}{K_e} s + f_r + K_e} \tag{2.34}$$

The equivalent voltage $V_e(t)$ is the voltage generated by the ESC according to the control signal u_{PWM} representing the width of the *Pulse Width Modulation* (PWM) signal received at the input. The relationship between $V_e(t)$ and u_{PWM} is given by:

$$V_e(t) = u_{PWM} V_B \tag{2.35}$$

where V_B is the voltage available at the input.

The expression of u_{PWM} is given by:

$$u_{PWM}(t) = \frac{P(t) - P_{max}}{P_{max} - P_{min}} \tag{2.36}$$

where $P(t)$ is the pulse width at time t, P_{max} is the maximum pulse width, and P_{min} is the minimum pulse width.

Equation (2.34) finally becomes:

$$\frac{\omega(s)}{u_{PWM}(s)} = \frac{K}{Ts + 1} \tag{2.37}$$

where: $K = \frac{V_B}{K_e + f_r}$, and $T = \frac{R_m J_r}{K_e(K_e + f_r)}$.

2.5 Conclusion

This chapter presented the dynamic modeling of the quadrotor with two attitude parameterizations, namely, Euler angles and quaternions. We presented the quadrotor, the principle of its movement, and the different forces and moments that act on the aircraft. We conclude that the Quadrotor is an under-actuated nonlinear system since the number of control inputs is less than the number of degrees of freedom. The next chapter will provide a design of various linear and non-linear control techniques.

Chapter 3 DESIGN OF CONVENTIONAL NONLINEAR CONTROL APPROACHES

3.1 Introduction

There is always a difference between the mathematical process model and reality. This difference is due to environmental phenomena neglected during modeling and errors in the precision of the values of parameters of the model. Therefore, the question is how to obtain the desired performance despite the presence of neglected dynamics in the process model. The study has three parts: first, we synthesize an integral backstepping controller based on the nonlinear model developed previously. The second part will be devoted to the presentation of a controller by sliding mode, and we will end with the third part where the controller is designed by nonlinear model predictive control.

3.2 Preliminaries

Before moving towards the designing controller, it is necessary to define Lyapunov and other stability notions, as they serve as a benchmark for appropriate performance-based evaluation.

Definition 1 (Lyapunov stability [111]) *Assume a time invariant system as follows:*

$$\dot{x} = f(x) \tag{3.1}$$

Consider this system has equilibrium point x_e, i.e. $f(x_e) = 0$ and it starts with initial state $x(0)$. We can say that equlibrium point x_e can be identified as:

- *Stable: if for each $\epsilon > 0$, there exists $\delta(\epsilon) > 0$ such that*

$$\|x(0) - x_e\| < \delta \quad and \quad \|x(t) - x_e\| < \epsilon \qquad \forall t \geq 0$$

- *Unstable: if above condition does not hold.*

Definition 2 (Asymptotic stability [111, 112]) *Lyapunov stability lacks the rigorousness in a sense that it does not even infer that $x(t)$ converges x_e as t reaches infinity. The only assurance is the hovering of states around x_e. Convergence requires a stronger conception and an augmented notion is the asymptotic stability. The dynamical system (18), is said to be asymptotically stable if it is Lyapunov stable and in addition, there exists $r > 0$ such that:*

$$\|x(0) - x_e\| < r \quad while \quad x(t) \longrightarrow x_e \ as \ t \longrightarrow \infty$$

Definition 3 (Exponential stability [111, 112]) *The dynamical system presented by (18) is supposed to be exponentially stable if there exist positive constants λ and μ such that:*

$$\|x(t)\| < \lambda \|x(0)\| e^{-\mu t}$$

The definitions of stability are said to be global if these hold for all initial states. From the definitions, it can be seen that the stronger notion of stability is the exponential stability.

Considering the system expressed in (2.25), we are assuming here that disturbances $d_i(t) = 0$, (2.25) becomes:

$$\dot{x} = \begin{cases} \dot{x}_{2i-1} = x_{2i}, & i = 1, 2, ..., 6 \\ \dot{x}_{2i} = f_i + v_i \end{cases} \tag{3.2}$$

where, $f_i = f_i(x)$ are the same as defined in (2.26), v_i are defined as follow:

$$\begin{aligned} v_1 = g_1(x)u_1, \quad v_2 = g_2(x)u_1, \quad v_3 = g_3(x)u_1, \\ v_4 = g_4(x)u_2, \quad v_5 = g_5(x)u_3, \quad v_6 = g_6(x)u_4 \end{aligned} \tag{3.3}$$

3.3 Integral Backstepping Approach

The backstepping control is based on the Lyapunov Stability Theory. Backstepping design involves recursively selecting some appropriate functions of state variables as ”virtual control” inputs to subsystems with single inputs and single outputs of the overall system. Following this, the backstepping design is divided into different design steps. Adding an integral term ensures the convergence to the desired trajectory and increases the robustness of the system when the parameters are unknown.

Step 1: The error variable and its integral are defined as follows:

$$\begin{aligned} e_{1_i} &= r_{2i-1} - x_{2i-1} \\ p_i(t) &= \int_0^t e_{1_i}(\tau)d\tau, \qquad i = 1, \dots, 6 \end{aligned} \tag{3.4}$$

where, r is the desired trajectory and is defined as follows:

$$\begin{aligned} r &= \begin{bmatrix} x_d & \dot{x}_d & y_d & \dot{y}_d & z_d & \dot{z}_d & \phi_d & \dot{\phi}_d & \theta_d & \dot{\theta}_d & \psi_d & \dot{\psi}_d \end{bmatrix}^\top \\ &= \begin{bmatrix} r_1 & r_2 & r_3 & r_4 & r_5 & r_6 & r_7 & r_8 & r_9 & r_{10} & r_{11} & r_{12} \end{bmatrix}^\top \end{aligned} \tag{3.5}$$

and in compact form:

$$r = \begin{cases} r_{2i-1} \\ r_{2i} \qquad i = 1, 2, \dots, 6 \end{cases} \tag{3.6}$$

This definition specifies the control objective, where the recursive methodology will systematically drive the tracking error to zero. Herein, Lyapunov candidate functions V_{1_i}, which are positive definite around the desired position, are used for stabilizing the tracking errors e_{1_i}:

$$V_{1_i} = \frac{1}{2}e_{1_i}^2 + \beta_i p_i^2 \tag{3.7}$$

where $\beta_i \in \mathbb{R}^+$ are tuning parameters which will be fixed later.

The derivatives of V_{1_i} are written as:

$$\dot{V}_{1_i} = e_{1_i}\left(r_{2i} - x_{2i} + \beta_i p_i\right) \tag{3.8}$$

Stabilization of e_{1_i} can be guaranteed by introducing new virtual control inputs χ_i:

$$\chi_i = r_{2i} + \beta_i p_i + \alpha_{1_i} e_{1_i} \tag{3.9}$$

where α_{1_i} are positive numbers that determine the error convergence speed. Thus the derivative of the Lyapunov functions are negative definite, consequently, the error converges to zero ($\dot{V}_{1_i} = -\alpha_{1_i} e_{1_i}^2 \leqslant 0$).

Step 2: Let e_{2_i} be the difference between the quadrotor velocity (linear and angular velocities) and the virtual command χ_i:

$$e_{2_i} = \chi_i - x_{2i} \tag{3.10}$$

From equations (3.9), we get the dynamics of the error $\dot{e}_{2_i}$.

$$\begin{aligned} \dot{e}_{2_i} &= \dot{\chi}_i - \dot{x}_{2i} \\ &= \dot{r}_{2i} + \beta_i e_{1_i} + \alpha_{1_i} \left(e_{2_i} - \alpha_{1_i} e_{1_i} - \beta_i p_i \right) - \dot{x}_{2i} \end{aligned} \tag{3.11}$$

If $e_{2_i} \to 0$, the equation (3.10) will be satisfied.

At this step, the error e_{2_i} must be stabilized. For this, we consider the augmented Lyapunov functions:

$$V_{2_i} = \frac{1}{2} e_{1_i}^2 + \beta_i p_i^2 + \frac{1}{2} e_{2_i}^2 \tag{3.12}$$

and their time derivatives as follows:

$$\begin{aligned} \dot{V}_{2_i} &= e_{1_i} \dot{e}_{1_i} + \beta_i p_i e_{1_i} + e_{2_i} \dot{e}_{2_i} \\ &= e_{1_i} \left(-\alpha_{1_i} e_{1_i} + e_{2_i} \right) + e_{2_i} \left(\dot{r}_{2i} + \beta_i e_{1_i} + \alpha_{1_i} \left(-\alpha_{1_i} e_{1_i} + e_{2_i} \right) - \dot{x}_{2i} \right) \\ &= -\alpha_{1_i} e_{1_i}^2 + e_{2_i} \left(\dot{r}_{2i} + \left(1 - \alpha_{1_i}^2 + \beta_i \right) e_{1_i} + \alpha_{1_i} e_{2_i} - \alpha_{1_i} \beta_i p_i - \dot{x}_{2i} \right) \end{aligned} \tag{3.13}$$

with $\dot{V}_{2_i}$ being designed as

$$\dot{V}_{2_i} = -\alpha_{1_i} e_{1_i}^2 - \alpha_{2_i} e_{2_i}^2 \tag{3.14}$$

where α_{2_i} are positive numbers for tunning.

Finally, the following controller is obtained

$$v_i = -\alpha_{1_i}\beta_i p_i + \left(\beta_i + 1 - \alpha_i^2\right) e_{1_i} + \left(\alpha_{1_i} + \alpha_{2_i}\right) e_{1_i} + \dot{r}_{2i} - f_i, \quad i = 1, 2, \ldots, 6. \quad (3.15)$$

Step 3: The obtained commands in (3.15) are virtual commands. In this step, we will determine the reel commands,u_1, u_2, u_3 and u_4 from the virtual one $v = \begin{bmatrix} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{bmatrix}^\top$. According to (3.2), we have:

$$\begin{pmatrix} \cos(x_1)\cos(x_5)\sin(x_3) + \sin(x_1)\sin(x_5) \\ \cos(x_1)\sin(x_3)\sin(x_5) - \sin(x_1)\cos(x_5) \\ \cos(x_1)\cos(x_3) \end{pmatrix} \frac{u_1}{m} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} \quad (3.16)$$

$$\begin{pmatrix} \frac{l}{J_x} u_2 \\ \frac{l}{J_y} u_3 \\ \frac{1}{J_z} u_4 \end{pmatrix} = \begin{pmatrix} v_4 \\ v_5 \\ v_6 \end{pmatrix} \quad (3.17)$$

We have three equations in (3.16) to determine the command u_1. For this, we note that:

$$\begin{pmatrix} \cos(x_1)\cos(x_5)\sin(x_3) + \sin(x_1)\sin(x_5) \\ \cos(x_1)\sin(x_3)\sin(x_5) - \sin(x_1)\cos(x_5) \\ \cos(x_1)\cos(x_3) \end{pmatrix} \frac{u_1}{m} = Re_3 \frac{u_1}{m} \quad (3.18)$$

Knowing according to the properties of the rotation matrices:

$$\left\| Re_3 \frac{u_1}{m} \right\| = \left\| \frac{u_1}{m} \right\| \quad (3.19)$$

By taking the modulus of both sides of (3.16), we get:

$$u_1 = m\sqrt{v_1^2 + v_2^2 + v_3^2} \quad (3.20)$$

From (3.17), we easily obtain the three commands u_2, u_3, u_4:

$$u_2 = \frac{J_x}{l} v_4, \quad u_3 = \frac{J_y}{l} v_5, \quad u_4 = J_z v_6 \quad (3.21)$$

Considering (3.16), it is noticeable that the quadrotor's orientation determines the direction of the thrust force in space. As a consequence, it is essential to have a correct orientation of the quadrotor to be able to reach the desired position. Therefore, it is necessary to design a high-performance attitude controller. The desired roll and pitch angles ϕ_d and θ_d are generated from equation:

$$\begin{aligned} m(v_1 c_{\psi_d} + v_2 s_{\psi_d}) &= s_{\theta_d} u_1 \\ m(v_1 s_{\psi_d} - v_2 c_{\psi_d}) &= s_{\phi_d} c_{\theta_d} u_1 \\ m(v_3) &= c_{\phi_d} c_{\theta_d} u_1 \end{aligned} \tag{3.22}$$

that yields:

$$\begin{aligned} \phi_d &= \arctan\left(\frac{v_1 s_{\psi_d} - v_2 c_{\psi_d}}{v_3}\right) \\ \theta_d &= \arctan\left(\frac{v_1 c_{\psi_d} + v_2 s_{\psi_d}}{\sqrt{(v_1 s_{\psi_d} - v_2 c_{\psi_d})^2 + v_3^2}}\right) \end{aligned} \tag{3.23}$$

3.3.1 Controller Results

In the following, the synthesized command in (3.20) and (3.21) is applied to the dynamics of the quadrotor. Figures 3.1 and 3.2 present the simulation results of the IBC technique applied to the quadrotor model. The simulation was performed in MATLAB/SIMULINK® with the initial conditions $x_0 = [0, 0, 1, 0, 1, 0.1, -10°, 0, 35°, 0, 30°, 0]^\top$. The dynamic controller parameters which stabilize the quadrotor are listed in Table 3.1. As shown in Figure 3.1, the position and rotation of the quadrotor track ideally the reference trajectory.

Table 3.1: Controller's parameters of IBC.

Controller	**Symbol**	**Value**					
	β_i	50	50	18	10^2	10^2	10
IBC	α_{i_1}	7	7	15	70	70	6
	α_{i_2}	5	5	0.5	0.5	0.5	5

3.4 Sliding Mode Control Approach

Control in the presence of uncertainty is one of the issues of modern control theory. The uncertainties in the controlled system are often caused by the differences between the plant's actual dynamics and its mathematical model dynamics used to design the controller. The most common differences are unknown plant pa-

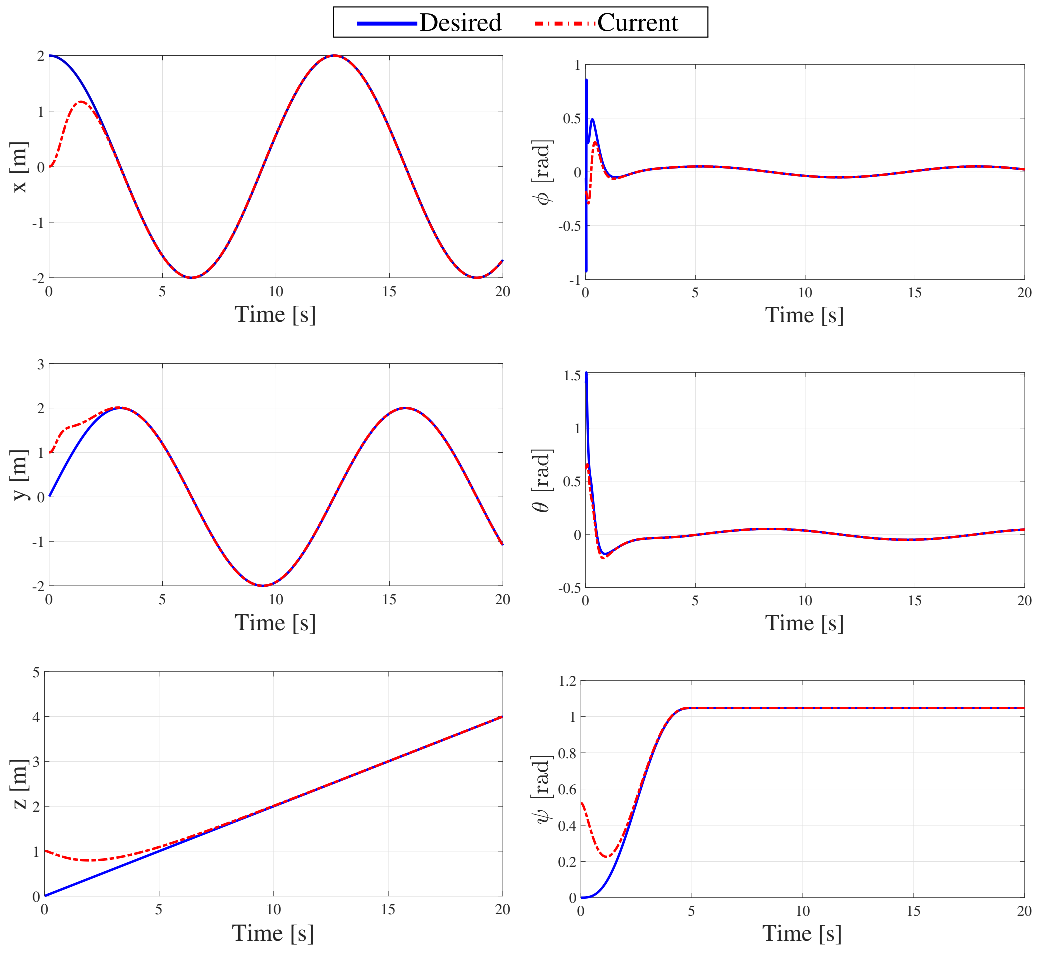


Figure 3.1: Helix trajectory tracking results with the IBC approach.

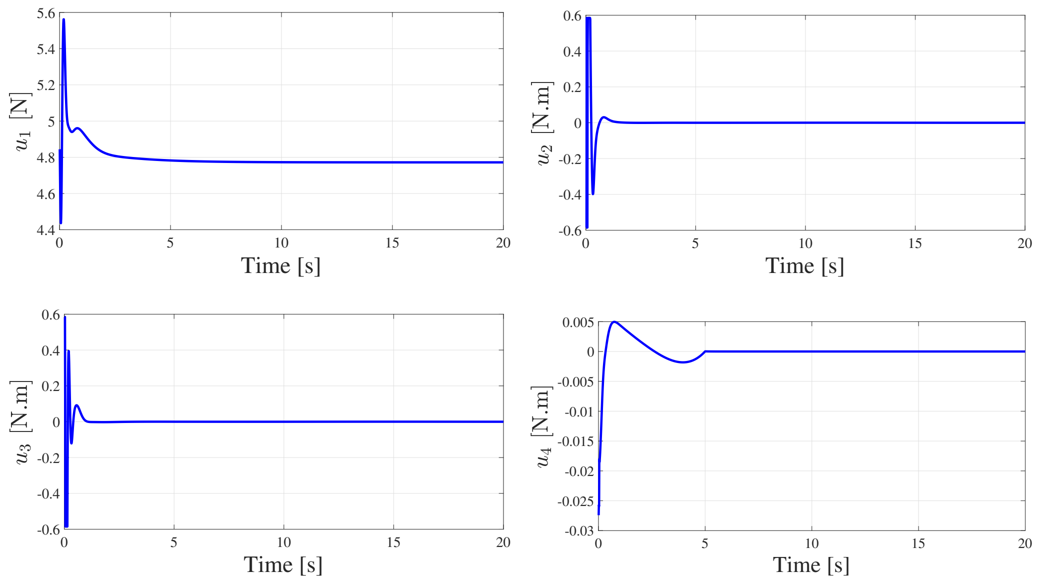


Figure 3.2: Control inputs with the IBC approach.

rameters and external disturbances. In such a case, achieving desired closed-loop system performance is a very challenging problem, and there has been enormous interest in finding robust control laws to solve this problem; Among the approaches to designing robust controllers is SMC which is a particular kind of *Variable Structure System* (VSS). Control laws for sliding mode control consist of two essential phases: (i) a reaching phase, and (ii) a sliding mode phase. The reaching phase drives the system state from any initial condition to reach the switching manifold in a finite time; while sliding phase drives the system states to equilibrium or origin. Figure 3.3 shows the sliding mode phases configuration with s as a continuous-time sliding function given by:

$$S = \{x \in X | S(x,t) = 0\} \tag{3.24}$$

In order to achieve the sliding mode following properties should be ensured: i) The

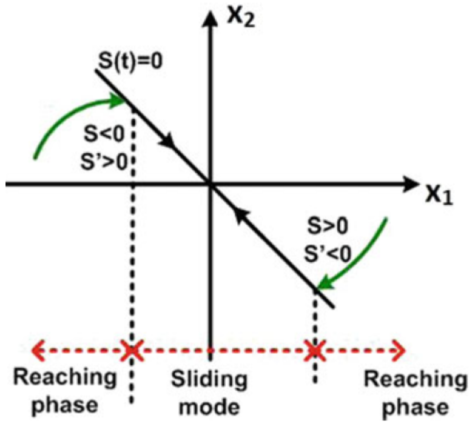


Figure 3.3: Sliding mode phases.

system stability is strictly restricted to the sliding surface., ii) Sliding mode should occur within a finite time. The sufficient condition for the occurrence of sliding motion on a sliding surface is given by

$$S\dot{S} < 0 \tag{3.25}$$

where S is the sliding surface and $\dot{S}$ is the rate of change of distance from the sliding surface. The condition in (3.25) is known as a reachability condition that is not sufficient for the sliding mode. The main drawback of this condition is that $S(t)$ takes an infinite time to reach the sliding surface. Thus, to overcome this problem another condition is defined:

$$S\dot{S} < -\Theta|S|, \quad \Theta > 0 \tag{3.26}$$

This condition is known as "Θ-reachability" condition that ensures the finite time convergence to $S = 0$.

The briefing of proposed reaching laws in literature [113, 114, 115] is

- Constant-proportional rate

$$\dot{S} = -\kappa\, sign(S) - \mu S, \quad \kappa, \mu > 0$$

- Power rate reaching law

$$\dot{S} = -\mu S - \kappa |S|^{\zeta} sign(S), \quad 0 < \zeta < 1$$

- Power rate exponential reaching law

$$\dot{S} = -\frac{\kappa}{N(S)},$$

  where $N(S) = \delta_0 + (1 - \delta_0)^{-\zeta |S|^{p_0}}$, δ_0 is strictly positive offset less than 1, and $p_0 \in \mathbb{N}^+$ is a strictly positive integer.

3.4.1 SMC for Quadrotor

Each state is second relative order with respect to its order, a second-order Slotine surface is chosen

$$S_i = \dot{\varepsilon}_i + \lambda_i \varepsilon_i \quad i = 1, 2, \dots, 6 \tag{3.27}$$

where $\varepsilon_i = r_{2i-1} - x_{2i-1}$, and $\lambda_i \in \mathbb{R}^+$ is tunable parameter.

The purpose of the control is to force the system onto the sliding surface and prevent it to get out, i.e.

$$S_i = 0 \tag{3.28}$$

The dynamic of the surface (3.27) is

$$\dot{S}_i = \ddot{\varepsilon}_i + \lambda_i \dot{\varepsilon}_i \tag{3.29}$$

Or

$$\dot{S}_i = \dot{r}_{2i} - f_i - v_i + \lambda_i \left(r_{2i} - \dot{x}_{2i} \right) \tag{3.30}$$

The following desired dynamics are

$$\dot{S}_i = -\kappa\, sign\left(S_i\right) - \mu_i S_i \tag{3.31}$$

where the $\kappa \in \mathbb{R}^+$ and $\mu_i \in \mathbb{R}^+$ are tunable parameters, and the function $sign(.)$ is defined as follows [116]

$$sign(x) = \begin{cases} 1 & if \quad x > 0 \\ 0 & if \quad x = 0 \\ -1 & if \quad x < 0 \end{cases} \tag{3.32}$$

Equating (3.30) and (3.31) the following command signal is obtained

$$v_i = \dot{r}_{2i} - f_i - \lambda_i \left(r_{2i} - \dot{x}_{2i}\right) - \kappa_i\, sign\left(S_i\right) - \mu_i S_i \tag{3.33}$$

The real control inputs (thrust force u_1, and the torques $\{u_2, u_3, u_4\}$) are obtained similarly to (3.20) and (3.21).

The desired roll and pitch angles are obtained similarly to (3.23).

Remark 4 *The discontinuity of the signum function in the SMC law may lead to chattering. In order to reduce the chattering phenomenon, we replace the discontinuous function* $sign(x)$ *with the following continuous pseudo-sign function, Figure 3.4 represents the sign function and pseudo function with a different value of delta, it can be seen that the more delta is small, the more the pseudosign becomes identical to the sign function, a choice has been made for* $\eta = 0.05$

$$psign(x, \eta) = \frac{x}{|x| + \eta} \tag{3.34}$$

3.4.2 Stability analysis

To study the stability of each of its commands, we consider the following Lyapunov functions:

$$V_i = \frac{1}{2} S_i^2 > 0, \quad i = 1, 2, \ldots, 6 \tag{3.35}$$

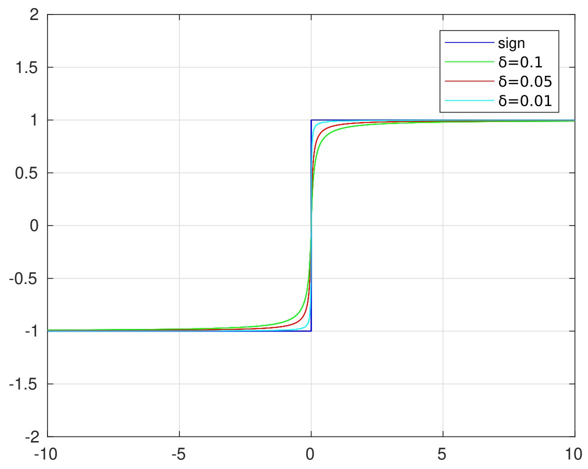


Figure 3.4: Sign and pseudo-sign function with different values of the delta.

In this case, it is considered that the disturbances are not equal to zero, i.e. $d_i(t) \neq 0$. By deriving each of these candidate Lyapunov functions and by considering equations (3.30) and (3.33), we obtain:

$$\dot{V}_i = S\left(-\kappa_i sign(S_i) - \mu_i S_i - d_i\right) \tag{3.36}$$

Recalling that the disturbances d_i are bounded ($|d_i| < \epsilon_i$), which gives us:

$$\dot{V}_i < S\left(-\kappa_i sign(S_i) - \mu_i S_i - \epsilon\right) \tag{3.37}$$

By choosing the constants κ and μ_i such that:

$$\begin{cases} \mu_i > 0, \\ \kappa_i > \epsilon_i \quad i = 1, \ldots, 6 \end{cases} \tag{3.38}$$

yields

$$\dot{V}_i = -\mu_i S_i^2 - S_i(\kappa_i sign(S_i)) - \epsilon_i) < 0 \tag{3.39}$$

because:

$$-\mu_i S_i^2 < 0, \quad \forall \mu_i > 0 \tag{3.40}$$

and:

$$\begin{cases} S_i > 0 \Rightarrow sign(S_i) = 1 \Rightarrow \\ -S_i(\kappa_i sign(S_i) - \epsilon_i) = -S_i(\kappa_i - \epsilon_i) < 0, \quad \forall \kappa_i > \epsilon_i \\ S_i < 0 \Rightarrow sign(S_i) = -1 \Rightarrow \\ -S_i(\kappa_i sign(S_i) - \epsilon_i) = S_i(\kappa_i + \epsilon_i) < 0, \quad \forall \kappa_i > \epsilon_i > 0 \end{cases} \tag{3.41}$$

3.4.3 Controller Results

Figures 3.5 to 3.6 present the simulation results of the SMC technique applied to the quadrotor model. All simulations were performed with the initial positions $x_0 = [0, 0, 1, 0, 1, 0.1, -10°, 0, 35°, 0, 30°, 0]^\top$. The parameters of the SMC are defined as follows: $\lambda = \left[20, 20, 35, 10, 10, 10^{-5}\right]$, $\kappa = \left[5, 5, 3, 0.7, 0.7, 10^{-6}\right]$, and $\mu = \left[1, 1, 3.5, 10^{-2}, 10^{-2}, 5 \times 10^{-5}\right]$.

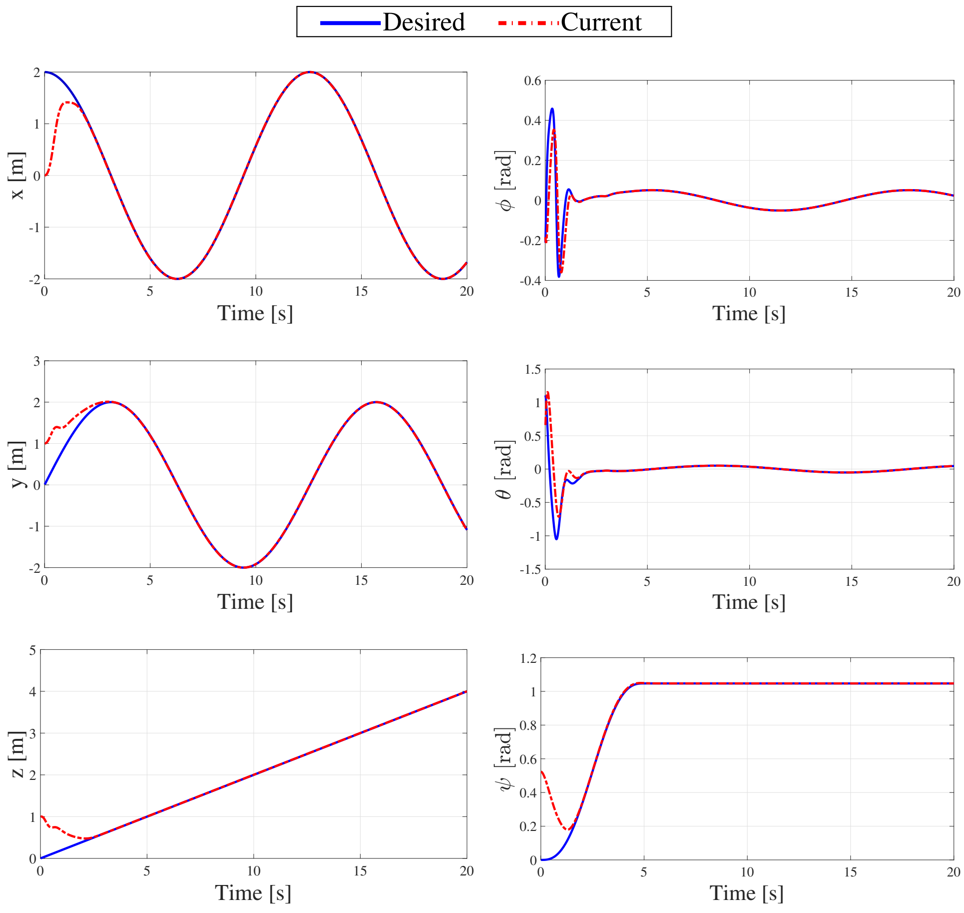


Figure 3.5: Helix trajectory tracking results with the SMC approach.

