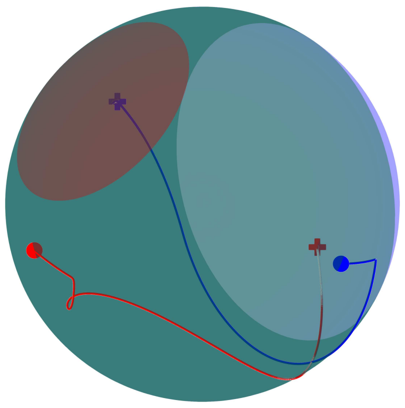


Computationally Efficient and Safe Constrained Attitude Control for Spacecraft Slewing with Control Barrier Functions

José Henrique Faísco Pereira

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisors: Prof. Daniel de Matos Silvestre
Profa. Rita Maria Mendes de Almeida Correia da Cunha

Examination Committee

Chairperson:
Supervisor: Prof. Daniel de Matos Silvestre
Members of the Committee: Prof. António Pedro Rodrigues de Aguiar
Profa. Rita Maria Mendes de Almeida Correia da Cunha

June 2023

Declaration

Acknowledgments

I would like to express my heartfelt gratitude to those who have supported and inspired me throughout my academic journey. First and foremost, I am deeply grateful to my parents for their unwavering support and for giving me the opportunity to pursue higher education. Your selfless sacrifices have allowed me to achieve my dreams, and for that, I am forever indebted. To my sister and brother-in-law, thank you for being a constant source of motivation and for always believing in my abilities. Your faith in me has been a driving force behind my perseverance.

I am especially grateful to two professors whose courses have had a significant impact on my studies and are central to this thesis. Their ability to present complex topics in an accessible manner has deeply influenced the theoretical sections of my work. First, I would like to thank Prof. João Xavier. His courses on Convex Optimization made me realize the practicality of good theory. Convex Optimization is a wonderful field, but it was Prof. Xavier's passion and pedagogical skills that revealed its beauty to me. Without his enthusiasm and clear explanations, the elegance of this subject might have remained hidden from me. I would also like to extend my gratitude to Prof. José Natário. His courses on Geometry provided me with a deeper understanding of the mathematics essential to this thesis. Thank you for your patience, open-door policy, and willingness to help students grasp challenging concepts.

I am sincerely thankful to my dissertation supervisor, Prof. Daniel Silvestre, for his invaluable guidance and support throughout this process. Additionally, I would like to thank Pedro Lourenço from GMV for his insights and contributions to my research.

I would like to express my profound appreciation to my friends and colleagues from Formula Student Técnico. Working, suffering, and celebrating alongside you has produced some of the most satisfying moments of my life. If I can wish for anything in my future career, it is to work alongside driven and capable individuals like you again.

This work was partially supported by the Portuguese Fundação para a Ciência e a Tecnologia (FCT) through project FirePuma (DOI: 10.54499/PCIF/MPG/0156/2019), through LARSyS FCT funding (DOI: 10.54499/LA/P/0083/2020, 10.54499/UIDP/50009/2020, and 10.54499/UIDB/50009/2020) and through COPELABS, University Lusófona project UIDB/04111/2020.

Abstract

Constraints of an optimization problem provide a natural framework to encode operational and safety requirements of aerospace missions. However, the use of real-time optimization is rare in the aerospace industry, despite several academic publications that attempt to apply these methods in aerospace applications. This academic-industry gap is due to the high investment cost of aerospace missions, which require provable safety and stability guarantees that are very hard to obtain with real-time optimization methods. Additionally, the limitations of computational hardware used in space missions, and the nonconvex constraints of spacecraft dynamics and mission requirements result in NP-hard optimization problems, with no guarantees of real-time solutions.

This thesis attempts to bridge the academic-industry gap by showing how Control Lyapunov Functions (CLFs) and Control Barrier Functions (CBFs) can be used to implement real-time optimization-based control laws that are provably safe, stable and computationally efficient. It provides an intuitive description of CLFs and CBFs for engineers and details their use in designing an attitude controller for constrained attitude slew maneuvers, along with a concise theory of attitude representation using quaternions. Simulations validate safety and stability claims, and the approach is compared to a state-of-the-art method using Successive Convexification (SCvx) and Model Predictive Control/Controller (MPC).

Keywords

Control Barrier Functions; Control Lyapunov Functions; Successive Convexification; Model Predictive Control; Attitude Control; Safety Constraints

Resumo

As restrições de um problema de otimização proporcionam uma estrutura natural para codificar os requisitos operacionais e de segurança das missões aeroespaciais. No entanto, a utilização de otimização em tempo real é rara na indústria aeroespacial, apesar de várias publicações académicas que tentam aplicar estes métodos em aplicações aeroespaciais. Esta lacuna entre a academia e a indústria deve-se ao elevado custo de investimento das missões aeroespaciais, que exigem garantias de segurança e estabilidade prováveis que são muito difíceis de obter com métodos de otimização em tempo real. Adicionalmente, as limitações do hardware computacional utilizado nas missões espaciais, e as restrições não convexas da dinâmica das naves espaciais e dos requisitos da missão resultam em problemas de otimização *NP-hard*, sem garantias de soluções em tempo real.

Esta tese tenta colmatar a lacuna entre a academia e a indústria, mostrando como Control Lyapunov Functions (CLFs) e Control Barrier Functions (CBFs) podem ser usados para implementar leis de controlo baseadas em otimização em tempo real que são comprovadamente seguras, estáveis e computacionalmente eficientes. Fornece uma descrição intuitiva de CLFs e CBFs para engenheiros e detalha a sua utilização na conceção de um controlador de atitude para manobras de alteração de atitude sujeitas a restrições, juntamente com uma teoria concisa da representação da atitude usando quaterniões. Simulações validam as afirmações de segurança e estabilidade, e a abordagem é comparada com um método de estado-da-arte que usa Successive Convexification (SCvx) e Model Predictive Control/Controller (MPC).

Palavras Chave

Control Barrier Functions; Control Lyapunov Functions; Successive Convexification; Model Predictive Control; Controlo de Atitude; Restrições de Segurança

Contents

List of Figures

List of Tables

Notation

$\mathbb{R}^n$	Euclidean n-space.
$\boldsymbol{\alpha}, ..., \boldsymbol{\omega}, \mathbf{a}, ..., \mathbf{z}$	Lowercase bold letters denote elements of $\mathbb{R}^n$ or functions whose domain is $\mathbb{R}^n$, represented as either n-tuples or column matrices.
$\boldsymbol{\Delta}, ..., \boldsymbol{\Omega}, \mathbf{A}, ..., \mathbf{Z}$	Uppercase bold letters denote $\mathbb{R}^{n\times m}$ matrices or functions whose domain is $\mathbb{R}^{n\times m}$, with $m > 1$.
$\mathbf{I}_n$	The $n \times n$ identity matrix, usually we will leave the dimension implicit and simply write $\mathbf{I}$.
$\mathbf{0}_{n\times m}$	The zero element of $\mathbb{R}^{n\times m}$, usually we will leave the dimension implicit and simply write $\mathbf{0}$.
${}^{\mathcal{B}}\mathbf{p}$	The coordinate vector of vector $\mathbf{p} \in \mathbb{R}^n$ with respect to some basis $\mathcal{B}$.
$\mathbf{A}^\top$	The transpose of matrix $\mathbf{A}$.
$\| \circ \|_p$	The p-norm, $p \in \mathbb{N} \cup \{\infty\}$.
$\| \circ \|$	The Euclidean norm, $\| \circ \| = \| \circ \|_2$.
$\mathcal{S}^n$	Unit sphere of dimension n, embedded in $\mathbb{R}^{n+1}$.
$SO(n)$	The group of special orthogonal $n \times n$ matrices.
$\mathfrak{so}(n)$	The set of $n \times n$ skew-symmetric matrices.
$\partial\mathcal{C}$	The boundary of set $\mathcal{C}$.
int$(\mathcal{C})$	The interior of set $\mathcal{C}$.
$\overline{\mathcal{C}}$	The complement of set $\mathcal{C}$.
$[\,\circ\,]_\times$	The map from $\mathbb{R}^3 \to \mathfrak{so}(3)$ defined as $[\mathbf{u}]_\times\mathbf{v} = \mathbf{u} \times \mathbf{v}$.
∇V	The gradient of function $V : \mathbb{R}^n \to \mathbb{R}$. The gradient of a function is a vector field $\nabla V : \mathbb{R}^n \to \mathbb{R}^n$, so it is represented as a column matrix.
$\frac{\partial V}{\partial x}$	The partial derivative of $V : \mathbb{R}^n \to \mathbb{R}$ with respect to vector $\mathbb{R}^n$. We consider its codomain to be the dual space of $\mathbb{R}^n$, so it is represented as a row matrix, i.e. $\partial V/\partial\mathbf{x} = \nabla V^\top$.
$L_{\mathbf{f}}V$	The Lie derivative of a function $V : \mathbb{R}^n \to \mathbb{R}$ with respect to a vector field $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$. We consider it equivalent to $\nabla V \cdot \mathbf{f}$.
$\mathcal{L}_\beta$	The LogSumExp function with parameter β.
$\otimes$	The quaternion product defined on $\mathcal{S}^3$. An element $\mathbf{q} = (q_s, \mathbf{q}_v)$ of $\mathcal{S}^3$ endowed with $\otimes$ is called a quaternion, where q_s is the scalar part and $\mathbf{q}_v$ is the vector part of the quaternion. Two quaternions are multiplied using $\otimes$ according to Hamilton's definition.
$\angle(\mathbf{u}, \mathbf{v})$	The angle between vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^3$.

Acronyms

CG&C	Computational Guidance and Control
CBF	Control Barrier Function
HOCBF	High-Order Control Barrier Function
CLF	Control Lyapunov Function
ESA	European Space Agency
$\mathbf{H}_\infty$	H-infinity
IVP	Initial Value Problem
LQR	Linear Quadratic Regulator
MC	Monte Carlo
MPC	Model Predictive Control/Controller
ODE	Ordinary Differential Equation
OSAM	On-orbit Servicing, Assembly and Manufacturing
OSE	Optimization in Space Engineering
QP	Quadratic Program
SCvx	Successive Convexification
w.r.t.	with respect to
PWTB	passive world-to-body or global-to-local
PBTW	passive body-to-world or local-to-global
NASA	National Aerospace Agency
JPL	Jet Propulsion Laboratory
ISS	International Space Station
SOC-i	Satellite for Optimal Control and Imaging

SCP	Sequential Convex Programming
SOCP	Second Order Cone Program
RK4	Runge-Kutta Fourth-Order
ZOH	Zero-Order Hold
FOH	First-Order Hold
LTV	Linear Time-Varying

1 Introduction

Contents

1.1 Motivation

In the past couple of decades, improvements in computing hardware and optimization software libraries have sparked increased interest in implementing optimization techniques in the methods used for controlling spacecraft [3]. The current state-of-the-art in the field of optimization allows the implementation of feedback laws that are a solution of an optimization problem that can be carried out on onboard hardware. These strategies go beyond the computational requirements of tried and true methods for developing closed-form analytical control laws, such as the Linear Quadratic Regulator (LQR) and H-infinity ($\mathbf{H}_\infty$). Thus, the term Computational Guidance and Control (CG&C) has been coined to differentiate techniques of real-time optimization-based control from the closed-form methods used to obtain fixed feedback gains [4].

As an example of the growing interest of optimization-based methods in aerospace, the European Space Agency (ESA) has established the Optimization in Space Engineering (OSE) working group to disseminate existing optimization tools and techniques for space applications, which has resulted in five workshops from 2013 to 2019 [5]. Furthermore, the of use real-time optimization-based control in the domain of aerospace applications has been extensively researched, with a non-exhaustive list including: entry, descent, and landing guidance [6] [3] [7] [8]; autonomous rendezvous and docking [3] [9] [7]; formation flying and reconfiguration for swarms of spacecraft [10] [7]; geostationary satellite station-keeping [7]; spacecraft attitude reorientation [11] [12]; among others.

Regardless of the interest from the academia in optimization-based CG&C methods for space-related applications, deployment of optimization techniques is still rather limited in the industrial setting. Indeed, the few consistent operational uses of real-time optimization-based control software for critical systems are limited to the self-landing rocket boosters of SpaceX and Blue Origin [8] and the autonomous rendezvous capabilities of SpaceX's Dragon capsule [3]. There are two main factors that account for this separation between academia and industry practices, namely, the requirements for a guaranteed feasible solution in bounded time and for guaranteed stability and safety.

The difficulty in fulfilling the first requirement, is because most of the dynamics for spacecraft applications, such as translational and attitude dynamics, are modeled as nonlinear differential equations. Furthermore, several of the operational and safety requirements, with which the control systems have to comply, are most naturally expressed as nonconvex constraints. For example, in the previously mentioned applications, some of the nonconvex constraints translate impulsive inputs, collision-avoidance and forbidden/mandatory attitude constraints. This results in optimization problems that are nondeterministic polynomial hard (NP-hard), which means that there is no known time bound for the termination of the solver computation [7].

Thankfully, several of the aforementioned applications have convex reformulations or algorithms that attempt to solve the nonconvex problem by iteratively solving a series of convex approximations [7]. Convex optimization problems are polynomial hard, which translates in the existence of efficient algorithms for this class of problems. However, depending on the actual problem, the time to obtain the solution might be higher than the allotted slot to produce the control action.

Given the real-time requirement, we must have assurance that the time the algorithm takes to solve the optimization problem is lower than the sampling time of the controller. In some of the previously mentioned applications, such as docking, close proximity operations like On-orbit Servicing, Assembly and Manufacturing (OSAM), or maneuvers where safety is a concern or that must be performed quickly (such as quick attitude reorientations or slew maneuvers), the sampling rates for the controller are generally higher than $1\,\mathrm{Hz}$, requiring optimization-based techniques to be able to be solved quickly. Moreover, computational resources are placed at an even higher premium when we are dealing with deep space applications, where performance of the hardware must be traded off for robustness against the harsh environment.

The imperative of the second of the previously mentioned requirements, is due to the high investment cost associated with aerospace applications. As such, real-time control algorithms must have guaranteed stability, robustness and safety margins. These guarantees must be theoretically proven and cannot be limited to simple numerical simulations [13]. As optimization-based control laws lack a closed-form analytical expression, the endeavour of finding a theoretical result becomes a challenging issue.

The constrained attitude control problem makes for an interesting case study for the application of CG&C methods. It is a practically relevant problem in science missions, such as the Cassini and the New Horizons missions [14] [11] [15]. These science missions possess sensitive optical instruments that cannot point to strong light sources, such as the Sun. Other practical reasons to have attitude constraints also include maintaining an antenna pointing to another satellite, or the Earth, to maintain communication without requiring expensive omnidirectional antennas. Another reason is to maximize a star tracker sensor's utility, by keeping it pointed away from the Earth, Sun or Moon [16]. Besides being of practical interest, the nonlinear attitude kinematics and rotational dynamics, along with the high sampling rates required for attitude controllers, make it a challenging archetypical control problem on which to test the performance of CG&C methods.

Historically, control algorithms to address the constrained attitude problem have relied on heuristic methods. For example, for the Cassini spacecraft, a ground-commanded attitude trajectory has to be tracked and a constraint monitor verifies if the commanded attitude and angular velocities do not result in the attitude constraints being violated. If the constraint monitor concludes a violation is imminent, an avoidance algorithm then attempts to modify the commanded attitude and angular velocities such that the attitude moves away from the forbidden attitude region, while trying to stay close to the commanded trajectory [14]. The limited safety guarantees of such methods usually require a human-in-the-loop, to monitor possible failures, such as when an imaging instrument on New Horizons was nearly destroyed when an attitude maneuver inadvertently pointed it at the sun [11]. This limits the autonomy of the system. The reactive nature of these heuristic methods also makes them sub-optimal [12].

1.2 Contribution

In this thesis, we propose a control algorithm for the constrained attitude control problem that has theoretical guarantees of stability and safety, while also taking milliseconds to solve on modern personal computing hardware. We do this by leveraging two tools from nonlinear control theory: Control Lyapunov Functions (CLFs) and Control Barrier Functions (CBFs). A CLF is a nonlinear control design method based on Lyapunov functions and, hence, is provably stable [17]. A CBF is a method which (when "safety" is expressed as the system being able to avoid a set of undesirable states) is able to provide constraints on the control which makes the set of safe states positively invariant and hence results in a provably safe controller [18] [19]. For control affine systems, CLFs and CBFs can be used together as constraints for a Quadratic Program (QP), which is the second easiest form of convex optimization problems to solve [3].

We compare our CLF-CBF-QP constrained attitude controller to a different method that uses Successive Convexification (SCvx), a state-of-the-art trajectory optimization algorithm. This method is currently the only one scheduled to be implemented in real-life use cases, to the best of our knowledge. One of the projects that will make use of SCvx is the Satellite for Optimal Control and Imaging (SOC-i) CubeSat, a project from the University of Washington planned for launch this year [11]. The SCvx method is also a proposed solution for ESA's Comet Interceptor mission planned for launch in 2028 [12].

1.3 Outline

In Chapter 2, we lay the theoretical foundations necessary to address the constrained attitude control problem. The purpose of this chapter is to derive the quaternion attitude representation, along with its kinematics equations, and the rigid body dynamics, starting from first principles.

In Chapter 3, we delve into CLFs and CBFs, which are pivotal to the thesis. We provide an intuitive explanation of CLFs, deriving it from Lyapunov's stabillity criterion. We then show how the CLF constraint can be implement in optimization-based controllers. Subsequently, we present CBFs in a similar intuitive manner, deriving it from Nagumo's theorem for positive invariance. We then show how the CBF constraints can be used in conjuction with the CLF constraints, to formulate an optimization-based controller that is provably stable and safe. This chapter also addresses the limitations of these methods and introduces High-Order Control Barrier Functions (HOCBFs) to address the problem of controlling system of relative degree greater than 1.

After laying out the necessary theoretical foundations in Chapters 2 and 3, we address the constrained attitude control problem in Chapter 4. We begin by formulating the problem in Section 4.1. Next, in Section 4.2, we describe the state-of-the-art method based on SCvx and Model Predictive Control/Controller (MPC), which we will use for comparison with our CLF-CBF-QP-based solution, detailed in Section 4.3. Finally, in Section 4.4, we perform Monte Carlo simulations to analyze and compare the results of our proposed solution with those obtained from the SCvx and MPC solution.

Chapter 5 encapsulates the key findings of the dissertation, underscoring the efficacy of CBFs in enforcing safety constraints in attitude control. It reflects on the contributions of this research and concludes with suggestions for future research directions, building upon the outcomes of this study.

2 Theory of Attitude and Rotations

Contents

L'algèbre n'est qu'une géométrie écrite, la géométrie n'est qu'une algèbre figurée.

Sophie Germain

In this chapter, we present the theoretical foundations necessary to mathematically model the constrained attitude problem. During the research phase for this work, we found that most resources that present the theory of 3D rotations and attitude representations for spacecraft applications lack a proper theoretical explanation for the formulas they use, presenting them *ex nihilo*.

It is understandable that engineering focused resources would not dwell on mathematical derivations. However, as we will see, the mathematical theory of rotations and attitude representations allow for different conventions to be used when describing what a rotation is or how to represent attitude. This leads to subtle differences in the resulting formulas and, as a consequence, different conventions are used across different fields and sometimes even within the same field [20] [21]. Thus, the formulas for rotation and attitude representations cannot be treated as "plug-and-play", as they require proper understanding of how the different conventions relate to each other, especially in interdisciplinary domains, where one has to consolidate the conventions used by publications from different fields [20].

The objective of this chapter is to present a complete treatment of the theory of rotations and the quaternion representation of attitude, from first principles. The only theoretical background that the reader requires is in Linear Algebra and Multivariable Calculus. Much of the material presented in this chapter is consolidated from the excellent resources by Altmann [22], Shuster [23] and Solà [24], but with additional theoretical contextualization and more derivation of formulas.

In Section 2.1 we present the model of 3D space on which rotations will be described. This model is based on the properties of Linear Algebra and, as we shall see, the description of positions in 3D space is highly dependent on the basis one chooses. In Section 2.2 we show how to translate between different orthonormal bases, a key step in understanding rotations. It turns out rotations can be modeled as a change of basis between two bases that have the same orientation, a property described in Section 2.3.

By defining a rigid body in Section 2.4, we can connect the abstract operation of change of basis to the physical phenomenon of a rotation of a physical object. In Section 2.5, we begin the treatment of describing rotations proper, in both possible conventions: rotations as active transformations; or rotations as passive transformations. After describing how one can take a rigid body from one configuration in space to a rotated configuration, in Section 2.6 we describe how one can model the motion between these two configurations, i.e. how to model the rotation kinematics. In Section 2.7 we present the possible alternative representations to describe rotations, focusing on the Euler-Rodrigues symmetric parameters, also known as the quaternion representation, which we will use for the rest of this work.

Finally, in Section 2.8, we introduce the theory of rotation dynamics.

2.1 Euclidean space

To present the theory of rotations in 3D space, we first need a mathematical model to describe 3D space. As the quote from french mathematician Sophie Germain states, the geometry of 3D space is described by the properties of linear algebra. We model 3D space as the inner product space called the Euclidean 3-space [25], defined as follows:

Definition 2.1.1 (Euclidean 3-space)**.** Euclidean 3-space is the set of all triples of real numbers,

$$\mathbb{R}^3 = \{(p_1, p_2, p_3) : p_1, p_2, p_3 \in \mathbb{R}^3\},$$

endowed with the operation of vector addition, defined component-wise,

$$\mathbf{p} + \mathbf{q} = (p_1 + q_1, p_2 + q_2, p_3 + q_3),\ p, q \in \mathbb{R}^3,$$

the operation of scalar multiplication over the real numbers, also defined component-wise

$$\alpha\mathbf{p} = (\alpha \cdot p_1, \alpha \cdot p_2, \alpha \cdot p_3),\ p \in \mathbb{R}^3, \alpha \in \mathbb{R}$$

and with the Euclidean inner product (also known as the dot product)[a],

$$\mathbf{p} \cdot \mathbf{q} = p_1 \cdot q_1 + p_2 \cdot q_2 + p_3 \cdot q_3,\ p, q \in \mathbb{R}^3.$$

[a]the symbol $\cdot$ will be overloaded to refer to scalar multiplication, matrix multiplication and the dot product. For the cases of scalar multiplication and matrix multiplication, it will be included where necessary to improve readability and left implicit otherwise.