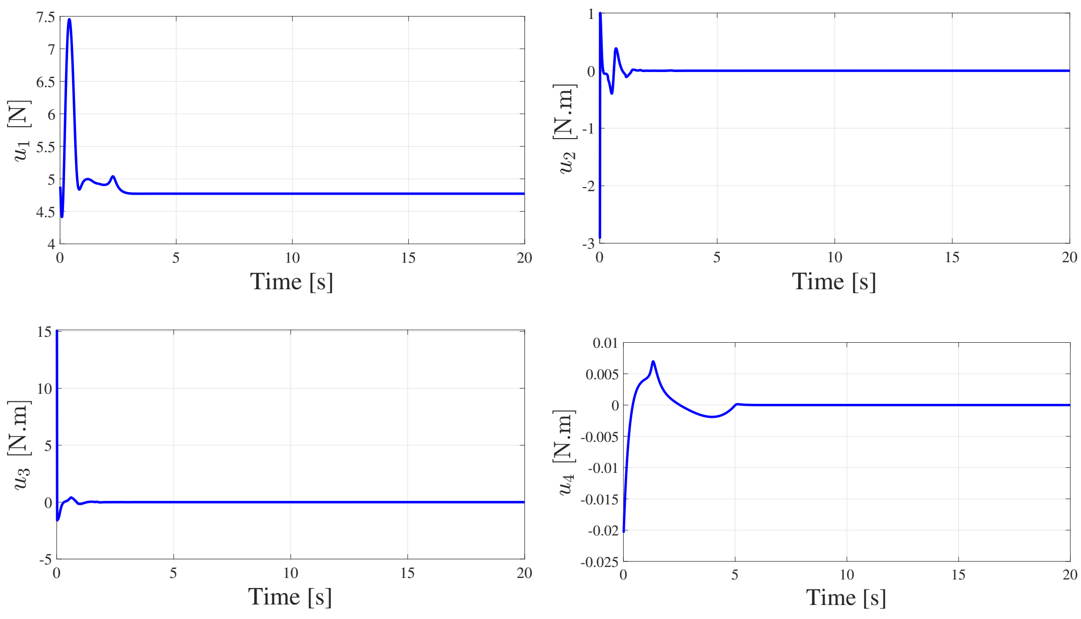


Figure 3.6: Control inputs with the SMC approach.

3.5 Nonlinear Model Predictive Control NMPC

The predictive control problem consists of determining the control vector u that minimizes the selected cost function, while ensuring the satisfaction of the constraints. Its principle makes it possible to transform this optimization problem in infinite horizon continuous-time into a finite horizon optimal control problem. The predictive approach methodology can be illustrated in the diagram shown in Figure 3.7:

- At each sampling time k, the future system outputs are predicted over a prediction horizon N using the preceding inputs and outputs. These predictions are noted $x[k+j|k], j = 0, 1, \ldots, N$ to indicate the value of the output at instant $k + j$, calculated at the instant k.

- The sequence of future commands $u[k+j|k], j = 0, 1, \ldots, N-1$ is calculated by optimizing a certain determined criterion so that the predicted output $x[k+j|k]$ is as close as possible to the reference trajectory $r[k + j|k], j = 1, \ldots N$,while minimizing the control effort. If the criterion is quadratic, and if the system is linear as well as there are no constraints on the output nor the input, the solution is obtained explicitly. Otherwise, an iterative optimization method must be used.

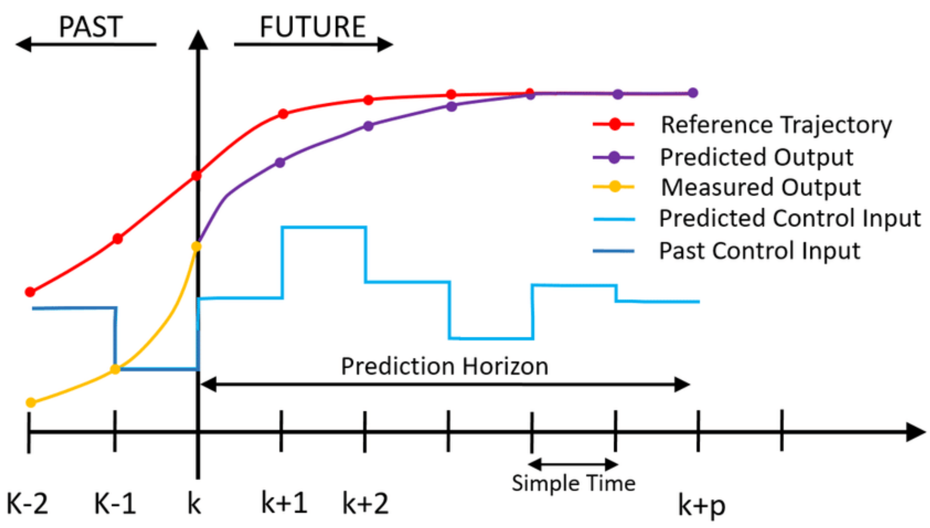


Figure 3.7: Basic concept for Model Predictive Control.

- Finally, Only the first component $u[k|k]$ of the optimal control sequence $u[k|k+j]$ is applied to the system. At the next sampling time $k+1$, the resolution begins again with step one by taking into account the new updated measurements of the system $x[k+1]$ and a new control sequence $u[k+1|k+1+j], j = 0, \ldots, N-1$ is determined. The control sequence is improved at each sampling period since new measurements could be taken and consequently the vector of the control signal $u[k+1|k+1+j], j = 0, 1, .., N-1$ will be in principle different from $u[k+j|k], j = 0, 1, .., N$. This is known as the concept of receding horizon.

Based on the above definition, the discrete-time NMPC formulation with multiple shooting is as follows:

$$\min_{u[k+j|k],x[k+j|k]} \sum_{j=0}^{N-1} L_r\left(x[j], u[j], r[j]\right) + L_t\left(x[N]\right) \tag{3.42a}$$

$$\textbf{s.t}: x[0] - x_0 = 0 \tag{3.42b}$$

$$x[j+1] - x[j] - \boldsymbol{f_{RK4}}(x[j], u[j]) = 0, \qquad j = 0, \ldots, N \tag{3.42c}$$

$$x[j] \in \mathcal{X}, \qquad j = 0, \ldots, N \tag{3.42d}$$

$$u[j] \in \mathcal{U}, \qquad j = 0, \ldots, N-1 \tag{3.42e}$$

where (3.42d) and (3.42e) are respectively, the sets of constraint on states (map margins and Euler-angles limitations i.e. $-\frac{\pi}{2} \prec \phi \prec \frac{\pi}{2}$,$-\frac{\pi}{2} \prec \theta \prec \frac{\pi}{2}$, $-\pi \prec \psi \prec \pi$) and on inputs that were defined in equation (2.23). x_0 is the current state.

The running cost function denotes $L_r\left(x[j], u[j], r[j]\right)$ and is equal to:

$$L_r\left(x[j], u[j], r[j]\right) = \|r[j] - x[j]\|^2_{\mathbf{Q}} + \|u_{ref}[j] - u[j]\|^2_{\mathbf{R}} \tag{3.43}$$

and $L_t\left(x[N]\right)$ being the terminal cost function and is equal to:

$$L\left(x[N]\right) = \|r[N] - x[N]\|^2_{\mathbf{H}} \tag{3.44}$$

where $\mathbf{Q}, \mathbf{H} \in \mathbb{R}^{12\times 12}$ are positive-definite tuning matrices, $\mathbf{R} \in \mathbb{R}^{4\times 4}$ is positive semi-definite matrix.

The control input reference u_{ref} is taken to obtain better tracking performance based on desired trajectory acceleration and defined as: $u_{ref} = \left[m\sqrt{a_1^2 + a_2^2 + (a_3+g)^2},\ 0,\ 0,\ 0\right]^\top$, where a_1, a_2 and a_3 are the discrete time of desired trajectory acceleration $\{\ddot{x}_d, \ddot{y}_d, \ddot{z}_d\}$.

Herein (3.42c), the concept of direct multiple shooting [117] is defined as an equality constraint, where $\boldsymbol{f_{RK4}}(.)$ is the Runge Kutta 4^{th} integration and is defined as :

$$\begin{aligned}
k_1 &= \boldsymbol{f}(x[k], u[k]) \\
k_2 &= \boldsymbol{f}(x[k] + \frac{h}{2}k_1, u[k]) \\
k_3 &= \boldsymbol{f}(x[k] + \frac{h}{2}k_2, u[k]) \\
k_4 &= \boldsymbol{f}(x[k] + hk_3, u[k]) \\
\boldsymbol{f_{RK4}}(x[k], u[k]) &= 1/6\left(k_1 + 2k_2 + 2k_3 + k_4\right)
\end{aligned} \tag{3.45}$$

and, $\boldsymbol{f}(x[k], u[k]) = \left[x_{2i}\ ,\ f_i(x[k]) + g_i(x[k])u[k]\right]^\top,\ i = 1, \ldots, 6.$

Remark 5 *Multiple shooting takes it one step further: it not only introduces a grid of control values but also introduces additional decision variables for the states on the same grid.*

3.5.1 Controller Results

In order to implement the proposed quadrotor system we need to solve the Optimization Control Problem (OCP) (4.39) in the previous section. This can be done in multiple ways, however, the two main classes of methods are sequential methods, such as direct single shooting [118], and simultaneous methods such as direct multiple shooting [119], and direct collocation [120]. In this work, we chose to use direct multiple shooting, in where implicit numerical integration of the ODE constraints (3.42b) and (3.42c), as well as the objective function (3.42a), is performed as part of the nonlinear optimization. Multiple shooting works by breaking up a trajectory into some number of segments and using the single shooting to solve for each segment. As the segments get shorter, the relationship between the decision variables and the objective function and constraints becomes more linear. The shooting intervals are then connected to create the full-time horizon, by enforcing constraints on the shooting gaps between intervals.

For this problem, we opted to use direct multiple shooting for several reasons. Comparing multiple shooting with direct collocation, they both offer the same stability in terms of optimization, however, multiple shooting offers more flexibility in terms of the integrator used, and can cope even with strong nonlinearity [121]. Comparing single shooting to direct multiple shooting the single shooting problem has much fewer decision variables, however, the problem often becomes very dense, and hence increases the computation time, single shooting is also more unstable, as propagating the gradients through a long time horizon often cause them to become very small (vanish) or very large (explode), and hence the optimization steps may be oscillatory and unstable.

The OCP in (4.39) is transformed into a nonlinear programming problem (NLP) and simulated using CasADi toolkit [122]. Furthermore, a Interior Point Optimizer (IPOPT) [123] is used to solve the NLP, using up to 2000 iterations, a tolerance of 10^{-6}. The NMPC is running at time samples of $h = 0.01$s and the prediction horizon N set to 20. The weighting matrices for the optimal cost functional were defined as follows: $Q = diag([150, 150, 100, 19, 19, 10, 1, 2, 2, 2, 2, 0.2, 0.2, 0.05])$, $H = 10 \times Q$, $R = 10^{-4} \times I_4$. The initial states and the reference trajectory are chosen similarly to the section 3.3.1. The constraints on inputs and states are taken as follows

$$\mathcal{X} := \begin{cases} x_1 \in [-2, +2] \\ x_2 \in \,]-\infty, +\infty[\\ x_3 \in [-2, +2] \\ x_4 \in \,]-\infty, +\infty[\\ x_5 \in \,]-\infty, +\infty[\\ x_6 \in \,]-\infty, +\infty[\\ x_7 \in [-deg2rad(10), +deg2rad(10)] \\ x_8 \in \,]-\infty, +\infty[\\ x_9 \in [-deg2rad(15), deg2rad(35)] \\ x_{10} \in \,]-\infty, +\infty[\\ x_{11} \in [-\pi, \pi] \\ x_{12} \in \,]-\infty, +\infty[\end{cases}, \mathcal{U} := \begin{cases} u_1 \in [0, 9.3585] \\ u_2 \in [-0.5849, 0.5849] \\ u_3 \in [-0.5849, 0.5849] \\ u_4 \in [-0.0507, 0.0507] \end{cases} \tag{3.46}$$

Figure 3.8 presents the tracking responses of the translation coordinates (x,y, and z) and the rotation coordinates (ϕ, θ, and ψ) of NMPC. Figure 3.9 depicts the deduced control signals from the optimization of the constrained objective function (4.39). We notice that all control signals are within the saturation limits.

Many factors influence the NMPC algorithm regarding system performance, such as the weighting matrices Q and R, the prediction horizon N, and the sampling time. Figure 3.10 shows the performance of the closed-loop system for NMPC with different prediction horizons. The sampling times and tunning matrices, i.e., Q and R of the NMPC for different prediction horizons N are similar. From Figure 3.10(a), we notice that the position error stabilized faster with a larger number of the prediction horizon. For different prediction horizons, the computational time increases as the prediction horizon increases, Figure 3.10(b). The effect of different values of prediction horizons on the cost function is shown in Figure 3.10(c); the more the value of the prediction horizon increases more the cost function goes faster to the vicinity of zero. Consequently, choosing $N = 20$ was good to compromise between minimum tracking error and computational time.

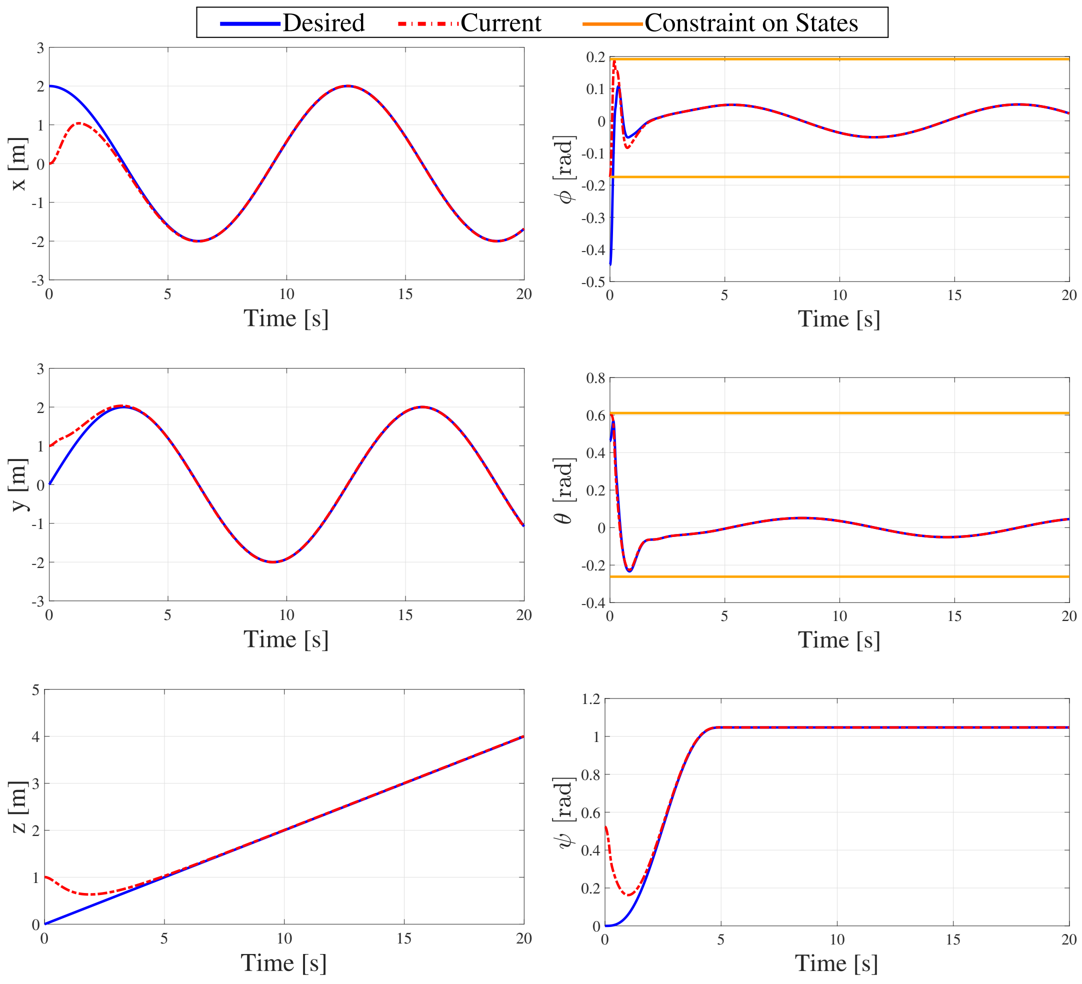


Figure 3.8: Helix trajectory tracking results with the NMPC approach.

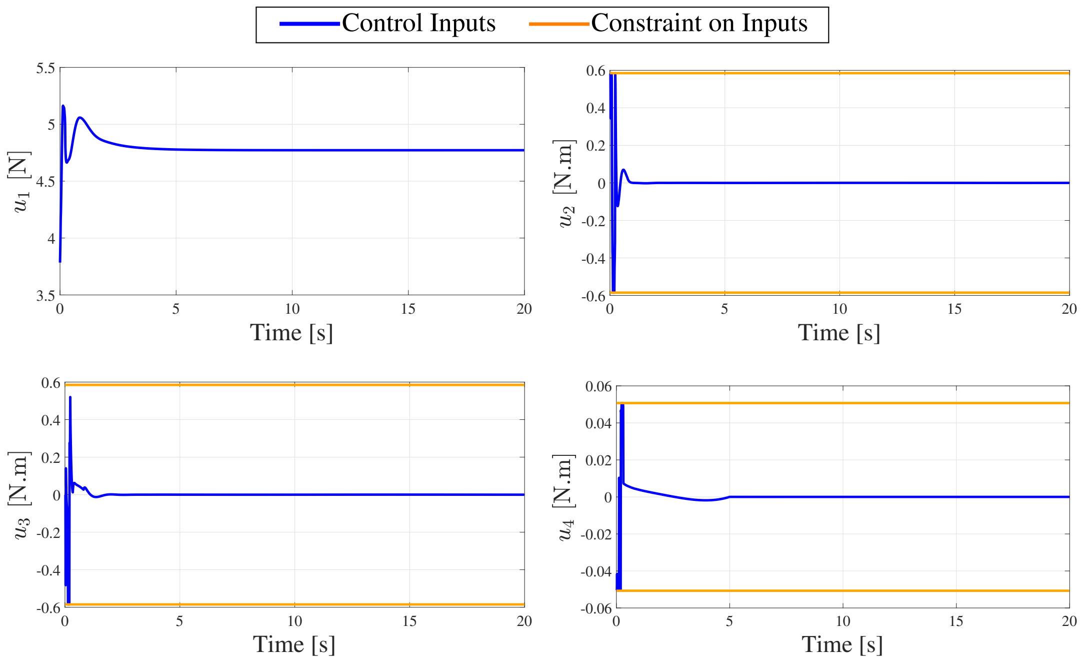


Figure 3.9: The control inputs with the NMPC approach.

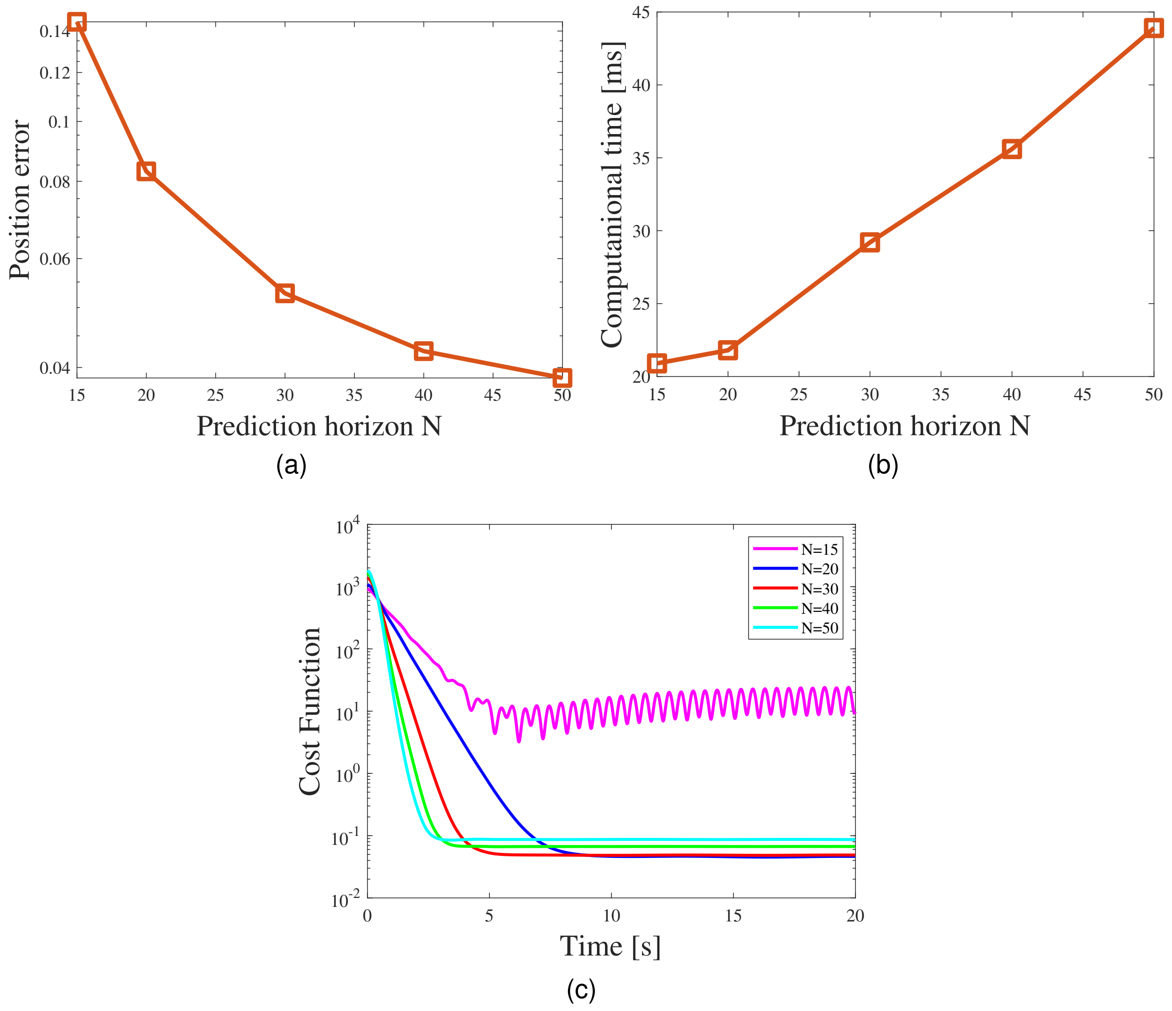


Figure 3.10: Effect of prediction horizon on: (a) Position error, (b) Computational time, (c) Cost function.

3.6 Robustness and Performance Analysis

This section treats the robustness and performance of the proposed controller. It begins with an analysis of an IBC and SMC. It is shown that seemingly reasonable design choices. Then, different metrics are used to compare IBC, SMC, and NMPC.

3.6.1 Numerical Results of IBC and SMC

In this section, numerical simulations are performed to validate and teste the effectiveness of the proposed controllers. The non-linear model from equation (2.25) is used to carry out all simulations. The sampling period of the simulation is set to 10ms. The simulations are performed using Matlab/Simulink with the parameters of quadrotor given in Table 2.1. The initial position and attitude angle values of the quadrotor for simulation tests are set to zero. The dynamic controllers' parameters that stabilize are listed the same as in subsections 3.3.1 and 3.4.3.

The evaluation of the designed controllers is done under disturbances, model uncertainties and wind effect conditions. The quadrotor has to follow the line, helix and spiral trajectories. The equations of the desired trajectories used in the simulation are defined in Appendix A.

To verify the controllers' robustness, it is assumed that there is a payload weighing 50% of the quadrotor's mass and uncertainties 50% on the values of the inertias in the time interval [5s 10s]. External disturbances are also injected into the inputs as steps: 2.5 [N] along x-, y-, and z-directions from the time 15s to 18s, 1 [N.m] in ϕ-direction from the time 25s to 28s, 1 [N.m] in θ-direction from the time 35s to 38s, and 0.05 [N.m] along ψ-direction from the time 45s to 48s. In order to further assess the effectiveness of the controller scheme; the Simulink Dryden Wind Turbulence Model is used to operate the quadrotor under the effect of the wind and generated stochastic velocities disturbance added to the dynamics of the quadrotor. This has a great influence on the dynamics of the aircraft, in particular, the linear, and angular velocities. The angular and linear velocity of the wind gusts are shown in Figure 3.11.

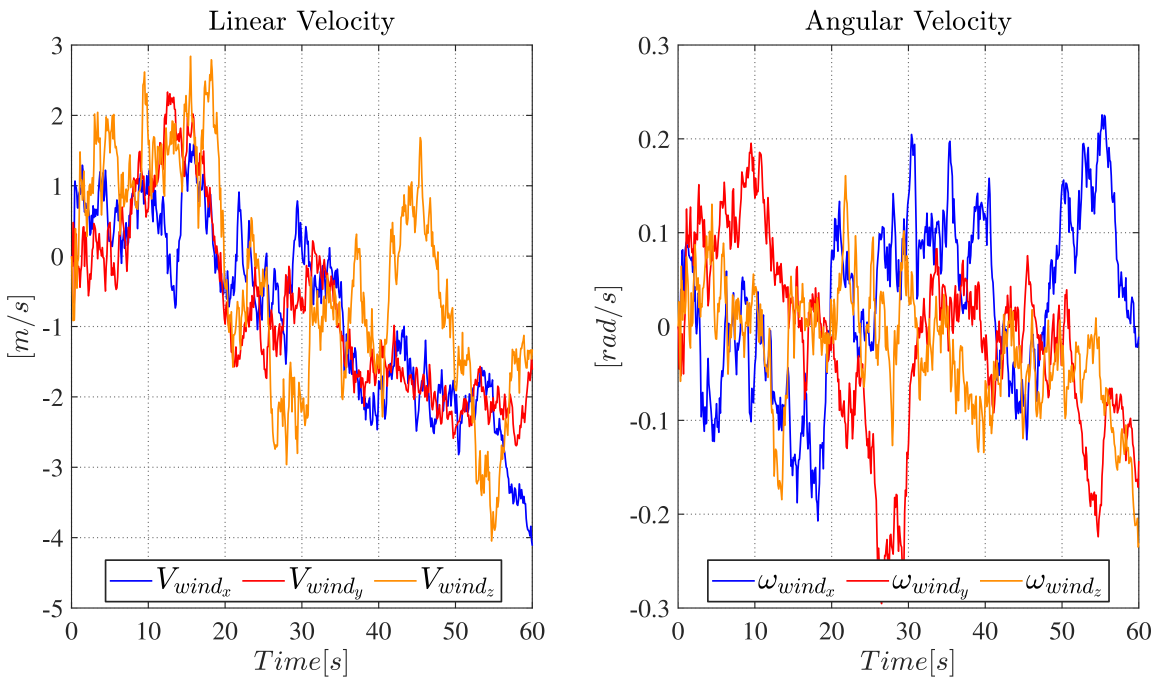


Figure 3.11: The linear and the angular velocities of the wind gusts effect applied to quadrotor.

Figure 3.12 shows the results of tracking the helix trajectory under disturbances, model uncertainties, and wind effects. Both controllers achieve successful tracking. Integral backstepping control has a significantly larger error with respect to the sliding mode control. The comparison of the tracking error is depicted in Figure 3.13. The tracking performance of each controller and with different trajectories of the quadrotor in 3D is shown in Figure 3.14. Figure 3.15 shows the control effort of each command, it can be seen that the sliding mode control has a high effort compared to the effort of integral the backstepping. Under wind gusts, disturbances and unmodeled dynamics both Integral backstepping and sliding mode performed well with an acceptable amount of tracking errors. After increasing the mass (from the time 5s to the 10s), in Figure 3.12 and Figure 3.13, we notice that there are variations in the behavior for the simulation of the SMC and IBC. We also notice in Figure 3.15 an increase in the thrust force (u_1) from 4.8N to 7.4N due to the increase in mass.

It can be seen also from the figures that the sliding mode significantly outperforms the integral backstepping with regard to transient performance. In contrast, the sliding mode has a large control effort and more chattering with respect to the integral backstepping in the roll and pitch torques.