A position in 3D space is thus an element $\mathbf{p}$ of Euclidean 3-space (or 3-space, for short), which we call a point. We denote Euclidean 3-space by $\mathbb{R}^3$, when it is implicitly understood that the set has the properties defined above. The definition of Euclidean 3-space naturally extends to the set of n-tuples of real numbers, $\mathbb{R}^n$, which we call the Euclidean n-space.

We emphasize that a position in 3D space is not modeled as 3 numbers, rather, it is modeled by the relations between these three numbers. The relations are given by an inner product space with the dot product as its inner product. If this statement is immediately obvious to the reader, then they are free to skip the rest of this section. For the reader that does not understand the implications of the previous statement, we dedicate the rest of this section to clarifying what it means and illustrating how each property of Definition 2.1.1 relates to our intuition of 3D space.

Endowing the set $\mathbb{R}^3$ with vector addition and scalar multiplication means that it is a vector space, i.e. points are vectors. The purpose of defining positions in 3D space as points, is not that we describe position based on 3 numbers, since we could do this with polar coordinates (but this would not be a vector space). The purpose is that we can use the abstract properties of vector spaces in describing position.

The operations of vector and scalar multiplication on a set endow it with the geometrical notion of “direction”, expressed by the algebraic property of linear independence [26]. Two vectors $\mathbf{p}, \mathbf{q} \in \mathbb{R}^3$ are linearly dependent if there exists some scalars $\alpha, \beta \in \mathbb{R}$ not equal to zero such that

$$\alpha\mathbf{p} + \beta\mathbf{q} = 0.$$

in which case we say that “$\mathbf{p}$ and $\mathbf{q}$ have the same direction”. If the scalars α and β do not exist, then $\mathbf{p}$ and $\mathbf{q}$ are linearly independent and we say that they have “different directions”.

The property of linear independence leads to one of the most important results in linear algebra, which is that any vector in an n-dimensional vector space can be described as the linear combination of n linearly independent vectors [27].

For Euclidean 3-space, we can pick 3 linearly independent points, i.e. 3 different directions, to span the entire space. For example, if we take $\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3} \in \mathbb{R}^3$ to be linearly independent vectors, then any other vector $\mathbf{p} \in \mathbb{R}^3$ can be described in terms of these 3 vectors as

$$\mathbf{p} = x_1 \cdot \mathbf{e_1} + x_2 \cdot \mathbf{e_2} + x_3 \cdot \mathbf{e_3},\ \ x_1, x_2, x_3 \in \mathbb{R}\ .$$

The set of 3 linearly independent vectors $\mathcal{E} = \{\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3}\}$ are a basis of $\mathbb{R}^3$, and each of the 3 scalars x_1, x_2, x_3 that multiply each basis vector is called a coordinate of $\mathbf{p}$. These three coordinates can be compiled into an ordered triple of real numbers (x_1, x_2, x_3), which means that they are also a point in $\mathbb{R}^3$.

The notion of defining position to be a point and then describing it in terms of a coordinate vector, which is another point, might sound like mathematical pedantry, but it is an important distinction. A position is a point $\mathbf{p} \in \mathbb{R}^3$, which is an abstract vector, and so are its basis vectors, i.e. the components will never actually have numerical values, we are only interested in their properties as vectors. The coordinates of $\mathbf{p}$ with respect to (w.r.t.) basis $\mathcal{E}$ is a vector with numerical values, which will depend on geometrical quantities (angles and distances) measured w.r.t. to the basis. We will write the coordinate vector of a point $\mathbf{p}$ as ${}^{\mathcal{E}}\mathbf{p} = (x_1, x_2, x_3)$ where the left superscript denotes the basis in which the coordinates are written, and refer to it as the coordinate vector of $\mathbf{p}$ w.r.t. the basis $\mathcal{E}$. It is worth clarifying that given another basis $\mathcal{F}$, then ${}^{\mathcal{E}}\mathbf{p} = (x_1, x_2, x_3)$ and ${}^{\mathcal{F}}\mathbf{p} = (x'_1, x'_2, x'_3)$ refer to the same point $\mathbf{p} \in \mathbb{R}^3$ but with respect to different bases, hence they have different values.

As we mentioned before, the coordinates of a point will be obtained by measurements of the geometrical quantities of distance and angle. The angle is a purely geometrical concept and cannot be described algebraically [26]. However, we can define algebraically the notion of perpendicularity (more generally called orthogonality) of two vectors using an operation called the inner product, defined as:

Definition 2.1.2 (Inner Product)**.** An inner product on a vector space V over $\mathbb{R}$[a] is an operator $\langle \circ, \circ \rangle : V \times V \to \mathbb{R}$ that has the following properties:

- Symmetry: $\langle \mathbf{v}, \mathbf{u} \rangle = \langle \mathbf{u}, \mathbf{v} \rangle, \ \forall \mathbf{u}, \mathbf{v} \in V$
- Linearity in the first argument: $\langle \alpha\mathbf{u} + \beta\mathbf{v}, \mathbf{w} \rangle = \langle \alpha\mathbf{u}, \mathbf{w} \rangle + \langle \beta\mathbf{v}, \mathbf{w} \rangle, \ \forall \alpha, \beta \in \mathbb{R}, \mathbf{u}, \mathbf{v}, \mathbf{w}$
- Positive definiteness: $\langle \mathbf{v}, \mathbf{v} \rangle \geq 0, \forall \mathbf{v} \in V$ and $\langle \mathbf{v}, \mathbf{v} \rangle = 0 \Leftrightarrow \mathbf{v} = \mathbf{0}$

[a] I.e. a vector space where the scalars are real numbers.

It is straightforward to check that the dot product satisfies the properties in Definition 2.1.2, so it is an inner product on $\mathbb{R}^3$. A vector space endowed with an inner-product is called an inner-product space. Since the dot product is an inner product, we will also use the generic operator symbol $\langle \circ, \circ \rangle : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$ to refer to it, i.e. $\langle \mathbf{p}, \mathbf{q} \rangle = \mathbf{p} \cdot \mathbf{q}$, $\mathbf{p}, \mathbf{q} \in \mathbb{R}^3$.

Two vectors $\mathbf{p}, \mathbf{q} \in \mathbb{R}^3$ are said to be orthogonal if $\langle \mathbf{p}, \mathbf{q} \rangle = 0$. We will see further ahead how we can relate the algebraic notion of orthogonality with the geometric notion of perpendicularity.

Any inner product space V is also a normed space, with norm defined as

$$\|\mathbf{p}\| = \sqrt{\langle \mathbf{p}, \mathbf{p} \rangle}, \ \mathbf{p} \in V.$$

For the case of Euclidean 3-space, we get $\|\mathbf{p}\| = \sqrt{p_1^2 + p_2^2 + p_3^2}$. The norm of a vector is also called its "length". The length of a vector is its distance to the point $\mathbf{0} = (0, 0, 0)$, which means that the norm in turn induces a distance function on the normed space, i.e. any normed space is a metric space. In Euclidean 3-space, the distance $d : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}_0^+$ is given by

$$d(\mathbf{p}, \mathbf{q}) = \|\mathbf{p} - \mathbf{q}\| = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2}, \mathbf{p}, \mathbf{q} \in \mathbb{R}^3$$

which is called the Euclidean distance, it is the length of the straight line joining two points, as shown in Figure 2.1.

An alternative and equivalent definition of the Euclidean inner product can be obtained if we consider the Law of Cosines, which tells us that the side of length $\|\mathbf{p} - \mathbf{q}\|$ of the triangle shown in Figure 2.2, is given by

$$\|\mathbf{p} - \mathbf{q}\|^2 = \|\mathbf{p}\|^2 + \|\mathbf{q}\|^2 - 2\|\mathbf{p}\|\|\mathbf{q}\| \cos(\theta).$$

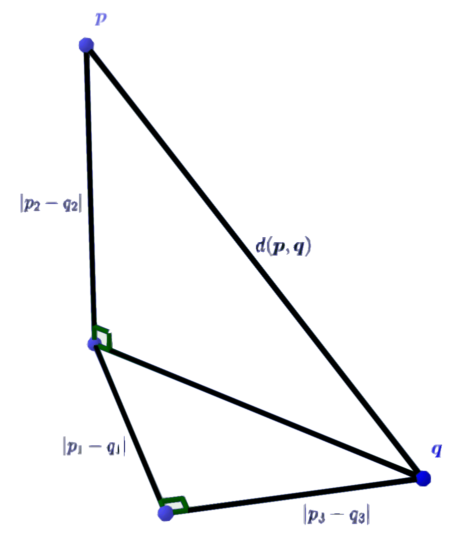


Figure 2.1: Euclidean distance between two points.

But from our algebraic definition we have

$$\|\mathbf{p}-\mathbf{q}\|^2 = (\mathbf{p}-\mathbf{q})\cdot(\mathbf{p}-\mathbf{q}) = \mathbf{p}\cdot\mathbf{p}+\mathbf{q}\cdot\mathbf{q}-2\mathbf{p}\cdot\mathbf{q},$$

from which we get

$$\langle \mathbf{p},\mathbf{q}\rangle = \mathbf{p}\cdot\mathbf{q} \Leftrightarrow \tag{2.1}$$

$$\langle \mathbf{p},\mathbf{q}\rangle = \|\mathbf{p}\|\|\mathbf{q}\|\cos(\theta). \tag{2.2}$$

It becomes clear from (2.1) and (2.2) that if $\mathbf{p}\neq\mathbf{0}$ and $\mathbf{q}\neq\mathbf{0}$, then we must have that $\langle \mathbf{p},\mathbf{q}\rangle = 0 \Leftrightarrow \theta = 90^\circ$. So two non-zero vectors of $\mathbb{R}^3$ being orthogonal is equivalent to them being perpendicular.

Since the definition of the Euclidean inner product given by (2.2) depends only on geometrical quantities of length and angle, and not on the components of the vector, it is called the geometric dot product. The dot product given by (2.1) is called the algebraic dot product. We will use $\langle\circ,\circ\rangle$ to refer to either of these equivalent definitions of the dot product.

In an inner product space, not all bases are created equal. A particularly useful class of bases are the orthonormal bases. An orthonormal basis $\mathcal{E} = \{\mathbf{e_1},\mathbf{e_2},\mathbf{e_3}\}$ of Euclidean 3-space is one such that

$$\langle \mathbf{e}_i,\mathbf{e}_j\rangle = \delta_{ij},$$

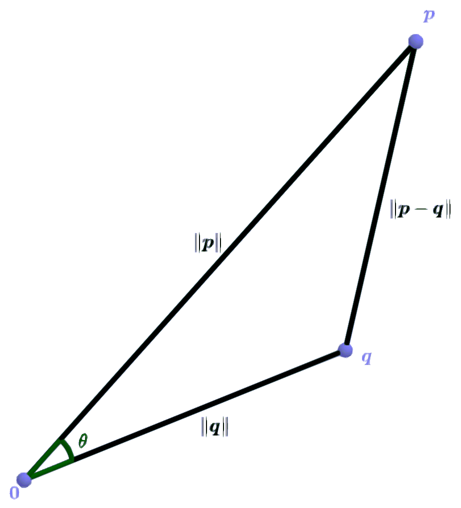


Figure 2.2: Law of cosines.

where δ_{ij} is the Dirac delta function, defined as

$$\delta_{ij} = \begin{cases} 1 \text{ if } i = j \\ 0 \text{ otherwise} \end{cases} .$$

This means that the basis vectors are unit length and are perpendicular to each other. An orthonormal basis is also called a frame in geometry [25].

The advantage of using an orthonormal basis is that the ith coordinate of a point $\mathbf{p} \in \mathbb{R}^3$ w.r.t. the basis $\mathcal{E}$ is obtained by the inner product

$$x_i = \langle \mathbf{p}, \mathbf{e}_i \rangle,$$

allowing us to describe the point by the sum

$$\mathbf{p} = \sum_{i=1}^{3} \langle \mathbf{p}, \mathbf{e}_i \rangle \mathbf{e}_i$$

called the orthonormal expansion of vector $\mathbf{p}$. This expression is incredibly useful, as it means we only need the inner product between $\mathbf{p}$ and each of the basis vectors to determine its coordinates.

2.2 Orthogonal Transformations

In the previous section, we defined Euclidean 3-space, which is our mathematical model of physical 3D space. Further, we saw how any position in 3-space can be represented by a coordinate vector w.r.t. some frame $\mathcal{E}$. The choice of a frame which allows us to describe 3D space with coordinates is subjective and any two descriptions of position are equally valid.

In this section, we describe how to transform the coordinates of a position from one frame to another, an action that in linear algebra is generally known as a change a basis. In particular, for a change of basis between two frames, the transformation that performs this action belongs to a group of linear transformations which are called orthogonal transformations, which are the topic of this section. We will see in the next section how rotations are related to the orthogonal transformations. The reader that is familiar with the concept of orthogonal transformations and the direction cosine-matrix is free to skip to the next section.

Let us start by considering in addition to frame $\mathcal{E} = \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$, another frame $\mathcal{F} = \{\mathbf{e}'_1, \mathbf{e}'_2, \mathbf{e}'_3\}$ that represents point $\mathbf{p}$ with the coordinates ${}^{\mathcal{F}}\mathbf{p} = \{x'_1, x'_2, x'_3\}$. If we know the coordinates ${}^{\mathcal{E}}\mathbf{p}$, how can we find the coordinates ${}^{\mathcal{F}}\mathbf{p}$? The two frames allow us to describe the vector in two different but equivalent ways

$$\begin{cases} \mathbf{p} = \sum_{i=1}^{3} x_i \mathbf{e}_i \\ \mathbf{p} = \sum_{i=1}^{3} x'_i \mathbf{e}'_i \end{cases} \Leftrightarrow \tag{2.3}$$

$$\sum_{j=1}^{3} x_j \mathbf{e}_j = \sum_{i=1}^{3} x'_i \mathbf{e}'_i. \tag{2.4}$$

If we wish to isolate the coordinate x'_i from (2.4), we can just apply $\langle \mathbf{e}'_i, \circ \rangle$ to both sides of the expression and obtain

$$x'_i = \sum_{j=1}^{3} x_j \langle \mathbf{e}'_i, \mathbf{e}_j \rangle = \sum_{j=1}^{3} C_{ij} x_j$$

with $C_{ij} = \langle \mathbf{e}'_i, \mathbf{e}_j \rangle = \cos\left(\angle(\mathbf{e}'_i, \mathbf{e}_j)\right)$, where $\angle(\mathbf{e}'_i, \mathbf{e}_j)$ is the angle between vectors $\mathbf{e}'_i$ and $\mathbf{e}_j$.

So the transformation $\mathbf{C} : \mathbb{R}^3 \to \mathbb{R}^3$ that maps the coordinate vector ${}^{\mathcal{E}}\mathbf{p}$ to the coordinate vector ${}^{\mathcal{F}}\mathbf{p}$ is given by

$${}^{\mathcal{F}}\mathbf{p} = \mathbf{C}({}^{\mathcal{E}}\mathbf{p}) = \left(\sum_{j=1}^{3} C_{1j} x_j, \sum_{j=1}^{3} C_{2j} x_j, \sum_{j=1}^{3} C_{3j} x_j \right)$$

which is a linear transformation. Every linear transformation can be represented by matrix operations, so representing elements of $\mathbb{R}^3$ as column matrices, we can write succinctly the linear transformation that relates ${}^{\mathcal{E}}\mathbf{p}$ with ${}^{\mathcal{F}}\mathbf{p}$ as a matrix operation

$$\begin{bmatrix} x_1' \\ x_2' \\ x_3' \end{bmatrix} = \begin{bmatrix} \langle \mathbf{e}_1', \mathbf{e}_1 \rangle & \langle \mathbf{e}_1', \mathbf{e}_2 \rangle & \langle \mathbf{e}_1', \mathbf{e}_3 \rangle \\ \langle \mathbf{e}_2', \mathbf{e}_1 \rangle & \langle \mathbf{e}_2', \mathbf{e}_2 \rangle & \langle \mathbf{e}_2', \mathbf{e}_3 \rangle \\ \langle \mathbf{e}_3', \mathbf{e}_1 \rangle & \langle \mathbf{e}_3', \mathbf{e}_2 \rangle & \langle \mathbf{e}_3', \mathbf{e}_3 \rangle \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \Leftrightarrow {}^{\mathcal{F}}\mathbf{p} = \mathbf{C}\, {}^{\mathcal{E}}\mathbf{p}.$$

Since $\mathbf{C}$ is a matrix consisting of the cosine of angles between different vectors, it is called the direction cosine matrix.

Likewise, the linear transformation that relates the vector ${}^{\mathcal{E}}\mathbf{p}$ with ${}^{\mathcal{F}}\mathbf{p}$ is given by the matrix operation

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \langle \mathbf{e}_1, \mathbf{e}_1' \rangle & \langle \mathbf{e}_1, \mathbf{e}_2' \rangle & \langle \mathbf{e}_1, \mathbf{e}_3' \rangle \\ \langle \mathbf{e}_2, \mathbf{e}_1' \rangle & \langle \mathbf{e}_2, \mathbf{e}_2' \rangle & \langle \mathbf{e}_2, \mathbf{e}_3' \rangle \\ \langle \mathbf{e}_3, \mathbf{e}_1' \rangle & \langle \mathbf{e}_3, \mathbf{e}_2' \rangle & \langle \mathbf{e}_3, \mathbf{e}_3' \rangle \end{bmatrix} \begin{bmatrix} x_1' \\ x_2' \\ x_3' \end{bmatrix}.$$

However, if we apply the symmetry of inner product to each of the elements of the matrix, we notice that this expression is

$${}^{\mathcal{E}}\mathbf{p} = \mathbf{C}^\top\, {}^{\mathcal{F}}\mathbf{p},$$

from where we get that

$${}^{\mathcal{E}}\mathbf{p} = \mathbf{C}^\top \mathbf{C}\, {}^{\mathcal{E}}\mathbf{p} \implies \mathbf{C}^\top \mathbf{C} = I \Leftrightarrow \mathbf{C}^{-1} = \mathbf{C}^\top,$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{C}^{-1}$ is the inverse of $\mathbf{C}$. Matrices with the property that its transpose is equal to its inverse are called orthogonal. They are called such because they preserve the inner product (and thus also orthogonality), which also means that distances are preserved, i.e. they are isometries. We denote by $O(3)$ the set of all 3×3 orthogonal matrices, which is called the orthogonal group

$$O(3) = \{\mathbf{A} \in \mathbb{R}^{3\times 3} : \mathbf{A}^\top \mathbf{A} = I\}.$$

From the properties of the determinant we obtain

$$\det(\mathbf{C}^\top \mathbf{C}) = \det(\mathbf{I}) \Leftrightarrow \det(\mathbf{C}^\top)\det(\mathbf{C}) = 1 \Leftrightarrow \det(\mathbf{C})^2 = 1 \implies \det(\mathbf{C}) = \pm 1.$$

Topologically, the fact that the determinant can only be ± 1 means that the orthogonal group is made out of two disjoint subsets. To be able to differentiate between transformations from these two subsets, we need to introduce the concept of orientation, which is the subject of the next section.

2.3 Orientation

In the previous section, we saw how the orthogonal group has two disjoint subsets, one of which is the subset of orthogonal matrices with determinant 1. Orthogonal matrices with determinant 1 are transformations that preserve an essential geometrical property of 3-space, called the orientation. These orientation preserving orthogonal transformations is what we use to model rotations.

A plane is uniquely defined by three points. In Euclidean 3-space, we have the property that a plane divides 3-space in two halfspaces. Take the plane defined by the origin and the points $\mathbf{e}_1$ and $\mathbf{e}_2$, as shown in Figure 2.3. The line that passes through the origin and makes a $90°$ angle with the plane is called the line normal to the plane. On the line normal to the plane, we have two choices of points that are unit norm and ortogonal to both $\mathbf{e}_1$ and $\mathbf{e}_2$. These are points $\mathbf{a}$ and $\mathbf{b}$ in Figure 2.3. Any one of these two points would complete our frame, whichever we choose defines the orientation of our frame.

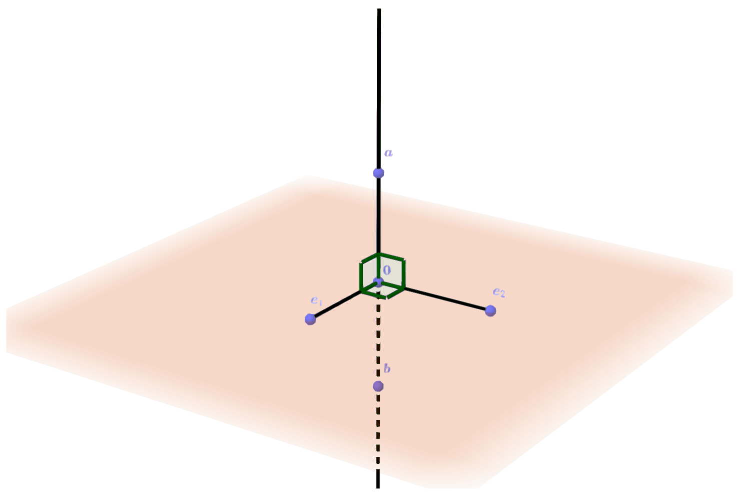


Figure 2.3: Two choices of orientation for the frame with vectors $\mathbf{e_1}$ and $\mathbf{e_2}$.

Algebraically, orientation in Euclidean 3-space is given by the cross-product defined as follows:

Definition 2.3.1 (Cross Product)**.** The cross product is the operator $\times : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^3$ that has the properties:

- Antisymmetry: $\mathbf{u} \times \mathbf{v} = -\mathbf{v} \times \mathbf{u},\ \forall \mathbf{u}, \mathbf{v} \in \mathbb{R}^3$
- Bilinearity:
  $(\alpha\mathbf{u} + \beta\mathbf{v}) \times \mathbf{w} = \alpha(\mathbf{u} \times \mathbf{w}) + \beta(\mathbf{v} \times \mathbf{w})$ and $\mathbf{u} \times (\alpha\mathbf{v} + \beta\mathbf{w}) = \alpha(\mathbf{u} \times \mathbf{v}) + \beta(\mathbf{u} \times \mathbf{w})$,
  $\forall \alpha, \beta \in \mathbb{R}, \mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{R}^3$
- Jacobi identity: $\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) + \mathbf{b} \times (\mathbf{c} \times \mathbf{a}) + \mathbf{c} \times (\mathbf{a} \times \mathbf{b}) = \mathbf{0},\ \forall \mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^3$

Further $\|\mathbf{u} \times \mathbf{v}\| = \|\mathbf{u}\|\|\mathbf{v}\| \sin(\theta), \forall \mathbf{u}, \mathbf{v} \in \mathbb{R}^3$, where θ is the angle between vectors $\mathbf{u}$ and $\mathbf{v}$, $\theta = \angle(\mathbf{u}, \mathbf{v})$.