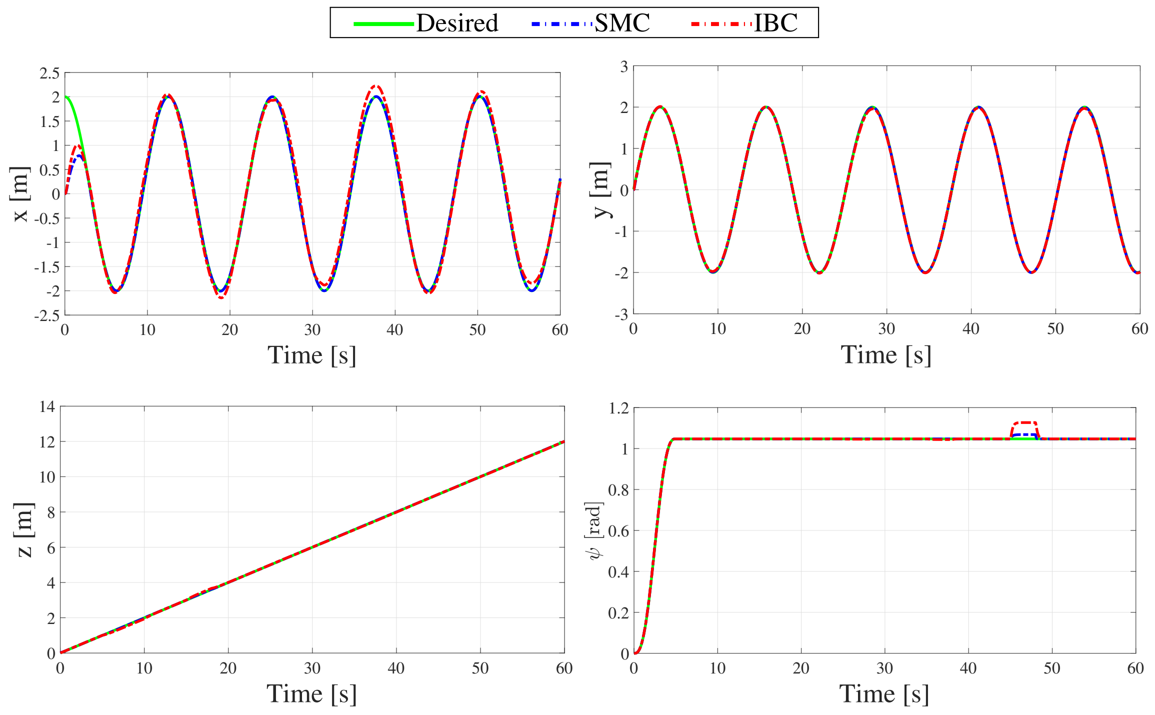


Figure 3.12: Tracking helix trajectory under parameter uncertainties, wind gusts effect and disturbances.

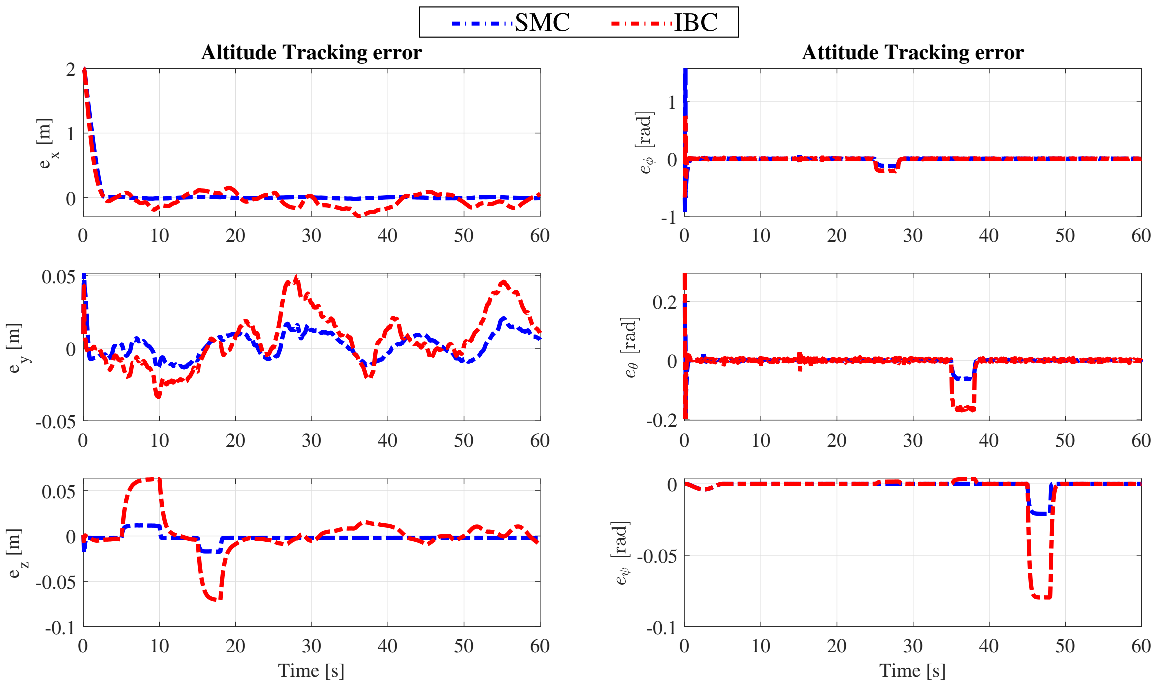


Figure 3.13: Tracking errors in altitude and attitude of helix trajectory with IBC and SMC under parameter uncertainties, wind gusts effect and disturbances.

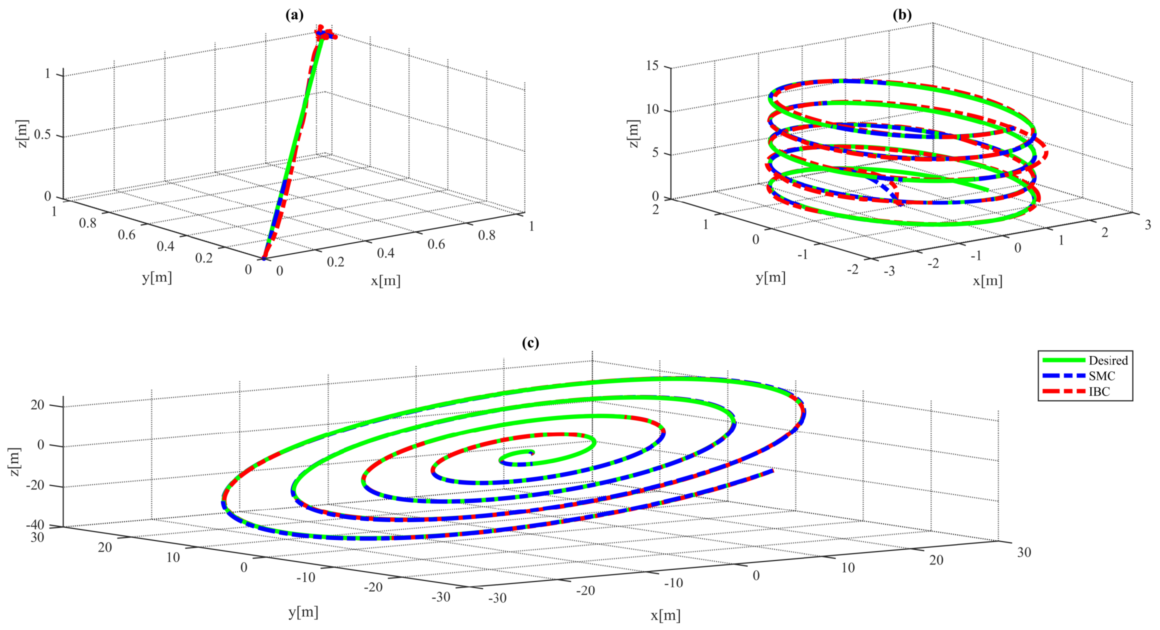


Figure 3.14: (a) Straight line, (b) Helix (c) Spiral tracking trajectory with IBC and SMC under parameter uncertainties, wind gusts effect and disturbances.

Furthermore, the performance of each controller is evaluated using the *Root Mean Square Error* (RMSE). Table 3.2 shows the comparison between the desired trajectory and achieved trajectory values with respect to time for straight line, helix, and spiral trajectory with disturbance, model uncertainties, and wind gusts. From

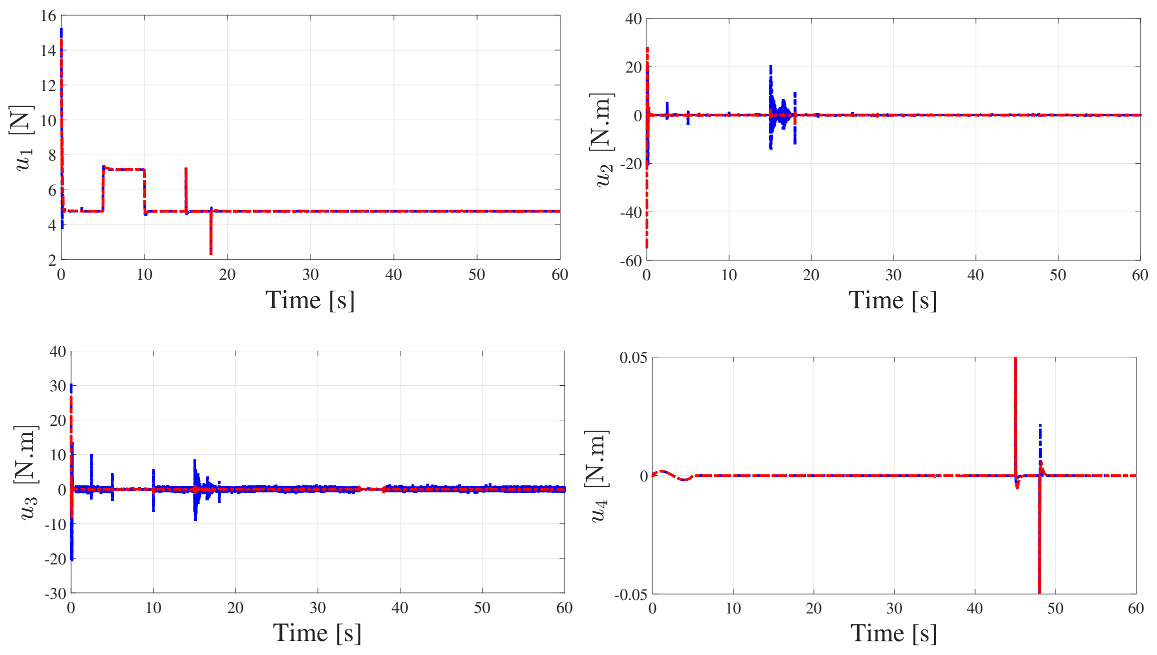


Figure 3.15: Controller inputs of the IBC (red) and SMC (blue) under parameter uncertainties, wind gusts effect, and disturbances.

this table, it is found that the RMSE is less than 5% for the three trajectories for each controller, which is considered tolerable. It can be seen that the robustness to disturbance and model uncertainties of SMC is better than IBS. SMC has more accuracy and robustness than IBC in tracking the three desired paths.

Table 3.2: RMSE values for straight line, helix and spiral trajectories with disturbances, model uncertainties, and wind gusts effect.

Trajectory	**Controller**	**RMSE values with disturbance and model uncertainties**			
		$x\%$	$y\%$	$z\%$	$\psi\%$
Straight line	SMC	**0.2046**	**0.8615**	**0.4035**	**0.5882**
	IBC	1.1736	1.9060	1.8607	1.7171
Helix	SMC	**23.1752**	**0.9165**	**0.5326**	**0.4519**
	IBC	23.6137	1.9755	2.3083	1.7172
Spiral	SMC	**3.1073**	**3.1405**	**3.0838**	**0.4519**
	IBC	3.5068	4.6514	3.0092	1.7171

3.6.2 Comparaison Between IBC, SMC and NMPC

With the aim of carrying out a comparative study between the SMC, IBC, and NMPC controllers, the following performance metrics are used:

1. The control signal energy (CSE)

$$CSE = \sum_{k=1}^{n} u^2[k] \tag{3.47}$$

2. The control effort energy (CEE)

$$CEE = \sum_{k=1}^{n} (u[k] - u[k-1])^2 \tag{3.48}$$

3. The average computational time

4. The position rout mean square error (PRMSE)

$$PRMSE(cm) = \sqrt{\frac{\sum_{k=1}^{n} \left((x_d[k] - x[k])^2 + (y_d[k] - y[k])^2 + (z_d[k] - z[k])^2\right)}{n}} \tag{3.49}$$

5. The attitude rout mean square error (ARMSE)

$$ARMSE(^\circ) = \sqrt{\frac{\sum_{k=1}^{n} \left((\phi_d[k] - \phi[k])^2 + (\theta_d[k] - \theta[k])^2 + (\psi_d[k] - \psi[k])^2\right)}{n}} \tag{3.50}$$

Results of the comparative study of the three commands are shown in Table 3.3. Moreover, the disturbances, wind turbulence, and uncertainties used in this case are mentioned in this paper [124]. Regarding the criteria that indicates the amount of energy consumed by the controllers, it can be seen that the smallest CSE values with respect to u_1, u_2, u_3, and u_4 is determined based on the NMPC approach for both cases (with or without disturbances) compared to SMC and IBC. Besides, NMPC provides the lowest fluctuations and smoothness at control inputs which are revealed by the CEE values. Nevertheless, the NMPC shows a high computational burden compared to other controllers. As a result of the chattering phenomena, the SMC approach has a high effort (CEE and CSE values) compared to the other controllers. For the three controllers without disturbances, it can be noticed that the PRMSE and ARMSE values are less than $0.06cm$ and $0.09deg$ respectively which are considered tolerable. While, in the presence of disturbances, the SMC outperforms the IBC and NMPC showing good tracking ability in terms of ARMSE and PRMSE.

Table 3.3: A comparison between SMC, IBC, and NMPC tracking of straight-line trajectory. It is done with CSE and CEE of $\{u_1, u_2, u_3, u_4\}^\top$, Average time, PRMS, and ARMS criterions

	Controller	CSE	CEE	Average Time[ms]	PRMSE [cm]	ARMSE [deg]
without disturbances	IBC	4.5407e+04	2.2729e-04	0.8528	**0.0051**	0.0718
		0.0021	8.9765e-05			
		0.0069	2.3942e-04			
		4.7647e-04	5.6564e-08			
	SMC	4.5407e+04	2.3160e-04	**0.6636**	0.0151	0.0459
		0.0074	9.2390e-04			
		0.0457	0.0022			
		5.2894e-04	5.7366e-08			
	NMPC	4.5407e+04	**2.1768e-04**	16	0.0584	**0.0047**
		4.4816e-04	**1.4897e-05**			
		5.7965e-04	**1.5780e-05**			
		4.7172e-04	**5.5041e-08**			
with disturbances	IBC	6.0236e+04	2.9813	1.6	0.8133	1.9414
		13.2886	1.1165			
		8.1007	0.6525			
		3.3047	0.0016			
	SMC	6.0241e+04	7.9003	**1.5**	**0.1470**	**0.4943**
		30.8017	5.1253			
		21.1184	3.8687			
		3.3084	0.0021			
	NMPC	**6.0191e+04**	**0.2527**	26.7	4.6709	1.6334
		0.8102	**9.6626e-04**			
		0.8088	**0.0010**			
		3.2275	**0.0008**			

3.7 Conclusion

An integral backstepping and a sliding mode controller are presented and utilized to control the quadrotor to track different desired trajectories. Furthermore, a comparison between these two approaches has been conducted using RMS error. The sliding mode controller achieved has better robustness against external disturbances, unmodeled dynamics and wind gust turbulence with high control effort compared to the integral backstepping controller. Although, it should be noted that integral backstepping control is a powerful nonlinear control approach and has a good performance tracking under nominal condition, which preserves the stability of the system under external disturbances and also, has a lower control effort compared to sliding mode control. After that, Another Comparative study between the SMC, IBC, and NMPC controllers presented with the different performance metrics that were used. The SMC controller exhibited the robustness against disturbances, while the NMPC has shown lower control effort.

Chapter 4 DESIGN OF HYBRID NONLINEAR CONTROL APPROACHES

4.1 Introduction

Many researchers have combined the SMC and MPC. In [125], the surface parameters of sliding mode control have been determined and updated using the nonlinear model predictive control. In [126], the sliding mode predictive control has been used to control the boiler-turbine system that deals with uncertainties and system constraints. The adopted control strategy was based on the dual-mode law that is constructed of two-part: the discrete sliding mode control law where the sliding surface was in the terminal sliding region, and the receding horizon optimization law where the sliding surface was out from the terminal sliding region. A comparative study between DSMC with predictive sliding function PSF and PSMC is done in [127]. These strategies are applied to the linearised isothermal Van de Vussen systems. The simulation results have shown that both of the combination controllers have outperformed the NMPC and SMC. As well as, the DSMC with PSF has more ability to eliminate the chattering compared to the PSMC. While this latter has strong robustness to disturbances.

In this chapter, two nonlinear approaches are described and synthesized. The first one is predictive sliding mode control which is a combination of DSMC and NMPC. We start the synthesis of the discrete sliding mode control DSMC. Then, the combined Sliding mode with predictive control, i.e. PSMC is designed [128]. A comparative study is done with different scenarios to assess the robustness of the proposed controller is performed. The second control approach concerns the quaternion-based NMPC for quadrotor tracking trajectory and avoiding obstacles

[129]. This method consists of a proportional derivative controller to calculate the desired quaternion and NMPC for the tracking trajectory.

4.2 Design of Discrete Sliding Mode Control DSMC approach

The objective of the SMC law is to constrain the system state trajectory (2.31) to reache and then to maintain it on the sliding surface even in the presence of uncertainties in the system.

Let a second-order Slotine surface [130] is chosen as:

$$s_i[k] = e_{2i}[k] + \lambda_i e_{2i-1}[k], \ \ i = 1, 2, ..., 6 \tag{4.1}$$

where the $\lambda_i \in R^+$ are tunable parameters, $e[k]$ are the tracking error which is the difference between the actual state $x[k]$ and the desired one $r[k]$ and is defined as follows:

$$e[k] = \begin{cases} e_{2i-1}[k] = r_{2i-1}[k] - x_{2i-1}[k] \\ e_{2i}[k] = r_{2i}[k] - x_{2i}[k] \end{cases} \tag{4.2}$$

where $r[k]$ is the discrete-time of the desired trajectory $r(t) = \left[x_d, \dot{x}_d, y_d, \dot{y}_d, z_d, \dot{z}_d, \phi_d, \dot{\phi}_d, \theta_d, \dot{\theta}_d, \psi_d, \dot{\psi}_d\right]$. $\{x_d, y_d, z_d, \psi_d\}$ and its derivatives are provided from the trajectory generator, while $\{\phi_d, \theta_d\}$ and its derivative can be deduced from the position controller.

The purpose of the control is to force the system to evolve on the sliding surface and prevent it from getting out of it, i.e.:

$$S = \{e[k] \mid s_i\left(e[k]\right) = 0, i = 1, 2, ..., 6\} \tag{4.3}$$

We introduce the virtual command $v_i[k]$ in such a way that $x_{2i}[k+1] = v_i[k]$, which gives us:

$$v_i[k] = x_{2i}[k] + h\left(f_i(x[k]) + g_i(x[k])u[k]\right) \tag{4.4}$$

The dynamic of the surface (4.1) is:

$$\begin{aligned} s_i[k+1] &= s_i[k] + \left(e_{2i}[k+1] + \lambda_i e_{2i-1}[k+1]\right) \\ &= s_i[k] + \left(r_{2i}[k+1] - v_i[k] + \lambda_i e_{2i-1}[k+1]\right) \end{aligned} \tag{4.5}$$

Motivated by the reaching law presented by Weibing Gao et *al.* in [131], the following exponential reaching law is adopted:

$$s_i[k+1] = (1 - h\sigma_i)\, s_i[k] - h\mu_i sign\,(s_i[k])\,, \quad i = 1, 2, ..., 6 \tag{4.6}$$

where $sign()$ is the signum function, σ_i and μ_i are tuning parameters and satisfying $0 \leqslant h\sigma_i < 1$ and $\mu_i > 0$.

In equation (4.6), the sliding manifolds are bounded as:

$$|s_i[k]| \leqslant \Delta_i, \quad i = 1, \ldots, 6 \tag{4.7}$$

where Δ_i is called quasi-sliding mode band width and is:

$$\Delta_i = \frac{h\mu_i}{1 - h\sigma_i}, \quad i = 1, \ldots, 6 \tag{4.8}$$

For equation (4.6), the system states will converge to the desired values only if $s_i[k] \to 0$. On the other hand, sliding manifolds will approach zero only if $h\mu_i \to 0$. As we know, $h\mu_i$ is nonzero, then the tracking errors will not converge to the origin. However, it will approach near the origin if h is given a small value. Moreover, there is no guarantee for inequality (4.7) in the presence of model uncertainty and external disturbance.

By equating (4.6) and (4.5), the following virtual commands signal are obtained:

$$\begin{aligned} v_i[k] = {} & \sigma_i s_i[k] + \mu_i sign\,(s_i[k]) + \lambda_i e_{2i-1}[k+1] + r_{2i}[k+1], \\ & i = 1, 2, ..., 6 \end{aligned} \tag{4.9}$$

By applying the properties of the rotation matrix [132], we determine the real commands as follow:

$$\begin{aligned} u_1[k] &= m\sqrt{(V_1[k])^2 + (V_2[k])^2 + (V_3[k])^2} \\ u_2[k] &= \frac{V_4[k]}{g_4([x_k])}, u_3[k] = \frac{V_5[k]}{g_5([x_k])}, u_4[k] = \frac{V_6[k]}{g_6([x_k])} \end{aligned} \tag{4.10}$$

where $V_i[k] = v_i[k] - f_i(x[k])$, $i = 1, 2, ..., 6$. The desired roll and pitch angles ϕ_d and

θ_d are generated from equation:

$$\begin{cases} m\left(V_1[k]c_\psi + V_2[k]s_\psi\right) = s_\theta u_1[k] \\ m\left(V_1[k]s_\psi + V_2[k]c_\psi\right) = s_\phi c_\theta u_1[k] \\ mV_3[k] = c_\phi c_\theta u_1[k] \end{cases} \tag{4.11}$$

We draw from equation (4.11):

$$\begin{cases} \phi_d = arctan\left(\dfrac{V_1[k]s_{\psi_d} - V_2[k]c_{\psi_d}}{V_3[k]}\right) \\ \theta_d = arctan\left(\dfrac{V_1[k]c_{\psi_d} + V_2[k]s_{\psi_d}}{\sqrt{(V_1[k]s_{\psi_d} - V_2[k]c_{\psi_d})^2 + V_3[k]^2}}\right) \end{cases} \tag{4.12}$$

To alleviate the chattering problem caused by the discontinuous sign function. we replace this latter by a pseudo-sign function which is defined as follows:

$$psign(x, \eta) = \frac{x}{|x| + \eta} \tag{4.13}$$

where $0 < \eta << 1$ has been chosen equal to 0.05.

4.2.1 Stability Analysis

To evaluate the stability condition of DSMC, direct Lyapunov stability analysis is used. The positive definite Lyapunov functions are chosen as

$$V_i[k] = |s_i[k]|, \quad i = 1, \dots, 6 \tag{4.14}$$

The Lyapunov function at $[k+1]$-th instant is

$$V_i[k+1] = |s_i[k+1]|, \quad i = 1, \dots, 6 \tag{4.15}$$

which gives

$$\Delta V_i[k] = |s_i[k+1]| - |s_i[k]|, \quad i = 1, \dots, 6 \tag{4.16}$$

For the system to be stable, Eq. (4.16) must be a negative definite function. This results into

$$|s_i[k+1]| < |s_i[k]|, \tag{4.17}$$

which is known as the Sarpturk condition [133], and can be represented as:

$$The\ existance\ condition: \quad (s_i[k+1] - s_i[k])\, sign(s[k]) < 0 \tag{4.18a}$$

$$The\ reaching\ condition: \quad (s_i[k+1] + s_i[k])\, sign(s[k]) > 0 \tag{4.18b}$$

that are known as Sarpturk's reaching laws.

By substituting the control law (4.9) in equation (2.31), the error dynamics will be in the following form:

$$e_{2i-1}[k+1] = e_{2i-1}[k] + h\, e_{2i}[k] \tag{4.19a}$$

$$\begin{aligned} e_{2i}[k+1] = e_{2i}[k] + hr_{2i}[k+1] - h\sigma_i s_i[k] - h\mu_i sign(s_i[k]) \\ - (1+h)e_{2i}[k] - e_{2i-1}[k] - h\, r_{2i}[k+1] + h\, d[k] - d[k] \end{aligned} \tag{4.19b}$$

Substituting (4.19a) and (4.19b) into(4.1) yields

$$s_i[k+1] = s_i[k] - h\left(\sigma_i s_i[k] + \mu_i sign(s_i[k]) + d_i[k]\right) \tag{4.20}$$

From (4.18) and (4.20), we obtain the following bounds that achieved the sliding mode function $s(k)$ will definitely converge to a vicinity of zero:

$$\begin{cases} \|d_i[k]\| \leq d_{max,i} \\ \mu_i \geq \dfrac{d_{max,i}}{1 - h\sigma_i} \\ 0 < h\sigma_i < 1 \end{cases} \tag{4.21}$$

4.3 Non-linear Predictive Sliding Mode Control PSMC

The non-linear PSMC control law is used in this work for the quad-rotor trajectory tracking problems. This hybrid approach is based on the NMPC and the DSMC in order to provide the best trade-off between minimum effort energy control, tracking trajectory, and rejection of disturbance. The main objective of PSMC is to generate the optimum control input where the PSMC concept [134] is illustrated in Figure 4.1.The sliding function trajectory to be tracked by the PSMC should show the attraction to the sliding surface $s_i = 0$, as well as its discontinuous switching behavior when the system states are already on the surface.

Firstly, at each sampling time k, the DSMC part calculates the reference sliding surfaces $s_{ref}[k+j|k], j = 0, \dots, N$ over the horizon N, invoking the equations (4.1) and (4.6) yields:

$$\begin{aligned} s_{ref}[j] &= s[j] \\ s_{ref}[j+1] &= (1-h\sigma)s_{ref}[j] - h\mu s_{ref}[j] \\ &\vdots \\ s_{ref}[j+N] &= (1-h\sigma)s_{ref}[j+N-1] - h\mu s_{ref}[j+N-1] \end{aligned} \tag{4.22}$$

Then, the NMPC computes the control sequence $u[k|k+j]$ using the plant-model. The computations optimize the tracking of the predicted sliding functions $s_{pred}[k+j|k], j = 0, \dots, N$ while minimizing the control effort. In the end, the first element of the calculated control sequence is applied to the quadrotor model Figure 4.2.

The mathematical formulation of the non-linear PSMC can be written as follows:

$$\min_{u[k+j|k],x[k+j|k],s[k+j|k]} \quad \sum_{j=0}^{N-1} J_r\left(x[j], s[j], u[j], r[j]\right) + J_t(x[N], s[N]) \tag{4.23a}$$

$$\textbf{s.t}: x[0] - x_0 = 0, \tag{4.23b}$$

$$x[j+1] = x[j] + h\boldsymbol{f}(x[j], u[j]) \tag{4.23c}$$

$$s_{pred_i}[j+1] = s_{pred_i}[j] + e_{2i}[j+1] + \lambda_i e_{2i-1}[j+1] \tag{4.23d}$$

$$x[j] \in \mathcal{X}, \qquad j = 0, \dots, N \tag{4.23e}$$

$$u[j] \in \mathcal{U}, \qquad j = 0, \dots, N-1 \tag{4.23f}$$

$$s_{pred}[j] \in \mathcal{S}, \qquad j = 0, \dots, N \tag{4.23g}$$

where $J_r(.)$ is the running cost function of PSMC, and is defined as:

$$J_r\left(x, s, u, r\right) = \|u_{ref}[j] - u[j]\|^2_{\boldsymbol{R}} + \|s_{pred}[j] - s_{ref}[j]\|^2_{\boldsymbol{\lambda}} \tag{4.24}$$

and, $J_t(.) = \|s_{pred}[N] - s_{ref}[N]\|^2_{\boldsymbol{\eta}}$ is the terminal cost function, $\boldsymbol{\lambda}, \boldsymbol{\eta} \in \mathbb{R}^{6\times 6}$ are a positive-definite tuning matrices which penalize the tracking surface functions. $\mathcal{X} \subseteq \mathbb{R}^{12}$,$\mathcal{U} \subseteq \mathbb{R}^4$ are the same specified in 3.5, $\mathcal{S}$ is the set of terminal sliding region that is defined as [131, 135]:

$$\mathcal{S} = \bigcup_{i=1}^{6} \mathcal{S}_i, \qquad \mathcal{S}_i = \{x \mid |s_i(x)| \le \Delta_i, x \in \mathcal{X}, u \in \mathcal{U}, \Delta_i = h\mu_i\} \tag{4.25}$$

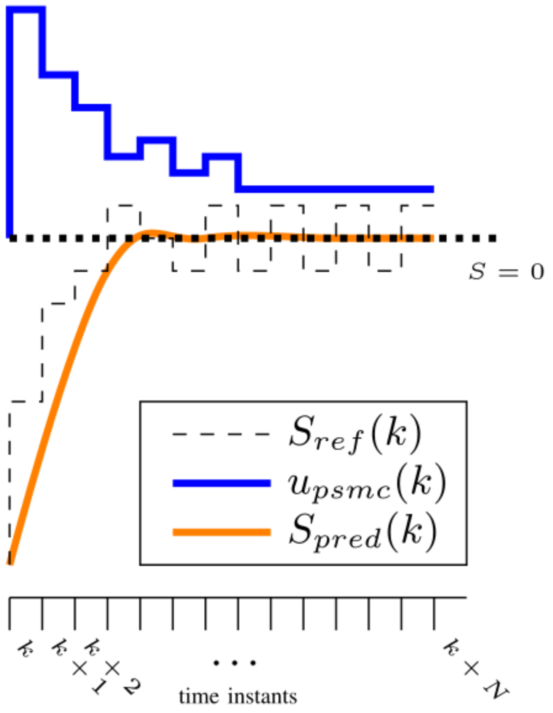


Figure 4.1: Concept of predictive sliding mode control strategy.