We say that the orthonormal frame $\mathcal{E} = \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ is positively oriented if

$$\mathbf{e}_1 \times \mathbf{e}_2 = \mathbf{e}_3 \Leftrightarrow \mathbf{e}_2 \times \mathbf{e}_3 = \mathbf{e}_1 \Leftrightarrow \mathbf{e}_3 \times \mathbf{e}_1 = \mathbf{e}_2 \tag{2.5}$$

and it is negatively oriented if

$$\mathbf{e}_1 \times \mathbf{e}_2 = -\mathbf{e}_3 \Leftrightarrow \mathbf{e}_2 \times \mathbf{e}_3 = -\mathbf{e}_1 \Leftrightarrow \mathbf{e}_3 \times \mathbf{e}_1 = -\mathbf{e}_2 \tag{2.6}$$

Geometrically, orientation differentiates between the “handedness” of a frame in space. If one can position their right hand such that the index, middle finger and thumb is aligned with the directions of the frame, then the frame is said to be “right handed” or positively oriented, otherwise it is said to be “left handed” or negatively oriented. It should be intuitive that the only transformation that can take a right handed frame and convert it to a left handed frame is a reflection about a plane of symmetry.

By using the properties of the cross product for any two vectors written in the same positively oriented orthonormal basis, we have

$$\begin{aligned}\mathbf{a} \times \mathbf{b} &= (a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + a_3\mathbf{e}_3) \times (b_1\mathbf{e}_1 + b_2\mathbf{e}_2 + b_3\mathbf{e}_3) \\ &= (a_2b_3 - a_3b_2)\mathbf{e}_1 + (a_3b_1 - a_1b_3)\mathbf{e}_2 + (a_1b_2 - a_2b_1)\mathbf{e}_3,\end{aligned}$$

which can be written as the matrix operation

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{bmatrix}.$$

We define the operator $[\,\circ\,]_\times : \mathbb{R}^3 :\rightarrow \mathfrak{so}(3)$ as

$$[\mathbf{a}]_\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} = a_1\boldsymbol{\mathcal{I}} + a_2\boldsymbol{\mathcal{J}} + a_3\boldsymbol{\mathcal{K}}$$

where

$$\boldsymbol{\mathcal{I}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}, \quad \boldsymbol{\mathcal{J}} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}, \quad \boldsymbol{\mathcal{K}} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

and $\mathfrak{so}(3)$ is the set of skew-symmetric matrices

$$\mathfrak{so}(3) = \{\mathbf{A} \in \mathbb{R}^{3\times 3} : \mathbf{A} = -\mathbf{A}^\top\}.$$

From the first equalities of expressions (2.5) and (2.6) one obtains the expression

$$\mathbf{e}_3 \cdot (\mathbf{e}_1 \times \mathbf{e}_2) = \pm 1 \tag{2.7}$$

where the equality is $+1$ if the frame is right handed and -1 if it is left handed. The expression on the left of (2.7) is called the scalar triple product. Writing the basis vectors as components $\mathbf{e}_i = (e_{i1}, e_{i2}, e_{i3})$ and expanding the scalar triple product, we get

$$\mathbf{e}_3 \cdot (\mathbf{e}_1 \times \mathbf{e}_2) = e_{31}(e_{12} \cdot e_{23} - e_{13} \cdot e_{22}) - e_{32}(e_{11}e_{23} - e_{13}e_{21}) + e_{33}(e_{11}e_{22} - e_{12}e_{21}),$$

which is the expression for the Laplace expansion, along the third row, of the determinant of the matrix

$$\begin{bmatrix} e_{11} & e_{21} & e_{31} \\ e_{12} & e_{22} & e_{32} \\ e_{13} & e_{23} & e_{33} \end{bmatrix},$$

which we call the attitude matrix of the frame $\mathcal{E}$ [25]. So the frame being right-handed (respectively left-handed) is equivalent to the determinant of the attitude matrix of the frame being 1 (respectively -1).

Let $\mathbf{A}$ (respectively $\mathbf{A}'$) be the attitude matrix of frame $\mathcal{E}$ (respectively $\mathcal{F}$). Expressing the orthonormal expansion of $\mathbf{p}$ with matrix operations, we have that

$$\mathbf{p} = \mathbf{A}'\,{}^{\mathcal{F}}\mathbf{p} = \mathbf{A}\,{}^{\mathcal{E}}\mathbf{p} \Leftrightarrow {}^{\mathcal{F}}\mathbf{p} = \mathbf{A}'^{\top}\mathbf{A}\,{}^{\mathcal{E}}\mathbf{p} \implies \mathbf{C} = \mathbf{A}'^{\top}\mathbf{A} \implies \det(\mathbf{C}) = \det(\mathbf{A}')\det(\mathbf{A}),$$

from which we get $\det(\mathbf{C}) = 1 \Leftrightarrow (\det(\mathbf{A}) = 1 \wedge \det(\mathbf{A}') = 1) \vee (\det(\mathbf{A}) = -1 \wedge \det(\mathbf{A}') = -1)$, i.e. $\det(\mathbf{C}) = 1$ if and only if both basis have the same orientation. Likewise, $\det(\mathbf{C}) = -1$ if and only if the bases have opposite orientations. The subset of $O(3)$ with determinant -1 are the transformations that represent reflections. Orthogonal matrices that preserve orientation (i.e with determinant 1) are called special and the set of all special orthogonal matrices is called the special orthogonal group

$$SO(3) = \{\mathbf{A} \in \mathbb{R}^{3\times 3} : \mathbf{A}^{\top}\mathbf{A} = \mathbf{I},\ \det(\mathbf{A}) = 1\}.$$

These special orthogonal matrices represent rotations, so we will refer to elements of $SO(3)$ as rotation matrices. In the next sections, we will see how they can be used to model the rotation of rigid bodies.

2.4 Rigid Bodies

Before proceeding to explain how transformations in $SO(3)$ can be used to represent the rotation of physical objects, such as spacecraft, we need a mathematical model for physical objects. We assume that physical objects are described by rigid bodies.

To define a rigid body, we start by considering the model for a particle, which is an object that at any given time occupies a single point in space. We define a particle to be a function $\mathbf{p} : \mathbb{R}_0^+ \to \mathbb{R}^3, \mathbf{p}(t) = \big(p_1(t), p_2(t), p_3(t)\big)$. For a given time $t \in \mathbb{R}_0^+$, $\mathbf{p}(t) \in \mathbb{R}^3$ is the position of the particle at time t and the image of the function $\mathbf{p}$ is called the path of the particle. A rigid body is a set of particles, which are related by the property that their distances are fixed in relation to each other at any given time, i.e. given any two particles $\mathbf{p}$ and $\mathbf{q}$ of a rigid body, we have that $\|\mathbf{p}(0) - \mathbf{q}(0)\| = \|\mathbf{p}(t) - \mathbf{q}(t)\|, \forall t \in \mathbb{R}_0^+$.

We define a rotating frame to be a set of particles $\{\mathbf{e}_1(t), \mathbf{e}_2(t), \mathbf{e}_3(t)\}$ that are always orthonormal and have the same orientation, i.e.

$$\mathbf{e}_i(t) \cdot \mathbf{e}_j(t) = \delta_{ij},\ \forall t \in \mathbb{R}_0^+ \quad \text{and} \quad \mathbf{e}_3(t_1) \cdot (\mathbf{e}_1(t_1) \times \mathbf{e}_2(t_1)) = \mathbf{e}_3(t_2) \cdot (\mathbf{e}_1(t_2) \times \mathbf{e}_2(t_2)),\ \forall t_1, t_2 \in \mathbb{R}_0^+.$$

Given a rigid body $\mathcal{O}$, we define a body-fixed frame to be a rotating frame $\mathcal{B}$, such that, for a given point $\mathbf{c} \in \mathcal{O}$, which we call the origin of the body-fixed frame, we have that $(\mathcal{B} + \mathbf{c}) \cup \mathcal{O}$ is a rigid body, where $\mathcal{B} + \mathbf{c}$ is the translation of set $\mathcal{B}$ by vector $\mathbf{c}$, i.e. for $\mathcal{B} = \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$, $\mathcal{B} + \mathbf{c} = \{\mathbf{e}_1 + \mathbf{c}, \mathbf{e}_2 + \mathbf{c}, \mathbf{e}_3 + \mathbf{c}\}$. This means that the translated frame moves sympathetically with the rigid body $\mathcal{O}$.

Since for any $\mathbf{e}_i \in \mathcal{B}$ and $\mathbf{p} \in \mathcal{O}$, we have that $\|\mathbf{e}_i + \mathbf{c} - \mathbf{p}\|$ and $\|\mathbf{p} - \mathbf{c}\|$ are constants, thus

$$\|\mathbf{e}_i + \mathbf{c} - \mathbf{p}\|^2 = \|\mathbf{e}_i + \mathbf{c}\|^2 + \|\mathbf{p}\|^2 - 2\mathbf{e}_i \cdot \mathbf{p} - 2\mathbf{c} \cdot \mathbf{p} \tag{2.8}$$

$$= \|\mathbf{e}_i\|^2 + 2\mathbf{e}_i \cdot \mathbf{c} - 2\mathbf{e}_i \cdot \mathbf{p} + \|\mathbf{p} - \mathbf{c}\|^2 = 1 + \|\mathbf{p} - \mathbf{c}\|^2 - 2\mathbf{e}_i(\mathbf{p} - \mathbf{c}) \tag{2.9}$$

$$\Leftrightarrow \mathbf{e}_i \cdot (\mathbf{p} - \mathbf{c}) = (1 + \|\mathbf{p} - \mathbf{c}\|^2 - \|\mathbf{e}_i + \mathbf{c} - \mathbf{p}\|^2)/2 = cte \tag{2.10}$$

$$\implies x_i = \langle \mathbf{e}_i, \mathbf{p} - \mathbf{c} \rangle = \langle \mathbf{e}_i, \mathbf{p}' \rangle = cte,\ \forall t \in \mathbb{R}_0^+ \tag{2.11}$$

where $\mathbf{p}' = \mathbf{p} - \mathbf{c}$. Taking $\mathbf{B}$ to be the attitude matrix of $\mathcal{B}$, we can write

$$\mathbf{p}' = \mathbf{B}\,{}^{\mathcal{B}}\mathbf{p}' \Leftrightarrow \tag{2.12}$$

$$\mathbf{p} = \mathbf{B}\,{}^{\mathcal{B}}\mathbf{p}' + \mathbf{c}, \tag{2.13}$$

and taking the derivative, given that the coordinate vector is constant, we get

$$\dot{\mathbf{p}} = \dot{\mathbf{B}}\,{}^{\mathcal{B}}\mathbf{p}' + \dot{\mathbf{c}}. \tag{2.14}$$

Equation (2.14) shows that the motion of any particle of the rigid body is composed of two terms, one term that is only related to the rotational motion of the body fixed frame (given by $\dot{\mathbf{B}}$), and one term that is only related to the translational motion of the origin of the body fixed frame (given by $\dot{\mathbf{c}}$). The fact that we can decompose motion of an object into decoupled motions of translation and rotation is known as Chasle's theorem [28]. Given this fact, and since we are only interested in controlling the attitude of a spacecraft, we will from now on consider that the origin of the body-fixed frame is fixed at $\mathbf{0}$.

2.5 Rotation of Rigid Bodies

In this section, we begin the treatment of modeling rotations of rigid bodies. Figure 2.4 shows a rigid body in the form a parallelepiped and a body-fixed frame with origin in the centroid of the parallelepiped. Without loss of generality, we take the basis vectors of the body-fixed frame to be aligned with the principal axis of inertia. There is another frame that is aligned with the body-fixed frame at $t = t_1$, $\mathcal{N} = \{\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3\} = \{\mathbf{e}_1(t_1), \mathbf{e}_2(t_1), \mathbf{e}_3(t_1)\}$, which remains fixed in place for all time, this type of constant frame is called an inertial (or Newtonian) reference frame.

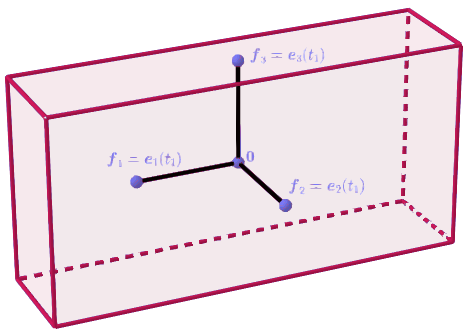


Figure 2.4: Rigid body aligned with the inertial reference frame.

If the rigid body rotates, depending on one's perspective, two equivalent actions are observed. Figure 2.5 shows what happens if the observer has the point of view of the inertial reference frame. In this case, the coordinates of the particles w.r.t. the inertial reference frame have changed, and we write the rotation as a transformation of the coordinates w.r.t. $\mathcal{N}$. This is the description of rotation as an alibi or active transformation. Figure 2.6 shows the case where the rigid body rotates but an observer is viewing from the perspective of the body-fixed frame. From their perspective, it is not the body that is rotating, but the inertial reference frame. This describes rotation as a transformation of the coordinates of the inertial reference frame w.r.t. the body-fixed frame. This type of transformation is called an alias or passive transformation.

In this section we will go over the theory of each of these types of transformations one by one.

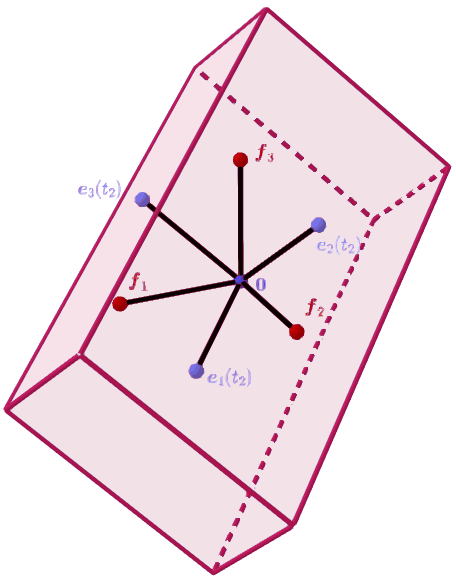


Figure 2.5: Rotation as an active transformation.

2.5.1 Rotation as an active transformation

Take a particle $\mathbf{p}$ of the rigid body. The particle can be represented by the coordinate vector w.r.t. the body-fixed frame $\mathcal{B}$, ${}^{\mathcal{B}}\mathbf{p}$, which is constant, since $\langle \mathbf{p}(t), \mathbf{e}(t) \rangle = cte, \forall \mathbf{e} \in \mathcal{B}$ and $t \in \mathbb{R}_0^+$. However, the particle $\mathbf{p}$ is not constant, since the attitude matrix of the body-fixed frame $\mathbf{B}(t)$ is time-changing. Likewise, the particle can also be represented by the coordinate vector w.r.t. the inertial reference frame $\mathcal{N}$, ${}^{\mathcal{N}}\mathbf{p}(t)$, which is time changing since the attitude matrix of the inertial reference frame $\mathbf{N}$ is constant.

We can thus write

$$\mathbf{p}(t) = \sum {}^{\mathcal{B}}p_i \mathbf{e}_i(t) = \sum {}^{\mathcal{N}}p_i(t)\mathbf{f}_i \Leftrightarrow \mathbf{p}(t) = \mathbf{B}(t)\,{}^{\mathcal{B}}\mathbf{p} = \mathbf{N}\,{}^{\mathcal{N}}\mathbf{p}(t).$$

For our attitude control problem, we will assume that the body-fixed frame at the initial time t_1 is aligned with the inertial reference frame[1]. As such, we have that ${}^{\mathcal{N}}\mathbf{p}(t_1) = {}^{\mathcal{B}}\mathbf{p}$ from where we get

$$\mathbf{B}(t_2)\,{}^{\mathcal{N}}\mathbf{p}(t_1) = \mathbf{N}\,{}^{\mathcal{N}}\mathbf{p}(t_2) \Leftrightarrow \mathbf{N}^\top \mathbf{B}(t_2)\,{}^{\mathcal{N}}\mathbf{p}(t_1) = {}^{\mathcal{N}}\mathbf{p}(t_2).$$

The matrix $\mathbf{N}^\top \mathbf{B}(t_2)$ is the matrix that transforms the coordinate vector of the particles of the rigid body, w.r.t. the inertial reference frame, from its initial unrotated position at t_1 to its rotated position at t_2, i.e. it is the matrix that represents rotation as an active transformation. It is a direction cosine matrix

[1] If it is not, then we need to find the direction cosine matrix that takes coordinates from the inertial reference frame to the body fixed frame at t_1 which is an attitude estimation problem and will not be handled in this work.

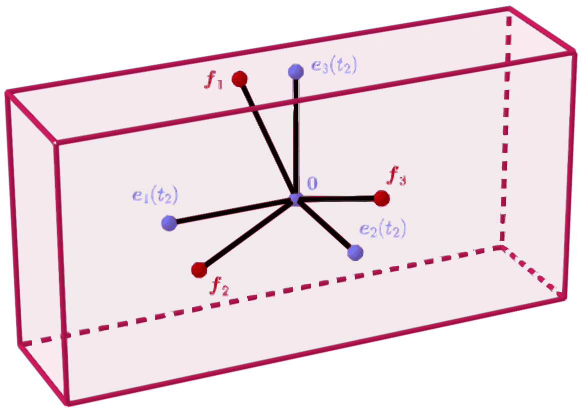


Figure 2.6: Rotation as a passive transformation.

where the element in the ith row and jth column is $(\mathbf{N}^\top \mathbf{B})_{ij}(t_2) = \langle \mathbf{f}_i, \mathbf{e}_j(t_2) \rangle$, thus, its columns are the coordinates of vector $\mathbf{e}_j(t_2)$ w.r.t. the basis $\mathcal{N}$, for this reason we write it as ${}^{\mathcal{N}}\mathbf{B}(t_2)$, which is the attitude matrix of $\mathcal{B}$ w.r.t. $\mathcal{N}$

$${}^{\mathcal{N}}\mathbf{B}(t_2) = \mathbf{N}^\top \mathbf{B}(t_2) = \begin{bmatrix} | & | & | \\ {}^{\mathcal{N}}\mathbf{e}_1(t_2) & {}^{\mathcal{N}}\mathbf{e}_2(t_2) & {}^{\mathcal{N}}\mathbf{e}_3(t_2) \\ | & | & | \end{bmatrix}.$$

Thus, in the active transformation, the rotation of the rigid body from its configuration at t_1 to its configuration at t_2 is represented by the matrix $\mathbf{R} = {}^{\mathcal{N}}\mathbf{B}(t_2) = \mathbf{N}^\top \mathbf{B}(t_2)$, which is a change of basis matrix from $\mathcal{B}$ to $\mathcal{N}$, since ${}^{\mathcal{B}}\mathbf{p} = {}^{\mathcal{N}}\mathbf{p}(t_1)$.

2.5.2 Rotation as a Passive Transformation

In the description of rotation as a passive transformation, the rotation transforms the coordinates of the inertial reference frame basis vectors, w.r.t. to the body-fixed basis, at time t_1 to the coordinate vector at time t_2. Writing

$${}^{\mathcal{B}}\mathbf{N}(t_2) = \mathbf{R}\,{}^{\mathcal{B}}\mathbf{N}(t_1)$$

where

$${}^{\mathcal{B}}\mathbf{N}(t) = \begin{bmatrix} | & | & | \\ {}^{\mathcal{B}}\mathbf{f}_1(t) & {}^{\mathcal{B}}\mathbf{f}_2(t) & {}^{\mathcal{B}}\mathbf{f}_3(t) \\ | & | & | \end{bmatrix} = \mathbf{B}^\top \mathbf{N}(t)$$

is the attitude matrix of $\mathcal{N}$ w.r.t. $\mathcal{B}$. Since at t_1, $\mathbf{f}_i = \mathbf{e}_i(t_1)$, we have immediately that ${}^{\mathcal{B}}\mathbf{N}(t_1) = \mathbf{I} \Leftrightarrow \mathbf{R} = {}^{\mathcal{B}}\mathbf{N}(t_2)$., i.e. the rotation matrix in the passive transformation case is given by the attitude matrix of $\mathcal{N}$ w.r.t. $\mathcal{B}$ (at time t_2). This gives us

$${}^{\mathcal{B}}\mathbf{N}(t) = \left(\mathbf{N}^{\top}\mathbf{B}(t)\right)^{\top} = \left({}^{\mathcal{N}}\mathbf{B}(t)\right)^{\top},$$

and from the previous section we know that ${}^{\mathcal{N}}\mathbf{B}(t)$ is the change of basis matrix from $\mathcal{B}$ to $\mathcal{N}$. Rotations are special orthogonal matrices (so the transpose is the inverse transformation), which means matrix ${}^{\mathcal{B}}\mathbf{N}(t)$ is a change of basis from $\mathcal{N}$ to $\mathcal{B}$ that describes the rotation as a passive transformation.

2.5.3 Our Rotation Convention

The choice of describing rotations as a passive or active transformation can lead to subtle differences in subsequent equations of kinematics and on parametrizations of the rotation matrix, so clarifying which convention one is following, either as an active or passive transformation, is essential. This is unfortunately not always the case in publications about attitude. To add additional confusion, some authors separate passive into passive world-to-body or global-to-local (PWTB), and passive body-to-world or local-to-global (PBTW) [24] [20], where one is just the inverse transformation of the other. The passive transformation that we have derived is what the authors of [24] [20] call PWTB, and just as we have seen, and the authors of [20] themselves state, the inverse of the PWTB is just the active transformation, so even though it has the word "passive" in it, PBTW is actually a rotation as an active transformation!

The choice of which kind of transformation represents a rotation is arbitrary. Physics and dynamics studies use the active transformation, aerospace use passive transformation [23]. Interdisciplinary areas such as robotics have publications that use both conventions [20], although active transformations are preferred.

In this work, we will describe rotations as active transformations, which means we describe the rotation of the spacecraft as the attitude matrix of the body-fixed frame w.r.t. the inertial reference frame, which we will denote by $\mathbf{Q}$ and call the spacecraft's attitude.