4.3.1 Robustness Assessment of the Designed Controllers

Simulation results using MATLAB/SIMULINK® are developed in this section to corroborate the proposed controllers' effectiveness. The quad-rotor dynamic model from equations (2.19) is used to perform all simulations. The sampling period of the simulation is set to $h = 10ms$, and the initial conditions are set to zero except in the case 4.3.1.1. The quad-rotor and controllers parameters are given in Tables 2.1 and 4.1, respectively.

In addition, the constraints on inputs, states and sliding mode band are taken as

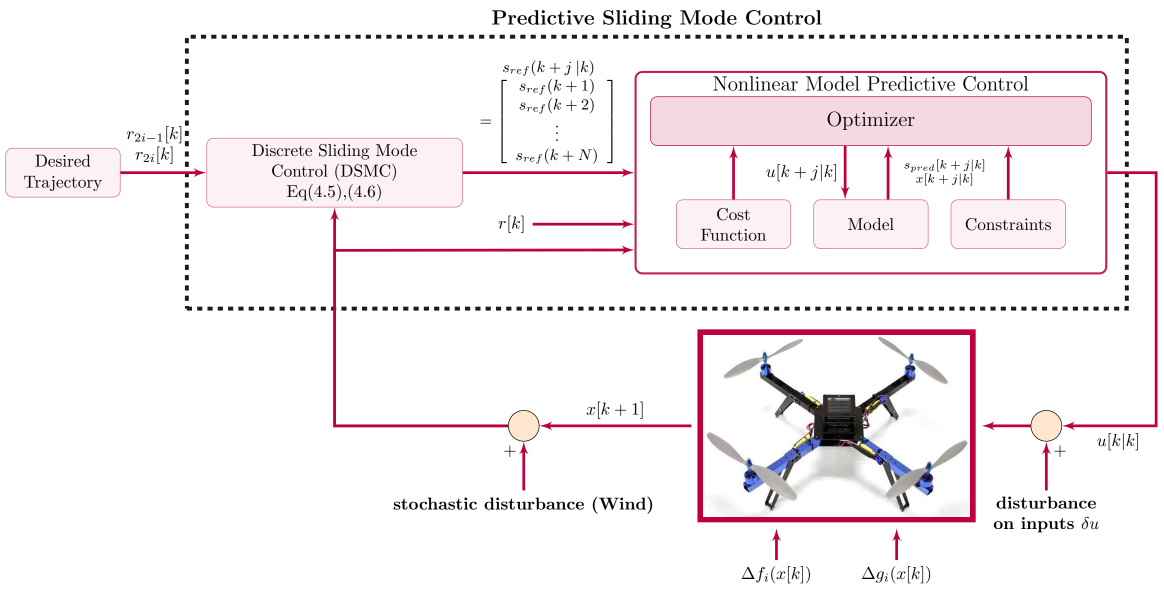

Predictive Sliding Mode Control
Desired Trajectory
$r_{2i-1}[k]$
$r_{2i}[k]$
Discrete Sliding Mode Control (DSMC) Eq(4.5),(4.6)
$s_{ref}(k+j\,|k) = [s_{ref}(k+1), s_{ref}(k+2), \vdots, s_{ref}(k+N)]$
$r[k]$
Nonlinear Model Predictive Control
Optimizer
$u[k+j|k]$
$s_{pred}[k+j|k]$
$x[k+j|k]$
Cost Function
Model
Constraints
$x[k+1]$
$u[k|k]$
stochastic disturbance (Wind)
disturbance on inputs δu
$\Delta f_i(x[k])$
$\Delta g_i(x[k])$

Figure 4.2: Block diagram of the predictive sliding mode control strategy.

follows:

$$\mathcal{X} := \begin{cases} x_1 \in \,]-\infty,+\infty[\\ \quad\vdots \\ x_6 \in \,]-\infty,+\infty[\\ x_7 \in \left[-\dfrac{\pi}{2},\dfrac{\pi}{2}\right] \\ x_8 \in \,]-\infty,+\infty[\\ x_9 \in \left[-\dfrac{\pi}{2},\dfrac{\pi}{2}\right] \\ x_{10} \in \,]-\infty,+\infty[\\ x_{11} \in [-\pi,\pi] \\ x_{12} \in \,]-\infty,+\infty[\end{cases}, \quad \mathcal{U} := \begin{cases} u_1 \in [0, 9.3585] \\ u_2 \in [-0.5849, 0.5849] \\ u_3 \in [-0.5849, 0.5849] \\ u_4 \in [-0.0507, 0.0507] \end{cases}, \quad \mathcal{S} := \begin{cases} |s_i| < h\mu_i \\ i = 1,\dots,6. \end{cases} \tag{4.26}$$

The OCP in (4.23) is transformed into a NLP and simulated using CasADi toolkit [122]. Furthermore, a Interior Point Optimizer (IPOPT) [123] is used to solve the NLP, using up to 2000 iterations, a tolerance of 10^{-6}, and the horizon prediction N set to 20.

Table 4.1: Controller Parameters

Controller	Symbol	Value
DSMC	λ_i	71 71 18.5 10 10 25
	μ_i	7.9 7.9 0.9 1.9 1.9 6.9
	σ_i	0.02 0.02 0.18 0.2 0.2 0.5
NMPC	Q	$diag\left(\left[\begin{array}{cccccccccccc} 0.5, & .05, & .5, & .05, & 60, & 20, & 20, & 3, & 20, & 3, & 65, & 53.5 \end{array}\right]\right)$
	H	$10 \times Q$
	R	$diag\left(\left[\begin{array}{cccc} 1, & 10^{-2}, & 10^{-2}, & 10^{-3} \end{array}\right]\right)$
PSMC	R	$diag\left(\left[\begin{array}{cccc} 1, & 10^{-2}, & 10^{-2}, & 10^{-3} \end{array}\right]\right)$
	λ_i	0.05 0.05 18.5 10 10 11
	μ	$diag\left(\left[\begin{array}{cccccc} 0.85, & 0.85, & 0.005, & 0.22, & 0.22, & 0.35 \end{array}\right]\right)$
	σ	$diag\left(\left[\begin{array}{cccccc} 1, & 1, & 1.5, & 0.55, & 0.55, & 1.85 \end{array}\right]\right)$

To demonstrate the effectiveness of the PSMC control, this latter is compared to the NMPC and DSMC controls with different following scenarios. All the reference trajectories used in these cases are defined in the appendix A.

4.3.1.1 Case 1: Nominal Performance comparison

The simulation is done here performed using nominal conditions, to track an inclined 8-shaped trajectory without any considering disturbances or parametric uncertainties, and with an initial condition different from the equilibrium point ($x_0 = [-15°, 0, 35°, 0, 40°, 0, 1, 0, -0.5, 0, 0.5, 0]^\top$). Simulation results in this case are presented in Figures 4.3, 4.4, and 4.5. As can be shown, all controllers achieve successful tracking. In contrast, the DSMC and PSMC exhibit a response time faster than NMPC, Figures 4.3 and 4.5. While in Figure 4.4, the DSMC has a large control effort exceeding the control limits for the yaw torque in comparison with PSMC and NMPC.

4.3.1.2 Case 2: Wind gusts rejection ability

In this case, the quadrotor is undergoing sudden wind gusts as external disturbances in the interval $[10, 30]s$. The Dryden Wind Turbulence Model [136] provided by the “Aerospace Blockset” toolbox of Simulink is used to generate a stochastic velocities disturbance added to the dynamics of the quad-rotor. This has a great

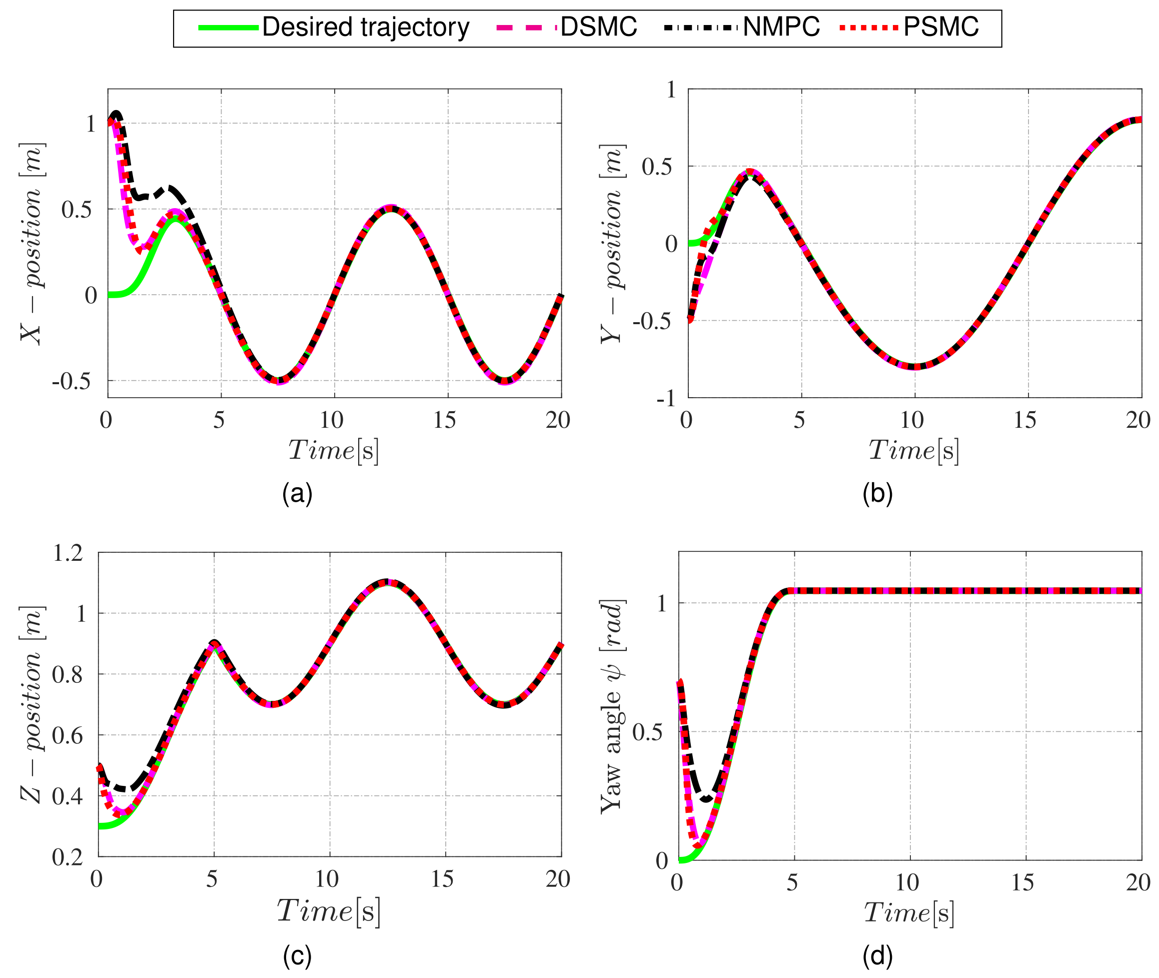
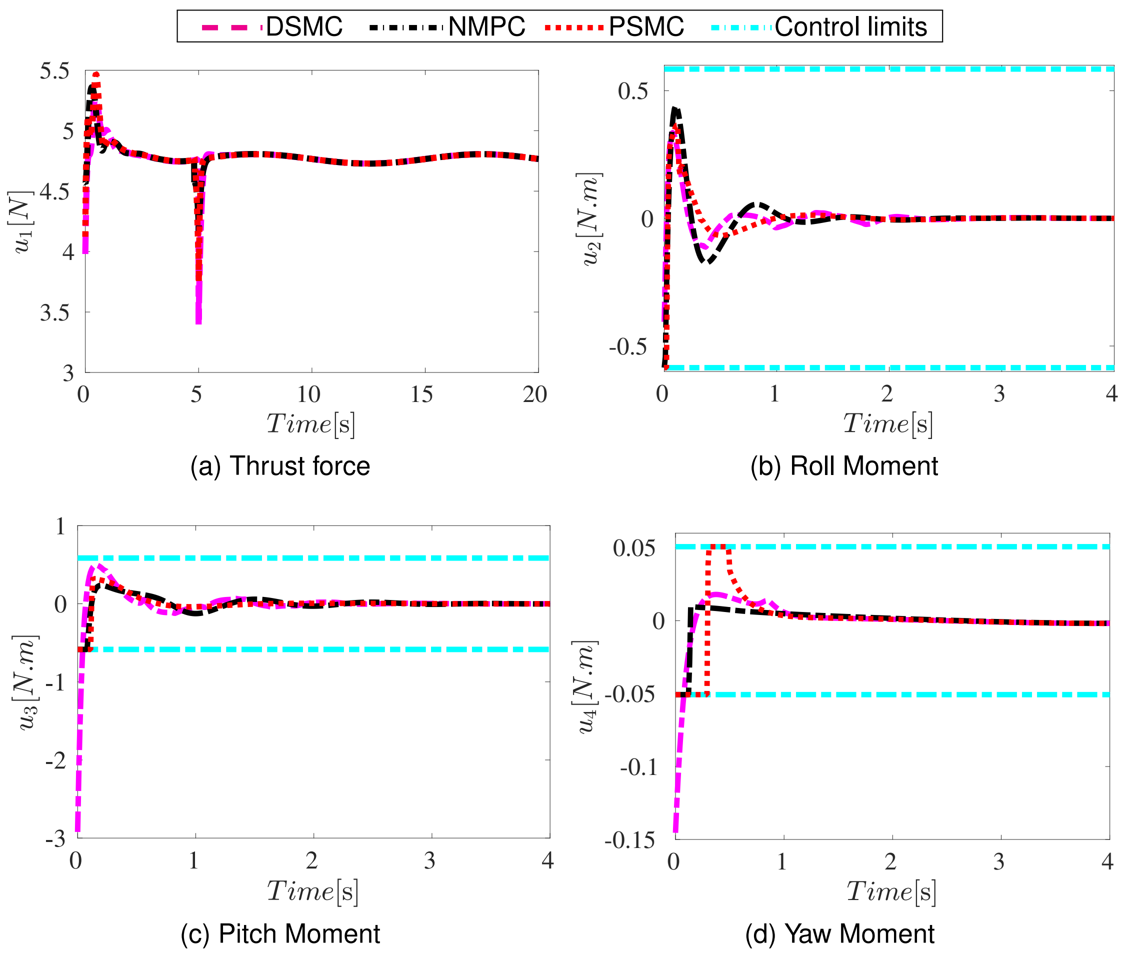


Figure 4.3: Results of tracking an inclined 8-shaped trajectory with the nominal condition.

influence on the dynamics of the aircraft, in particular, the linear, and angular velocities. Figures 4.6 illustrates the linear and angular velocity components of the applied wind turbulence. Figure 4.7 depicts the quadrotor response to track the square trajectory against wind gusts effect with the three controllers. The NMPC fails to track the reference trajectory in the presence of wind, particularly in $x-$ and $y-$positions have a noticeable error that reaches 0.4 m (Figures 4.7 (d) and 4.7(f)). PSMC and DSMC exhibit strong tracking ability against wind gusts and outperform the NMPC.

Figure 4.8 shows the control inputs in case of wind gusts for the three controllers. As for the control effort, the NMPC has a minimum effort even in the presence of wind. Although, the DSMC's good tracking, it has a large control effort; more chattering phenomena and exceeds the control limits Figures 4.8(b)-4.8(c). While, the PSMC control effort remains within a limits control, and has minimum chattering



(a) Thrust force (b) Roll Moment

(c) Pitch Moment (d) Yaw Moment

Figure 4.4: The control effort applied to the quadrotor with nominal condition.

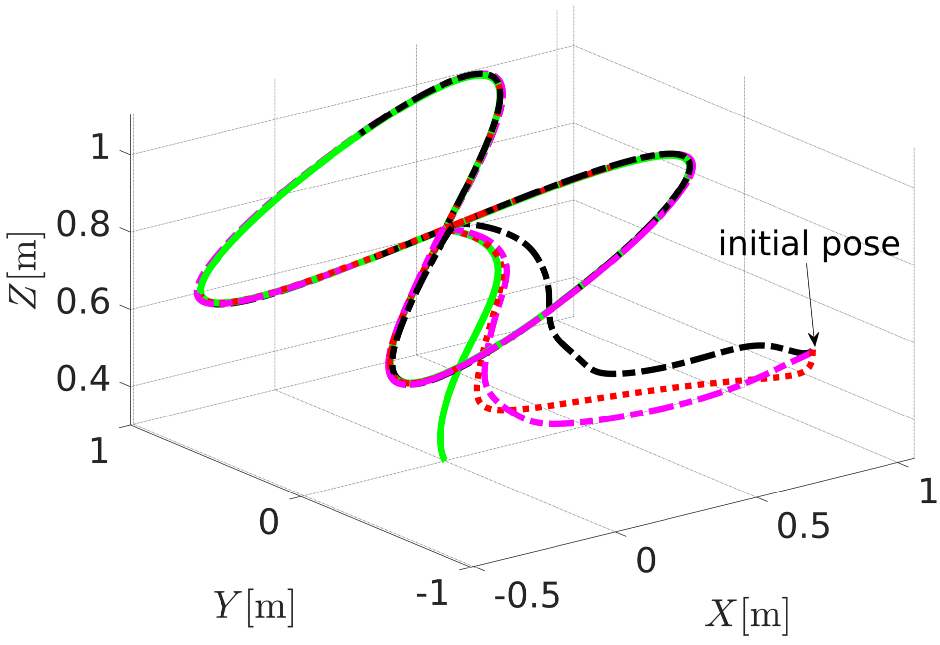


Figure 4.5: 3D tracking trajectory in the nominal case.

compared to DSMC.

On the other hand, the DSMC shows some interesting robustness properties, but in the presence of saturation on inputs, the stability cannot be ensured. Figures 4.7(d) to 4.7(d) show how the quadrotor deviates when it is controlled via the DSMC (represented by a blue dashdotted line) with saturation on inputs. The Figure 4.9 shows 3D tracking square trajectory, both of PSMC and DSMC are successfully tracking the desired trajectory even in the wind presence, while the NMPC can't follow the desired trajectory and deviate from it.

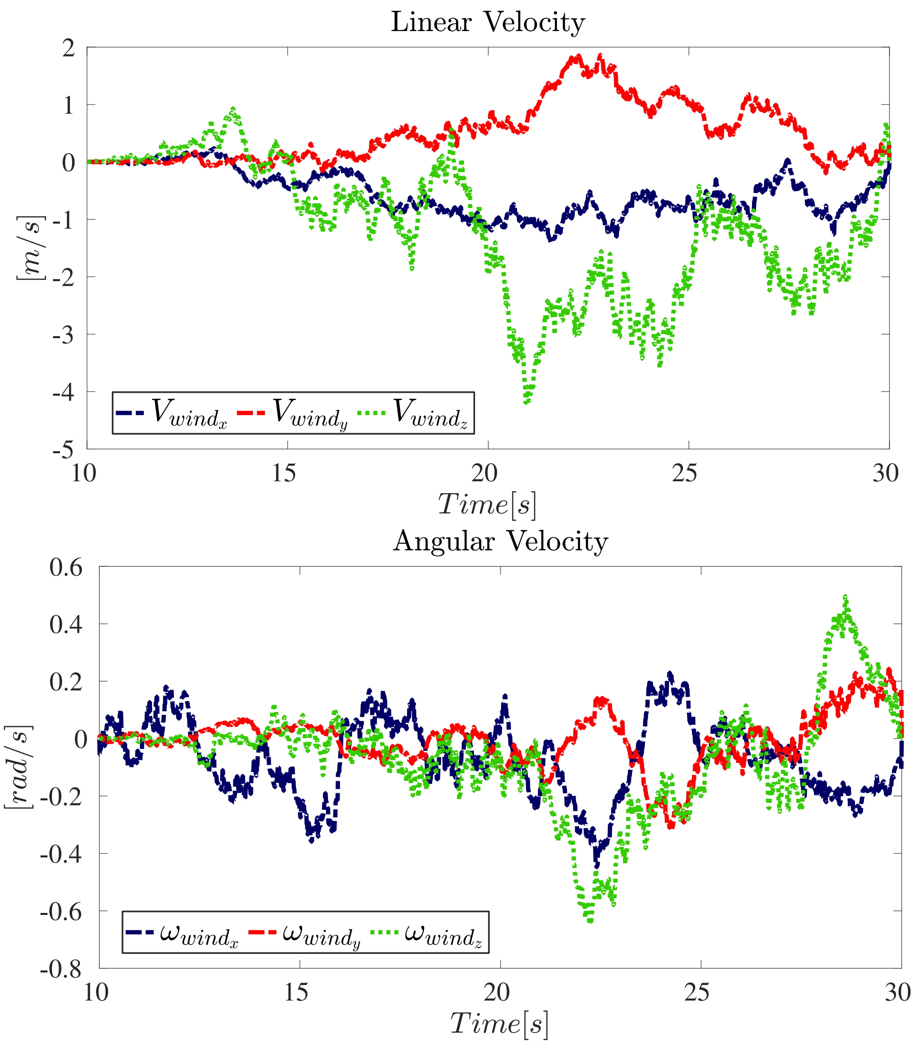


Figure 4.6: Linear V_{wind} and angular ω_{wind} velocity components of the applied wind turbulence in the interval $[10, 30]s$.

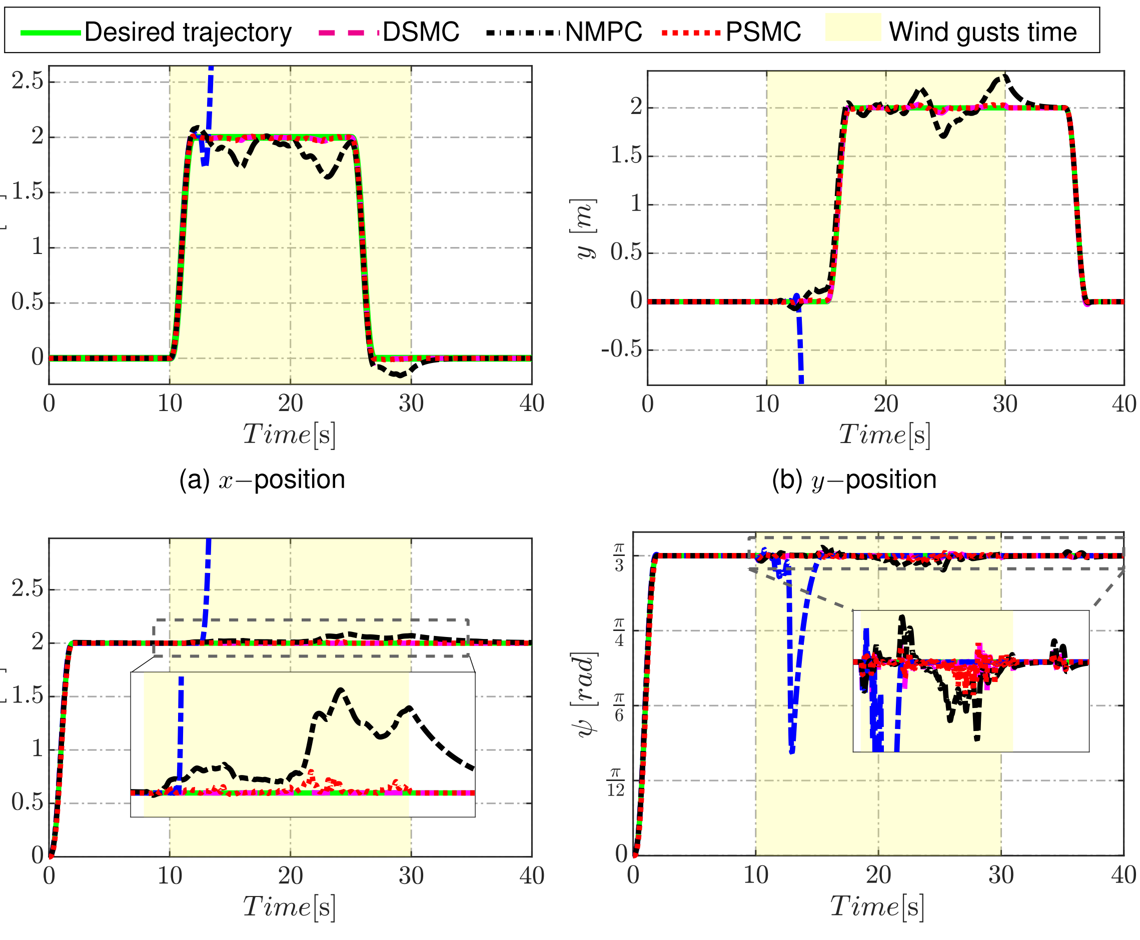


Figure 4.7: Simulation results showing tracking of square references under wind turbulence in the interval $[10, 30]s$. The marked area indicates the turbulence wind period, and the blue dash-dotted line indicates the DSMC with saturation on inputs.

4.3.1.3 Case 3: robustness comparison in the presence of model mismatch

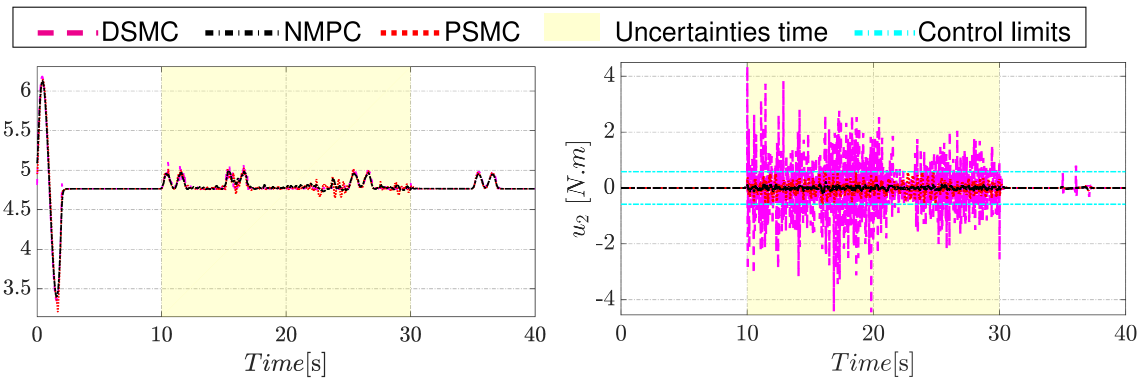
In this case, to check the controllers' robustness, the unmodeled dynamics are included in the mathematical model of the quadrotor. Since the mass m and the inertia matrix $I = diag(J_x, J_y, J_z)$ are time-variant in the first at interval [10-30]s, 40% variations of these parameters which are unknown to the controllers, and they are assumed:

$$\tilde{m} = m\,(1 + 0.4\sin(0.5t) + \gamma)$$

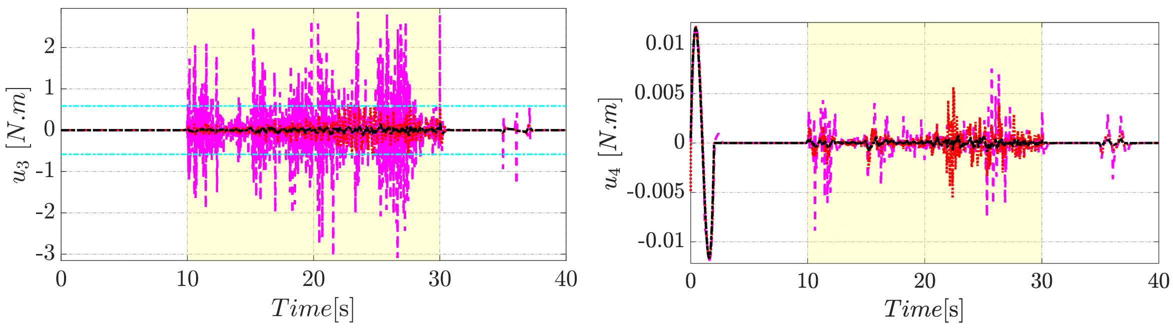
$$\tilde{J} = I_3 \times (1 + 0.4\sin(0.5t))\,J$$

where $\gamma = -0.125 + 0.25 \times rand(1)$ and rand(.) is a MATLAB function that generates a random number between 0 and 1, and I_3 is (3×3) identity matrix.

In the second period [40-50]s, we assume that there are uncertainties on the drag and thrust coefficients which are ordinarily difficult to identify. From equation (2.21),

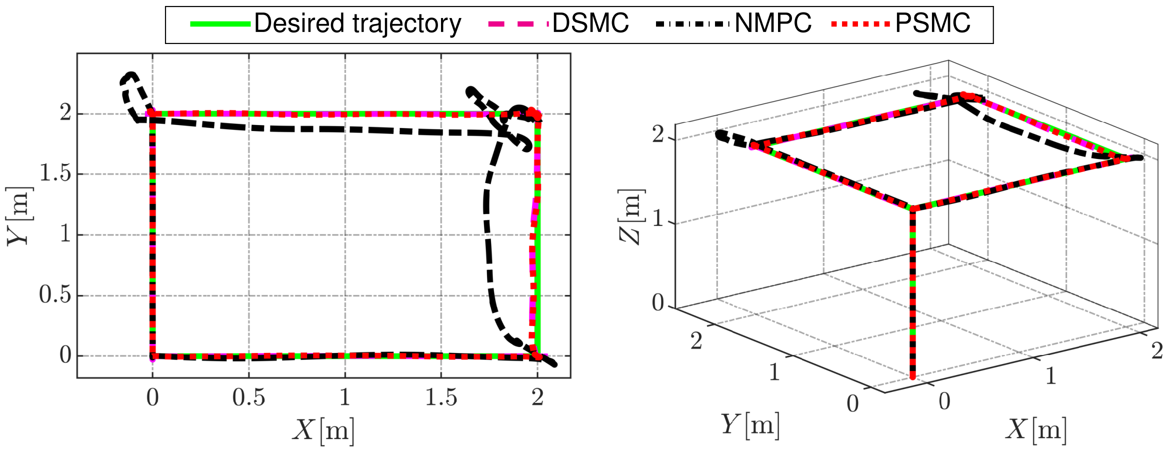


(a) Thrust force (b) Roll torque



(c) Pitch torque (d) Yaw torque

Figure 4.8: Control effort in case of wind gusts.



(a) XY view (b) 3D view

Figure 4.9: 3D Trajectory Tracking in case of wind gusts.

the variations on d and b parameters induce a disturbances on the control inputs as follow:

$$\tilde{u} = u + \delta u$$

where δu is the added disturbances caused by mismatches thrust and drag coefficients on the control inputs, and is equal to $\delta u = [2, 0.5, 0.5, 0.05]^\top$.

Figure 4.10 shows the responses of tracking helix trajectory for three nonlinear controllers under uncertainties. Figure 4.11 presents the control inputs of the non-linear controllers under parametric uncertainties. Figure 4.12 illustrates the position (x,y, and z) and yaw errors. Figure 4.13 shows the linear and angular velocities errors. As it can be seen the PSMC preserves its good tracking performance with small tracking errors Figures 4.10(a) to 4.10(a). From Figure 4.13, DSMC has noticeable errors in angular velocity at the beginning of the simulation. In Figures 4.11(a), 4.11(b), 4.11(c) and 4.11(d), it's observed that DSMC inputs exceeds the limitations on control inputs and has more chattering compared to PSMC in the presence of mismatched mass and inertia, while NMPC and PSMC preserve the control inputs within the bounded constrains.

Figure 4.14 displays the sliding surfaces evolution $s(k)$ and illustrates the contribution of the proposed PSMC in terms of convergence rate and oscillations reduction compared to DSMC.

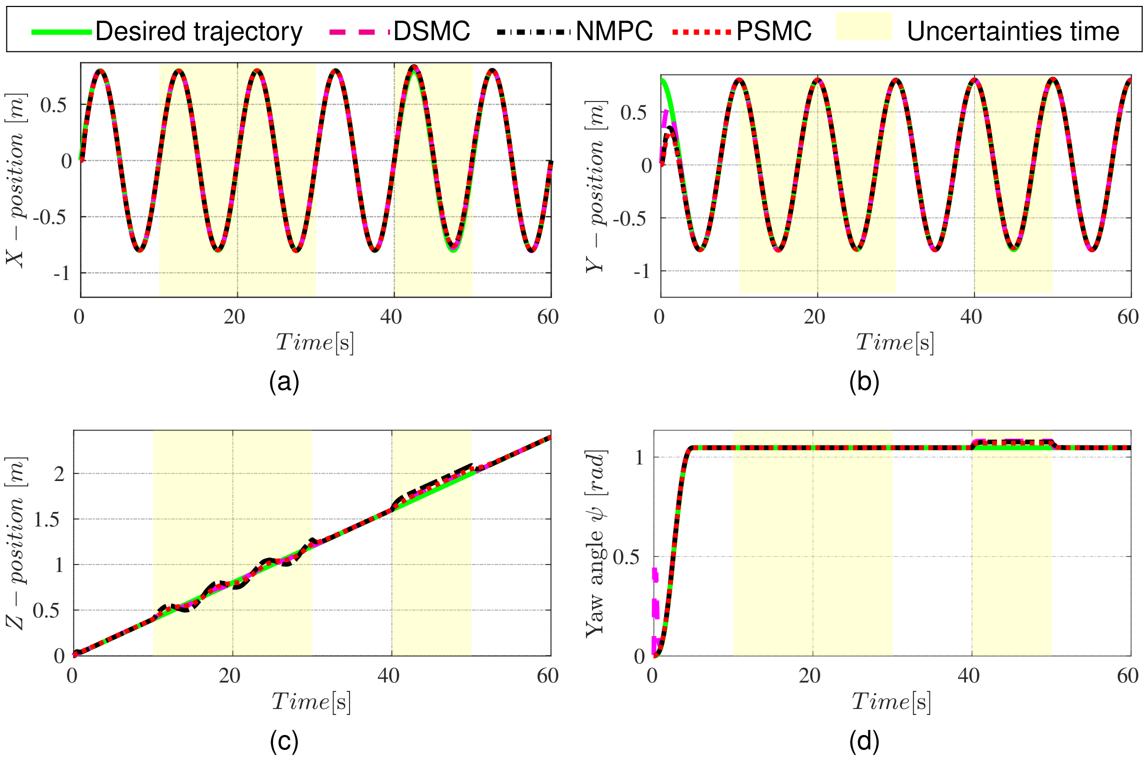


Figure 4.10: Responses of tracking the helix trajectory under parameter uncertainties.

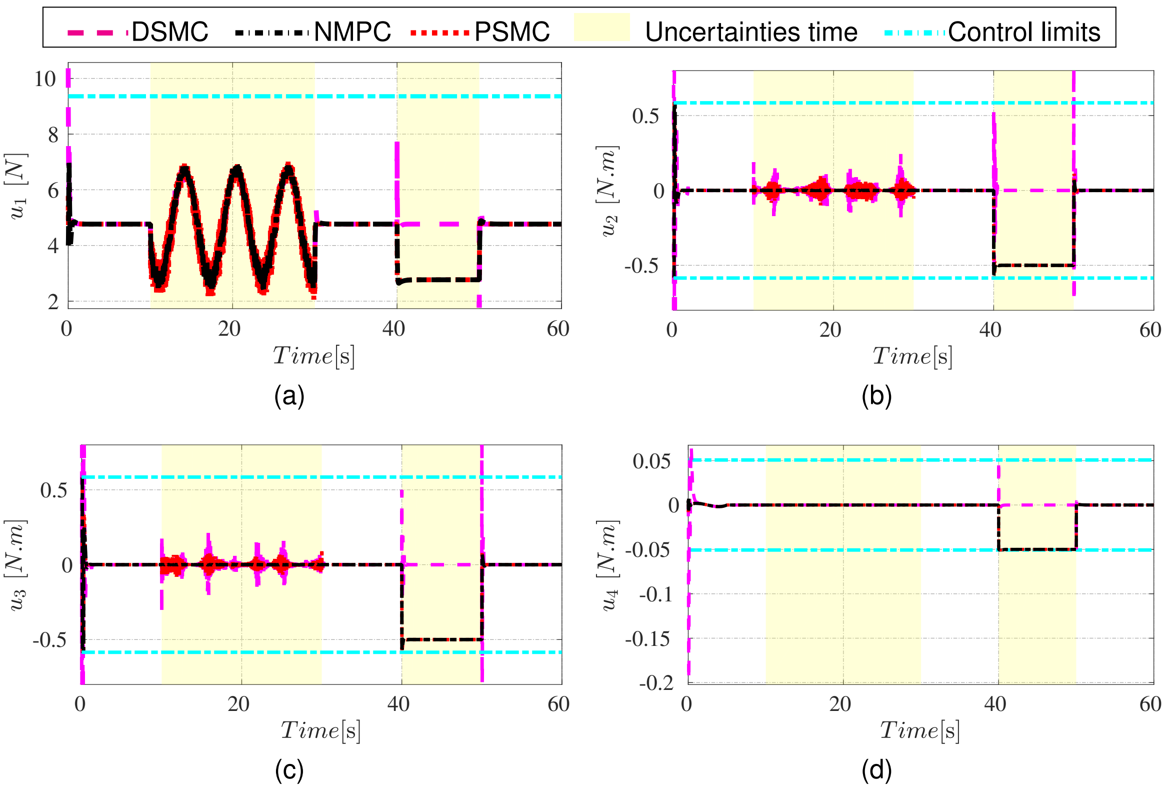


Figure 4.11: Control performance under parameters uncertainties with helix trajectory.

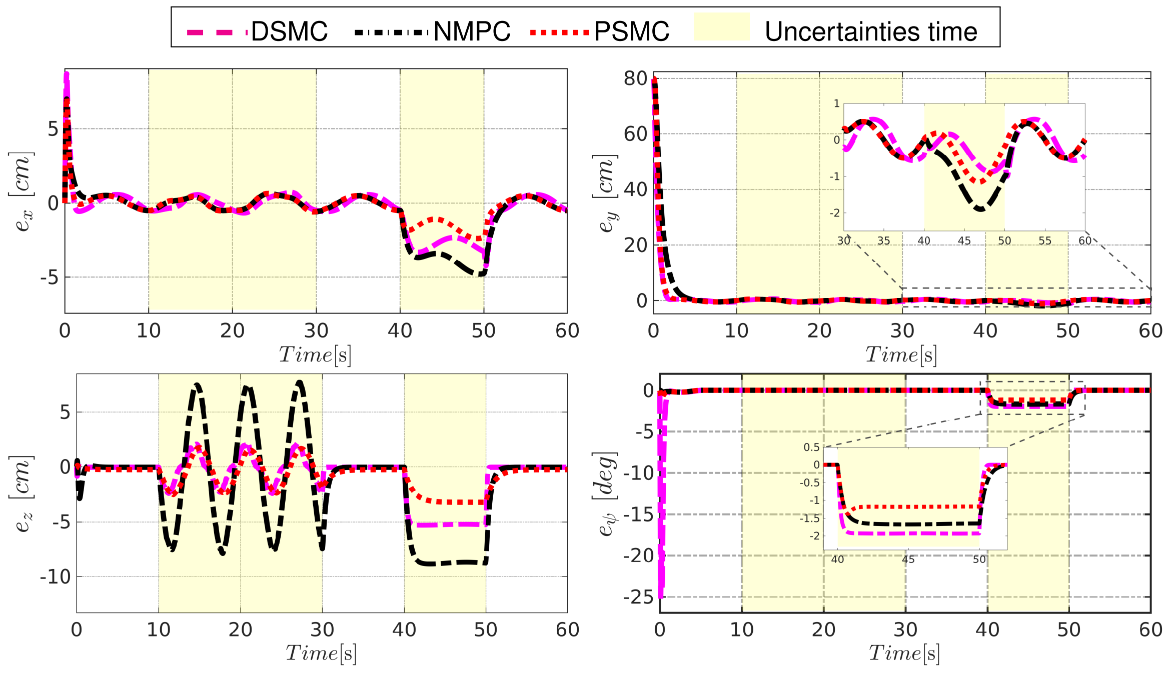


Figure 4.12: Simulation results of position (x,y, and z) and yaw errors.

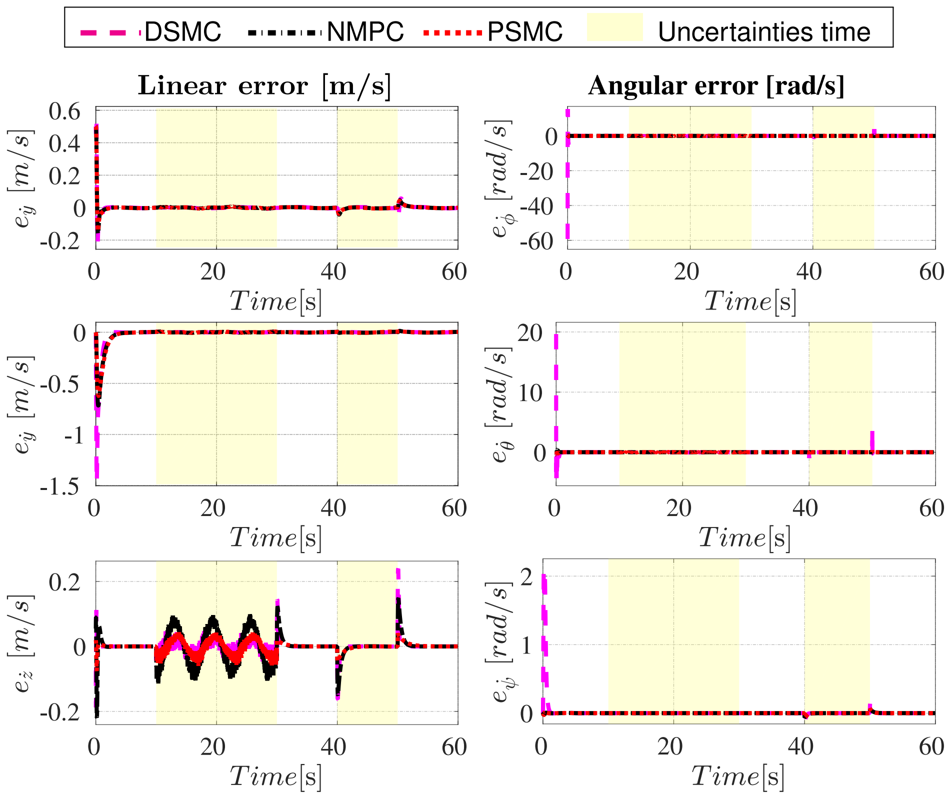


Figure 4.13: Simulation results of linear and angular error velocities.

4.4 Quaternion-based Nonlinear MPC

Quadrotor applications have become increasingly complicated as they require faster and more accurate movements to improve efficiency in obstacle avoidance and reduce flight time. This section presents an NMPC for tracking the trajectory of the quadrotor with obstacle avoidance; This command was chosen for its features in handling inputs and states constraints. Besides, given the nonlinearities and computational costs usually associated with the Euler angles and rotation matrix, the section proposes a quaternion-based approach to represent the vehicle's attitude; Finally, a numerical simulation was performed to evaluate the controller's performance with time-variant uncertainty.

4.4.1 Definition of the Orientation Error

Let q_e defines the quaternion error which descibes the rotation of the current quaternion q to the desired quaternion q_d and is given by:

$$q_e = q^* \otimes q_d \tag{4.27}$$

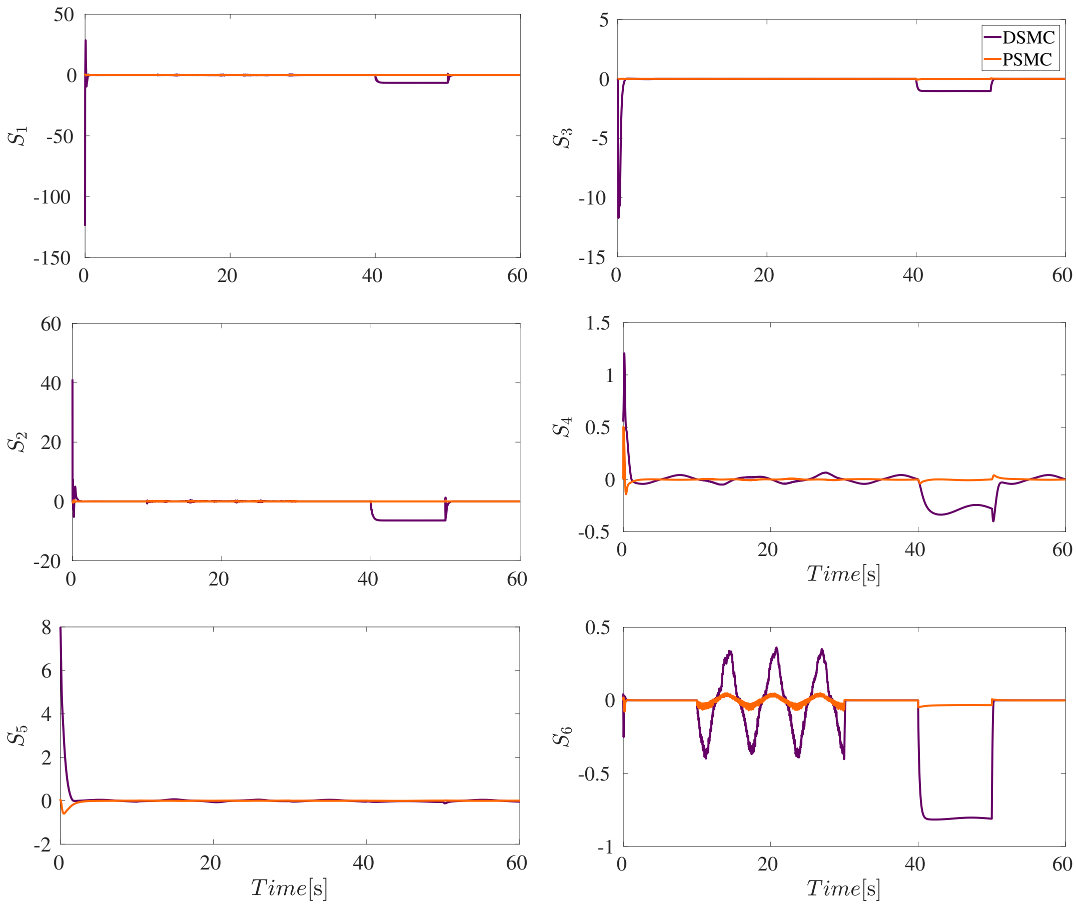


Figure 4.14: Evaluations of sliding surfaces.

The calculation of the quaternion error is divided into two parts [137, 138]:

Reduced attitude error It corresponds to the calculation of the misalignment of the quadrotor's thrus direction and is defined as the shortest rotation $\tilde{q}_e$, that aligns the quadrocopter's current direction of thrust with the desired one, Fig.4.15a.

Heading Correction Recalling that the quadrotor is not a fully actuated system, it can not move instantaneously in all directions. However, it can accelerate and generate a thrust force only along the b_z direction. Consequently, the acceleration vector direction in the body-fixed frame is always collinear with the b_z-axis ${}^{\mathcal{B}}b_z = [0, 0, 1]^\top$. The desired acceleration is corrected by PD control and is given by:

$$ {}^{\mathcal{I}}a = a_d + K_p\xi_e + K_d v_e + g e_3 \tag{4.28}$$

and its direction vector given as:

$$^{\mathcal{I}}\mathbf{a} = \frac{a_x e_x + a_y e_y + a_z e_z}{\sqrt{a_x^2 + a_y^2 + a_z^2}} \tag{4.29}$$

Note that the normalized vector in the inertial frame in (4.28) has been taken to define the direction vector. To describe it in the body-fixed frame, we multiplied it by the rotation matrix R_q:

$$^{\mathcal{B}}a = R_q{}^{I}\mathbf{a} \tag{4.30}$$

The intent is to tilt the vector b_z until it is aligned with the vector $^{\mathcal{I}}a$. The rotation axis η around which the quadrotor is rotated as well as the rotation angle γ can be found thus

$$b_z \times {}^{\mathcal{B}}a = \eta \sin(\gamma) \tag{4.31}$$

and

$$b_z \cdot {}^{\mathcal{B}}a = \cos(\gamma) \tag{4.32}$$

where $(\times)$ is the cross product and $(.)$ is the dot product.

From equations (2.14c), (4.30) and (4.31) the pseudo quaternion error is obtained as:

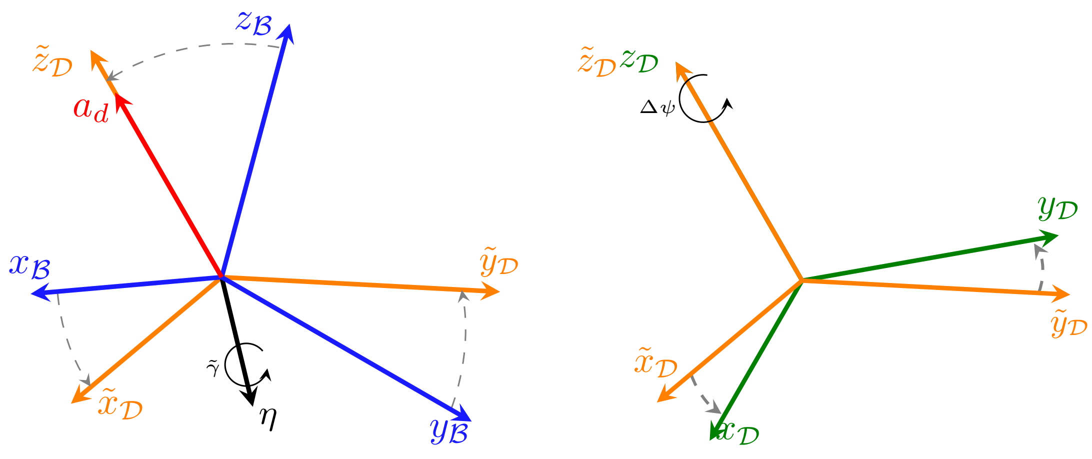
$$\begin{aligned} \tilde{q}_e &= \begin{bmatrix} \tilde{q}_{e0} & \tilde{q}_{e1} & \tilde{q}_{e2} & \tilde{q}_{e3} \end{bmatrix}^{\top} \\ &= \begin{bmatrix} \sqrt{\frac{1+{}^{\mathcal{B}}a_z}{2}} & -\frac{{}^{\mathcal{B}}a_y}{2\tilde{q}_{e0}} & \frac{{}^{\mathcal{B}}a_x}{2\tilde{q}_{e0}} & 0 \end{bmatrix}^{\top} \end{aligned} \tag{4.33}$$

The full quaternion error is obtained by multiplying $\tilde{q}_e$ by $q_z = \begin{bmatrix} \cos\left(\frac{\Delta\psi}{2}\right) & 0 & 0 & \sin\left(\frac{\Delta\psi}{2}\right) \end{bmatrix}^{\top}$

$$q_e = q_z \otimes \tilde{q}_e \tag{4.34}$$

where $\Delta\psi = \psi_d - \psi$ is the error in heading.

Visualization of the rotations is represented by the reduced attitude error $\tilde{q}_e$ and the yaw error q_z yaw in Figure 4.15. The orange coordinate frame $\mathcal{B}$ represents the quadrotor's current attitude q whereas the green coordinate frame d represents the desired attitude q_d. Rotating the $\mathcal{B}$-frame by $\tilde{q}_e$ (a rotation of γ about η) yields the auxiliar orange coordinate frame $\tilde{d}$, whose z-axis coincides with the desired z-axis. A subsequent rotation by q_z (a rotation of $\Delta\psi$ about the desired z-axis) then also aligns the x- and y-axis with the desired coordinate frame $\mathcal{D} = \{d_x, d_y, d_z\}$.


$\tilde{z}_{\mathcal{D}}$
$z_{\mathcal{B}}$
a_d
$x_{\mathcal{B}}$
$\tilde{y}_{\mathcal{D}}$
$\tilde{\gamma}$
η
$\tilde{x}_{\mathcal{D}}$
$y_{\mathcal{B}}$
$\tilde{z}_{\mathcal{D}} z_{\mathcal{D}}$
$\Delta\psi$
$y_{\mathcal{D}}$
$\tilde{y}_{\mathcal{D}}$
$\tilde{x}_{\mathcal{D}}$
$x_{\mathcal{D}}$

(a) Tilting the thrust vector (b) Heading correction

Figure 4.15: Visualization of the rotations represented by the reduced attitude error $\tilde{q}_e$ and the yaw error q_z yaw.

The angular velocity error is defined as follows:

$$\Omega_e = {}^{\mathcal{B}}\Omega - {}^{\mathcal{D}}\Omega_d \tag{4.35a}$$

$$= {}^{\mathcal{B}}\Omega - R_{q_e}^{\top}\Omega_d \tag{4.35b}$$

4.4.2 Controller Formulation

The quadcopter's controller has typically based on a cascade feedback strategy which divides the system into two loops containing an outer loop that generates the desired attitude and thrust force; and an inner loop that calculates the applied torques [139, 140]. The control framework is schematized in Figure 4.16. It is divided into two main stages; The first stage is the desired quaternion calculation using PD Controller. In the second stage, the NMPC controller is used to calculate the control inputs within constraints. The main task of optimal predictive control is to find the control input vector u that minimizes an elected cost function while ensuring the satisfaction of the constraints on states and inputs. Therefore, It is necessary to define the components of the cost function as well as the constraints affecting the system.

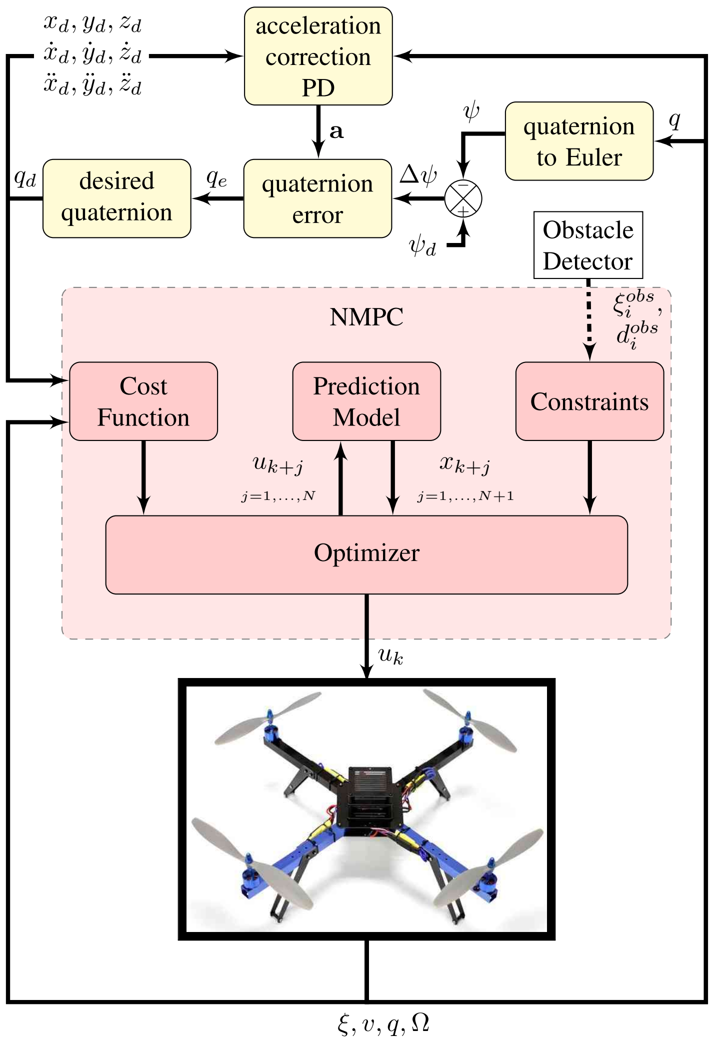


Figure 4.16: NMPC strategy scheme.

4.4.2.1 System Discretization

The continuous-time quadrotor dynamics in the equation (2.20) can be written in discrete-time using the $4-th$ order of an explicit Runge-Kutta method. The following steps should be used for integrating $\dot{x}$ given an initial state x_k and an input u_k, with an integration step T_h

$$k_1 = \boldsymbol{f}(x_k, u_k),\ k_2 = \boldsymbol{f}(x_k + \frac{T_h}{2}k_1, u_k)$$
$$k_3 = \boldsymbol{f}(x_k + \frac{T_h}{2}k_2, u_k),\ k_4 = \boldsymbol{f}(x_k + T_h k_3, u_k) \tag{4.36}$$
$$\boldsymbol{f_{RK4}}(x_k, u_k) = 1/6\,(k_1 + 2k_2 + 2k_3 + k_4)$$

However, this classical numerical integration method does not preserve the predicted quaternion q_{k+1} on its manifold. Then, the Crouch-Grossman (CG) approach evolved in [141] is used in Eq (2.16). As a consequence, the integration of quaternion is obtained with the third order of CG and is given by the following calculations rule

$$\begin{aligned}K_1 &= \frac{1}{2}\Gamma(\Omega_k)\\ K_2 &= \frac{1}{2}\Gamma(\Omega_{k+\frac{3}{4}T_h})\\ K_3 &= \frac{1}{2}\Gamma(\Omega_{k+\frac{17}{24}T_h})\\ q_{k+1} &= \exp\left(\frac{24}{17}T_h K_3\right)\exp\left(-\frac{2}{3}T_h K_2\right)\exp\left(\frac{13}{51}T_h K_1\right) q_k\end{aligned} \tag{4.37}$$

4.4.2.2 Constraints

Input constraints Since it is sometimes necessary to protect the electronic devices, and specifically quadrotor motors, from overvoltage. The control input has been constrained between the minimum $\underline{\omega}$ and maximum $\bar{\omega}$ angular velocity that can be expressed as the first inequality constraint in the optimal control problem OCP.

Obstacle avoidance constraints The second inequality constraints consist of the avoidance obstacle. We assumed that both the quadrotor and obstacle i are spheres with diameter d^q and d^{obs}. The position of obstacle-i denotes $\xi_i^{obs} = [x_i^{obs}, y_i^{obs}, z_i^{obs}]$. For safety reasons, a minimum distance ϵ between the robot and these obstacles must be maintained.
Therefore, the mathematical condition that the quadrotor evades the obstacles is as follow

$$\sqrt{(x_{k+j} - x_i^{obs})^2 + (y_{k+j} - y_i^{obs})^2 + (z_{k+j} - z_i^{obs})^2} - \frac{1}{2}(d^q + d_i^{obs}) \geqslant \epsilon \tag{4.38}$$

Separate constraints should be added for each obstacle and each time-step.

4.4.2.3 Cost Function

The NMPC tracking problem can be expressed as follows:

$$\min_{x_{k+j},u_{k+j}} \sum_{j=0}^{N-1} \underbrace{x_{e,j}^\top\, Q\, x_{e,j} + u_{e,j}^\top\, R\, u_{e,j}}_{\text{running cost}} + \underbrace{x_{e,N}^\top\, Q_T\, x_{e,N}}_{\text{terminal cost}} \tag{4.39a}$$

$$\textbf{s.t}: x_0 = \bar{x}_0 \tag{4.39b}$$

$$x_{j+1} = x_j + f_{RK4}(x_j, u_j), \quad j = 0, \dots, N \tag{4.39c}$$

$$x_j \in \mathcal{X}, \quad j = 0, \dots, N \tag{4.39d}$$

$$u_j \in \mathcal{U}, \quad j = 0, \dots, N-1 \tag{4.39e}$$

$$(4.38) \tag{4.39f}$$

where $x_e = [\xi_e, v_e, q_e, w_e] \in \mathbb{R}^{13}$ is the state error, $u_{e,j} \in \mathbb{R}^4$ is the diffrence between the input u_j and the reference input $u_{ref} = [m\sqrt{a_1^2 + a_2^2 + (a_3 + g)^2},\ 0,\ 0,\ 0]^\top$, a_1,a_2 and a_3 are the discrete-time of desired trajectory acceleration $\{\ddot{x}_d, \ddot{y}_d, \ddot{z}_d\}$. $\mathcal{X} \subseteq \mathbb{R}^{13}$ and $\mathcal{U} \subseteq \mathbb{R}^4$ are respectively the sets of feasible states and control inputs. $\bar{x}_0$ The state of the system at time step $j = 0$. The parameters $Q, Q_T \in \mathbb{R}^{13\times13}$ and $R \in \mathbb{R}^{4\times4}$ are diagonal matrix which stand for tuning.