2.6 Rotation Kinematics

In the previous section, we described rotation only by considering the initial ($t = t_1$) and final ($t = t_2$) position of the rigid body. In this section, we want to consider rotation as a "motion", which means considering the position of the rigid body at every time instance $t \in [t_1, t_2]$. More specifically, we want to study how the motion "evolves" as the body rotates, i.e. the kinematics of rotation of the rigid body. We assume that the motion is "smooth" meaning that basis vectors of the body-fixed frame are infinitely differentiable functions, or C^∞.

First lets assume a single particle moving under rotation. Since rotations are isometries, its norm is constant. So we have that for all $t \in \mathbb{R}_0^+$, the following relation holds

$$\|\mathbf{p}(t)\| = c \implies \|\mathbf{p}(t)\|^2 = c^2 \implies \mathbf{p}^\top(t)\mathbf{p}(t) = c^2$$

This relation means that the particle is constrained to move on a sphere of radius c. We denote the set of points that satisfy this relation by M

$$M = \{\mathbf{p} \in \mathbb{R}^3 : \mathbf{p}^\top \mathbf{p} = c^2\} = c \cdot \mathcal{S}^2$$

where $\mathcal{S}^2 = \{\mathbf{p} \in \mathbb{R}^3 : \mathbf{p}^\top \mathbf{p} = 1\}$ is the unit sphere and $c \cdot \mathcal{S}^2$ is just multiplying every vector in the set $\mathcal{S}^2$ by c.

If we take the derivative of this condition, we obtain

$$\dot{\mathbf{p}}^\top \mathbf{p} + \mathbf{p}^\top \dot{\mathbf{p}} = 0 \implies \dot{\mathbf{p}}^\top \mathbf{p} = 0.$$

This condition means that any velocity vector of $\mathbf{p}$ lives on a plane that is perpendicular (i.e. orthogonal) to $\mathbf{p}$. If one considers this plane to have its origin at $\mathbf{p}$, then the plane is tangent to the sphere on which $\mathbf{p}$ is constrained to move, as shown in Figure 2.7. For this reason we call the velocity vector a tangent vector and draw it as an arrow starting at $\mathbf{p}$, which we call its point of application [25]. We also call the plane on which it lives the tangent space to M at $\mathbf{p}$ and denote it by $T_\mathbf{p}M$.

For any $\mathbf{p} \in M$, the tangent space is a subspace of $\mathbb{R}^3$ that locally approximates M as $\mathbb{R}^2$. Therefore, we say that M is "locally Euclidean" of dimension 2. More generally, any subset M of Euclidean space $\mathbb{R}^n$, which at every point $\mathbf{p} \in M$ can be locally approximated as $\mathbb{R}^m$ (with $m \leq n$), is called an (embedded) m-dimensional manifold [29]. Specifically for the case of $m = 2$, we call it a surface.

As we saw in Section 2.4, to study the rotational kinematics of the rigid body one only needs to look at the basis vectors of the body-fixed frame, since every other particle of the rigid body can be defined as an orthonormal expansion of the basis vectors. If we take the time derivative of the relations of the positively oriented, body-fixed frame, we get

$$\begin{cases} \mathbf{e}_1 = \mathbf{e}_2 \times \mathbf{e}_3 \\ \mathbf{e}_2 = \mathbf{e}_3 \times \mathbf{e}_1 \\ \mathbf{e}_3 = \mathbf{e}_1 \times \mathbf{e}_2 \end{cases} \underset{\substack{\text{product rule} \\ +\text{antisymmetry}}}{\implies} \begin{cases} \dot{\mathbf{e}}_1 = \dot{\mathbf{e}}_2 \times \mathbf{e}_3 - \dot{\mathbf{e}}_3 \times \mathbf{e}_2 \\ \dot{\mathbf{e}}_2 = \dot{\mathbf{e}}_3 \times \mathbf{e}_1 - \dot{\mathbf{e}}_1 \times \mathbf{e}_3 \\ \dot{\mathbf{e}}_3 = \dot{\mathbf{e}}_1 \times \mathbf{e}_2 - \dot{\mathbf{e}}_2 \times \mathbf{e}_1 \end{cases} \Leftrightarrow \begin{cases} \dot{\mathbf{e}}_1 = (-\dot{\mathbf{e}}_1 \times \mathbf{e}_1 - \dot{\mathbf{e}}_2 \times \mathbf{e}_2 - \dot{\mathbf{e}}_3 \times \mathbf{e}_3) \times \mathbf{e}_1 \\ \dot{\mathbf{e}}_2 = (-\dot{\mathbf{e}}_1 \times \mathbf{e}_1 - \dot{\mathbf{e}}_2 \times \mathbf{e}_2 - \dot{\mathbf{e}}_3 \times \mathbf{e}_3) \times \mathbf{e}_2 \\ \dot{\mathbf{e}}_3 = (-\dot{\mathbf{e}}_1 \times \mathbf{e}_1 - \dot{\mathbf{e}}_2 \times \mathbf{e}_2 - \dot{\mathbf{e}}_3 \times \mathbf{e}_3) \times \mathbf{e}_3 \end{cases}$$

$$\implies \begin{cases} \dot{\mathbf{e}}_1 = \boldsymbol{\omega} \times \mathbf{e}_1 \\ \dot{\mathbf{e}}_2 = \boldsymbol{\omega} \times \mathbf{e}_2 \\ \dot{\mathbf{e}}_3 = \boldsymbol{\omega} \times \mathbf{e}_3 \end{cases} \implies \dot{\mathbf{B}}(t) = \left[\boldsymbol{\omega}(t)\right]_\times \mathbf{B}(t),$$

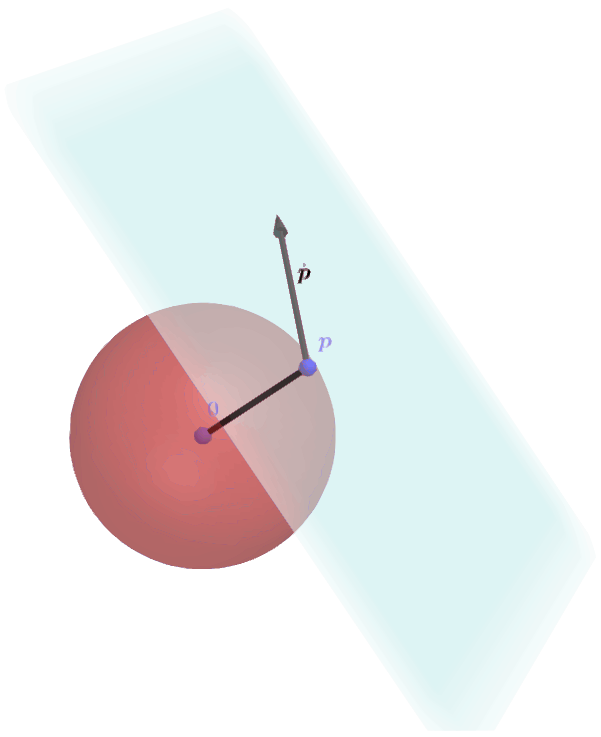


Figure 2.7: The tangent space to point $\mathbf{p}$ on the sphere of radius c.

where $\boldsymbol{\omega} = \mathbf{e}_1 \times \dot{\mathbf{e}_1} + \mathbf{e}_2 \times \dot{\mathbf{e}_2} + \mathbf{e}_3 \times \dot{\mathbf{e}_3}$. So knowing this single vector $\boldsymbol{\omega}$ determines the velocity of every basis vector and, by consequence, the velocity of every particle of the rigid body. For this reason, we call $\boldsymbol{\omega}$ the angular velocity of the rigid body. More generally, the velocity of any vector $\mathbf{p}$ whose coordinates are fixed w.r.t. the body-fixed frame is given by $\dot{\mathbf{p}} = \boldsymbol{\omega} \times \mathbf{p}$.

Note that since $\dot{\mathbf{p}} = \boldsymbol{\omega} \times \mathbf{p}$, the movement of the particle is perpendicular to $\boldsymbol{\omega}$. Moreover, any particle that is parallel to $\boldsymbol{\omega}$ does not move at all. The direction of $\boldsymbol{\omega}$ defines the axis of rotation and all motion happens on planes perpendicular to $\boldsymbol{\omega}$, as shown in Figure 2.8.

Writing $\boldsymbol{\omega}$ in terms of its body-fixed frame coordinates, we get

$$\begin{cases} \dot{\mathbf{e}}_1 = {}^{\mathcal{B}}\omega_3\mathbf{e}_2 - {}^{\mathcal{B}}\omega_2\mathbf{e}_3 \\ \dot{\mathbf{e}}_2 = {}^{\mathcal{B}}\omega_1\mathbf{e}_3 - {}^{\mathcal{B}}\omega_3\mathbf{e}_1 \\ \dot{\mathbf{e}}_3 = {}^{\mathcal{B}}\omega_2\mathbf{e}_1 - {}^{\mathcal{B}}\omega_1\mathbf{e}_2 \end{cases} \Leftrightarrow$$

$$\dot{\mathbf{B}}(t) = \begin{bmatrix} | & | & | \\ \dot{\mathbf{e}}_1(t) & \dot{\mathbf{e}}_2(t) & \dot{\mathbf{e}}_3(t) \\ | & | & | \end{bmatrix} = \begin{bmatrix} | & | & | \\ \mathbf{e}_1(t) & \mathbf{e}_2(t) & \mathbf{e}_3(t) \\ | & | & | \end{bmatrix} \begin{bmatrix} 0 & -{}^{\mathcal{B}}\omega_3 & {}^{\mathcal{B}}\omega_2 \\ {}^{\mathcal{B}}\omega_3 & 0 & -{}^{\mathcal{B}}\omega_1 \\ -{}^{\mathcal{B}}\omega_2 & {}^{\mathcal{B}}\omega_1 & 0 \end{bmatrix} = \mathbf{B}(t)\left[{}^{\mathcal{B}}\boldsymbol{\omega}(t)\right]_\times .$$

The attitude of the spacecraft $\mathbf{Q}$ at time t is given as a function of $\mathbf{B}(t)$, $\mathbf{Q}(t) = \mathbf{N}^\top\mathbf{B}(t)$. Since the basis vectors $\mathbf{e}_i, i = 1, 2, 3$ are differentiable functions, then $\mathbf{B}$ is also a differentiable function, and by consequence $\mathbf{Q} : \mathbb{R}_0^+ \to SO(3)$ is also a differentiable function. Taking the time derivative of $\mathbf{Q}$, we find

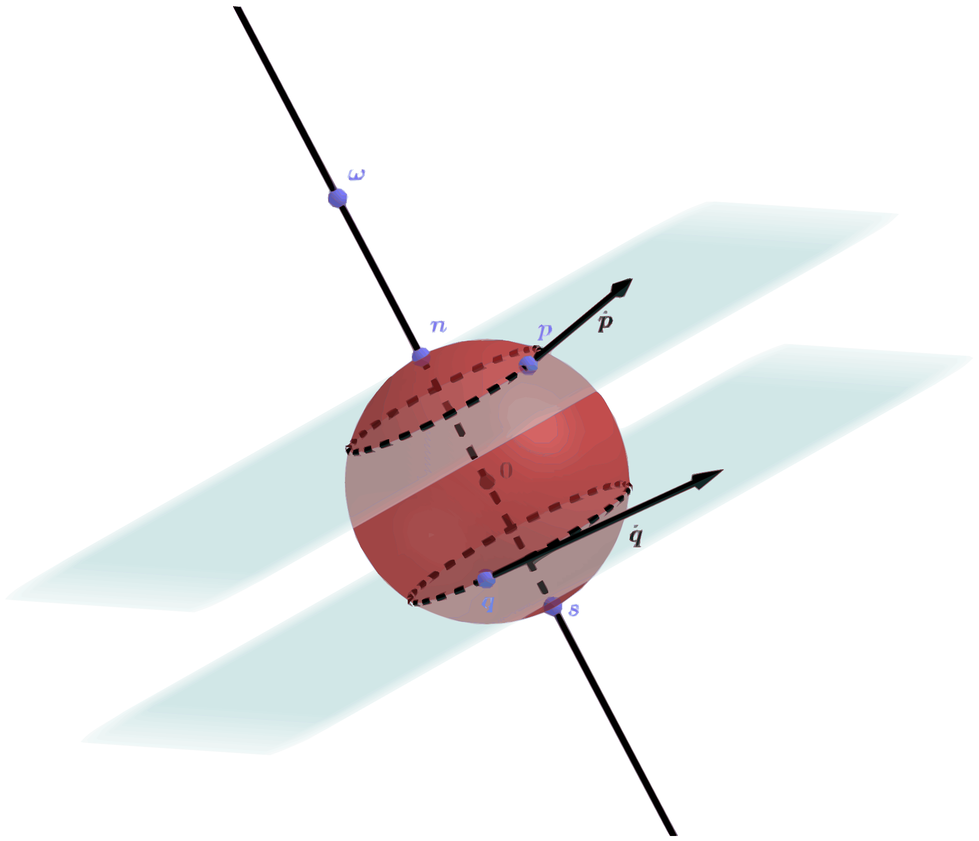


Figure 2.8: The axis of rotation defined by angular velocity vector $\boldsymbol{\omega}$.

that the spacecraft's attitude velocity is given by

$$\dot{\mathbf{Q}} = \mathbf{N}^\top \dot{\mathbf{B}} = \mathbf{N}^\top \mathbf{B}\left[{}^{\mathcal{B}}\boldsymbol{\omega}\right]_\times \Leftrightarrow \dot{\mathbf{Q}} = \mathbf{Q}[{}^{\mathcal{B}}\boldsymbol{\omega}]_\times.$$

There are very important consequences from this expression. First of all, $SO(3)$ is a subset of the 3×3 square matrices, $\mathbb{R}^{3\times 3}$, which is an inner product space with the inner product $\langle \mathbf{A}, \mathbf{B}\rangle = \text{tr}(\mathbf{A}^\top \mathbf{B})$. For any $\mathbf{Q} \in SO(3)$, we have that

$$\langle \mathbf{Q}, \dot{\mathbf{Q}}\rangle = \text{tr}(\mathbf{Q}^\top \dot{\mathbf{Q}}) = \text{tr}\big(\mathbf{Q}^\top \mathbf{Q}[{}^{\mathcal{B}}\boldsymbol{\omega}]_\times\big) = \text{tr}\big([{}^{\mathcal{B}}\boldsymbol{\omega}]_\times\big) = 0,$$

which means that the attitude's velocity is always orthogonal to $\mathbf{Q}$, or one might say that the set of all possible attitude velocities is tangent to $SO(3)$ at $\mathbf{Q}$. Secondly, much like the sphere in Figure 2.7, $SO(3)$ is not a vector space. However, the set of velocities of the spacecraft's attitude at $\mathbf{Q}$ is a vector space. For example, at $\mathbf{Q} = \mathbf{I}$, we have that $\dot{\mathbf{Q}} \in \mathfrak{so}(3)$ and $\mathfrak{so}(3)$ is a vector space spanned by the orthonormal basis

$$\mathfrak{so}(3) = \text{span}\left\{\frac{1}{2}\mathcal{I}, \frac{1}{2}\mathcal{J}, \frac{1}{2}\mathcal{K}\right\}$$

which means that $\mathfrak{so}(3)$ is a vector space of dimension 3. Therefore, $\mathfrak{so}(3)$ is isomorphic[2] to $\mathbb{R}^3$ (written $\mathfrak{so}(3) \cong \mathbb{R}^3$), which means there is a bijective map

$$\begin{gathered}\varphi : \mathfrak{so}(3) \to \mathbb{R}^3 \\ \frac{a_1}{2}\boldsymbol{\mathcal{I}} + \frac{a_2}{2}\boldsymbol{\mathcal{J}} + \frac{a_3}{2}\boldsymbol{\mathcal{K}} \mapsto (a_1, a_2, a_3)\end{gathered}$$

which preserves the algebraic structure of the two vector spaces

$$\varphi(\mathbf{A} + \mathbf{B}) = \varphi(\mathbf{A}) + \varphi(\mathbf{B}),\ \forall \mathbf{A}, \mathbf{B} \in \mathfrak{so}(3).$$

At $\mathbf{Q} = \mathbf{I}$, we have that $\dot{\mathbf{Q}}$ is an element of the tangent space $T_{\mathbf{I}}SO(3) = \mathfrak{so}(3) \cong \mathbb{R}^3$, so one can say that at the identity, $SO(3)$ is "locally like" $\mathbb{R}^3$.

At every other point $\mathbf{Q} \in SO(3)$, the set of all possible velocities at that point is just the image of $\mathfrak{so}(3)$ by the bijective linear transformation $\mathbf{Q}$, so the set of velocities at $\mathbf{Q}$ is also a vector space of dimension 3, tangent to $SO(3)$ at $\mathbf{Q}$, i.e. the tangent space $T_{\mathbf{Q}}SO(3) = \{\mathbf{Q}\boldsymbol{\Omega} : \boldsymbol{\Omega} \in \mathfrak{so}(3)\}$. So $SO(3)$ is "locally like" Euclidean 3-space at every point, i.e. it is a manifold.

The fact that $SO(3)$ is both a group[3] and a manifold means that it has both algebraic and geometric structure. Mathematical objects of this kind were first studied by Norwegian mathematician Sophus Lie and hence get the name Lie groups.

2.7 Alternative Representations

As we have seen in the previous sections, rotations are linear transformations and as such the correct mathematical formalism to approach rotations is as matrices. However, doing such has computational drawbacks, notably its kinematic equations

$$\begin{aligned}\dot{\mathbf{Q}} = \mathbf{Q}[\boldsymbol{\omega}]_\times \Leftrightarrow \begin{bmatrix} | & | & | \\ \dot{\mathbf{q}}_1 & \dot{\mathbf{q}}_2 & \dot{\mathbf{q}}_3 \\ | & | & | \end{bmatrix} &= \begin{bmatrix} | & | & | \\ \mathbf{q}_1 & \mathbf{q}_2 & \mathbf{q}_3 \\ | & | & | \end{bmatrix} \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \\ \implies \begin{cases} \dot{\mathbf{q}}_1 &= \omega_3 \mathbf{q}_2 - \omega_2 \mathbf{q}_3 \\ \dot{\mathbf{q}}_2 &= \omega_1 \mathbf{q}_3 - \omega_3 \mathbf{q}_1 \\ \dot{\mathbf{q}}_3 &= \omega_2 \mathbf{q}_1 - \omega_1 \mathbf{q}_2 \end{cases}\end{aligned} \tag{2.15}$$

with $\boldsymbol{\omega}$ some vector in $\mathbb{R}^3$, are highly coupled, consisting of 9 differential equations which depend only on a 3-dimensional input $\boldsymbol{\omega}$, so there is a lot of redundant computation. Another problem is that due to numerical errors, the spacecraft's attitude matrix $\mathbf{Q}$, will eventually stop being a special orthogonal

[2] From the greek for "equal form".

[3] A group is a set of elements that are related by a certain algebraic structure, for more on groups consult [30] and [31].

matrix. This means that while performing numerical integration of the system of differential equations, we will have to constantly check that $\det(\mathbf{Q}) \approx 1$ and that $\sqrt{\langle \mathbf{Q}^\top \mathbf{Q} - \mathbf{I}, \mathbf{Q}^\top \mathbf{Q} - \mathbf{I} \rangle} \approx 0$, and if the values deviate too much, we will have to find the special orthogonal matrix that best approximates the current $\mathbf{Q}$, adding further computational overhead.

This motivates using parametrizations of $\mathbf{Q}$ that allow us to parsimoniously write down the kinematic equations. One of the most popular parametrizations is the quaternion parametrization which is a hypercomplex number system that describe 3D rotations analogously to how complex numbers describe 2D rotations. In this section, we will derive the quaternion parametrization through another, more intuitive, but equivalent parametrization, called the Euler-Rodrigues parameters.

2.7.1 A Brief Note on 3-Dimensional Parametrizations of $SO(3)$

As we mentioned in Section 2.6, the $SO(3)$ group is a 3-dimensional manifold embedded in 9-dimensional Euclidean space. This fact would lead us to believe it is possible to parametrize $SO(3)$ using 3 parameters. Indeed there are several 3-dimensional parametrizations of $SO(3)$, the most famous and oldest of which are the Euler Angles (which is actually a family of 12 different parametrizations) [23]. Other 3-dimensional parametrizations exist, such as the Axis-Azimuth representation, the Gibbs vector and the Modified Rodrigues Parameters. Although each of these parametrizations have their applications, they have largely fallen out of use as general purpose descriptions of $SO(3)$ for computational applications [23].

There are two main reasons why 3-dimensional parametrizations of $SO(3)$ have fallen out of use. Firstly, because it is not possible to have a global 3-dimensional parametrization of $SO(3)$ that does not have singular points [32]. Singular points of a parametrization $\mathbf{g}$ of a m-dimensional manifold M are points $\mathbf{p} \in \mathbb{R}^m$ where the Jacobian of $\mathbf{g}$ does not have rank m. The Jacobian of a parametrization of M is a mapping from the set of velocities on $\mathbb{R}^m$ at $\mathbf{p}$, i.e. the tangent space $T_{\mathbf{p}}\mathbb{R}^m$, to the set of velocities of M at $\mathbf{g}(\mathbf{p})$, i.e. the tangent space $T_{\mathbf{g}(\mathbf{p})}M$ [25], the fact that the rank of the Jacobian is not equal to its dimension, means that there stops being a one-to-one correspondence between motion in the parameter space and motion in the manifold. This means that even though the spacecraft might have moved in real life, this is not reflected as a change in the value of our parameters.

Secondly, all 3-dimensional parametrizations of $SO(3)$ have nonlinear composition rules, which leads to nonlinear kinematic equations [23]. For example, the kinematics equation for a set of Euler Angles is given by [32]

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\phi) & \sin(\phi)\cos(\theta) \\ 0 & -\sin(\phi) & \cos(\phi)\cos(\theta) \end{bmatrix}^{-1} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}$$

which is highly nonlinear. Contrast it with the kinematic equations in (2.15), which although has 6 more equations and is still nonlinear, each equation simpler because it is bilinear (i.e. linear if we keep the

$\mathbf{q}_i$'s or $\boldsymbol{\omega}$ constant), so the computational advantage we gain from having to solve less equations when using a 3-dimensional representation is lost by each equation being more complex.

In the following sections we will discuss a parametrization of $SO(3)$ that is nonminimal, i.e. it has more parameters than the dimension of $SO(3)$, but we only have to add one extra parameter to have a global and non-singular parametrization of $SO(3)$ with bilinear kinematics.