4.4.3 Simulation Results

This section examines the effectiveness of the proposed controller using MATLAB/SIMULINK® software [1]; The NMPC is running at time samples of $T_h = 0.01$ s with prediction horizon $N = 30$. The states are initialized as follows, $x_0 = [0, 0.8, -0.5, zeros(1,3), 1, zeros(1,6)]^\top$. The weighting matrices for the optimal cost functional are chosen as follows: $Q = diag([150, 150, 100, 19, 19, 10, 1, 2, 2, 2, 0.2, 0.2, 0.05])$, $Q_T = diag([75, 75, 10, 1, 2, 2, 2, 0.2, 0.2, 0.05])$, $R = 10^{-5} \times I_4$. The PD parameters are chosen as: $K_p = diag(3,3,5), K_d = diag(0.3, 0.3, 0.5)$. The reference trajectory is defined by an eight-shaped trajectory in $\mathbb{R}^3$ space with $\omega_n = \pi/10$

$$x_d = -0.5\sin(\omega_n t), \qquad y_d = 0.8\cos(\frac{\omega_n}{2}t),$$
$$z_d = 0.2\sin(\omega_n t), \qquad \psi_d = atan2(\dot{y}_d, \dot{x}_d).$$

Along trajectory, three obstacles are placed at the following positions: $\xi_1^{obs} = [0, 0.8, 0]$, $\xi_2^{obs} = [0,0,0]$ and $\xi_3^{obs} = [0, -0.8, 0]$ with the diameters $d_1^{obs} = 0.1m$, $d_2^{obs} = 0.15m$ and $d_3^{obs} = 0.1m$.

In addition, unmodeled dynamics are incorporated into the mathematical model

[1] This video shows the experiments that were performed: `https://youtu.be/EsFEsBK5aYw`

to examine the controller's robustness. Since the inertia matrix J and the mass m are time-variant by 20% variations that are assumed to be as follows: $\tilde{J} = I_3 \times (1 + 0.2\sin(0.5t))\, J, \tilde{m} = m\,(1 + 0.2\sin(0.5t) + \sigma)$, where, $I_3 \in \mathbb{R}^{3\times3}$ is the identity matrix, $\sigma = -0.125 + 0.25 \times rand(1)$ and $rand(.)$ is a matlab function that generates a random number between 0 and 1.

In Figure 4.17, the simulation shows the success of following the reference trajectory while avoiding obstacles. The light purple tube indicates the volume occupied by the quadrotor during the flight; It is evident that the quadrotor avoids the obstacles smoothly while preserving a safe distance $\epsilon = 5cm$.

The position and linear velocity error of tracking eight-shape trajectory are presented in Figure 4.18. Both cases with or without uncertainties achieved good tracking, and they are observed to be asymptotically stable. The quadrotor has changed its trajectory around 0, 5, 10, 15, and 20 seconds on the simulation, while avoiding colliding with obstacles.

On Figure 4.19, it can be seen that the error quaternion stabilizes to the equilibruim point $\boldsymbol{q} = [1, 0, 0, 0]^\top$.

Motors quadrotor speeds are shown in Figure 4.20. As can it be shown, speeds are maintained whithin the constraints imposed by inputs where $\omega_i \in [0, 600]$. No constraint is violated by the controller, thus in turn supporting the control framework's ability to apply in real-time.

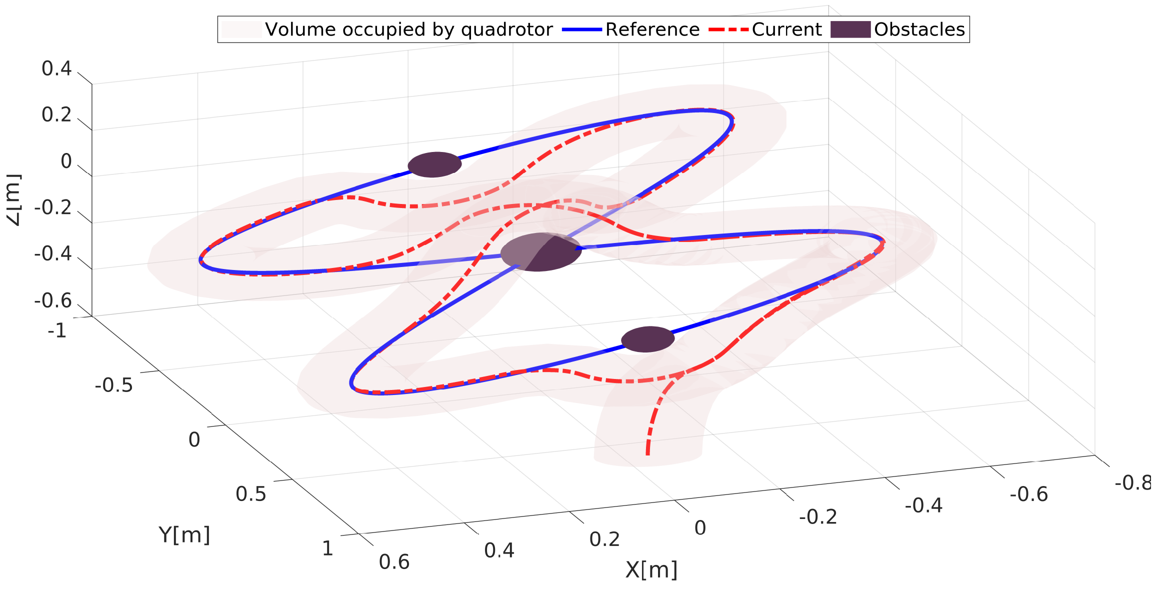


Figure 4.17: Tracking trajectory in 3D.

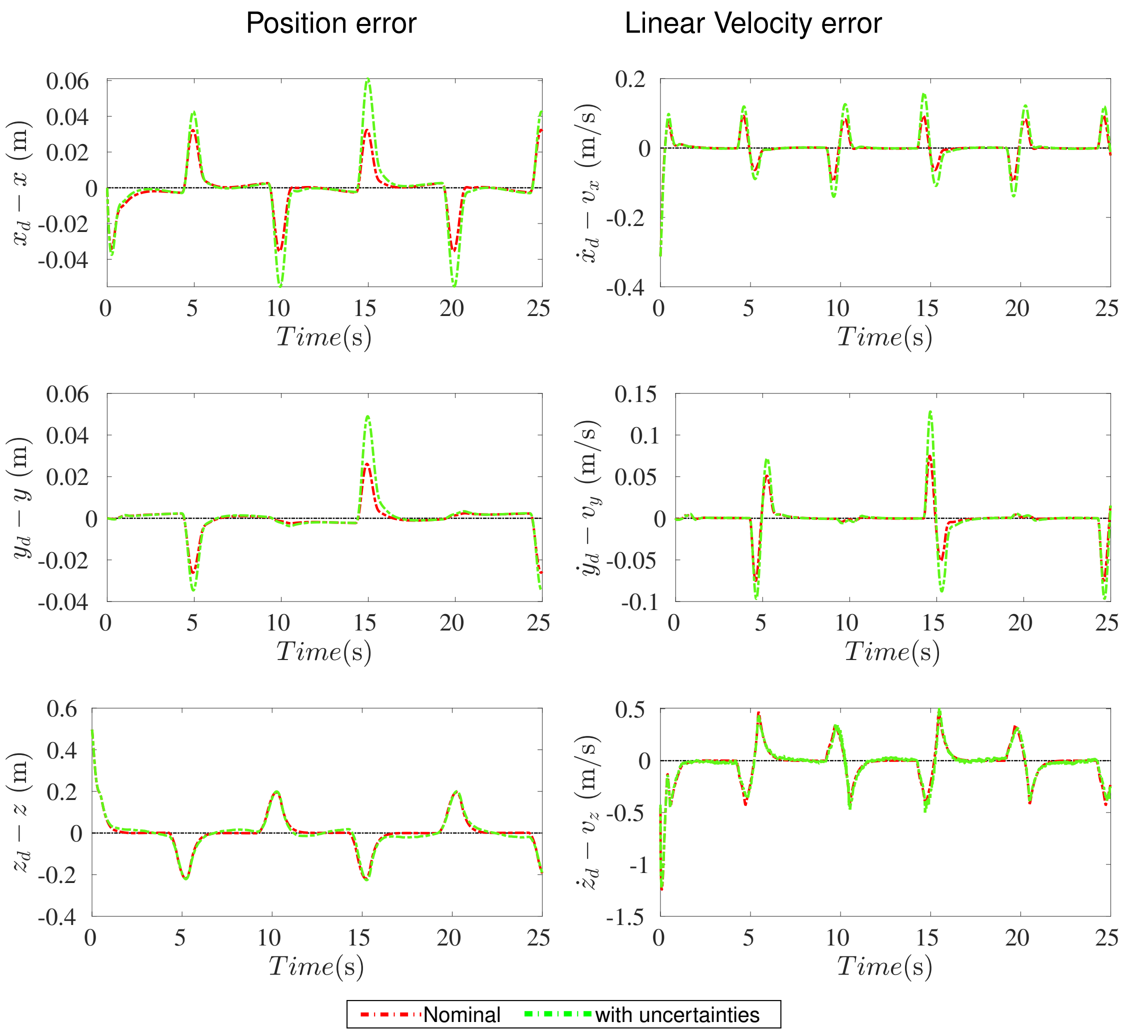


Figure 4.18: Position and linear velocity error.

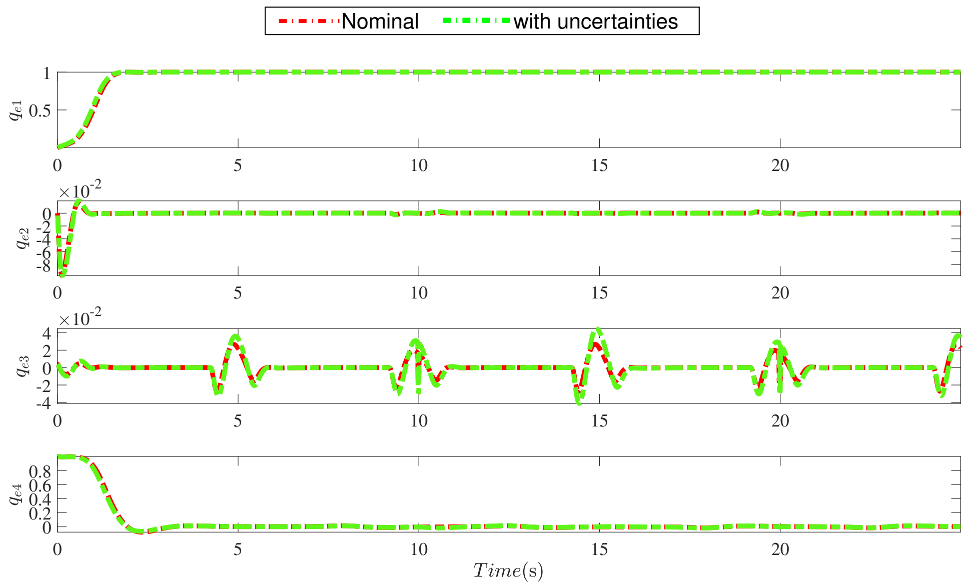


Figure 4.19: Quaternion error.

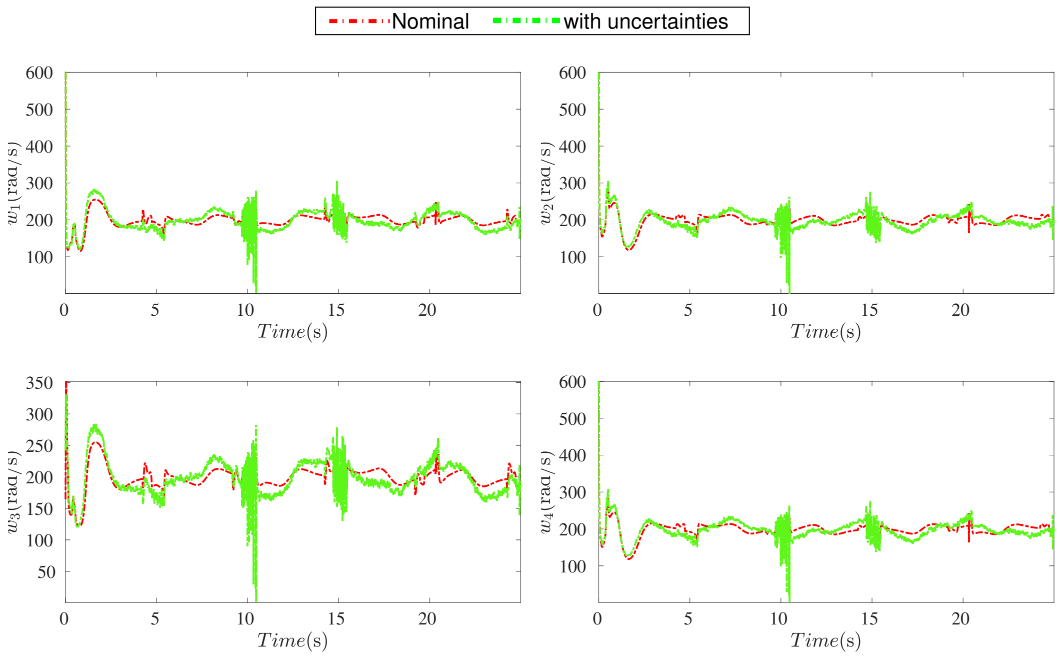


Figure 4.20: The speeds of the quadcopter motors.

4.5 Conclusion

In this chapter, the PSMC scheme is proposed for the quadrotor tracking-trajectory problem in the first part. This control has been designed by integrating the merits of both model predictive control and sliding mode control approaches. Simulation results illustrate the excellent tracking performance and strong robustness of the proposed PSMC approach in different scenarios. The second part has proposed quaternion-based NMPC for quadrotor trajectory tracking and obstacle avoidance, making it different from others working on quadrotors that mostly used Euler angles to represent the orientation. The simulation results have shown the proposed control's performances concerning time-variant parametric uncertainties and the presence of obstacles.

Chapter 5 QUADROTOR SETUP AND ASSEMBLY

5.1 Introduction

Due to technological advancements, flight controllers have become more affordable and have a higher performance which has allowed the general public to develop their own quadcopter. In this chapter, we present an implementation of the quadcopter, which will be used to verify the mathematical model and controller described in the previous chapter. We did not implement all of the simulated scenarios on actual hardware due to insufficient time and the lack of some sensors.

5.2 Hardware

Defining the components used and how they can be connected is the first step to building a quadcopter from scratch.

ArduPilot Mega board ArduPilot Mega (APM) is an Arduino Mega-based autopilot system developed by DIY Drones community as an upgrade of ArduPilot flight control. This compact system gives the users chance to develop their own autopilot software. One can design an autopilot for any fixed wing planes, traditional helicopters, multi-rotor vehicles, ground rovers, submarines and even boats. The project is open source under GPLv3 license. Figure 5.1(a) showsArduPilot Mega (APM) v2.8 unit and Figure 5.1(b) shows its circuitboard.

In summary, APM 2.8 board characteristics include the following

– It contains an ATmega2560 main processor, an 8bit Atmel processor with 16 MHz RAM.

– It includes a six degree of freedom MEMS IMU (MPU-6000) which contains a 3 axis gyroscope (angular velocity measurement), a 3 axis accelerometer (acceleration measurement), and a temperature sensor. Moreover, APM has a MEMS pressure sensor/ barometer (MS5611-01BA) that is used to measure altitude and magnetometer (heading information) which can be integrated internally or via external compass.

– A I2C (Inter-Integrated Circuits) serial port, typically used to connect additional sensors like an external barometer.

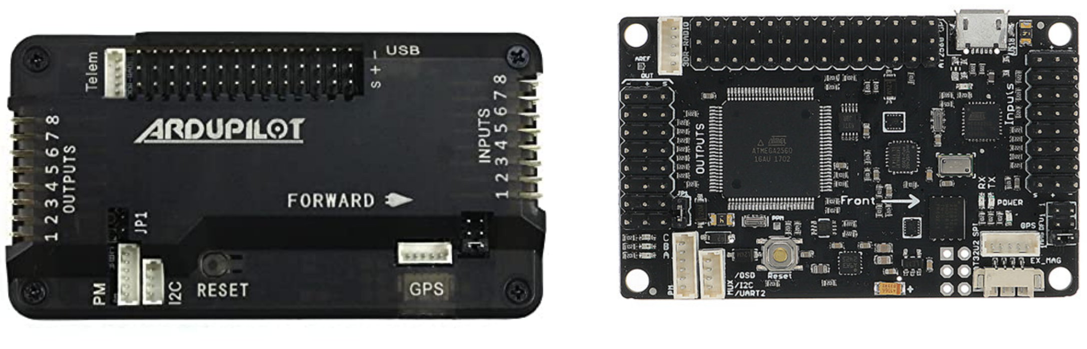
– A Micro-USB, which allows attaching directly the board to a computer. This allows not only to get telemetry and interact with the autopilot, but also to upload the firmware you may need. However, and as it is a wired connection, it may not be possible in all scenarios.

– A Power Module port, to allow powering the board through an external power supply.

– A port for a GPS Module where, although 3D Robotics recommends the UBlox LEA-6H GPS (precision of 2.5m), you can place any module you would like.

– And the last one is to plug in a custom serial radio device running at 57600bps, which can be used to communicate directly to a ground station (to receive telemetry, upload flight plans, etc.).

Raspberry Pi The Raspberry Pi [142] is small single-board computer developed in the United Kingdom by the Raspberry Pi Foundation for education purposes. Several variants of free operating system can be run through the SD card image. The Raspberry Pi has 64Bit Quad Core ARM Cortex-A53 Processor running at 1.2GHz. It is also equipped with a General Purpose Input Output (GPIO) connector, pins for communication protocols such as UART, The I2C and the SPI, besides it four USB ports, one Ethernet port, an HDMI port, SD card slot and a microUSB for providing power, Figure 5.2.



(a) Enclosured (b) Circuit board

Figure 5.1: APM Flight Controller.

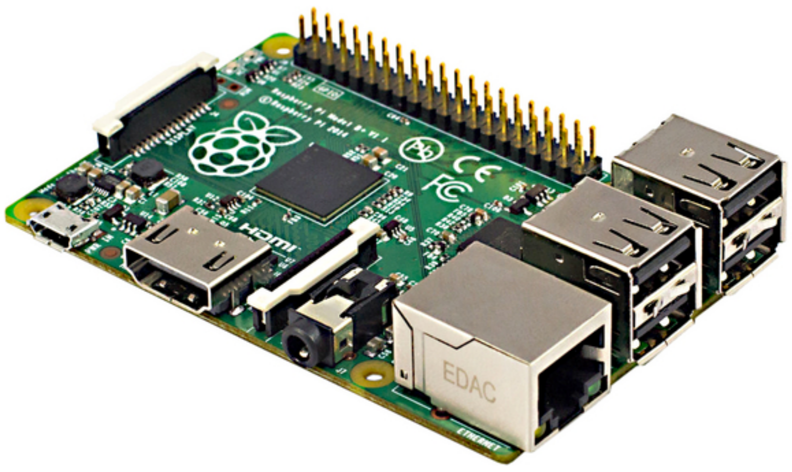


Figure 5.2: Raspberry Pi Model B.

Frame The structure that holds all the components together. They need to be designed to be strong, rigid and lightweight. In order to have good flight characteristics, it is necessary to choose a frame which is symmetrical and which has the least possible deformation and flexion so that the quadcopter can give maximum performance Figure 5.3.

Propellers Propellers come in various lengths and pitches, the larger the propeller or larger its pitch, the more power it will require and more thrust it will generate. Propellers used should match with the frame of the quadrotor. A pair of CW (Counter Clockwise) and CCW (Counter Clockwise) propellers are affixed to the motor, they

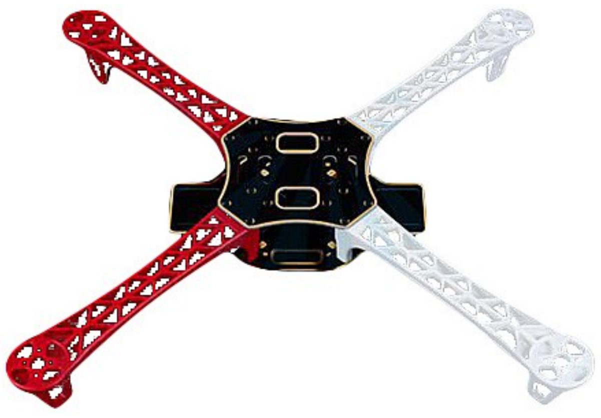


Figure 5.3: DJI 450 frame.

generate thrust and achieve lift by sending the air downwards. For our project, we chose propellers with a diameter of 22 cm, Figure 5.4. Figure CW and CCW propellers

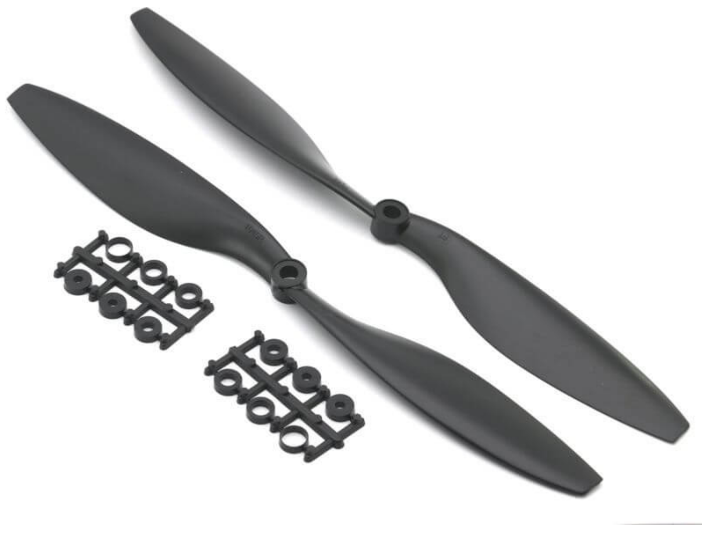


Figure 5.4: CW and CCW propellers

Brushless Motors (BLDC) Brushless Motors (BLDC) Brushless motors are synchronous motors powered by DC current, they are widely used in industry, aerospace, automotive and modeling applications. BLCD motor to be mounted at the end of each arm to rotate the propellers in quadrotors. Comparing with brushed motors they are more reliable, more eficient, and less noisy BLCD motors are composed of two parts stator and rotor, stator is the part where all the windings are located and it does not turn but produces electromagnetic fields to influence permanent magnets placed on a rotor to make it turn. In our project, we will use Lynxmotion Brushless Motor 28x30 1000kv, Figure 5.5, having 1000 KV with a mass of 58g.

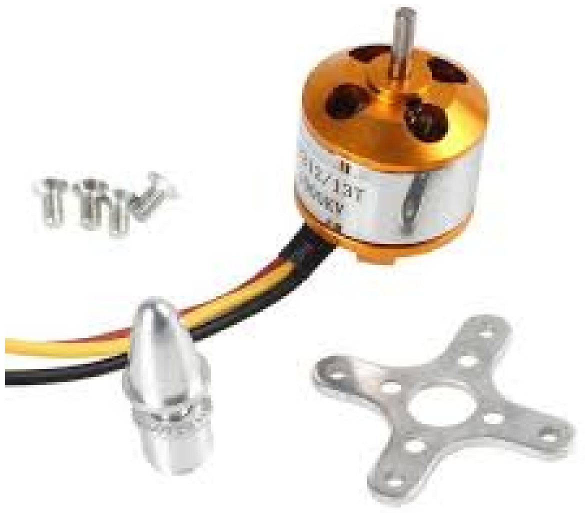


Figure 5.5: Brushless Motor 1000kv

Kv is the rating of a motor that relates to how fast it will rotate for a given voltage. It is indicated in RPM (rotations per minute) per volt with no load condition. If the KV rating for a particular motor is 650rpm/V, then at 11.1V, the motor will be rotating at 11.1V x 650 = 7215rpm.

Electronic speed control (ESCs) it is used to control the speed by which the motors are rotating. Usually, it has 5 wires on one side and 3 on the other. The 3 wires go to a BLDC motor (i.e. the 3 phase signal from ESC) and the five wires have two power wires for battery, two for powering the Flight Controller and last one is for PWM signals. In our project, we use four Lynxmotion 30A Multirotor ESC, Figure 5.6. it has a dimension equal to 27 x 22 x 7mm and a mass of 9g.

Li-Po Battery Stands for ”Lithium Polymer”, these batteries are the most used when building a drone because they are lightweight and are capable of storing sufucient energy. It can come in various sizes and can last from 10-30 minutes depending on the size you use for the battery. The battery we chose is of capacity of a 3800mAh and a voltage of 11.1 volt with 3 cell battery that means 3.7 Volts each, Figure 5.7.

Figure 5.6: 30A Multirotor ESC.

Figure 5.7: Battery Li-Polymer (Li-Po) 3S.

Power Module It should be connected to the ”POWER” port on Ardupilot through 8-pin connector to provide a stable voltage (5.3 Volts) to this latter. Power Module has two ends one (XT60 connector female) is connected to the power distribution board and the other end (XT60 connector male) is connected to the battery.

RC Receiver and transmitter *Radio Control* (RC) system uses a radio signal to control a device at a distance. They are therefore used for manual quadrotor flight. An RC consists of two parts, a receiver attached to the frame, and a transmitter (joy-stick) that sends control signals to the receiver. The transmitter converts signals into radio signals. Then, the receiver at the vehicle collects these signals and generates appropriate PWM values to drive the vehicle. To recognize the transmitter signals, the transmitter and receiver must be bounded together before the first flight. Receivers come in a wide variety of protocols and frequencies; It should be noted that receivers are not all created equally. Despite receivers having the same frequency and protocol, they can have very different performance specifications.

Until recently, RC receivers operated in the *very high-frequency* (VHF) band and used either *Pulse Position Modulation* (PPM) or *Pulse Code Modulation* (PCM). The PCM protocol is more robust because it sends error-checking information with every packet, but it also suffers from “lockout”. Lockout occurred when synchronization was lost with the receiver, for whatever reason, and the radio-controlled model no longer responds to the controller. The latest RC equipment operates on the 2.4 GHz *Industrial Scientific Medical* (ISM) and utilizes a more robust modulation protocol than either PCM or PPM.

FlySky FS-i6 2.4 GHz radio control (RC) with ten channels is used for radio communication in our quadrotor platform as shown in Figure 5.8. However, only 5 channels have been used in our quadrotor system for sending the commands (PWM signals) of the roll, pitch, yaw, throttle, and mode switch. According to specific stick movements, each channel of the RC transmitter provides a throttle value in the range of (1000-2000). It is necessary to calibrate the RC to determine the desired operating range.

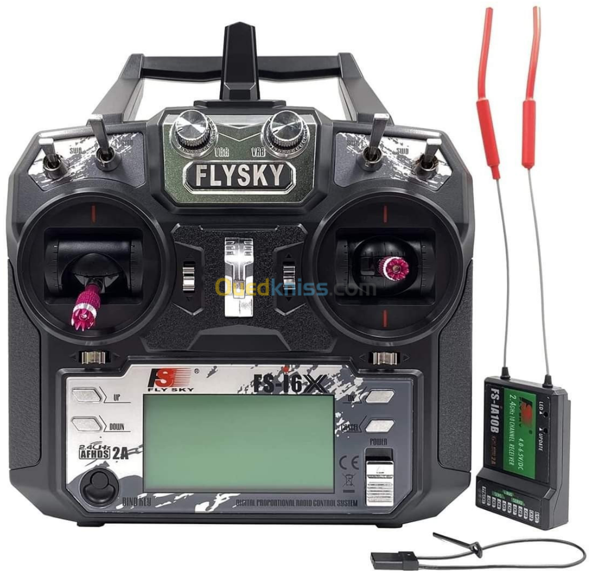


Figure 5.8: FlySky FS-i6 RC Transmitter and Receiver.

5.3 Software

GCS (Ground Control Station) It is a ground-based computer that communicates with the drone via wireless telemetry. It displays real-time data on the drone’s performance and position and can be considered as virtual cockpit, it can be used for uploading a new mission commands, setting parameters and control the drone. Among the GCSs choices listed here [143], we chose Mission Planner, it is free open source software for Windows created by Micheal Oborne, it allows to plan autonomous mission and making a full control of the drone, after completing a mission a log file containing all internal parameters and state variable will be created for later analysis of the autopilot, it has a lot of features like : full setup/configuration of autopilot, displaying maps, waypoints, videos, vehicle position, instruments and its sensors measurements . . . etc.

It interacts with UAV through the Mavlink protocol by exchanging mavlink messages to allow the control of the UAV and to monitor its status.

Mavlink Protocol It is simply a communication protocol that allows a UAV to send and get messages from a GCS, it was released for the first time in 2009 by Lorenz Meier. Mavlink message has a specific format and it is usually implemented in autopilots to govern and control the motion of the UAV.