2.7.2 The Euler-Rodrigues Symmetric Parameters

As mentioned in Section 2.6, at a given time instance, the motion of a particle $\mathbf{p}$ under rotation is perpendicular to the axis defined by the angular velocity vector. It is intuitive that if the direction of the angular velocity is fixed between t_1 and t_2, then the final position, $\mathbf{p}(t_2)$, will be a 2-dimensional rotation on the plane perpendicular to $\boldsymbol{\omega}$ that passes through $\mathbf{p}(t_1)$, and that the angle of that rotation will be $\theta = \int_{t_1}^{t_2} \|\boldsymbol{\omega}(t)\| dt$. A little less obvious is the fact that even if the direction of the angular velocity changes over time, the final result of any 3-dimensional rotation, no matter how complicated, is always a 2-dimensional rotation of some angle θ on some plane defined by the unit normal vector $\mathbf{u}$, which we call the axis vector, this result is called Euler's theorem on the axis of rotation [33].

Using this fact, consider the construction shown in Figure 2.9, which shows the clockwise rotation of $\mathbf{p}$, by an angle θ, about the axis defined by the unit vector $\mathbf{u}$. This rotation is represented by matrix $\mathbf{Q}$. On an inner product space, given two vectors $\mathbf{p}$ and $\mathbf{u}$, one can always write $\mathbf{p}$ as

$$\mathbf{p} = \mathbf{p}_{||} + \mathbf{p}_{\perp},$$

where $\mathbf{p}_{||}$ is a vector parallel to $\mathbf{u}$ and $\mathbf{p}_{\perp}$ is a vector perpendicular to $\mathbf{u}$. For $\mathbf{u}$ unit norm, we have

$$\mathbf{p}_{||} = \langle \mathbf{p}, \mathbf{u} \rangle \mathbf{u} = \mathbf{u}\mathbf{u}^\top \mathbf{p}$$

and

$$\mathbf{p}_{\perp} = \mathbf{p} - \mathbf{p}_{||} = (\mathbf{I} - \mathbf{u}\mathbf{u}^\top)\mathbf{p}.$$

Under the rotation $\mathbf{Q}$, only the perpendicular component changes, as shown in Figure 2.9,

$$\mathbf{p}' = \mathbf{Q}\mathbf{p} = \mathbf{p}_{||} + \mathbf{Q}\mathbf{p}_{\perp} = \mathbf{u}\mathbf{u}^\top \mathbf{p} + \mathbf{p}'_{\perp}. \tag{2.16}$$

With respect to the orthogonal basis $\{\mathbf{u}, \mathbf{p}_{\perp}, \mathbf{u} \times \mathbf{p}\}$, $\mathbf{p}'_{\perp}$ is written as

$$\mathbf{p}'_{\perp} = \cos(\theta)\mathbf{p}_{\perp} + \sin(\theta)\mathbf{u} \times \mathbf{p} = \cos(\theta)(\mathbf{I} - \mathbf{u}\mathbf{u}^\top)\mathbf{p} + \sin(\theta)[\mathbf{u}]_\times \mathbf{p},$$

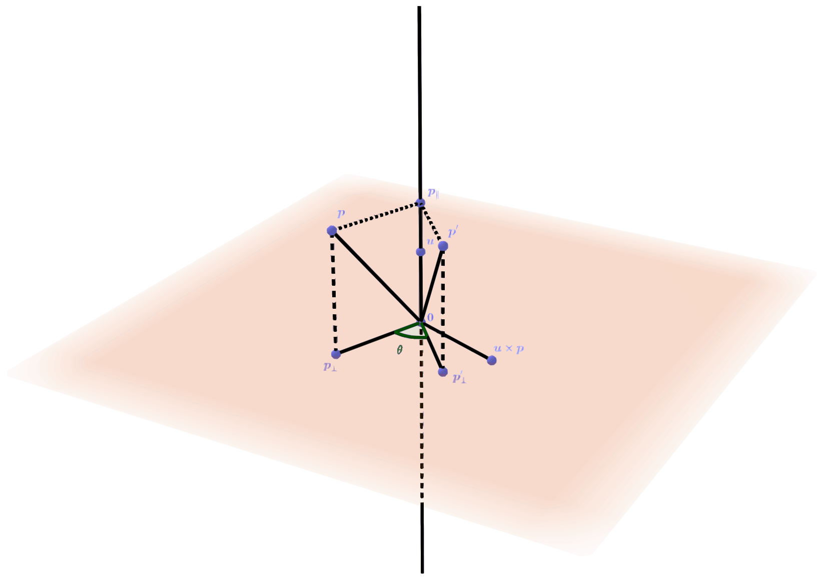


Figure 2.9: Rotation of $\mathbf{p}$ by angle θ about the axis defined by $\mathbf{u}$.

which substituting into (2.16) gives

$$\mathbf{Q}\mathbf{p} = (\cos(\theta)\mathbf{I} + (1 - \cos\theta)\mathbf{u}\mathbf{u}^{\top} + \sin(\theta)[\mathbf{u}]_{\times})\mathbf{p}.$$

This means that we can parametrize the rotation by the angle of rotation and its axis vector

$$\mathbf{Q}(\theta, \mathbf{u}) = \cos(\theta)\mathbf{I} + (1 - \cos\theta)\mathbf{u}\mathbf{u}^{\top} + \sin(\theta)[\mathbf{u}]_{\times}, \tag{2.17}$$

where $\theta \in [-\pi, \pi)$ and $\mathbf{u} \in \mathcal{S}^2$. The parametrization is, of course, not unique. Given a rotation of angle θ and axis vector $\mathbf{u}$, then the rotation has another parametrization, given by the rotation angle $-\theta$ and axis vector $-\mathbf{u}$.

Using the vector tripe product expansion[4]

$$\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) = \mathbf{b}(\mathbf{a} \cdot \mathbf{c}) - \mathbf{c}(\mathbf{a} \cdot \mathbf{b}), \tag{2.18}$$

with $\mathbf{a} = \mathbf{u}$, $\mathbf{b} = \mathbf{u}$, $\mathbf{c} = \mathbf{v}$ for some vector $\mathbf{v}$, we get

$$\mathbf{u} \times (\mathbf{u} \times \mathbf{v}) = \mathbf{u}\mathbf{u}^{\top}\mathbf{v} - \mathbf{v} \Leftrightarrow [\mathbf{u}]_{\times}^2\mathbf{v} = (\mathbf{u}\mathbf{u}^{\top} - \mathbf{I})\mathbf{v} \implies [\mathbf{u}]_{\times}^2 + \mathbf{I} = \mathbf{u}\mathbf{u}^{\top}.$$

[4] Also known by its mnemonic, the “bac cab” identity.

Performing the substitution in (2.17) results in

$$\mathbf{Q} = \mathbf{I} + (1 - \cos\theta)[\mathbf{u}]_\times^2 + \sin(\theta)[\mathbf{u}]_\times.$$

Using familiar trigonometric identities, we can arrive at

$$\mathbf{Q} = \mathbf{I} + \Big(\underbrace{1 - \cos^2\frac{\theta}{2}}_{\sin^2\frac{\theta}{2}} + \sin^2\frac{\theta}{2}\Big)[\mathbf{u}]_\times^2 + 2\cos\frac{\theta}{2}\sin\frac{\theta}{2}[\mathbf{u}]_\times = \mathbf{I} + 2[\mathbf{q}_v]_\times^2 + 2q_s[\mathbf{q}_v]_\times$$

where

$$q_s = \cos\frac{\theta}{2} \qquad \mathbf{q}_v = \sin\frac{\theta}{2}u = \sin\frac{\theta}{2}u_1\mathbf{i} + \sin\frac{\theta}{2}u_2\mathbf{j} + \sin\frac{\theta}{2}u_3\mathbf{k}$$

and $\{\mathbf{i},\mathbf{j},\mathbf{k}\} = \{(1,0,0),(0,1,0),(0,0,1)\}$ is called the natural, canonical or standard basis on $\mathbb{R}^3$. The scalar q_s along with the three components of $\mathbf{q}_v$ are called the Euler-Rodrigues symmetric parameters, which we will call Euler-Rodrigues parameters for short[5].

2.7.3 The Composition of Euler-Rodrigues Parameters

If we write the Euler-Rodrigues parameters as a vector

$$\mathbf{q} = \left(\cos\frac{\theta}{2}, \sin\frac{\theta}{2}u_1, \sin\frac{\theta}{2}u_2, \sin\frac{\theta}{2}u_3\right),$$

then we can easily check that any set of Euler-Rodrigues parameters result in a unit norm vector, which means that any vector of Euler-Rodrigues parameters is in the 3-dimensional sphere in $\mathbb{R}^4$

$$\mathbf{q} \in \mathcal{S}^3 = \{(a,b,c,d) \in \mathbb{R}^4 : a^2 + b^2 + c^2 + d^2 = 1\},$$

which is a 3-dimensional manifold embedded in Euclidean 4-space.

Even though we consider $\mathbf{q} \in \mathbb{R}^4$, we will write it as separate scalar and vector components

$$q = (q_s, \mathbf{q}_v) = \left(\cos\frac{\theta}{2}, \sin\frac{\theta}{2}u_1\mathbf{i} + \sin\frac{\theta}{2}u_2\mathbf{j} + \sin\frac{\theta}{2}u_3\mathbf{k}\right) \in \mathbb{R} \times \mathbb{R}^3$$

and treat $\mathbf{q}_v$ as a vector in $\mathbb{R}^3$. We from now on consider any rotation to be given by the parameterization $\mathbf{R} : \mathcal{S}^3 \to SO(3)$,

$$\mathbf{R}(\mathbf{q}) = \mathbf{I} + 2[\mathbf{q}_v]_\times^2 + 2q_s[\mathbf{q}_v]_\times. \tag{2.19}$$

Using these parameters, french mathematician Olinde Rodrigues discovered that given any two ro-

[5]Although the inclusion of Euler in the name is debatable [34].

tations $\mathbf{Q}_1$ and $\mathbf{Q}_2$, with Euler-Rodrigues parameters $\mathbf{q} = (q_s, \mathbf{q}_v)$, $\mathbf{p} = (p_s, \mathbf{p}_v)$, respectively, the Euler-Rodrigues parameters $\mathbf{r} = (r_s, \mathbf{r}_v)$ of their composition

$$\mathbf{Q}_3 = \mathbf{Q}_2\mathbf{Q}_1 \Leftrightarrow \mathbf{R}(\mathbf{r}) = \mathbf{R}(\mathbf{p})\mathbf{R}(\mathbf{q})$$

are given by the formulas

$$\cos\frac{\theta_3}{2} = r_s = q_s \cdot p_s - \mathbf{q}_v \cdot \mathbf{p}_v \ , \qquad (2.20)$$

$$\sin\frac{\theta_3}{2}\mathbf{u}_3 = \mathbf{r}_v = q_s\mathbf{p}_v + p_s\mathbf{q}_v - \mathbf{p}_v \times \mathbf{q}_v \ . \qquad (2.21)$$

A derivation of these formulas following Rodrigues original method of spherical trigonometry can be seen in [22], a more modern derivation using vector algebra not available to Rodrigues can be seen in [35].

Writing the Euler-Rodrigues parameters in vector form

$$\mathbf{q}_1 = \begin{bmatrix} q_s \\ \mathbf{q}_v \end{bmatrix}, \qquad \mathbf{q}_2 = \begin{bmatrix} p_s \\ \mathbf{p}_v \end{bmatrix}, \qquad \mathbf{q}_3 = \begin{bmatrix} r_s \\ \mathbf{r}_v \end{bmatrix},$$

we can express the Rodrigues formulas as matrix operations

$$\mathbf{q}_3 = \underbrace{\begin{bmatrix} p_s & -\mathbf{p}_v^\top \\ \mathbf{p}_v & p_sI - [\mathbf{p}_v]_\times \end{bmatrix}}_{[\mathbf{q}_2]_R} \begin{bmatrix} q_s \\ \mathbf{q}_v \end{bmatrix} = \underbrace{\begin{bmatrix} q_s & -\mathbf{q}_v^\top \\ \mathbf{q}_v & q_sI + [\mathbf{q}_v]_\times \end{bmatrix}}_{[\mathbf{q}_1]_L} \begin{bmatrix} p_s \\ \mathbf{p}_v \end{bmatrix},$$

or more succinctly as the binary operator $\otimes : \mathbb{R}^4 \times \mathbb{R}^4 \to \mathbb{R}^4$, $\mathbf{q}_1 \otimes \mathbf{q}_2 = [\mathbf{q}_2]_R \cdot \mathbf{q}_1 = [\mathbf{q}_1]_L \cdot \mathbf{q}_2$. Note that the composition of Euler-Rodrigues parameter vectors using the $\otimes$ operator is done left-to-right while the composition of rotation matrices is right-to-left. This is a source of much confusion when using quaternions to represent attitude and we will explain the reason for this when we relate the Euler-Rodrigues parameters to the quaternions in the next section

It is easily checked that $\otimes$ is associative and distributive. The inverse $\mathbf{Q}^\top$ of any rotation $\mathbf{Q}$ of angle θ about $\mathbf{u}$ is a rotation of angle $-\theta$ about $\mathbf{u}$. Using the standard trigonometric identities, this means that if

$$\mathbf{Q} \mapsto \mathbf{q} = \begin{bmatrix} q_s \\ \mathbf{q}_v \end{bmatrix}, \quad \text{then} \quad \mathbf{Q}^\top \mapsto \overline{\mathbf{q}} = \begin{bmatrix} q_s \\ -\mathbf{q}_v \end{bmatrix},$$

where $\overline{\mathbf{q}}$ is called the conjugate of $\mathbf{q}$. It is easily checked that if $\|\mathbf{q}\| = 1$, then

$$\mathbf{q} \otimes \overline{\mathbf{q}} = \overline{\mathbf{q}} \otimes \mathbf{q} = \mathbf{e}$$

where $\mathbf{e} = (1, 0) \in \mathcal{S}^3$ corresponds to the identity matrix and likewise the identity element for the $\otimes$ operation. From this we conclude that if $\mathbf{q}$ is the parametrization of the rotation $\mathbf{Q}$, $\overline{\mathbf{q}}$ is the parametrization of its inverse $\mathbf{Q}^\top$. Furthermore, we have that

$$\mathbf{r} = \mathbf{q} \otimes \mathbf{p} \implies \overline{\mathbf{r}} = \overline{\mathbf{p}} \otimes \overline{\mathbf{q}}. \tag{2.22}$$

2.7.4 Quaternions and the Symplectic Group

The fact that on the manifold $\mathcal{S}^3$ we can define an associative binary operation, that for every $\mathbf{q} \in \mathcal{S}^3$ there exists $\overline{\mathbf{q}} \in \mathcal{S}^3$, which is its multiplicative inverse, and that there is an identity element $\mathbf{e} \in \mathcal{S}^3$, and that $\mathcal{S}^3$ is closed under $\otimes$, means that to addition of $\mathcal{S}^3$ being a manifold, it is also a group. Like $SO(3)$, $\mathcal{S}^3$ is a Lie Group!

Much independently and a few years after Olinde Rodrigues had published his work on the parameterization of rotations, Irish mathematician William Hamilton attempted to create a number system that extended the complex numbers into 3-dimensions by adding another imaginary component j to form 3-dimensional complex number $a + bi + cj$. He failed at this task, realizing that for multiplication to work he needed to add another imaginary component, k. The relation between the imaginary components is given by

$$i^2 = j^2 = k^2 = ijk = -1.$$

The resulting 4-dimensional number system is almost like $\mathbb{R}$ and $\mathbb{C}$ except that multiplication is not commutative. Hamilton called the resulting number a quaternion, and we denote by $\mathbb{H}$ the set of all quaternions,

$$\mathbb{H} = \{a + bi + cj + dk : a, b, c, d \in \mathbb{R}\}.$$

For the pure imaginary part of the quaternion, Hamilton coined the term “vector” (which is why the three basis vectors of the standard basis of $\mathbb{R}^3$ are written as $\{\mathbf{i}, \mathbf{j}, \mathbf{k}\}$) [34]. The reason why this is relevant to our discussion, is that the algebraic structure of the multiplication of quaternions is exactly the same as the one that was just described for the composition of Euler-Rodrigues parameters!

The norm of a quaternion $q = a + bi + cj + dk$ is given by $|q| = a^2 + b^2 + c^2 + d^2$ and the set of all unit norm quaternions is called the Symplectic group (of dimension 1)

$$Sp(1) = \{q \in \mathbb{H} : |q| = 1\}$$

For $q_1, q_2 \in Sp(1)$, we have that the mapping $\varphi : Sp(1) \to \mathcal{S}^3$, $\varphi(a + bi + cj + dk) = (a, b, c, d)$ is bijective and that

$$\varphi(q_1 \cdot q_2) = \varphi(q_1) \otimes \varphi(q_2), \forall q_1, q_2 \in Sp(1)$$

i.e. the algebraic structure is preserved, so φ is an isomorphism.

This is the reason why the composition of Euler-Rodrigues parameters using the $\otimes$ operator is done in reverse order of the composition of rotations. It is to be consistent with the quaternion algebra as defined by Hamilton. This does not need to be the case, as we can define the multiplication of a 4-dimensional number system with imaginary units i,j,k with the rules

$$i^2 = j^2 = k^2 = kji = -1,$$

as has been done by the author of [23] and gained adoption at National Aerospace Agency (NASA) Jet Propulsion Laboratory (JPL), with uses in the Space Shuttle and International Space Station (ISS) programs [20]. However, outside of the work done at JPL, most published papers in various areas of research use Hamilton's definition [20], so that is the definition that we will use. Alternatively, we could consider the quaternion to be the representation of the passive transformation $\mathbf{Q}^\top$, i.e. $q \mapsto \mathbf{Q}^\top$, as is done in [24], in which case we get $\mathbf{Q}_3 = \mathbf{Q}_2\mathbf{Q}_1 \mapsto q_3 = q_2 \cdot q_1$ using Hamilton's definition, by property (2.22). But we will not do this. We will consider the quaternion to be the representation of the active transformation as we have done so far.

The Lie Group $\mathcal{S}^3$ is isomorphic to the Lie group $Sp(1)$. For this reason, we call the vector of Euler-Rodrigues parameters a quaternion, the operator $\otimes$ the quaternion multiplication operator, and for $\mathbf{q} = (q_s, \mathbf{q}_v) \in \mathbb{R} \times \mathbb{R}^3$ we call q_s the scalar part of the quaternion and $\mathbf{q}_v$ the vector part of the quaternion. We will continue our treatment of describing rotations as quaternions using 4-dimensional unit vectors $\mathbf{q} \in \mathcal{S}^3$. For more on the fascinating history of Hamilton's quaternions see [34].

As mentioned in Section 2.7.2, when parametrizing a rotation using the angle of rotation θ and the axis vector $\mathbf{u}$, the parameters $-\theta$ and $-\mathbf{u}$ result in the same rotation. When using quaternions to parametrize rotations, we can see from (2.19) that the antipodal points $\mathbf{q}$ and $-\mathbf{q}$ parametrize the same rotation. Since the mapping $\mathcal{S}^3 \to SO(3)$ is 2-to-1, we call $\mathcal{S}^3 \cong Sp(1)$ a double cover of $SO(3)$ [31].

2.7.5 The Quaternion Rotation Formula

We denote the set of pure quaternions as $\mathfrak{sp}(1) = \{ai + bj + ck : a, b, c \in \mathbb{R}\} \cong \mathfrak{s}(3) = \{(0, a, b, c) : a, b, c \in \mathbb{R}\}$. There is a natural one-to-one correspondence between the Euclidean 3-space vectors and the pure quaternions, with the map

$$\begin{aligned} \widehat{\ } : \mathbb{R}^3 &\to \mathfrak{s}(3) \\ (a, b, c) &\mapsto (0, a, b, c) \end{aligned}$$

which we call the "hat" map.

We saw that unit quaternions represent rotations. Any non-unit quaternion $\mathbf{q} \in \mathbb{R}^4$ can be thought of as representing the linear transformations which consist of a rotation, represented by $\mathbf{q}/\|\mathbf{q}\|$, composed with a linear transformation that scales a vector by a factor of $\|\mathbf{q}\|$, called a dilation. If one considers a pure quaternion $\widehat{\mathbf{x}} \in \mathfrak{s}(3)$, this quaternion represents the linear transformation consisting of a rotation of angle π over the axis vector $\mathbf{x}/\|\mathbf{x}\|$, composed with the dilation by a factor of $\|\mathbf{x}\|$.

One of the most important results of linear algebra is that if $\mathbf{A}$ is a change of basis matrix from a basis $\mathcal{E}$ to a basis $\mathcal{E}'$, and $\mathbf{B}$ is the representation of some linear transformation in basis $\mathcal{E}'$, then the matrix $\mathbf{C}$ which represents the same transformation as $\mathbf{B}$ but in basis $\mathcal{E}$ is given by the conjugation [31]

$$\mathbf{C} = \mathbf{A}\mathbf{B}\mathbf{A}^{-1}.$$

Given the linear transformation that rotates a vector by π over the axis vector $\mathbf{x}/\|\mathbf{x}\|$ and then performs the dilation by $\|\mathbf{x}\|$, this is represented in the body-fixed frame by the matrix ${}^{\mathcal{B}}\mathbf{X} = \|\mathbf{x}\| \cdot \mathbf{R}(\pi, {}^{\mathcal{B}}\mathbf{x}/\|\mathbf{x}\|) \mapsto \widehat{{}^{\mathcal{B}}\mathbf{x}}$ and in the inertial reference frame, by ${}^{\mathcal{N}}\mathbf{X} = \|\mathbf{x}\| \cdot \mathbf{R}(\pi, {}^{\mathcal{N}}\mathbf{x}/\|\mathbf{x}\|) \mapsto \widehat{{}^{\mathcal{N}}\mathbf{x}}$. Since the spacecraft's attitude $\mathbf{Q}$ is the change of basis matrix from $\mathcal{B}$ to $\mathcal{N}$, we get

$${}^{\mathcal{N}}\mathbf{X} = \mathbf{Q}\,{}^{\mathcal{B}}\mathbf{X}\mathbf{Q}^{\top},$$

or in quaternion form

$$\widehat{{}^{\mathcal{N}}\mathbf{x}} = \overline{\mathbf{q}} \otimes \widehat{{}^{\mathcal{B}}\mathbf{x}} \otimes \mathbf{q}, \tag{2.23}$$

where choosing $\mathbf{q}$ or $-\mathbf{q}$ to represent $\mathbf{Q}$ is irrelevant, since the minus signs cancel. Equivalently, we have

$$\widehat{{}^{\mathcal{B}}\mathbf{x}} = \mathbf{q} \otimes \widehat{{}^{\mathcal{N}}\mathbf{x}} \otimes \overline{\mathbf{q}}. \tag{2.24}$$

Either (2.23) or (2.24), are called the quaternion rotation action or, more whimsically, the sandwich product [24]. Although, in the way it has been described, nothing is actually being rotated, we are just performing conjugation to describe a transformation in another basis, it just turns out that the transformation we are describing (a binary rotation composed with a dilation) has a very direct mapping to a vector in $\mathbb{R}^3$ [22]. An alternative geometrical explanation for the quaternion rotation action, where one can consider that a quaternion is in fact being rotated is presented in [24].