Mission Planner It is an open source autopilot software created in 2007 by the DIY Drones community , it allows the autonomous control of multirotors, rovers and fixed wing planes and other platforms that supports Mavlink communication. Emlid website provide all the steps for configuring and running the autopilot [?]. Since it supports all kind of vehicles, we went for ArduCopter according to our platform.
The list of features associated with the Mission Planner is as follows:

- Setup, configure, and tune drones to optimal performance;
- Load firmware into APM that controls the drone;
- Plan autonomous missions by selecting waypoints and targets;
- Download logs to analyze the flight data;
- Interface with flight simulators to create a full hardware-in–loop simulator;
- Monitor the vehicle status in operation;
- View and analyze the telemetry logs.

5.4 Sensors Calibration

Sensor calibration is a mandatory step to get the motors to arm because the Arducopter performs a list of checks including missed calibration, configuration, or bad sensor data so if something is wrong it will prevent the motors to arm. These checks help prevent crashes and flyaways [144]. We have confined ourselves to accelerometer and magnetometer calibration and we disabled all the other checks using Mission Planner. After connecting the mission planner to autopilot, it will provide a window for sensors calibration Under Initial Setup Mandatory Hardware after selecting the sensor to calibrate from the left-side menu. In accelerometer calibration, it will request to hold the board still in different positions (level, on the right side, left side, nose down, nose up, and on its back, while the calibration process is performed each time a key will be pressed to indicate that the autopilot is in position. In Magnetometer calibration, the mission planner will request you to flip the board in all directions to reach every possible orientation possibly within 30 seconds. After doing so the GCS will load the parameters into Ardupilot. For more detail, see Appendix B.

5.5 Quadcopter Setup

This section covers the quadrotor's setup and installation. Utilizing all the components mentioned previously, the quadcopter is assembled then the autopilot is tested using Ardupilot Mission Planner GCS.

- Assemble DJI-450 frame.
- Download the Raspbian OS image and burn it into the SD card.
- Screw down the Ardupilot on top of the Raspberry Pi and do the necessary configurations and settings, for more details, see Appendix C.
- Perform calibrations for Accelerometer, Magnetometer and disable the nonused sensors in order to avoid the pre-arm checks failure when arming the motors.

– Screw the Motors to the end of each arm and attach each ESC at around the middle of each arm.

– Connect all four motors to their respective ESCs Three motor wires should be connected to the three ESC wires.

– Connect the ESC wires for PWM signals to the Ardupilot board. The pin near to the edge of the board is Ground, the middle pin is +5V and the pin farthest from the board's edge is the output signal pin that sends PWM signals to control the motor speed.

– Connect all ESCs to the Power Distribution Harness, Make sure the propellers are not mounted for safety when connecting the battery to the Power Distribution Harness, some beeps will be heard telling that the ESCs are ready.

– Make sure the motors are spinning in the right direction. In case a motor is spinning in the wrong direction, then simply flip any two wires between the motor and its ESC with the propellers off.

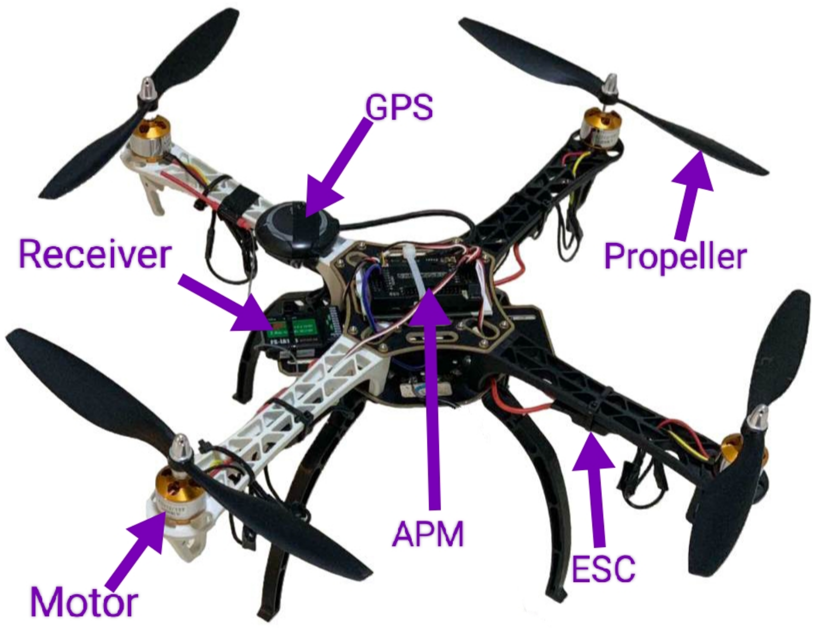


Figure 5.9: Drone assembly.

5.6 Flight test

An overview of the cascaded controllers used for our quadrotor system is shown in Figure 5.10. The lower level, the higher bandwidth, controls the four-rotor angular velocities. Another level is the onboard attitude control of the quadrotor at 100Hz. The processes of the onboard controller can be separated into three steps (from left to right):

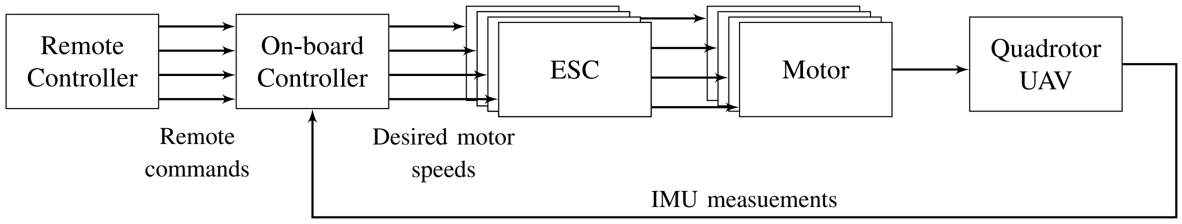


Figure 5.10: Overview of the quadrotor cascaded control loop.

- Firstly, the onboard controller generates the desired attitude (ϕ_d, θ_d, and ψ_d) and thrust (T_d) from the PWM signals provided by the remote controller.
- The onboard controller reads the IMU measurements and calibrates the measurements (see Section B.2). Then, the attitude control torque τ can be designed from the IMU measurements and/or the desired attitude generated in Step 1. The design of the attitude controller is based on the PID controller that can be tuned from WinGUI.
- Finally, the desired motor speeds $\omega_{i,d,}\ i = 1, 2, 3, 4$ can be generated by the control torque and thrust with the calibrated propeller efficiencies. The PWM signals can be generated and sent to the actuators (ESCs and motors), and the propellers will spin and provide different thrust forces that will determine the quadcopter maneuver.

For the flight test mission[1], firstly the drone is hung with a protective rope; this is for a safe flight test and to protect the drone from falling into instability. After several experiments and making sure of its stability during flight, we re-tested it without the protective rope as shown in Figure 5.11.

[1]Flight test video: `https://youtu.be/9IOhIv_foaU`

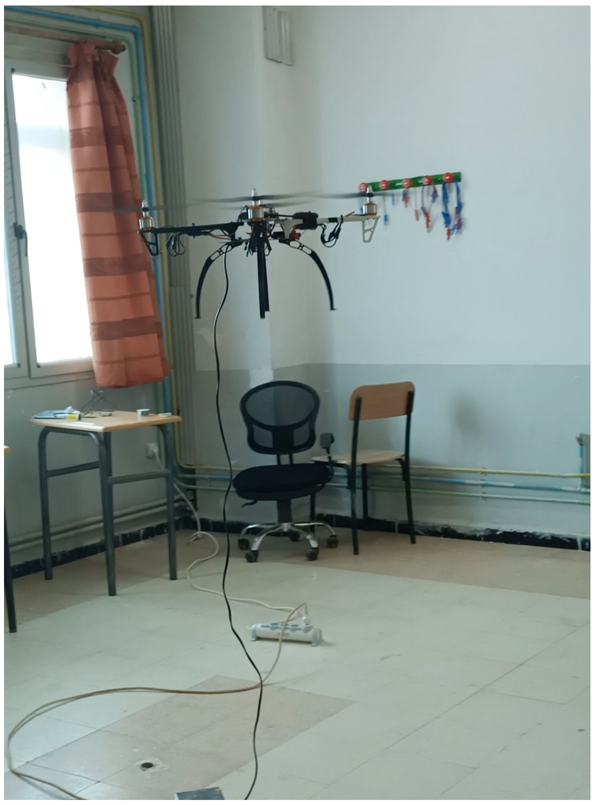
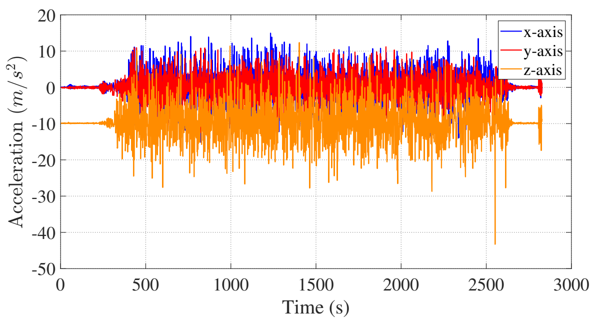


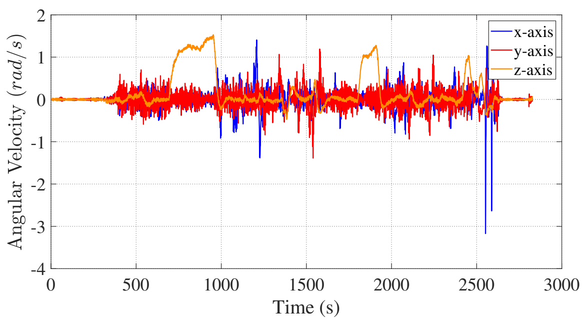
Figure 5.11: Flight test.

Figure 5.12 shows the data including 3-axis acceleration and angular rate measured by APM's IMU. It has been shown that the accelerometer stays quite stable when the quadcopter stops. The accelerometer measurement profiles also imply the diversity of vehicle jerks between a stop and a move.

Figure 5.13 illustrates the height measurement provided by barometer sensor during the test flight. The performance of the real-time attitude tracking is shown in Figure 5.14, which shows the Euler angles of the quadrotor.



(a) Accelerometer



(b) Gyroscope

Figure 5.12: IMU measurements.

Figure 5.15 shows the values of received PWM signals from the RC transmitter.

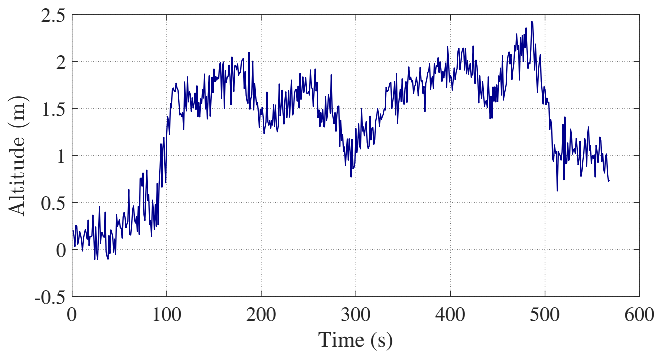
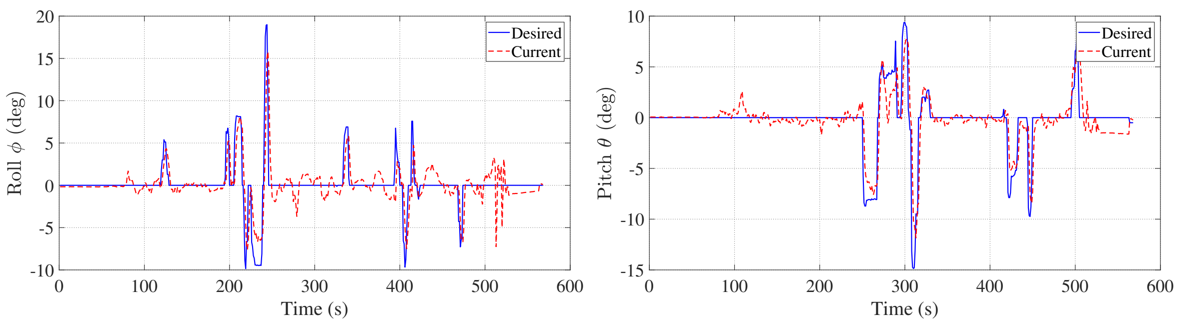
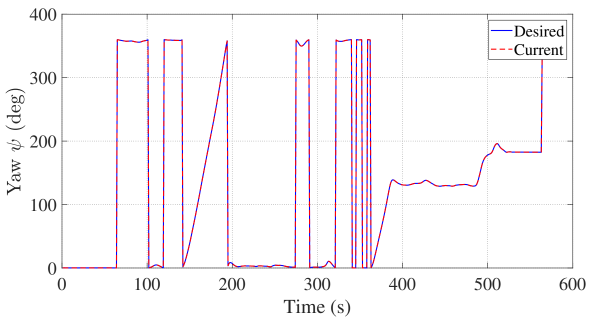


Figure 5.13: Barometer measurement.



(a) Roll

(b) Pitch



(c) Yaw

Figure 5.14: Results of Implementation of PID controller.

Profile of PWM sent to each ESC, as output of control strategy, is shown in Figure 5.16. In hovering, the PWM sent to ESC is about 1500. However, the value of PWM in hovering ideally should be equal for all motors. The different shown in Figure 5.16 is due to the unbalance of mechanical system.

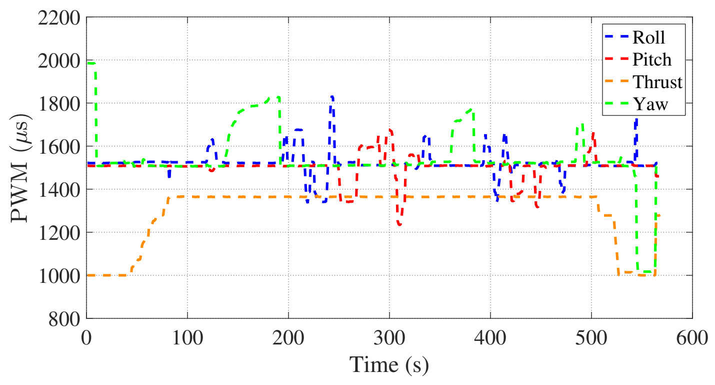


Figure 5.15: Received PWM signals from the RC transmitter.

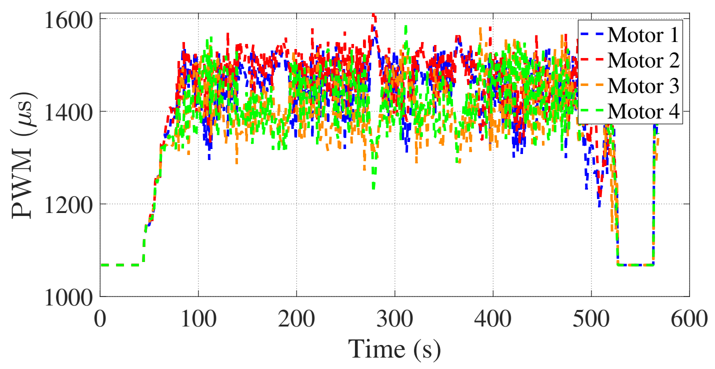


Figure 5.16: Profile of PWM on each ESC.

5.7 Conclusion

This chapter describes how to build a quadrotor from scratch where we presented the evolution of the quadrotor design, the material used as well as the software, communication protocols and the programming language used. We addressed sensors calibration driven by a specific requirements to get the quadcopter to fly and finally we explained the data communication between all the subsystems.

CONCLUSION

The quadcopter is one of the most popular mini drones due to its relative simplicity of manufacture and its dynamics. Therefore, it has attracted attention and become the subject of much research in recent years. It is a complex, nonlinear, multivariable, unstable system and exhibits strongly coupled dynamics which makes its control a great challenge. The mathematical model of a quadrotor UAV was developed in detail including its aerodynamic effects and rotor dynamics which we found lacking many kinds of literature.

Throughout this thesis, solutions were developed to provide full control of a quadcopter. To solve this problem, linear and nonlinear techniques were used, with different architectures studied for each technique. The developed strategies were all tested in simulation, which allowed us to determine the most auspicious.

In this thesis, the PSMC control strategy is proposed to ensure simultaneously the inputs constraint and robustness concerning sudden stochastic disturbances (wind turbulence), and time-variant parametric uncertainties. This work elaborated from a comparative study between different nonlinear control approaches. The controllers NMPC, IBC, and SMC have been tested in simulation. The SMC controller exhibited robustness against disturbances, while the NMPC has shown lower control effort. These results conduct us to propose PSMC that merges DSMC and NMPC advantages. The simulation results show the outperformed performances of the proposed PSMC with regards to NMPC, IBC, and SMC.

There has been considerable attention paid to unit quaternion-based attitude control laws since the unit quaternion is the minimal globally singular-free description of a rigid body's attitude. Nevertheless, each attitude is made up of two distinct unit quaternions covering the rotational configuration space SO(3). In addition, even though Euler's angle representation only has three parameters, it requires the evaluation of trigonometric functions, thereby making it computationally more demanding and more nonlinear. Another control strategy used in this thesis was based on quaternion parameterization to describe the quadrotor's attitude. This strategy adopted quaternion with MPC in a novel way, making it different from others working on quadrotors that mostly use Euler angles as orientations. The simulations result had shown the performances of the proposed control with regard to time-variant parametric uncertainties and the presence of obstacles. In addition, it is discovered that quaternions require significantly less computation time.

We finally focused on building an autonomous quadcopter using Raspberry Pi and ArduPilot as a flight controller, we presented the hardware, software, and programming languages used. Next, we explained sensor calibration to obtain good measurements, and also the steps to proceed with the construction of such a drone.

Futur Work

Future works comprise the incorporation of the adaptive mechanism with PSMC for parameter uncertainties problems to enhance tracking accuracy in presence of unmodeled dynamics. Further, stability and feasibility analysis will be investigated by including a nonlinear observer of the state.

As part of continuing this work, we suggest: Improving the sensor technology, like adding an ultrasonic sensor or an intelligent camera so we can take advantage of other ROS packages like Optitrack and Mavros-extra. The distance between the quadcopter and the ground station was only about 10m; this distance can be extended using telemetry with an antenna. Moreover, for autonomous mission planning, full autopilot implementation is needed using GPS for position estimation in outdoor scenarios.

Appendix A Reference Trajectories

In this appendix, we define all the reference trajectories used in this thesis.

The equations of the desired trajectories used in section for IBC and SMC contoller are:

- Line $x_d = 1$ m, $y_d = 1$ m, $z_d = 1$ m, $\psi_d = {}^{\pi}/_{3}$ rad
- Helix $x_d = \frac{1}{2}\cos\left(\frac{t}{2}\right)$ m, $y_d = \frac{1}{2}\sin\left(\frac{t}{2}\right)$ m, $z_d = 0.1t$ m, $\psi_d = {}^{\pi}/_{3}$ rad
- Spiral $x_d = \frac{1}{2}t\cos(\frac{t}{2})$ m, $y_d = \frac{1}{2}t\sin\left(\frac{t}{2}\right)$ m, $z_d = 0.1t\cos(t)$ m, $\psi_d = {}^{\pi}/_{3}$ rad

The equations of the desired trajectories used in section for NMPC, DSMC and PSMC contollers are:

- Inclined 8$-$shaped trajectory:

$$\begin{aligned} x_d &= 0.5 \times \sin(\omega_n t)(1 - e^{(-0.1t^3)}) \\ y_d &= 0.8 \times \cos(\omega_n t/2)(1 - e^{(-0.1t^3)}) \\ z_d &= \begin{cases} 0.3 + a_1 t^3 + a_2 t^4 + a_3 t^5 & if \quad t \leq 5 \\ 0.9 + 0.2\sin(\omega_n t) & if \quad t > 5 \end{cases} \end{aligned}$$

  where, $\omega_n = \frac{\pi}{5}$, $a_1 = 2.7894 \times 10^{-2}$,$a_2 = -7.3628 \times 10^{-3}$ and $a_3 = 5.4881 \times 10^{-4}$.

- Square trajectory

Initialization: $t_1 = t$, $t_2 = t$, $t_3 = t$, $t_4 = t$ and $t_5 = t$.

$$
\begin{aligned}
t_{max} &= 2 \\
t_1 &= max(0, min(t_1 - 10, t_{max})) \\
t_1 &= t_1/t_{max} \\
t_2 &= max(0, min(t_2 - 25, t_{max})) \\
t_2 &= t_2/t_{max}; \\
t_3 &= max(0, min(t_3 - 15, t_{max})) \\
t_3 &= t_3/t_{max} \\
t_4 &= max(0, min(t_4 - 35, t_{max})) \\
t_4 &= t_4/t_{max} \\
t_5 &= max(0, min(t_5 - 10, t_{max})) \\
t_5 &= t_5/t_{max} \\
x_d &= 2(10t_1^3 - 15t_1^4 + 6t_1^5 - 10t_2^3 + 15t_2^4 - 6t_2^5) \\
y_d &= 2(10t_3^3 - 15t_3^4 + 6t_3^5 - 10t_4^3 + 15t_4^4 - 6t_4^5) \\
z_d &= 2(10t_5^3 - 15t_5^4 + 6t_5^5) \\
\psi_d &= \frac{\pi}{3}(10t_5^3 - 15t_5^4 + 6t_5^5)
\end{aligned}
$$

Appendix B Components Calibration

B.1 ESC Calibration

Calibration of ESC is required for the first time and involves specifying the PWM signal range. It remains calibrated until it is used another time with another range. As ESCs operate with the APM board, there is either an automatic all-at-once calibration done by the APM board or manual one-by-one calibration which is done independently of the APM board. The required components for calibration are a fully charged LiPo battery, RC receiver and Transmitter, ESC, and a motor.

B.1.1 Calibration Steps

- Connect ESC to the throttle channel on the RC receiver.
- Connect high-voltage wires from the ESC to the motor.
- Turn on the transmitter and make sure that the throttle stick is set to maximum.
- Connect the LiPo battery to the ESC power wires.
- ESC releases two beeps. After these beeps, set the throttle stick to its minimum position.
- At the end of the beeping sequence, the ESC will release a long beep. It indicates the ESC has finally been set to the desired range by the long beep.
- Disconnect the battery and so the ESC control wires.

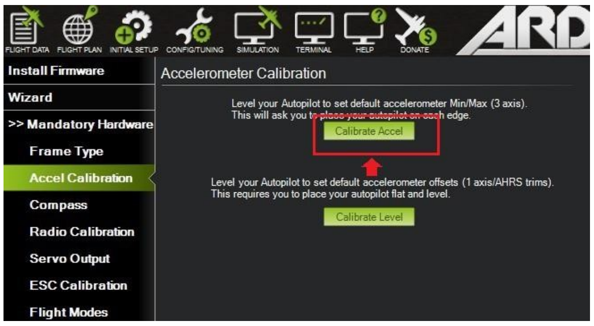
Same method is applied to all other ESCs. Keep in mind that the RC receiver must be powered synchronously with the ESC; this can be done by utilizing the APM connection itself.

B.2 IMU Sensors Calibration

Accelerometers and internal compass sensors need to be calibrated in the IMU. There is no need to calibrate the gyroscope.

B.2.1 Accelerometer Calibration

The accelerometer initially needs to be calibrated. For the first time, the accelerometer could not determine the direction of the quadcopter. Fortunately, there is an arrow at the top of the APM plate to easily idicate the orientation of the plate as well as facilitate the calibration steps. Assuming the APM is installed in the direction of the arrow, it can be calibrated using the APM task planning software. APM is associated with the Mission Planner software that details the calibration steps and assesses the correctness of the calibration process.



Accelerometer Calibration

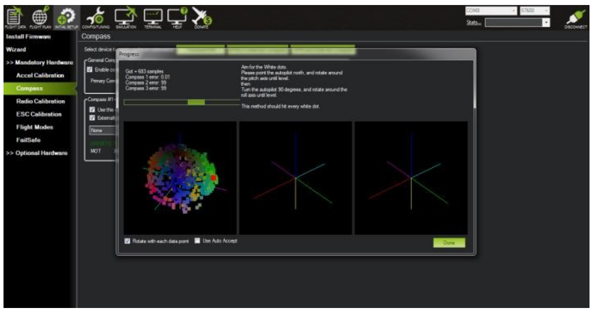
Level your Autopilot to set default accelerometer Min/Max (3 axis).
This will ask you to place your autopilot on each edge.

Level your Autopilot to set default accelerometer offsets (1 axis/AHRS trims).
This requires you to place your autopilot flat and level.

Figure B.1: Accelerometer Calibration.

B.2.2 Internal Compass Calibration

Compass calibration is necessary because the developed code relies on a sensors fusion algorithm that uses the compass and GPS to determine the attitude and altitude of the vehicle. Like the accelerometer, the compass must be calibrated for the first time through the APM mission planning software. The software provides the necessary calibration steps and verifies the correctness of the calibration.


Got = 683 samples
Compass 1 error: 0.01
Compass 2 error: 99
Compass 3 error: 99

Aim for the White dots.
Please point the autopilot north, and rotate around the pitch axis until level.
then
Turn the autopilot 90 degrees, and rotate around the roll axis until level.

This method should hit every white dot.

Figure B.2: Internal Compass Calibration.

B.2.3 RC Calibration

Calibration can be done using an Arduino board or using the APM mission planning software for the APM board, as mentioned earlier. The APM Mission Planner calibration procedure is a more reliable technique. The APM is connected to the channels of the RC receiver. The Mission Planner software provides the necessary steps for RC calibration. During calibration, the joysticks must be placed in their maximum and minimum positions to record the endpoints of the PWM signal. At the end of the calibration, the APM mission planning software provides a table with all the channel endpoints reached. This table can be used to check the operation of the RC.

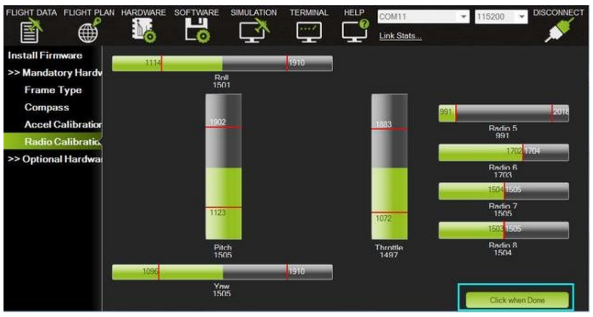


Figure B.3: RC Calibration.

Appendix C Communicating with Raspberry Pi via MAVLink

This appendix explains how to connect and configure a Raspberry Pi (RPi) so that it can communicate with an APM flight controller using the MAVLink protocol over a serial connection. Due to its memory requirements, RPi can be used to perform additional tasks such as image recognition which are not simply possible with APM.

C.1 Connecting the APM and RPi

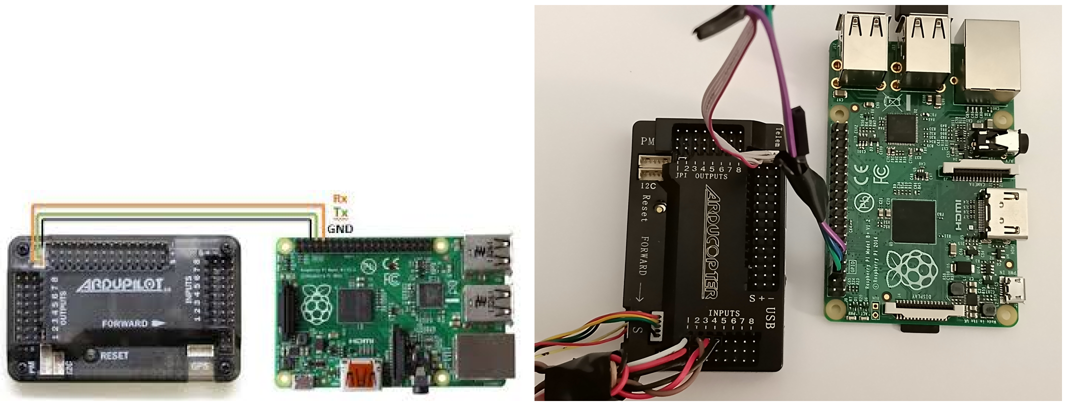


Figure C.1: Serial communication using Mavlink.

Remark 6 *Powering via USB is recommended as it is typically safer - because the input is regulated. If powering via USB, do not also connect the +5V pin as shown (still connect common ground).*

C.2 Setting up the flight controller

Connect to the flight controller with a ground station (i.e. APM Planner) and set the following parameters:

- SERIAL2_PROTOCOL = 2 (the default) to enable MAVLink 2 on the serial port.
- SERIAL2_BAUD = 921 so the flight controller can communicate with the RPi at 921600 baud.
- LOG_BACKEND_TYPE = 3 if you are using APSync to stream the dataflash log files to the RPi.

To test the RPi and APM are able to communicate with each other first ensure the RPi and Pixhawk are powered, then in a console on the RPi type:

```
sudo -s
mavproxy.py –master=/dev/ttyAMA0 –baudrate 57600 –aircraft MyCopter
```

Once MAVProxy has started you should be able to type in the following command to display the ARMING_CHECK parameters value

```
param show ARMING_CHECK
param set ARMING_CHECK 0
arm throttle
```

BIBLIOGRAPHY

[1] D. Popescu, F. Stoican, G. Stamatescu, L. Ichim, and C. Dragana, “Advanced uav–wsn system for intelligent monitoring in precision agriculture,” *Sensors*, vol. 20, no. 3, p. 817, 2020.

[2] R. Lopez Lopez, M. J. Batista Sanchez, M. Perez Jimenez, B. C. Arrue, and A. Ollero, “Autonomous uav system for cleaning insulators in power line inspection and maintenance,” *Sensors*, vol. 21, no. 24, p. 8488, 2021.

[3] V. Lippiello and J. Cacace, “Robust visual localization of a uav over a pipe-rack based on the lie group se(3),” *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 295–302, 2022.

[4] Z. Xu, D. Deng, and K. Shimada, “Autonomous uav exploration of dynamic environments via incremental sampling and probabilistic roadmap,” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2729–2736, 2021.

[5] M. Samir, S. Sharafeddine, C. M. Assi, T. M. Nguyen, and A. Ghrayeb, “Uav trajectory planning for data collection from time-constrained iot devices,” *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 34–46, 2019.

[6] Y. Chai, X. Liang, Z. Yang, and J. Han, “Energy-based nonlinear adaptive control for collaborative transportation systems,” *Aerospace Science and Technology*, vol. 126, p. 107510, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1270963822001845

[7] W. Stewart, E. Ajanic, M. Müller, and D. Floreano, “How to swoop and grasp like a bird with a passive claw for a high-speed grasping,” *IEEE/ASME Transactions on Mechatronics*, 2022.

[8] J. Fan, R. Lu, X. Yang, F. Gao, Q. Li, and J. Zeng, “Design and implementation of intelligent eod system based on six-rotor uav,” *Drones*, vol. 5, no. 4, p. 146, 2021.

[9] Q. Lindsey, D. Mellinger, and V. Kumar, “Construction with quadrotor teams,” *Autonomous Robots*, vol. 33, no. 3, pp. 323–336, 2012.

[10] C. Lenz, M. Schwarz, A. Rochow, J. Razlaw, A. S. Periyasamy, M. Schreiber, and S. Behnke, “Autonomous wall building with a ugv-uav team at mbzirc 2020,” in *2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2020, pp. 189–196.

[11] J. Borenstein, “The hoverbot–an electrically powered flying robot,” *Unpublished white paper, University of Michigan, Ann Arbor, MI. Available FTP: ftp://ftp. eecs. umich. edu/people/johannb/paper99. pdf*, 1992.

[12] N. Guenard, T. Hamel, and V. Moreau, “Dynamic modeling and intuitive control strategy for an” x4-flyer”,” in *2005 International Conference on Control and Automation*, vol. 1. IEEE, 2005, pp. 141–146.

[13] S. Bouabdallah, P. Murrieri, and R. Siegwart, “Design and control of an indoor micro quadrotor,” in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA’04. 2004*, vol. 5. IEEE, 2004, pp. 4393–4398.

[14] K. Rudin, M.-D. Hua, G. Ducard, and S. Bouabdallah, “A robust attitude controller and its application to quadrotor helicopters,” *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 10 379–10 384, 2011.

[15] G. Hoffmann, D. G. Rajnarayan, S. L. Waslander, D. Dostal, J. S. Jang, and C. J. Tomlin, “The stanford testbed of autonomous rotorcraft for multi agent control (starmac),” in *The 23rd Digital Avionics Systems Conference (IEEE Cat. No. 04CH37576)*, vol. 2. IEEE, 2004, pp. 12–E.

[16] “Draganflyer company,” Aug 2022. [Online]. Available: https://draganfly.com/

[17] “Ascending technologies humming-bird,” Aug 2022. [Online]. Available: http://www.asctec.de

[18] “Crazyflie,” Aug 2022. [Online]. Available: https://www.bitcraze.io/products/crazyflie-2-1/

[19] A. H. Ginting, O. Wahyunggoro, and A. I. Cahyadi, “Attitude control of quadrotor using pd plus feedforward controller on so (3),” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 1, pp. 566–575, 2018, https://doi.org/10.11591/ijece.v8i1.pp566-575.

[20] B. E. Jackson, K. Tracy, and Z. Manchester, “Planning with attitude,” *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5658–5664, 2021.

[21] J. Pliego-Jiménez, “Quaternion-based adaptive control for trajectory tracking of quadrotor unmanned aerial vehicles,” *International Journal of Adaptive Control and Signal Processing*, vol. 35, no. 5, pp. 628–641, 2021.

[22] J. Li and Y. Li, “Dynamic analysis and pid control for a quadrotor,” in *2011 IEEE International Conference on Mechatronics and Automation*. IEEE, 2011, pp. 573–578.

[23] C. S. Subudhi and D. Ezhilarasi, “Modeling and trajectory tracking with cascaded pd controller for quadrotor,” *Procedia computer science*, vol. 133, pp. 952–959, 2018.

[24] X. Heng, D. Cabecinhas, R. Cunha, C. Silvestre, and X. Qingsong, “A trajectory tracking lqr controller for a quadrotor: Design and experimental evaluation,” in *TENCON 2015-2015 IEEE region 10 conference*. IEEE, 2015, pp. 1–7.

[25] H. Wang, Z. Li, H. Xiong, and X. Nian, “Robust h attitude tracking control of a quadrotor uav on so (3) via variation-based linearization and interval matrix approach,” *ISA transactions*, vol. 87, pp. 10–16, 2019.

[26] A. Ataka *et al.*, “Controllability and observability analysis of the gain scheduling based linearization for uav quadrotor,” in *2013 International conference on robotics, biomimetics, intelligent computational systems*. IEEE, 2013, pp. 212–218.

[27] C. Fu, A. Sarabakha, E. Kayacan, C. Wagner, R. John, and J. M. Garibaldi, “Input uncertainty sensitivity enhanced nonsingleton fuzzy logic controllers for long-term navigation of quadrotor uavs,” *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 2, pp. 725–734, 2018.

[28] A. Al-Mahturi, F. Santoso, M. A. Garratt, and S. G. Anavatti, “Chapter 2 - modeling and control of a quadrotor unmanned aerial vehicle using type-2 fuzzy systems,” in *Unmanned Aerial Systems*, ser. Advances in Nonlinear Dynamics and Chaos (ANDC), A. Koubaa and A. T. Azar, Eds. Academic Press, 2021, pp. 25–46. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128202760000091

[29] M. Labbadi and M. Cherkaoui, “Robust adaptive global time-varying sliding-mode control for finite-time tracker design of quadrotor drone subjected to gaussian random parametric uncertainties and disturbances,” *International Journal of Control, Automation and Systems*, vol. 19, no. 6, pp. 2213–2223, 2021.

[30] X. Shi *et al.*, “Adaptive fractional-order smc controller design for unmanned quadrotor helicopter under actuator fault and disturbances,” *Ieee Access*, vol. 8, pp. 103 792–103 802, 2020.

[31] M. Bisheban and T. Lee, “Geometric adaptive control with neural networks for a quadrotor in wind fields,” *IEEE Transactions on Control Systems Technology*, vol. 29, no. 4, pp. 1533–1548, 2020.

[32] K. Liu, R. Wang, X. Wang, and X. Wang, “Anti-saturation adaptive finite-time neural network based fault-tolerant tracking control for a quadrotor uav with external disturbances,” *Aerospace Science and Technology*, vol. 115, p. 106790, 2021.

[33] R. Hedjar and M. A. Al Zuair, “Robust altitude stabilization of vtol-uav for payloads delivery,” *IEEE Access*, vol. 7, pp. 73 583–73 592, 2019.

[34] H. Zhao, W. Yang, and H. Zhu, “Unmanned aerial vehicles rescue system design and traffic model planning,” *Applied Sciences*, vol. 11, no. 21, p. 10481, 2021.

[35] H. Guan, X. Sun, Y. Su, T. Hu, H. Wang, H. Wang, C. Peng, and Q. Guo, “Uav-lidar aids automatic intelligent powerline inspection,” *International Journal of Electrical Power & Energy Systems*, vol. 130, p. 106987, 2021.

[36] S. Bouabdallah, “Design and control of quadrotors with application to autonomous flying,” Epfl, Tech. Rep., 2007.

[37] Q. Galvane, J. Fleureau, F.-L. Tariolle, and P. Guillotel, “Automated cinematography with unmanned aerial vehicles,” *arXiv preprint arXiv:1712.04353*, 2017.

[38] N. Elmeseiry, N. Alshaer, and T. Ismail, “A detailed survey and future directions of unmanned aerial vehicles (uavs) with potential applications,” *Aerospace*, vol. 8, no. 12, p. 363, 2021.

[39] D. Xia, L. Cheng, and Y. Yao, “A robust inner and outer loop control method for trajectory tracking of a quadrotor,” *Sensors*, vol. 17, no. 9, p. 2147, 2017.

[40] J. Park and N. Cho, “Collision avoidance of hexacopter uav based on lidar data in dynamic environment,” *Remote Sensing*, vol. 12, no. 6, p. 975, 2020.

[41] T. Ikeda, S. Yasui, M. Fujihara, K. Ohara, S. Ashizawa, A. Ichikawa, A. Okino, T. Oomichi, and T. Fukuda, “Wall contact by octo-rotor uav with one dof manipulator for bridge inspection,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 5122–5127.

[42] K. Bodie, M. Brunner, M. Pantic, S. Walser, P. Pfändler, U. Angst, R. Siegwart, and J. Nieto, “An omnidirectional aerial manipulation platform for contact-based inspection,” *arXiv preprint arXiv:1905.03502*, 2019.

[43] D. Aleksandrov and I. Penkov, “Energy consumption of mini uav helicopters with different number of rotors,” in *11th International Symposium” Topical Problems in the Field of Electrical and Power Engineering*, 2012, pp. 259–262.

[44] P. C. Garcia, P. Castillo, R. Lozano, and A. E. Dzul, *Modelling and control of mini-flying machines*. Springer Science & Business Media, 2005.

[45] J. Drew, “New search for vtol uavs may resurrect bell tiltrotor,” Dec 2019. [Online]. Available: https://www.flightglobal.com/new-search-for-vtol-uavs-may-resurrect-bell-tiltrotor/119404.article

[46] K. Nonami, F. Kendoul, S. Suzuki, W. Wang, and D. Nakazawa, *Development of Autonomous Quad-Tilt-Wing (QTW) Unmanned Aerial Vehicle: Design, Modeling, and Control*. Tokyo: Springer Japan, 2010, pp. 77–93. [Online]. Available: https://doi.org/10.1007/978-4-431-53856-1_4

[47] [Online]. Available: https://freewing.com/PivotingWing/

[48] “Honeywell rq-16 t-hawk,” Jul 2022. [Online]. Available: https://en.wikipedia.org/wiki/Honeywell_RQ-16_T-Hawk

[49] B. Li, W. Zhou, J. Sun, C.-Y. Wen, and C.-K. Chen, “Development of model predictive controller for a tail-sitter vtol uav in hover flight,” *Sensors*, vol. 18, no. 9, p. 2859, 2018.

[50] Festo, “Airjelly.” [Online]. Available: https://www.festo.com/us/en/e/about-festo/research-and-development/bionic-learning-network/highlights-from-2006-to-2009/airjelly-id_33841/

[51] I. Genuth, “Computer controlled pigeon,” Feb 2007. [Online]. Available: https://thefutureofthings.com/5459-computer-controlled-pigeon/

[52] M. H. Rosen, G. le Pivain, R. Sahai, N. T. Jafferis, and R. J. Wood, “Development of a 3.2 g untethered flapping-wing platform for flight energetics and control experiments,” in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 3227–3233.

[53] A. Abdullah, E. A. Bakar, and M. Z. M. Pauzi, “Monitoring of traffic using unmanned aerial vehicle in malaysia landscape perspective,” *Jurnal Teknologi*, vol. 76, no. 1, 2015.

[54] R. Daroya and M. Ramos, “Ndvi image extraction of an agricultural land using an autonomous quadcopter with a filter-modified camera,” in *2017 7th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*. IEEE, 2017, pp. 110–114.

[55] D. Falanga, K. Kleber, S. Mintchev, D. Floreano, and D. Scaramuzza, “The foldable drone: A morphing quadrotor that can squeeze and fly,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 209–216, 2018.

[56] R. Mahony, V. Kumar, and P. Corke, “Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor,” *IEEE Robotics and Automation magazine*, vol. 19, no. 3, pp. 20–32, 2012.

[57] S. Bouabdallah, A. Noth, and R. Siegwart, “Pid vs lq control techniques applied to an indoor micro quadrotor,” in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. IEEE, 2004, pp. 2451–2456.

[58] G. Hoffmann, H. Huang, S. Waslander, and C. Tomlin, “Quadrotor helicopter flight dynamics and control: Theory and experiment,” in *AIAA guidance, navigation and control conference and exhibit*, p. 6461.

[59] G. Szafranski and R. Czyba, “Different approaches of pid control uav type quadrotor,” 2011.

[60] I. D. Cowling, O. A. Yakimenko, J. F. Whidborne, and A. K. Cooke, “A prototype of an autonomous controller for a quadrotor uav,” in *2007 European Control Conference (ECC)*. IEEE, 2007, pp. 4001–4008.

[61] C. Liu, J. Pan, and Y. Chang, “Pid and lqr trajectory tracking control for an unmanned quadrotor helicopter: Experimental studies,” in *2016 35th Chinese Control Conference (CCC)*, 2016, pp. 10 845–10 850.

[62] H. Jafari, M. Zareh, J. Roshanian, and A. Nikkhah, “An optimal guidance law applied to quadrotor using lqr method,” *Transactions of the Japan Society for Aeronautical and Space Sciences*, vol. 53, no. 179, pp. 32–39, 2010.

[63] H. Lin, P. Sun, C. Cai, S. Lu, and H. Liu, “Secure lqg control for a quadrotor under false data injection attacks,” *IET Control Theory & Applications*, vol. 16, no. 9, pp. 925–934, 2022.

[64] O. Araar and N. Aouf, “Full linear control of a quadrotor uav, lq vs h_∞,” in *2014 UKACC International Conference on Control (CONTROL)*. IEEE, 2014, pp. 133–138.

[65] H. Liu, D. Li, Z. Zuo, and Y. Zhong, “Robust three-loop trajectory tracking control for quadrotors with multiple uncertainties,” *IEEE Transactions on Industrial Electronics*, vol. 63, no. 4, pp. 2263–2274, 2016.

[66] B. Mu, K. Zhang, and Y. Shi, “Integral sliding mode flight controller design for a quadrotor and the application in a heterogeneous multi-agent system,” *IEEE Transactions on Industrial Electronics*, vol. 64, no. 12, pp. 9389–9398, 2017.

[67] I. Ahmad, M. Liaquat, F. M. Malik, H. Ullah, and U. Ali, “Variants of the sliding mode control in presence of external disturbance for quadrotor,” *IEEE Access*, vol. 8, pp. 227 810–227 824, 2020.

[68] J.-J. Xiong and G. Zhang, “Discrete-time sliding mode control for a quadrotor uav,” *Optik*, vol. 127, no. 8, pp. 3718–3722, 2016.

[69] I. Kanellakopoulos, P. V. Kokotovic, and A. S. Morse, “Systematic design of adaptive controllers for feedback linearizable systems,” in *1991 American control conference*. IEEE, 1991, pp. 649–654.

[70] A. A. Mian and W. Daobo, “Modeling and backstepping-based nonlinear control strategy for a 6 dof quadrotor helicopter,” *Chinese Journal of Aeronautics*, vol. 21, no. 3, pp. 261–268, 2008.

[71] A. Das, F. Lewis, and K. Subbarao, “Backstepping approach for controlling a quadrotor using lagrange form dynamics,” *Journal of Intelligent and Robotic Systems*, vol. 56, no. 1, pp. 127–151, 2009.

[72] T. Madani and A. Benallegue, “Backstepping control for a quadrotor helicopter,” in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 3255–3260.

[73] W. Jasim and D. Gu, “Integral backstepping controller for quadrotor path tracking,” in *2015 International Conference on Advanced Robotics (ICAR)*. IEEE, 2015, pp. 593–598.

[74] J. Kim, S. A. Gadsden, and S. A. Wilkerson, “A comprehensive survey of control strategies for autonomous quadrotors,” *Canadian Journal of Electrical and Computer Engineering*, vol. 43, no. 1, pp. 3–16, 2019.

[75] V. Mistler, A. Benallegue, and N. M’sirdi, “Exact linearization and noninteracting control of a 4 rotors helicopter via dynamic feedback,” in *Proceedings 10th IEEE international workshop on robot and human interactive communication. Roman 2001 (Cat. no. 01th8591)*. IEEE, 2001, pp. 586–593.

[76] D. Lee, H. Jin Kim, and S. Sastry, “Feedback linearization vs. adaptive sliding mode control for a quadrotor helicopter,” *International Journal of control, Automation and systems*, vol. 7, no. 3, pp. 419–428, 2009.

[77] H. Voos, “Nonlinear control of a quadrotor micro-uav using feedback-linearization,” in *2009 IEEE International Conference on Mechatronics*. IEEE, 2009, pp. 1–6.

[78] A. Freddi, A. Lanzon, and S. Longhi, “A feedback linearization approach to fault tolerance in quadrotor vehicles,” *IFAC proceedings volumes*, vol. 44, no. 1, pp. 5413–5418, 2011.

[79] N. T. Nguyen, I. Prodan, and L. Lefevre, “Multi-layer optimization-based control design for quadcopter trajectory tracking,” in *2017 25th Mediterranean Conference on Control and Automation (MED)*. IEEE, 2017, pp. 601–606.

[80] G. Garimella, M. Sheckells, and M. Kobilarov, “Robust obstacle avoidance for aerial platforms using adaptive model predictive control,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5876–5882.

[81] P. Ru and K. Subbarao, “Nonlinear model predictive control for unmanned aerial vehicles,” *Aerospace*, vol. 4, no. 2, p. 31, 2017.

[82] K. J. Åström and B. Wittenmark, *Adaptive control*. Courier Corporation, 2013.

[83] S. Li, B. Li, and Q. Geng, “Adaptive sliding mode control for quadrotor helicopters,” in *Proceedings of the 33rd Chinese control conference*. IEEE, 2014, pp. 71–76.

[84] H. Hassani, A. Mansouri, and A. Ahaitouf, "A new robust adaptive sliding mode controller for quadrotor uav flight," in *2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS)*. IEEE, 2020, pp. 1–6.

[85] K. Eliker, S. Grouni, M. Tadjine, and W. Zhang, "Quadcopter nonsingular finite-time adaptive robust saturated command-filtered control system under the presence of uncertainties and input saturation," *Nonlinear Dynamics*, vol. 104, no. 2, pp. 1363–1387, 2021.

[86] C. Li, Y. Wang, and X. Yang, "Adaptive fuzzy control of a quadrotor using disturbance observer," *Aerospace Science and Technology*, vol. 128, p. 107784, 2022.

[87] H. Lee and H. J. Kim, "Robust control of a quadrotor using takagi-sugeno fuzzy model and an lmi approach," in *2014 14th International Conference on Control, Automation and Systems (ICCAS 2014)*. IEEE, 2014, pp. 370–374.

[88] J. Muliadi and B. Kusumoputro, "Neural network control system of uav altitude dynamics and its comparison with the pid control system," *Journal of Advanced Transportation*, vol. 2018, 2018.

[89] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[90] D. Bertsekas, *Reinforcement learning and optimal control*. Athena Scientific, 2019.

[91] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter, "Control of a quadrotor with reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2096–2103, 2017.

[92] Z. Jia, J. Yu, Y. Mei, Y. Chen, Y. Shen, and X. Ai, "Integral backstepping sliding mode control for quadrotor helicopter under external uncertain disturbances," *Aerospace Science and Technology*, vol. 68, pp. 299–307, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1270963817308878

[93] G. V. Raffo, M. G. Ortega, and F. R. Rubio, "Mpc with nonlinear h_∞ control for path tracking of a quad-rotor helicopter," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 8564–8569, 2008.

[94] S. Jalili, B. Rezaie, and Z. Rahmani, "A novel hybrid model predictive control design with application to a quadrotor helicopter," *Optimal Control Applications and Methods*, vol. 39, no. 4, pp. 1301–1322, 2018.

[95] L. Zhou, S. Xu, H. Jin, and H. Jian, "A hybrid robust adaptive control for a quadrotor uav via mass observer and robust controller," *Advances in Mechanical Engineering*, vol. 13, no. 3, p. 16878140211002723, 2021.

[96] A. Chovancová, T. Fico, Ľuboš Chovanec, and P. Hubinsk, "Mathematical modelling and parameter identification of quadrotor (a survey)," *Procedia Engineering*, vol. 96, pp. 172–181, 2014, modelling of Mechanical and Mechatronic Systems. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877705814031981

[97] G. Charland-Arcand, "Contrôle non linéaire par backstepping d'un hélicoptère de type quadrotor pour des applications autonomes," Ph.D. dissertation, École de technologie supérieure, 2014.

[98] I. Cheeseman and W. Bennett, “The effect of the ground on a helicopter rotor in forward flight,” 1955.

[99] X. Zhang, Y. Fang, X. Zhang, J. Jiang, and X. Chen, “A novel geometric hierarchical approach for dynamic visual servoing of quadrotors,” *IEEE Transactions on Industrial Electronics*, vol. 67, no. 5, pp. 3840–3849, 2019.

[100] J. Wehbeh and I. Sharf, “An mpc formulation on $so(3)$ for a quadrotor with bidirectional thrust and nonlinear thrust constraints,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4945–4952, 2022.

[101] P.-H. Chiang and J.-T. Yu, “Attitude control of quadrotor uavs using special orthogonal group so (3) and gaussian error function,” in *2021 International Automatic Control Conference (CACS)*. IEEE, 2021, pp. 1–5.

[102] S. R. Nekoo, J. Á. Acosta, and A. Ollero, “Quaternion-based state-dependent differential riccati equation for quadrotor drones: Regulation control problem in aerobatic flight,” *Robotica*, pp. 1–16, 2022.

[103] C. Zha, X. Ding, Y. Yu, and X. Wang, “Quaternion-based nonlinear trajectory tracking control of a quadrotor unmanned aerial vehicle,” *Chinese Journal of Mechanical Engineering*, vol. 30, no. 1, pp. 77–92, 2017.

[104] Q. Lu, B. Ren, and S. Parameswaran, “Uncertainty and disturbance estimator-based global trajectory tracking control for a quadrotor,” *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 3, pp. 1519–1530, 2020.

[105] Y.-C. Choi and H.-S. Ahn, “Nonlinear control of quadrotor for point tracking: Actual implementation and experimental tests,” *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 3, pp. 1179–1192, 2015, doi: 10.1109/TMECH.2014.2329945.

[106] B. Graf, “Quaternions and dynamics,” *arXiv preprint arXiv:0811.2889*, 2008.

[107] J. P. Ortiz, L. I. Minchala, and M. J. Reinoso, “Nonlinear robust h-infinity pid controller for the multivariable system quadrotor,” *IEEE Latin America Transactions*, vol. 14, no. 3, pp. 1176–1183, 2016.

[108] W. Mizouri, S. Najar, L. Bouabdallah, and M. Aoun, “Dynamic modeling of a quadrotor uav prototype,” in *New Trends in Robot Control*. Springer, 2020, pp. 281–299.

[109] M. E. Guerrero-Sánchez, D. A. Mercado-Ravell, R. Lozano, and C. D. García-Beltrán, “Swing-attenuation for a quadrotor transporting a cable-suspended payload,” *ISA Transactions*, vol. 68, pp. 433–449, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0019057817301635

[110] H. Khebbache, “Tolérance aux défauts via la méthode backstepping des systèmes non linéaires: application système uav de type quadrirotor,” Ph.D. dissertation, 2018.

[111] A. Polyakov and L. Fridman, “Stability notions and lyapunov functions for sliding mode control systems,” *Journal of the Franklin Institute*, vol. 351, no. 4, pp. 1831–1865, 2014, special Issue on 2010-2012 Advances in Variable Structure Systems and Sliding Mode Algorithms. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0016003214000040

[112] H. K. Khalil, *Nonlinear control*. Pearson New York, 2015, vol. 406.

[113] C. J. Fallaha, M. Saad, H. Y. Kanaan, and K. Al-Haddad, “Sliding-mode robot control with exponential reaching law,” *IEEE Transactions on industrial electronics*, vol. 58, no. 2, pp. 600–610, 2010.

[114] A. Mehta and B. Bandyopadhyay, *Frequency-shaped and observer-based discrete-time sliding mode control*. Springer, 2015.

[115] D. KB and S. Thomas, “Power rate exponential reaching law for enhanced performance of sliding mode control,” *International Journal of Control, Automation and Systems*, vol. 15, no. 6, pp. 2636–2645, 2017.

[116] P. Kachroo and M. Tomizuka, “Chattering reduction and error convergence in the sliding-mode control of a class of nonlinear systems,” *IEEE Transactions on automatic control*, vol. 41, no. 7, pp. 1063–1068, 1996.

[117] C. Kirches, “The direct multiple shooting method for optimal control,” in *Fast Numerical Methods for Mixed-integer Nonlinear Model-Predictive control*. Springer, 2011, pp. 13–29.

[118] G. Hicks and W. Ray, “Approximation methods for optimal control synthesis,” *The Canadian Journal of Chemical Engineering*, vol. 49, no. 4, pp. 522–528, 1971.

[119] P. Deuflhard, “A modified newton method for the solution of ill-conditioned systems of nonlinear equations with application to multiple shooting,” *Numerische Mathematik*, vol. 22, no. 4, pp. 289–315, 1974.

[120] T. Tsang, D. Himmelblau, and T. F. Edgar, “Optimal control via collocation and non-linear programming,” *International Journal of Control*, vol. 21, no. 5, pp. 763–768, 1975.

[121] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, “Fast direct multiple shooting algorithms for optimal robot control,” in *Fast motions in biomechanics and robotics*. Springer, 2006, pp. 65–93.

[122] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, “CasADi – A software framework for nonlinear optimization and optimal control,” *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[123] A. Wächter and L. T. Biegler, “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,” *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[124] D. Meradi, Z. A. Benselama, and R. Hedjar, “Trajectory tracking performance with two nonlinear controllers of quadrotor under wind effect,” in *2020 4th International Conference on Advanced Systems and Emergent Technologies (IC_ASET)*, 2020, pp. 50–55.

[125] L. C. McNinch and H. Ashrafiuon, “Predictive and sliding mode cascade control for unmanned surface vessels,” in *Proceedings of the 2011 American Control Conference*. IEEE, 2011, pp. 184–189.

[126] Z. Tian, J. Yuan, X. Zhang, L. Kong, and J. Wang, “Modeling and sliding mode predictive control of the ultra-supercritical boiler-turbine system with uncertainties and input constraints,” *ISA transactions*, vol. 76, pp. 43–56, 2018.

[127] H. B. Mansour, K. Dehri, and A. S. Nouri, “Comparison between predictive sliding mode control and sliding mode control with predictive sliding function,” in *International Conference on Electrical Engineering and Control Applications*. Springer, 2016, pp. 80–97.

[128] D. Meradi, Z. A. Benselama, and R. Hedjar, “A predictive sliding mode control for quadrotor’s tracking trajectory subject to wind gusts and uncertainties,” *International Journal of Electrical and Computer Engineering*, vol. 12, no. 5, p. 4861, 2022.

[129] D. Meradi, Z. A. Benselama, R. Hedjar, and N. E.-H. Gabour, “Quaternion-based nonlinear mpc for quadrotor’s trajectory tracking and obstacles avoidance,” in *2022 2nd International Conference on Advanced Electrical Engineering (ICAEE)*, 2022, pp. 1–6.

[130] J.-J. E. Slotine, W. Li *et al.*, *Applied nonlinear control*. Prentice hall Englewood Cliffs, NJ, 1991, vol. 199, no. 1.

[131] W. Gao, Y. Wang, and A. Homaifa, “Discrete-time variable structure control systems,” *IEEE transactions on Industrial Electronics*, vol. 42, no. 2, pp. 117–122, 1995.

[132] G. Costa and G. Fogli, *Symmetries and group theory in particle physics: An introduction to space-time and internal symmetries*. Springer Science & Business Media, 2012, vol. 823.

[133] S. Sarpturk, Y. Istefanopulos, and O. Kaynak, “On the stability of discrete-time sliding mode control systems,” *IEEE Transactions on Automatic Control*, vol. 32, no. 10, pp. 930–932, 1987.

[134] A. Musa, L. R. Sabug, and A. Monti, “Robust predictive sliding mode control for multiterminal hvdc grids,” *IEEE Transactions on Power Delivery*, vol. 33, no. 4, pp. 1545–1555, 2018.

[135] J.-S. Zhou, Z.-Y. Liu, and R. Pei, “A new nonlinear model predictive control scheme for discrete-time system based on sliding mode control,” in *Proceedings of the 2001 American Control Conference.(Cat. No. 01CH37148)*, vol. 4. IEEE, 2001, pp. 3079–3084.

[136] “Discrete wind gust model.” [Online]. Available: https://www.mathworks.com/help/aeroblks/drydenwindturbulencemodeldiscrete.html

[137] D. Brescianini, M. Hehn, and R. D’Andrea, “Nonlinear quadrocopter attitude control: Technical report,” ETH Zurich, Tech. Rep., 2013.

[138] D. Brescianini and R. D’Andrea, “Tilt-prioritized quadrocopter attitude control,” *IEEE Transactions on Control Systems Technology*, 2018.

[139] N. Cao and A. F. Lynch, “Inner–outer loop control for quadrotor uavs with input and state constraints,” *IEEE Transactions on Control Systems Technology*, vol. 24, no. 5, pp. 1797–1804, 2015.

[140] S. O. Ariyibi and O. Tekinalp, “Quaternion-based nonlinear attitude control of quadrotor formations carrying a slung load,” *Aerospace Science and Technology*, vol. 105, p. 105995, 2020.

[141] M. S. Andrle and J. L. Crassidis, “Geometric integration of quaternions,” *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 6, pp. 1762–1767, 2013.

[142] “Raspberry pi,” https://www.raspberrypi.org/, accessed: 17 mai 2022.

[143] “Choosing a ground station.” [Online]. Available: https://ardupilot.org/copter/docs/common-choosing-a-ground-station.html

[144] “Pre-arm safety checks.” [Online]. Available: https://ardupilot.org/copter/docs/common-prearm-safety-checks.html