2.7.6 Quaternion Kinematics

If $\mathbf{x}$ is a vector whose coordinates in the body-fixed frame are constant, we have that, taking the derivative of the sandwhich product from (2.24), we get

$$\begin{aligned}\widehat{{}^{\mathcal{N}}\dot{\mathbf{x}}} &= \dot{\overline{\mathbf{q}}} \otimes \widehat{{}^{\mathcal{B}}\mathbf{x}} \otimes \mathbf{q} + \overline{\mathbf{q}} \otimes \widehat{{}^{\mathcal{B}}\mathbf{x}} \otimes \dot{\mathbf{q}} \Leftrightarrow \\ \widehat{{}^{\mathcal{N}}\dot{\mathbf{x}}} &= \dot{\overline{\mathbf{q}}} \otimes \mathbf{q} \otimes \widehat{{}^{\mathcal{N}}\mathbf{x}} \otimes \underbrace{\overline{\mathbf{q}} \otimes \mathbf{q}}_{\mathbf{e}} + \underbrace{\overline{\mathbf{q}} \otimes \mathbf{q}}_{\mathbf{e}} \otimes \widehat{{}^{\mathcal{N}}\mathbf{x}} \otimes \overline{\mathbf{q}} \otimes \dot{\mathbf{q}},\end{aligned} \tag{2.25}$$

since $\|\mathbf{q}\| = 1$, we have that $\overline{\mathbf{q}} \otimes \mathbf{q} = \mathbf{e}$. Taking the derivative of this condition we have

$$\dot{\overline{\mathbf{q}}} \otimes \mathbf{q} + \overline{\mathbf{q}} \otimes \dot{\mathbf{q}} = 0 \Leftrightarrow \overline{\mathbf{q}} \otimes \dot{\mathbf{q}} = -\dot{\overline{\mathbf{q}}} \otimes \mathbf{q} = -\overline{\overline{\mathbf{q}} \otimes \dot{\mathbf{q}}}.$$

Since $\overline{\mathbf{q}} \otimes \dot{\mathbf{q}}$ is the the symmetric of its conjugate, it must be the case that it is a pure quaternion, which we write as

$$\widehat{\boldsymbol{\xi}} = \overline{\mathbf{q}} \otimes \dot{\mathbf{q}}, \; \boldsymbol{\xi} \in \mathbb{R}^3. \tag{2.26}$$

Substituting (2.26) into (2.25), we get

$$\widehat{{}^{\mathcal{N}}\dot{\mathbf{x}}} = \overline{\widehat{\boldsymbol{\xi}}} \otimes \widehat{{}^{\mathcal{N}}\mathbf{x}} + \widehat{{}^{\mathcal{N}}\mathbf{x}} \otimes \widehat{\boldsymbol{\xi}}$$

and writing $\mathbf{p} = \widehat{{}^{\mathcal{N}}\mathbf{x}} \otimes \widehat{\boldsymbol{\xi}} \implies \overline{\mathbf{p}} = -\overline{\widehat{\boldsymbol{\xi}}} \otimes \widehat{{}^{\mathcal{N}}\mathbf{x}}$, results in

$$\widehat{{}^{\mathcal{N}}\dot{\mathbf{x}}} = -\overline{\mathbf{p}} + \mathbf{p} \Leftrightarrow \begin{bmatrix} 0 \\ {}^{\mathcal{N}}\boldsymbol{\omega} \times {}^{\mathcal{N}}\mathbf{x} \end{bmatrix} = -\begin{bmatrix} p_s \\ -\mathbf{p}_v \end{bmatrix} + \begin{bmatrix} p_s \\ \mathbf{p}_v \end{bmatrix} = \begin{bmatrix} 0 \\ 2\mathbf{p}_v \end{bmatrix}. \tag{2.27}$$

From (2.27) and since $\mathbf{p}_v = -\boldsymbol{\xi} \times {}^{\mathcal{N}}\mathbf{x}$ from quaternion multiplication, we see that

$$\mathbf{p}_v = -\frac{1}{2}{}^{\mathcal{N}}\boldsymbol{\omega} \times {}^{\mathcal{N}}\mathbf{x} = -\boldsymbol{\xi} \times {}^{\mathcal{N}}\mathbf{x} \implies \boldsymbol{\xi} = -\frac{1}{2}{}^{\mathcal{N}}\boldsymbol{\omega},$$

and from (2.26) we get

$$-\frac{1}{2}\widehat{{}^{\mathcal{N}}\boldsymbol{\omega}} = \overline{\mathbf{q}} \otimes \dot{\mathbf{q}} \implies \dot{\mathbf{q}} = -\frac{1}{2}\mathbf{q} \otimes \widehat{{}^{\mathcal{N}}\boldsymbol{\omega}},$$

or by performing the conjugation $\widehat{{}^{\mathcal{N}}\boldsymbol{\omega}} = \overline{\mathbf{q}} \otimes \widehat{{}^{\mathcal{B}}\boldsymbol{\omega}} \otimes \mathbf{q}$, we arrive at the quaternion kinematics equation

$$\dot{\mathbf{q}} = -\frac{1}{2}\widehat{{}^{\mathcal{B}}\boldsymbol{\omega}} \otimes \mathbf{q}. \tag{2.28}$$

Note that the quaternion kinematics equation (2.28) is different from what might appear in other literature such as [24], since we are using the quaternion to represent the rotation as an active transformation. If, instead, we were considering rotation as a passive transformation, then the quaternion that would represent the rotation would be the conjugate of the one we are currently using. If we conjugate

(2.28) we obtain the kinematics equation for the quaternion representing the passive transformation

$$\dot{\overline{\mathbf{q}}} = \frac{1}{2}\overline{\mathbf{q}} \otimes \widehat{{}^{\mathcal{B}}\boldsymbol{\omega}}.$$

2.8 Rigid Body Dynamics

Given a particle $\mathbf{p}_i : \mathbb{R}_0^+ \to \mathbb{R}^3$, we associate to that particle a given mass m_i, which we assume to be constant. The linear momentum of the particle is given by

$$\boldsymbol{\rho}_i(t) = m_i \dot{\mathbf{p}}_i(t).$$

Newton's Second Law of Motion tells us that, in an inertial reference frame, the linear momentum of the particle obeys the differential equation

$${}^{\mathcal{N}}\dot{\boldsymbol{\rho}}_i(t) = {}^{\mathcal{N}}\mathbf{F}_i(t),$$

where ${}^{\mathcal{N}}\mathbf{F}_i(t)$ are the inertial coordinates of the force vector. The fact that this equation is only valid on inertial reference frames is the reason why they are also called Newtonian.

Now, consider a set of particles $\{\mathbf{p}_i\}_{i\in\mathcal{I}}$, which is called a system of particles in physics, where the index set $\mathcal{I}$ is finite. We define the center of mass of the system of particles to be given by

$$\mathbf{c} = \frac{\sum_{i\in\mathcal{I}} m_i \mathbf{p}_i}{m},$$

where $m = \sum_{i\in\mathcal{I}} m_i$ is the total mass of the system of particles.

The angular momentum of a particle $\mathbf{p}_i$ of the system of particles $\{\mathbf{p}_i\}_{i\in\mathcal{I}}$, about the center of mass, which we will call simply the angular momentum of particle $\mathbf{p}_i$, is given by

$$\boldsymbol{\ell}_i = \underbrace{(\mathbf{p}_i - \mathbf{c})}_{\mathbf{p}_i'} \times \boldsymbol{\rho}_i. \tag{2.29}$$

The moment of force, or torque, applied to particle $\mathbf{p}_i$ about the center of mass, which we will refer to simply as the torque applied particle $\mathbf{p}_i$, is given by

$$\boldsymbol{\tau}_i = (\mathbf{p}_i - \mathbf{c}) \times \mathbf{F}_i = \mathbf{p}_i' \times \dot{\boldsymbol{\rho}}_i.$$

Now, consider the rigid body $\mathcal{O} = \{\mathbf{p}_i\}_{i\in\mathcal{I}} \cup \mathbf{c}$ where $\mathbf{c}$ is the center of mass and let $\mathcal{B}$ be the body-fixed frame with origin at the center of mass of the rigid body. The relation between the coordinates in the inertial frame $\mathcal{N}$ and the body fixed frame $\mathcal{B}$ is given by the affine transformation

$$^{\mathcal{N}}\mathbf{p}_i = \mathbf{Q}\,^{\mathcal{B}}\mathbf{p}'_i + {}^{\mathcal{N}}\mathbf{c}.$$

Taking the derivative, we get that the velocity of the particle is

$$\begin{aligned}\dot{\mathbf{p}}_i &= \mathbf{N}\,^{\mathcal{N}}\dot{\mathbf{p}}_i = \mathbf{N}(\dot{\mathbf{Q}}\,^{\mathcal{B}}\mathbf{p}'_i + {}^{\mathcal{N}}\dot{\mathbf{c}}) = \mathbf{N}(\mathbf{N}^\top\mathbf{B})[^{\mathcal{B}}\boldsymbol{\omega}]_\times\,^{\mathcal{B}}\mathbf{p}'_i + \dot{\mathbf{c}} = [\omega]_\times\mathbf{B}\mathbf{Q}^\top\,^{\mathcal{N}}\mathbf{p}' + \dot{\mathbf{c}},\\ &= \boldsymbol{\omega}\times\mathbf{p}'_i + \dot{\mathbf{c}}.\end{aligned} \tag{2.30}$$

Using the product rule, we have that the derivative of the angular momentum of particle $\mathbf{p}_i$ is

$$\dot{\boldsymbol{\ell}}_i = \dot{\mathbf{p}_i}' \times \boldsymbol{\rho}_i + \mathbf{p}'_i \times \dot{\boldsymbol{\rho}}_i = (\dot{\mathbf{p}}_i - \dot{\mathbf{c}}) \times (m_i\dot{\mathbf{p}}_i) + \boldsymbol{\tau}_i = \boldsymbol{\tau}_i - \dot{\mathbf{c}} \times m_i\dot{\mathbf{p}}_i.$$

Summing over all the particles of the system, we get the derivative of the total angular momentum of the rigid body

$$\dot{\boldsymbol{\ell}} = \sum_{i\in\mathcal{I}}\dot{\boldsymbol{\ell}}_i = \sum_{i\in\mathcal{I}}\boldsymbol{\tau}_i - \dot{\mathbf{c}}\times\left(\sum_{i\in\mathcal{I}} m_i\dot{\mathbf{p}}_i\right) = \boldsymbol{\tau} - \dot{\mathbf{c}}\times(m\dot{\mathbf{c}}) \Leftrightarrow \dot{\boldsymbol{\ell}} = \boldsymbol{\tau},$$

where $\boldsymbol{\tau} = \sum_{i\in\mathcal{I}}\boldsymbol{\tau}_i$ is the total torque applied to the rigid body.

Using (2.29) and (2.30), we get that the total momentum of the system of particles is

$$\boldsymbol{\ell} = \sum_{i\in\mathcal{I}}\boldsymbol{\ell}_i = \sum_{i\in\mathcal{I}} m_i\mathbf{p}'_i \times \boldsymbol{\omega} \times \mathbf{p}'_i + \left(\sum_{i\in\mathcal{I}} m_i\mathbf{p}'_i\right)\times\dot{\mathbf{c}}, \tag{2.31}$$

which can be simplified, first by applying the vector triple product expansion from (2.18) on the first term

$$\sum_{i\in\mathcal{I}} m_i\mathbf{p}'_i\times\boldsymbol{\omega}\times\mathbf{p}'_i = \left(\sum_{i\in\mathcal{I}} m_i(\|\mathbf{p}'_i\|^2 I - \mathbf{p}'_i\mathbf{p}'^\top_i)\right)\boldsymbol{\omega} = \mathbf{J}\boldsymbol{\omega},$$

where

$$\mathbf{J} = \sum_{i\in\mathcal{I}} m_i(\|\mathbf{p}'_i\|^2 I - \mathbf{p}'_i\mathbf{p}'^\top_i)$$

is called the **inertia matrix** (or inertia tensor) of the rigid body. Further, we have that the second term of (2.31) vanishes, since

$$\sum_{i\in\mathcal{I}} m_i\mathbf{p}'_i = \underbrace{\sum_{i\in\mathcal{I}} m_i\mathbf{p}_i}_{m\mathbf{c}} - \underbrace{\left(\sum_{i\in\mathcal{I}} m_i\right)}_{m}\mathbf{c} = \mathbf{0}.$$

Hence, we get that the total angular momentum of the system of particles is given by

$$\boldsymbol{\ell} = \mathbf{J}\boldsymbol{\omega}.$$

Note that since we are considering the origin of the body fixed frame to be $\mathbf{c}$, then the components of

$\mathbf{p}'_i$ in the body frame, ${}^{\mathcal{B}}\mathbf{p}'_i$ are constant, as mentioned in Section 2.4. That means that the inertia matrix w.r.t. the body frame, given by

$$ {}^{\mathcal{B}}\mathbf{J} = \sum_{i\in\mathcal{I}} m_i(\| {}^{\mathcal{B}}\mathbf{p}'_i\|^2\mathbf{I} - {}^{\mathcal{B}}\mathbf{p}'_i \, {}^{\mathcal{B}}\mathbf{p}'^{\top}_i) = \sum_{i\in\mathcal{I}} m_i(\|\mathbf{p}'_i\|^2\mathbf{B}^\top\mathbf{B} - \mathbf{B}^\top\mathbf{p}'_i\mathbf{p}'^{\top}_i\mathbf{B}) = \mathbf{B}^\top\mathbf{J}\mathbf{B} $$

is a constant matrix.

This gives us that the total angular momentum, w.r.t. the body frame is given by

$$ \begin{aligned} \boldsymbol{\ell} &= \mathbf{B}\,{}^{\mathcal{B}}\boldsymbol{\ell} = \mathbf{J}\mathbf{B}\,{}^{\mathcal{B}}\boldsymbol{\omega} \Leftrightarrow \\ {}^{\mathcal{B}}\boldsymbol{\ell} &= {}^{\mathcal{B}}\mathbf{J}\,{}^{\mathcal{B}}\boldsymbol{\omega}, \end{aligned} \tag{2.32} $$

and its derivative is given by

$$ \begin{aligned} \dot{\boldsymbol{\ell}} &= \dot{\mathbf{B}}\,{}^{\mathcal{B}}\boldsymbol{\ell} + \mathbf{B}\,{}^{\mathcal{B}}\dot{\boldsymbol{\ell}} \Leftrightarrow \boldsymbol{\tau} = \mathbf{B}({}^{\mathcal{B}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{\ell} + {}^{\mathcal{B}}\dot{\boldsymbol{\ell}}) \Leftrightarrow \\ {}^{\mathcal{B}}\dot{\boldsymbol{\ell}} &= {}^{\mathcal{B}}\boldsymbol{\ell} \times {}^{\mathcal{B}}\boldsymbol{\omega} + {}^{\mathcal{B}}\boldsymbol{\tau}. \end{aligned} \tag{2.33} $$

Combining (2.32) with (2.33), we get the equation for the rotational dynamics

$$ {}^{\mathcal{B}}\mathbf{J}\,{}^{\mathcal{B}}\dot{\boldsymbol{\omega}} = {}^{\mathcal{B}}\mathbf{J}\,{}^{\mathcal{B}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{\omega} + {}^{\mathcal{B}}\boldsymbol{\tau}, \tag{2.34} $$

which is called Euler's equation for rigid body motion [28]. Note that for this equation to be valid, the key assumptions made were that the body is rigid, the mass of each particle is constant, the body fixed frame has its origin at the center of mass of the rigid body and that the torque is applied about the center of mass of the rigid body.

3

Theory of Control Lyapunov Functions and Control Barrier Functions

Contents

3.1 Control Lyapunov Functions

Consider the control affine system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{G}(\mathbf{x}(t))\mathbf{u}(t) \ , \tag{3.1}$$

where $\mathbf{x}(t) \in \mathcal{X} \subset \mathbb{R}^n$ is the system state at time $t \in \mathbb{R}_0^+$, the set $\mathcal{X}$ is the state space, and $\mathbf{u}(t) \in \mathcal{U} \subset \mathbb{R}^m$ is the control input at time t, which is an element of the control space $\mathcal{U}$. The vector fields $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ and $\mathbf{G} : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are locally Lipschitz. The requirement that the vector fields are locally Lipschitz is imposed to have sufficient conditions to guarantee that for all $\mathbf{u} : \mathbb{R}_0^+ \to \mathbb{R}^m$ piece-wise continuous, the IVP (Initial Value Problem) given by equation (3.1) and the initial condition $\mathbf{x}(0) = \mathbf{x}_0$, has a unique solution $\mathbf{x} : \mathbb{R}_0^+ \to \mathbb{R}^n \in C^1[0,\infty)$ [36]. We wish to stabilize this possibly non-linear system. To do so, we will make use of the concept of control Lyapunov functions.

3.1.1 Intuition for the CLF

Before presenting the formal definition of the CLF, we will attempt to derive it by establishing the conditions a controller needs to fulfill so it stabilizes the system to the desired equilibrium point. Working with system (3.1), lets assume that we can create a feedback control law, $\mathbf{u} = \mathbf{k}(\mathbf{x})$, such that the closed loop system, given by

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{G}(\mathbf{x}(t))\mathbf{k}(\mathbf{x}(t)) = \mathbf{f}_{cl}(\mathbf{x}(t)) \ ,$$

is asymptotically stable at point $\mathbf{x}^* \in \mathcal{X}$. A sufficient condition for the existence of such a controller is that there exists a Lyapunov function $V : X \to \mathbb{R}$ which is positive definite (with $V(\mathbf{x}) = 0 \Leftrightarrow \mathbf{x} = \mathbf{x}^*$) and with $\dot{V}$ negative definite ($\dot{V}(x) < 0, \forall x \in \mathcal{X} \setminus x^*$) [36]. Explicitly deriving the time derivative of V, using the chain-rule results in

$$\begin{aligned}\dot{V} &= \frac{d}{dt}V(\mathbf{x}(t)) = \frac{\partial V}{\partial \mathbf{x}}\frac{d\mathbf{x}}{dt} = \nabla V^\top(\mathbf{x})\dot{\mathbf{x}} = \nabla V^\top(\mathbf{x})\mathbf{f}(\mathbf{x}) + \nabla V^\top(\mathbf{x})\mathbf{G}(\mathbf{x})\mathbf{k}(\mathbf{x}) \\ &= L_\mathbf{f}V(\mathbf{x}) + L_\mathbf{G}V(\mathbf{x})\mathbf{k}(\mathbf{x}).\end{aligned}$$

For our system to be stable, we thus have the requirement that

$$L_\mathbf{f}V(\mathbf{x}) + L_\mathbf{G}V(\mathbf{x})\mathbf{k}(\mathbf{x}) \leq 0, \ \forall \mathbf{x} \in \mathcal{X}. \tag{3.2}$$

Now, if our control law fulfills this condition, we have a stable system. However, this control law might not be practically useful, since, although it stabilizes the system, there is no bound on how long we will actually take to reach the stabilizing point. As such, we can instead enforce a stronger constraint, such as

$$L_{\mathbf{f}}V(\mathbf{x}) + L_{\mathbf{G}}V(\mathbf{x})\mathbf{k}(\mathbf{x}) \leq -\lambda V(\mathbf{x}),\ \forall \mathbf{x} \in \mathcal{X}. \tag{3.3}$$

For $\lambda \in \mathbb{R}^+$, since V is positive definite, we have that $-\lambda V(\mathbf{x}) \leq 0,\ \forall \mathbf{x} \in \mathcal{X}$. One of the simplest Lyapunov candidate functions is the quadratic function, so if we assume that the control law allows the closed loop system to have $V(\mathbf{x}) = 0.5\|\mathbf{x} - \mathbf{x}^*\|^2$ as a Lyapunov function, then we have that

$$\dot{V}(\mathbf{x}(t)) = \left.\frac{\partial V}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}(t)} \dot{\mathbf{x}}(t) = (\mathbf{x}(t) - \mathbf{x}^*)^\top \dot{\mathbf{x}}(t). \tag{3.4}$$

As such, the Initial Value Problem (IVP)

$$\begin{gathered}\dot{V}(\mathbf{x}(t)) = -\lambda V(\mathbf{x}(t))\ , \mathbf{x}(0) = \mathbf{x}_0\ , \\ (\mathbf{x}(t) - \mathbf{x}^*)^\top \dot{\mathbf{x}}(t) = -\frac{\lambda}{2}(\mathbf{x}(t) - \mathbf{x}^*)^\top (\mathbf{x}(t) - \mathbf{x}^*),\ \mathbf{x}(0) = \mathbf{x}_0\ ,\end{gathered}$$

has the solution

$$\mathbf{x}(t) = \begin{cases} \mathbf{x}^*, & \text{if } \mathbf{x}_0 = \mathbf{x}^* \\ \\ (1 - e^{-\frac{\lambda}{2}t})\mathbf{x}^* + e^{-\frac{\lambda}{2}t}\mathbf{x}_0, & \text{otherwise} \end{cases}. \tag{3.5}$$

Defining $\varepsilon(t) = \|\mathbf{x}(t) - \mathbf{x}^*\|$, we have for the non-trivial solution that

$$\varepsilon(t) = e^{-\frac{\lambda}{2}t} \underbrace{\|\mathbf{x}_0 - \mathbf{x}^*\|}_{\varepsilon_0}\ , \tag{3.6}$$

i.e. the distance between the state at time t and the desired state $\mathbf{x}^*$ decays exponentially. So the stronger constraint in the inequality (3.3) allows us to have an upper-bound, t_{des}, on the time it takes to be within a distance ε_{des} of $\mathbf{x}^*$ by choosing λ as

$$\lambda = -\frac{2}{t_{des}} \log\left(\frac{\varepsilon_{des}}{\varepsilon_0}\right)\ , \tag{3.7}$$

where t_{des} is the desired time to be within a distance of ε_{des} of the stabilizing point.

Of course, instead of having $-\lambda V(\mathbf{x})$ on the right-hand side of inequality (3.3), we could have had any expression, so long at it was less than or equal to zero for any $\mathbf{x} \in \mathcal{X}$. Hence, generically we use a class $\mathcal{K}$ function γ in inequality (3.8). The expression on the right-hand side of inequality (3.3) is just the particular case where $\gamma(r) = \lambda r$.

3.1.2 Definition and Application of the CLF

Based on the conceptual intuition gained in the previous section, we now formalize the concept of the CLF in the following definition.

Definition 3.1.1 (Control Lyapunov Function [37]). A positive definite function $V : \mathbb{R}^n \to \mathbb{R}$ is said to be a control Lyapunov function if, for the system described in equation (3.1) it satisfies

$$\inf_{u \in \mathcal{U}} \{L_{\mathbf{f}} V(\mathbf{x}) + L_{\mathbf{G}} V(\mathbf{x})\mathbf{u}\} \leq -\gamma(V(\mathbf{x})), \ \forall \mathbf{x} \in \mathcal{X} \ , \tag{3.8}$$

where $\gamma : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ is a class $\mathcal{K}_\infty$ function.

Definition 3.1.1 is equivalent to saying that

$$K_{CLF}(\mathbf{x}) = \left\{\mathbf{u} \in \mathcal{U} : L_{\mathbf{f}} V(\mathbf{x}) + L_{\mathbf{G}} V(\mathbf{x})\mathbf{u} \leq -\gamma(V(\mathbf{x}))\right\} \neq \emptyset, \forall \mathbf{x} \in \mathcal{X} \ , \tag{3.9}$$

i.e. we can create a feedback control law $\mathbf{u} = \mathbf{k}(\mathbf{x})$ where, $\mathbf{k} : \mathbf{x} \in \mathcal{X} \mapsto \mathbf{u} \in K_{CLF}(\mathbf{x})$.

3.1.2.A Min-norm CLF controller

One of the easiest and most practical ways to synthesize a controller using CLFs is in the form of an optimization problem, where at the state $\mathbf{x} \in \mathcal{X}$ we find the control input $\mathbf{u} \in \mathcal{U}$ with minimum norm that satisfies the CLF condition:

$$\min_{\mathbf{u} \in \mathbb{R}^m, \delta \in \mathbb{R}} \quad \|\mathbf{u}\|^2 + \rho\delta^2 \tag{3.10}$$

$$s.t. : \quad L_{\mathbf{f}} V(\mathbf{x}) + L_{\mathbf{G}} V(\mathbf{x})\mathbf{u} + \delta \leq -\gamma(V(\mathbf{x})) \tag{3.11}$$

$$\mathbf{u} \in \mathcal{U} \tag{3.12}$$

where $\delta \in \mathbb{R}$ is a slack variable, introduced to guarantee feasibility of the optimization problem with the input constraints (3.12), and $\rho \in \mathbb{R}^+ : \rho \geq 1$ is a penalty weight. Note that the addition of the slack variable δ in the CLF constraint (3.11) invalidates the theoretical guarantees for stabilization to the state $\mathbf{x}^*$ for which $V(\mathbf{x}^*) = 0$, since the slack variable allows the time derivative of V to be positive. In practice, however, an appropriately high penalty weight will make it such that $\delta \approx 0$ and thus $\dot{V}(\mathbf{x}) < 0$, $\forall x \in \mathcal{X} \setminus \{\mathbf{x}^*\}$, whenever the CLF constraint is feasible by itself.

Assuming that the set $\mathcal{U}$ is a convex polytope, i.e. is expressed as the intersection of half-spaces, then the optimization problem is a QP, since the objective function (3.10) is a positive definite quadratic function, and the CLF constraint (3.11) and actuation constraints are affine in the optimization variables.

The fact that the optimization problem is a QP is advantageous, since this is second-easiest class of optimization problems to solve, and as such there are several algorithms that can solve QPs in polynomial time.

The resulting control input at time t, $\mathbf{u}(t) = \mathbf{k}(\mathbf{x}(t))$, is optimal in a greedy sense, since it only considers the current time state to determine the control input, instead of an horizon of future states.

3.2 Control Barrier Functions

3.2.1 Safety and Positive Invariance

Informally, safety can be defined as being able to avoid dangerous situations. Using this as a working definition, we can express the notion of safety for a state space model of a dynamical system. Let us consider the autonomous system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) \; , \tag{3.13}$$

where $\mathbf{x}(t) \in \mathcal{X} \subset \mathbb{R}^n$. It is assumed that the IVP resulting from any initial state $\mathbf{x}(0) \in \mathcal{X}$ and the above system of Ordinary Differential Equations (ODEs) has a unique solution defined for all $t \geq 0$. Translating "avoiding dangerous situations" from our working definition to our model, means that there is a set of states $\mathcal{C} \subset \mathcal{X}$ which we consider to be "safe" and a set $\overline{\mathcal{C}}$ which we consider to be "unsafe". A system can thus be categorized as safe, if $\mathbf{x}(t) \in \mathcal{C}, \forall t \geq 0$, i.e. if the set $\mathcal{C}$ is positive invariant.

Definition 3.2.1 (Positive invariance [1]). The set $\mathcal{C} \subset \mathcal{X}$ is said to be positive invariant, with respect to the system (3.13), if the solution to the IVP with $\mathbf{x}(0)$ is defined for $t \geq 0$ and that $\mathbf{x}(t) \in \mathcal{C}, \; \forall t \geq 0$.

Sufficient conditions for the invariance of sets were first given by Mitio Nagumo in 1942. A modern version of Nagumo's theorem from [1] will be given, but first we need to present a definition of tangent cone.

Definition 3.2.2 (Bouligand's tangent cone [1]). Given a closed set $\mathcal{C} \subset \mathbb{R}^n$, the cone tangent to $\mathcal{C}$ at $\mathbf{x} \in \mathcal{C}$ is given by

$$\mathcal{T}_{\mathcal{C}}(\mathbf{x}) = \left\{ \mathbf{z} \in \mathbb{R}^n : \lim_{\tau \to 0} \; \inf \; \frac{d(\mathbf{x} + \tau \mathbf{z}, \mathcal{C})}{\tau} = 0 \right\}, \tag{3.14}$$

where

$$d(\mathbf{x}, \mathcal{C}) = \inf_{c \in \mathcal{C}} \|\mathbf{x} - \mathbf{c}\|,$$

and $\lim \; \inf$ is the limit inferior.

Note that, for $\mathbf{x} \in \mathrm{int}(\mathcal{C})$, $\mathcal{T}_{\mathcal{C}}(\mathbf{x}) = \mathbb{R}^n$ and if $\mathbf{x} \notin \mathcal{C}$, $\mathcal{T}_{\mathcal{C}}(\mathbf{x}) = \emptyset$, thus $\mathcal{T}_{\mathcal{C}}(\mathbf{x})$ is only non-trivial when $\mathbf{x} \in \partial\mathcal{C}$.

Theorem 3.2.1 (Nagumo's theorem [1]). *Consider the system (3.13) and assume that for each initial condition in an open set $\mathbf{x}(0) \in \mathcal{O}$ it admits a unique solution defined $\forall t \geq 0$. Let $\mathcal{C} \subset \mathcal{X}$ be a closed set. Then, $\mathcal{C}$ is positive (or forward) invariant if and only if the "velocity" vector $\dot{\mathbf{x}}$, satisfies the Nagumo condition (A.K.A sub-tangentiality condition), i.e.*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \in \mathcal{T}_{\mathcal{C}}(\mathbf{x}), \forall \mathbf{x} \in \partial C \tag{3.15}$$

Nagumo's condition can be geometrically interpreted as saying that for every point on the boundary of the set, the corresponding "velocity" vector is tangent or pointing to the inside of the set. An illustration of this is given in Figure 3.1.

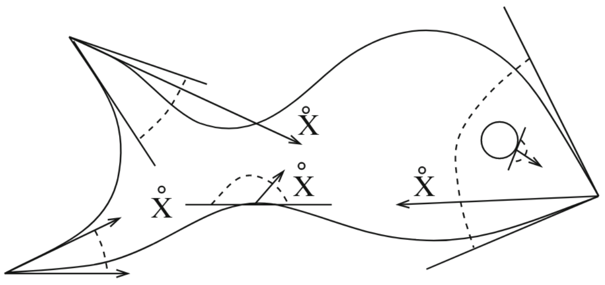


Figure 3.1: Nagumo's condition applied to a fish shaped set [1].

3.2.2 Control Barrier Functions

We are now ready to tackle the challenge of creating a feedback control law $\mathbf{k} : \mathcal{X} \to \mathcal{U}$ that can guarantee safety for an affine dynamical system of the form (3.1). To do so, we assume that the safe set $\mathcal{C}$ can be described as the superlevel set of a continuously differentiable function $h : \mathcal{C} \to \mathbb{R}$, i.e.

$$\mathcal{C} = \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) \geq 0\} \tag{3.16}$$

$$\partial\mathcal{C} = \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) = 0\} \tag{3.17}$$

$$\mathrm{int}(\mathcal{C}) = \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) > 0\} \tag{3.18}$$

3.2.2.A Intuition for the CBF

Like Section 3.1.1, before presenting the formal definition of the CBF, we will build an intuitive conceptual understanding by deducing it from first principles. We begin by assuming a priori that there exists a control law $\mathbf{k}$ that renders the safe set $\mathcal{C}$, described in (3.16), positive invariant, and we will see what conditions it must verify.

Since the function h is continuously differentiable, the tangent cone at any point $\mathbf{x} \in \partial\mathcal{C}$ is a halfspace that is tangent to $\partial\mathcal{C}$ (as can be seen in the smooth section of the belly of the fish shaped set in Figure 3.1). The boundary of the safe set ∂C, is a contour line of h, and it is a known result of multivariable calculus that the gradient of a function is always perpendicular to its contour lines. Moreover, it is also known that the gradient points in the direction that causes h to increase in value, so the gradient is pointing to the inside of $\mathcal{C}$.

Hence, Nagumo's condition for a set of the kind described in (3.16) is equivalent to having the "velocity" vector $\dot{\mathbf{x}}$ make an angle of at most $\pi/2$ radians with the gradient of h or, equivalently in terms of the Euclidean inner product,

$$\nabla h^\top(\mathbf{x})\dot{\mathbf{x}} \geq 0 \Leftrightarrow \nabla h^\top(\mathbf{x})\mathbf{f}(\mathbf{x}) + \nabla h^\top \mathbf{G}(\mathbf{x})\mathbf{k}(\mathbf{x}) \geq 0 \Leftrightarrow L_{\mathbf{f}}h(\mathbf{x}) + L_{\mathbf{G}}h(\mathbf{x})\mathbf{k}(\mathbf{x}) \geq 0,\ \forall \mathbf{x} \in \partial\mathcal{C}\ . \tag{3.19}$$

As it stands, this condition only activates when the current state is in $\partial\mathcal{C}$. This is undesirable for two reasons: firstly, this allows the state to get close to the unsafe set $\overline{\mathcal{C}}$, which is undesired as there is no safety margin in case there are disturbances and uncertainties in the model; secondly, for practical implementations, the finite precision of computers means that we will never actually measure $\mathbf{x}$ on being on $\partial\mathcal{C}$ (i.e. $\partial\mathcal{C}$ is a zero measure set in $\mathcal{X}$ and as such given any $\mathbf{x} \in \mathcal{X}$ the probability that $\mathbf{x} \in \partial\mathcal{C}$ is zero), so the condition will never actually be active. We need to extend the condition (3.19) to the whole of set $\mathcal{C}$.

Consider then the following, we allow $\nabla h^\top(\mathbf{x})\dot{\mathbf{x}} < 0$, i.e. for the value of $h(\mathbf{x})$ to decrease, so long as $h(\mathbf{x}) > 0$, i.e. so long as $\mathbf{x} \in \operatorname{int}(\mathcal{C})$, but also we need to guarantee that $\nabla h^\top(\mathbf{x})\dot{\mathbf{x}} = 0$ if $\mathbf{x} \in \partial\mathcal{C}$. Not strictly necessary for positive invariance of the safe set, but a feature we would wish for the safety of the system is that we would also like that, the closer we are to the boundary of the safe set, the slower $h(\mathbf{x})$ is allowed to decrease, i.e $\nabla h^\top(\mathbf{x})\dot{\mathbf{x}} \leq \nabla h^\top(\mathbf{y})\dot{\mathbf{y}},\ \forall \mathbf{x}, \mathbf{y} \in \mathcal{C} : d(\mathbf{x}, \overline{\mathcal{C}}) \geq d(\mathbf{y}, \overline{\mathcal{C}})$ where

$$d(\mathbf{x}, \mathcal{S}) = \inf_{\mathbf{y} \in \mathcal{S}} d(\mathbf{x}, \mathbf{y})\ ,$$

with d some distance function.

Usually, though not strictly necessary, $h(\mathbf{x})$ will be a function that encodes some distance measure between the state and the unsafe set, such as the norm squared, so a straightforward condition that fulfills the previously specified requirements is

$$L_{\mathbf{f}}h(x) + L_{\mathbf{G}}h(\mathbf{x})\mathbf{k}(\mathbf{x}) \geq -\alpha h(\mathbf{x}),\ \forall \mathbf{x} \in \mathcal{C}\ , \tag{3.20}$$

with $\alpha \in \mathbb{R}^+$. If $h(\mathbf{x})$ measures the distance to the unsafe set, the parameter α defines how close we allow the state to be near the unsafe set, since for the same value $h(\mathbf{x})$, a high value of α results in the function $h(\mathbf{x})$ being allowed to decrease, and thus get closer to $\overline{\mathcal{C}}$, more than for low a value of α.

Of course, much like our analysis for the CLF in Section 3.1.1, there is nothing particularly special about our choice for the right-hand side of inequality (3.20). Generically, we have

$$L_{\mathbf{f}}h(\mathbf{x}) + L_{\mathbf{G}}h(\mathbf{x})\mathbf{k}(\mathbf{x}) \geq -\alpha(h(\mathbf{x})),\ \forall \mathbf{x} \in \mathcal{C}\ , \tag{3.21}$$

with α a class $\mathcal{K}_\infty$ function.

3.2.3 Definition and Applications of the CBF

Definition 3.2.3 (Control Barrier Function [37]). Let $\mathcal{C}$ be the set given by (3.16) where h is a continuously differentiable function. The function, h, is a control barrier function for system (3.1) if there exists an extended class $\mathcal{K}_\infty$ function $\alpha : \mathbb{R} \to \mathbb{R}$, such that

$$\sup_{u\in\mathcal{U}} \{L_{\mathbf{f}}h(\mathbf{x}) + L_{\mathbf{G}}h(\mathbf{x})\mathbf{u}\} \geq -\alpha(h(\mathbf{x})),\ \forall \mathbf{x} \in \mathcal{C}\ . \tag{3.22}$$

Like Definition 3.1.1, Definition 3.2.3 means that at every state $\mathbf{x} \in \mathcal{X}$, there exists a set of control inputs such that

$$K_{CBF}(\mathbf{x}) = \left\{\mathbf{u} \in \mathcal{U} : L_{\mathbf{f}}h(\mathbf{x}) + L_{\mathbf{G}}h(\mathbf{x})\mathbf{u} + \alpha(h(\mathbf{x})) \geq 0\right\} \neq \emptyset, \forall \mathbf{x} \in \mathcal{C}\ . \tag{3.23}$$

Thus, we can create a control law $\mathbf{u}(t) = \mathbf{k}(\mathbf{x}(t))$ where $\mathbf{k} : \mathbf{x} \in \mathcal{X} \mapsto \mathbf{u} \in K_{CBF}(\mathbf{x})$ that renders the safe set $\mathcal{C}$ positive invariant.

3.2.3.A CBF-QP

Since CBFs are expressed as inequality constraints on the dynamical system, they can be incorporated into real-time optimization controllers in a number of ways. For example, if we have an already developed feedback control law $\mathbf{k}(\mathbf{x})$, which has no safety guarantees, for a given commanded input, we can find a provably safe approximation $\mathbf{k}_s(x) \in K_{CBF}(x)$, by solving:

$$\min_{\mathbf{u}\in\mathbb{R}^m} \quad \|\mathbf{u} - \mathbf{k}(\mathbf{x})\|^2 \tag{3.24}$$

$$s.t. : \quad L_{\mathbf{f}}h(\mathbf{x}) + L_{\mathbf{G}}h(\mathbf{x})\mathbf{u} \geq -\alpha(\mathbf{h}(\mathbf{x})) \tag{3.25}$$

$$\mathbf{u} \in \mathcal{U} \tag{3.26}$$

3.2.3.B CLF-CBF-QP

The optimization problems in Sections 3.1.2.A and 3.2.3.A can be formulated as a single optimization problem called a CLF-CBF-QP [18] [19].

$$\min_{\mathbf{u}\in\mathbb{R}^m,\delta\in\mathbb{R}} \quad \|\mathbf{u}\|^2 + \rho\delta^2 \tag{3.27}$$

$$s.t.: \quad L_{\mathbf{f}}V(\mathbf{x}) + L_{\mathbf{G}}V(\mathbf{x})\mathbf{u} + \delta \leq -\gamma(V(\mathbf{x})) \tag{3.28}$$

$$L_{\mathbf{f}}h(\mathbf{x}) + L_{\mathbf{G}}h(\mathbf{x})u \geq -\alpha(h(\mathbf{x})) \tag{3.29}$$

$$\mathbf{u} \in \mathcal{U} \tag{3.30}$$

3.2.4 Drawbacks

The CLF-CBF-QP, as well as the min-norm CLF-QP and CBF-QP in Sections 3.1.2.A and 3.2.3.A, respectively, have some drawbacks.

3.2.4.A Higher Order Degree functions

For the CLF-CBF-QP in (3.27) to make sense, we are assuming that $L_{\mathbf{G}}V(\mathbf{x}) \neq 0,\ \forall \mathbf{x} \in \mathcal{X} \setminus \{\mathbf{x}^*\}$ and $L_{\mathbf{G}}h(\mathbf{x}) \neq 0,\ \forall \mathbf{x} \in \mathcal{X} \cap \mathcal{C}$. If the Lie derivatives are zero in the aforementioned sets, then the optimization problem might have a trivial solution or not be feasible. The underlying assumption we are making is that the systems composed of the state equation, and outputs $V(\mathbf{x})$ and $h(\mathbf{x})$, have a relative degree of 1.

Definition 3.2.4 (Relative degree [36]). Consider the system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t)) + \mathbf{G}(\mathbf{x}(t))\mathbf{u}(t) \tag{3.31}$$

$$y(t) = h(\mathbf{x}(t)) \tag{3.32}$$

Where $\mathbf{f}$,$\mathbf{G}$ and h are sufficiently smooth in some domain $D \subset \mathbb{R}^n$ (i.e. we can differentiate them as many times as we want in domain D). This system is said to have relative degree 1 if $L_{\mathbf{G}}h(\mathbf{x}) \neq 0$ in a region $D_0 \subset D$, for all $\mathbf{x} \in D_0$, and relative degree ρ, $\rho \geq 2$ in a region $D_0 \subset D$ if

$$L_{\mathbf{G}}L_{\mathbf{f}}^{i-1}h(\mathbf{x}) = 0, \text{ for } i = 1, \dots, \rho - 1 \qquad \text{and} \qquad L_{\mathbf{G}}L_{\mathbf{f}}^{i-1}h(\mathbf{x}) \neq 0, \text{for } i = \rho \tag{3.33}$$

for all $\mathbf{x} \in D_0$, where

$$L_{\mathbf{f}}^{(0)}h(\mathbf{x}) = h(\mathbf{x}), \tag{3.34}$$

$$L_{\mathbf{f}}^{(i)}h(\mathbf{x}) = \frac{\partial(L_{\mathbf{f}}^{(i-1)}h)}{\partial \mathbf{x}}h(\mathbf{x}), \tag{3.35}$$

$$L_{\mathbf{G}}L_{\mathbf{f}}^{i}h(\mathbf{x}) = \frac{\partial(L_{\mathbf{f}}^{(i)}h)}{\partial \mathbf{x}}\mathbf{G}(\mathbf{x}). \tag{3.36}$$

For double integrator type dynamics, the system with output $y = V(\mathbf{x})$ or $y = h(\mathbf{x})$, where $V(\mathbf{x})$ and $h(\mathbf{x})$ are a CLF and CBF, respectively, that depend only on the configuration (i.e. position and/or attitude) of the spacecraft, is not of relative degree 1 for $D = \mathcal{X} \cap \mathcal{C}$, so there are states for which the controller might not stabilize and/or be safe.

Several proposals exist to solve this problem. Taylor et al. [38] propose the use of backstepping, however the implementation complexity of this approach increases immensely with the relative degree of the system. In [39], CBFs are extended to relative degree 2 and its method generalized in [40] for higher relative degrees, in a way that is simpler to implement compared to [38]. However the method of Higher-Order CBFs proposed by Wei Xiao and Calin Belta in [41] will be the one that we will use, as it is more general than the methods proposed in [39] [40] and easier to implement than the backstepping method of [38]. Moreover, while [41] is only concerned with solving the problem of generalizing CBFs for system with high relative degree, the method proposed is also easily extended to work with CLFs as well, which we will show in Section 3.3.

3.2.4.B Undesirable Equilibrium Points

According to Reis et al. [37], the CLF-CBF-QP might have undesirable asymptotically stable equilibria. The same paper presents a fix for the existence of these undesirable equilibrium points, by adding another constraint to the optimization problem which avoids the co-linearity of the vectors $\mathbf{f}(\mathbf{x})$, $\mathbf{G}(\mathbf{x})L_{\mathbf{f}}V(\mathbf{x})$, $\mathbf{G}(\mathbf{x})L_{\mathbf{G}}V(\mathbf{x})^{\top}$ and $\mathbf{G}(\mathbf{x})L_{\mathbf{G}}h(\mathbf{x})^{\top}$, which is a necessary condition for the existence of equilibrium points on the boundary of the safe set, and the use a rotating, non-radial Lyapunov function to remove the alignment between these vectors.

3.3 High-Order Control Barrier/Lyapunov Functions

In this section, we explain the method of High-Order CBFs presented in [41].

Consider the control affine system shown in (3.1) and assume that it has relative degree 2 for its CBFs, $h(\mathbf{x})$, which describes the safe set

$$\mathcal{C}_0 = \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) \geq 0\}.$$

In such a case, the dependency on the control input will only appear in the second time derivative

$$\dot{h}(\mathbf{x}) = L_{\mathbf{f}}h(\mathbf{x}) \tag{3.37}$$

$$\ddot{h}(\mathbf{x}) = L_{\mathbf{f}}^2 h(\mathbf{x}) + L_{\mathbf{G}}L_{\mathbf{f}}h(\mathbf{x})\mathbf{u} \tag{3.38}$$

in which case, the condition that enforces the positive invariance of $\mathcal{C}_0$ will have no dependency on $\mathbf{u}$

$$\dot{h}(\mathbf{x}) \geq -\alpha_0(h(\mathbf{x})) \Leftrightarrow +L_{\mathbf{f}}h(\mathbf{x}) + \alpha_0(h(\mathbf{x})) \geq 0, \tag{3.39}$$

where α_0 is a class $\mathcal{K}_\infty$ function. The solution proposed in [41] is that if one wants the condition in (3.39) to be enforced, and hence for $\mathcal{C}_0$ to be positive invariant, one simply defines a new CBF

$$\psi(\mathbf{x}) = \dot{h}(\mathbf{x}) + \alpha_0(h(\mathbf{x}))$$

called the HOCBF, which enforces the positive invariance of the set

$$\mathcal{C}_1 = \{\mathbf{x} \in \mathcal{X} : \psi(\mathbf{x}) \geq 0\}.$$

through the condition

$$\dot{\psi}(\mathbf{x}) \geq -\alpha_1(\psi(\mathbf{x})) \tag{3.40}$$

where α_1 is a class $\mathcal{K}_\infty$ function and

$$\dot{\psi}(\mathbf{x}) = \ddot{h}(\mathbf{x}) + \frac{\partial \alpha_0}{\partial x}_{|x=h(\mathbf{x})} \dot{h}(\mathbf{x})$$

so now the constraint one uses in the optimization problem is

$$L_{\mathbf{f}}^2 h(\mathbf{x}) + L_{\mathbf{G}} L_{\mathbf{f}} h(\mathbf{x})\mathbf{u} + \frac{\partial \alpha_0}{\partial h}_{|h(\mathbf{x})} L_{\mathbf{f}} h(\mathbf{x}) \geq -\alpha_1\big(L_{\mathbf{f}} h(\mathbf{x}) + \alpha_0(h(\mathbf{x}))\big) \tag{3.41}$$

which is control affine. Using the (3.41) as a constraint in our CLF-CBF-QP enforces the positive invariance of set $\mathcal{C}_1$, which in turn enforces the positive invariance of set $\mathcal{C}_0$.

If the system is of relative degree n, there is an obvious generalization. Simply use the HOCBF defined recursively as

$$\psi_i(\mathbf{x}) = \dot{\psi}_{i-1}(\mathbf{x}) + \alpha_{i-1}\big(\psi_{i-1}(\mathbf{x})\big),\ i = n, ..., 1$$

where $\psi_0 = h$ is the CBF, and use

$$\dot{\psi}_n(\mathbf{x}) \geq -\alpha_n\big(\psi_n(\mathbf{x})\big)$$

as the constraint in the CLF-CBF-QP, where the functions α_i, $i = 0, ..., n$ are class $\mathcal{K}_\infty$.

Although [41] does not mention it, this method is readily applied to CLFs as well, since for a given CLF, V, if one wants to enforce the CLF condition

$$\phi_0(\mathbf{x}) = -\dot{V}(\mathbf{x}) - \lambda_0\big(V(\mathbf{x})\big) \geq 0$$

which is equivalent to enforcing the positive invariance of the set

$$\mathcal{V}_0 = \{\mathbf{x} \in \mathcal{X} : \phi_0 \geq 0\}.$$

If the system with state equation (3.1) and output equation $y(t) = V(\mathbf{x}(t))$ if of relative degree n, then the system with output equation $y(t) = \phi_0(\mathbf{x}(t)$ is of relative degree $n-1$. One then uses the HOCBF defined recursively as

$$\phi_i(\mathbf{x}) = \dot{\phi}_{i-1}(\mathbf{x}) + \lambda_{i-1}\big(\phi_{-1}(\mathbf{x})\big),\ i = n-1, ..., 1$$

and the constraint

$$\dot{\phi}_{n-1}(\mathbf{x}) \geq -\lambda_{n-1}\big(\phi_{n-1}(\mathbf{x})\big)$$

with λ_i, $i = 0, ..., n-1$ class $\mathcal{K}_\infty$ functions, to enforce the CLF constraint.

4

Constrained Attitude Control Problem

Contents

4.1 Problem Formulation

We present the constrained attitude guidance problem as follows. Consider we wish to reorient a spacecraft from its current attitude $\mathbf{q}_0 \in \mathcal{S}^3$ to a final attitude $\mathbf{q}_f \in \mathcal{S}^3$. The trajectory of the attitude, $\mathbf{q} : \mathbb{R}_0^+ \to \mathcal{S}^3$, has boundary conditions $\mathbf{q}(0) = \mathbf{q}_0$ and $\mathbf{q}(t_f) = \mathbf{q}_f \otimes \varepsilon$, where $\varepsilon \in \mathcal{S}^3$ is some acceptable error. The trajectory must be such that a pointing vector ${}^{\mathcal{N}}\mathbf{v}(t) \in \mathcal{S}^2$ in the inertial reference frame, given by ${}^{\mathcal{N}}\mathbf{v}(t) = \mathbf{R}(\mathbf{q}(t))\,{}^{\mathcal{B}}\mathbf{v}$, at no point in time has an angle less that θ_{out} with some pointing vector ${}^{\mathcal{N}}\mathbf{s} \in \mathcal{S}^2$. Since ${}^{\mathcal{N}}\mathbf{v}$ and ${}^{\mathcal{N}}\mathbf{s}$ are both unit norm, this can be expressed as

$$
{}^{\mathcal{N}}\mathbf{v} \cdot {}^{\mathcal{N}}\mathbf{s} \leq \cos(\theta_{out}) \Leftrightarrow {}^{\mathcal{N}}\mathbf{s}^\top(t)\mathbf{R}(\mathbf{q}(t))\,{}^{\mathcal{B}}\mathbf{v}(t) \leq \cos(\theta_{out}),\ \forall t \in [0, \infty)\ . \tag{4.1}
$$

Likewise, we wish that a pointing vector ${}^{\mathcal{N}}\mathbf{d}(t) \in \mathcal{S}^2$ must at no point in time have an angle greater than θ_{in} with another pointing vector ${}^{\mathcal{N}}\mathbf{e} \in \mathcal{S}^2$, i.e.

$$
{}^{\mathcal{N}}\mathbf{d} \cdot {}^{\mathcal{N}}\mathbf{e} \geq \cos(\theta_{in}) \Leftrightarrow {}^{\mathcal{N}}\mathbf{e}^\top(t)\mathbf{R}(\mathbf{q}(t))\,{}^{\mathcal{B}}\mathbf{v}(t) \geq \cos(\theta_{in}),\ \forall t \in [0, \infty)\ . \tag{4.2}
$$

We consider the mission requirements in order of priority to be the following: the trajectory must be safe, in the sense that constraints (4.1) and (4.2) are not violated; the maneuver must be completed by the specified time t_f; the final attitude $\mathbf{q}(t_f)$ must have an error $\varepsilon = \arccos(\mathbf{q}_f \cdot \mathbf{q}(t_f))$ no greater than $0.2°$, so that it is within a region where a linear high-pointing accuracy controller can take over. The system state is composed of the attitude quaternion $\mathbf{q}$ and the angular velocity in the body fixed frame ${}^{\mathcal{B}}\boldsymbol{\omega}$, we thus get the state equation for our system by combining the quaternion kinematics equation (2.28) and the Euler equation for rigid body motion (2.34),

$$
\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{q}} \\ {}^{\mathcal{B}}\dot{\boldsymbol{\omega}} \end{bmatrix} = \underbrace{\begin{bmatrix} -\frac{1}{2}\begin{bmatrix} -\mathbf{q}_v^\top \\ q_s\mathbf{I} + [\mathbf{q}_v]_\times \end{bmatrix} {}^{\mathcal{B}}\boldsymbol{\omega} \\ {}^{\mathcal{B}}\mathbf{J}^{-1}({}^{\mathcal{B}}\mathbf{J}\,{}^{\mathcal{B}}\boldsymbol{\omega} \times {}^{\mathcal{B}}\boldsymbol{\omega}) \end{bmatrix}}_{\mathbf{f}(\mathbf{x})} + \underbrace{\begin{bmatrix} \mathbf{0} \\ {}^{\mathcal{B}}\mathbf{J}^{-1} \end{bmatrix}}_{\mathbf{G}(\mathbf{x})} {}^{\mathcal{B}}\boldsymbol{\tau}. \tag{4.3}
$$

The vector ${}^{\mathcal{B}}\boldsymbol{\tau}$ of coordinates of the total torque applied to the rigid body w.r.t. the body-fixed frame are the system's input. We consider that the state starts with zero angular velocity and we wish that the final angular velocity is zero. Thus, the initial state and desired final states are given by $\mathbf{x}_0 = (\mathbf{q}_0, \mathbf{0})$ and $\mathbf{x}_f = (\mathbf{q}_f, \mathbf{0})$, respectively. Additionally, we impose limits on the torque and angular velocity, as $\|{}^{\mathcal{B}}\boldsymbol{\tau}\|_\infty \leq \tau_{max} = \max\{|{}^{\mathcal{B}}\tau_i| : i = 1, 2, 3\} \leq \tau_{max}$ and $\|{}^{\mathcal{B}}\boldsymbol{\omega}\|_\infty \leq \omega_{max} = \max\{|{}^{\mathcal{B}}\omega_i| : i = 1, 2, 3\} \leq \omega_{max}$, with $\tau_{max}, \omega_{max} \in \mathbb{R}^+$. We thus can formulate the constrained attitude problem as the optimal control problem

$$\min_{{}^{\mathcal{B}}\boldsymbol{\tau}(t)\in\mathbb{R}^3, \mathbf{x}\in\mathcal{S}^3\times\mathbb{R}^3} \quad \Phi\big(\mathbf{x}(t_f)\big) + \int_0^{t_f} \|\,{}^{\mathcal{B}}\boldsymbol{\tau}(t)\|^2 dt \tag{4.4}$$

$$s.t. \quad \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{G}(\mathbf{x}(t))\,{}^{\mathcal{B}}\boldsymbol{\tau}(t) \quad t \in [0, t_f] \tag{4.5}$$

$$\mathbf{x}(0) = \mathbf{x}_0 \tag{4.6}$$

$${}^{\mathcal{N}}\mathbf{s}^\top(t)\mathbf{R}(\mathbf{q}(t))\,{}^{\mathcal{B}}\mathbf{v}(t) \leq \cos(\theta_{out}) \quad t \in [0, t_f] \tag{4.7}$$

$${}^{\mathcal{N}}\mathbf{e}^\top(t)\mathbf{R}(\mathbf{q}(t))\,{}^{\mathcal{B}}\mathbf{v}(t) \geq \cos(\theta_{in}) \quad t \in [0, t_f] \tag{4.8}$$

$$\|\,{}^{\mathcal{B}}\boldsymbol{\tau}(t)\|_\infty \leq \tau_{max} \quad t \in [0, t_f] \tag{4.9}$$

$$\|\,{}^{\mathcal{B}}\boldsymbol{\omega}(t)\|_\infty \leq \omega_{max} \quad t \in [0, t_f] \tag{4.10}$$

$$\tag{4.11}$$

where $\Phi\big(\mathbf{x}(t_f)\big)$ is a terminal cost defined as $\Phi\big(\mathbf{x}(t_f)\big) = \gamma \cdot \big(\max\{0, \mathbf{q}_f \cdot \mathbf{q}(t_f) - \varepsilon\}\big)^2 + \eta \cdot \|\,{}^{\mathcal{B}}\boldsymbol{\omega}(t_f)\|^2$ with appropriate penalty weights $\gamma \in \mathbb{R}^+$ and $\eta \in \mathbb{R}^+$ and where ε is the cosine of the maximum angle we wish between the final quaternion $\mathbf{q}(t_f)$ and target quaternion $\mathbf{q}_f$.

4.2 State-of-the-Art

In this section, we present a state-of-the-art method used to perform constrained attitude control and which we will use as a baseline to compare our proposed solution to. The method is based on trajectory optimization using SCvx and path following using MPC. We choose to highlight this method as it was one of the proposed solutions to perform constrained attitude control in the Comet Interceptor mission and it is the method used in the SOC-i CubeSat [12] [11].

4.2.1 Trajectory optimization with SCvx

As opposed to the CLF and CBF based methods, discussed in Section 3.1 and 3.22, which determines the input to be applied online, i.e. as the spacecraft is performing the maneuver, SCvx is a direct collocation, trajectory optimization algorithm. It is a trajectory optimization algorithmn because we first generate a trajectory offline, by minimizing some cost function subjected to constraints, given by the dynamics and mission requirements (such as safety). Only after the full trajectory has been obtained does the vehicle perform the maneuver, using a path following controller such as a MPC. Direct collocation refers to the fact that we are directly optimizing the placement of N nodes of the trajectory, and obtain the rest of the trajectory between the nodes by interpolation.

Trajectory optimization is typically a nonconvex problem. To address this, SCvx linearizes the dynamics and nonconvex constraints around a reference trajectory, thereby transforming the problem into a convex subproblem. This convex subproblem is solved to generate the next reference trajectory, which, in turn, defines a new convex subproblem. This iterative process continues until convergence is achieved. This strategy of solving a sequence of convex subproblems categorizes SCvx within a family of algorithms known as Sequential Convex Programming (SCP) algorithms. Additionally, SCvx incorporates trust regions in the convex subproblem, which constrain the nodes to remain within a neighborhood of the reference trajectory where the linearization is valid. This approach places SCvx within a family of algorithms called trust-region methods.

While the SCvx algorithm guarantees convergence to a trajectory, it does not guarantee that the resulting trajectory is dynamically feasible or respects the nonconvex constraints [42]. This limitation arises because SCvx introduces virtual control and virtual buffer terms as optimization variables to handle infeasibility issues resulting from the linearization of the dynamics and nonconvex constraints. The cost function optimized in the convex subproblem comprises the original cost and an additional term that penalizes the use of these virtual terms. Consequently, this augmented cost function may converge to a local minimum where the virtual terms are non-zero.

Moreover, even if the resulting trajectory is dynamically feasible and satisfies the nonconvex constraints at the optimization nodes, there is no guarantee that these constraints will be met along the entire interpolated trajectory.

A detailed explanation of how SCvx works can be found in [43]. For our purposes, we will use the SCPToolbox, a Julia module developed by Malyuta et al. [2], which includes SCvx as one of its SCP algorithms. This toolbox allows us to solve continuous-time optimal control problems of the form:

$$\min_{\mathbf{x}(\xi)\in\mathbb{R}^n,\mathbf{u}(\xi)\in\mathbb{R}^m,\mathbf{p}\in\mathbb{R}^d} \quad \phi(\mathbf{x}(1)) + \int_0^1 \Gamma(\mathbf{x}(\xi),\mathbf{u}(\xi),\mathbf{p})d\xi \tag{4.12}$$

$$s.t.: \quad \frac{d\mathbf{x}}{d\xi}(\xi) = \mathbf{f}_\xi(\xi,\mathbf{x}(\xi),\mathbf{u}(\xi),\mathbf{p}) \tag{4.13}$$

$$(\mathbf{x}(\xi),\mathbf{p}) \in \mathcal{X}(\xi) \tag{4.14}$$

$$(\mathbf{u}(\xi),\mathbf{p}) \in \mathcal{U}(\xi) \tag{4.15}$$

$$\mathbf{s}(\xi,\mathbf{x}(\xi),\mathbf{u}(\xi),\mathbf{p}) \leq 0 \tag{4.16}$$

$$\mathbf{g}_{ic}(\mathbf{x}(0),\mathbf{p}) = 0 \tag{4.17}$$

$$\mathbf{g}_{tc}(\mathbf{x}(1),\mathbf{p}) = 0 \tag{4.18}$$

Here, $\mathbf{p} \in \mathbb{R}^d$ is a vector of parameters that can be used, for instance, to allow a variable final time for the trajectory [2]. In order to have a direct comparison with the CLF-CBF-QP method, we will not utilize these parameters.

In this formulation, ϕ represents the terminal cost, Γ is the running cost function, and $\mathbf{f}_\xi$ encodes the dynamics. The sets $\mathcal{X}(\xi)$ and $\mathcal{U}(\xi)$ are convex sets specifying the convex state and control constraints, respectively. The function s specifies the nonconvex constraints, while $\mathbf{g}_{ic}$ and $\mathbf{g}_{tc}$ define the initial and terminal boundary conditions, respectively.

The toolbox parameterizes the state and control trajectories by $\xi \in [0,1]$. However, we can reparameterize using $t = \xi \cdot t_f$ to obtain the usual parameterization of the state and control trajectories by time t in the interval $t \in [0, t_f]$. With this reparameterization, the dynamics become

$$\frac{d\mathbf{x}}{d\xi}(\xi) = \frac{d\mathbf{x}}{dt}\big(t(\xi)\big)\frac{dt}{d\xi}(\xi) = \Big[\mathbf{f}\big(\mathbf{x}(t(\xi))\big) + \mathbf{G}\big(\mathbf{x}(t(\xi))\big)\mathbf{u}(t(\xi))\Big]t_f \tag{4.19}$$

$$\implies \mathbf{f}_\xi(\xi, \mathbf{x}, \mathbf{u}, \mathbf{p}) = \big[\mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u}\big]t_f. \tag{4.20}$$

We could use the function s to encode the attitude constraints as formulated in (4.1) and (4.2). However, forbidden attitude constraints are a rare type of nonconvex constraints where one can formulate equivalent convex constraints that describe the same feasible set. This process is known as lossless convexification [2].

In [12], it is demonstrated how to perform convex relaxation for quaternions representing rotation as a passive transformation. However, since we use quaternions to represent rotation as an active transformation, their expressions do not apply. We take this as a pedagogical opportunity to show how these expressions can be derived. Consider two unit vectors $\mathbf{v} \in \mathcal{S}^2$ and $\mathbf{r} \in \mathcal{S}^2$, their dot product is given by

$$\mathbf{r}\cdot\mathbf{v} = \cos(\theta) \Leftrightarrow {}^{\mathcal{N}}\mathbf{r}\cdot{}^{\mathcal{N}}\mathbf{v} = {}^{\mathcal{N}}\mathbf{r}\cdot\mathbf{R}(\mathbf{q})\,{}^{\mathcal{B}}\mathbf{v} = \cos(\theta) \underset{(2.19)}{\Leftrightarrow} \tag{4.21}$$

$${}^{\mathcal{N}}\mathbf{r}\big(\mathbf{I} - 2[\mathbf{q}_v]_\times^\top[\mathbf{q}_v]_\times + 2q_s[\mathbf{q}_v]_\times\big)\,{}^{\mathcal{B}}\mathbf{v} = \cos(\theta) \Leftrightarrow \tag{4.22}$$

$${}^{\mathcal{N}}\mathbf{r}^\top\,{}^{\mathcal{B}}\mathbf{v} + 2\big([\mathbf{q}_v]_\times\,{}^{\mathcal{N}}\mathbf{r}\big)^\top[{}^{\mathcal{B}}\mathbf{v}]_\times\mathbf{q}_v - 2q_s\,{}^{\mathcal{N}}\mathbf{r}^\top[{}^{\mathcal{B}}\mathbf{v}]_\times\mathbf{q}_v = \cos(\theta) \Leftrightarrow \tag{4.23}$$

$${}^{\mathcal{N}}\mathbf{r}^\top\,{}^{\mathcal{B}}\mathbf{v}\mathbf{q}^\top\mathbf{q} - 2\mathbf{q}^\top[{}^{\mathcal{N}}\mathbf{r}]^\top[{}^{\mathcal{B}}\mathbf{v}]_\times\mathbf{q}_v - 2q_s\,{}^{\mathcal{N}}\mathbf{r}^\top[{}^{\mathcal{B}}\mathbf{v}]_\times\mathbf{q}_v = \cos(\theta) \Leftrightarrow \tag{4.24}$$

$$\mathbf{q}^\top\underbrace{\left({}^{\mathcal{N}}\mathbf{r}^\top\,{}^{\mathcal{B}}\mathbf{v}\mathbf{I} + \begin{bmatrix}\mathbf{0} & -2\,{}^{\mathcal{N}}\mathbf{r}^\top[{}^{\mathcal{B}}\mathbf{v}]_\times \\ \mathbf{0} & 2[{}^{\mathcal{N}}\mathbf{r}]_\times[{}^{\mathcal{B}}\mathbf{v}]_\times\end{bmatrix}\right)}_{\mathbf{P}'({}^{\mathcal{N}}\mathbf{r},{}^{\mathcal{B}}\mathbf{v})}\mathbf{q} = \cos(\theta) \Leftrightarrow \tag{4.25}$$

$$\frac{1}{2}\mathbf{q}^\top\mathbf{P}'({}^{\mathcal{N}}\mathbf{r},{}^{\mathcal{B}}\mathbf{v})\mathbf{q} + \frac{1}{2}\mathbf{q}^\top\mathbf{P}'^\top({}^{\mathcal{N}}\mathbf{r},{}^{\mathcal{B}}\mathbf{v})\mathbf{q} = \cos(\theta) \Leftrightarrow \tag{4.26}$$

$$\mathbf{q}^\top\mathbf{P}({}^{\mathcal{N}}\mathbf{r},{}^{\mathcal{B}}\mathbf{v})\mathbf{q} = \cos(\theta). \tag{4.27}$$

Here, $\mathbf{P} : \mathcal{S}^2 \times \mathcal{S}^2 \to \{\mathbf{A} \in \mathbb{R}^{4\times4} : \mathbf{A}^\top = \mathbf{A}\}$ is given by

$$\mathbf{P}({}^{\mathcal{N}}\mathbf{r},{}^{\mathcal{B}}\mathbf{v}) = \begin{bmatrix}{}^{\mathcal{N}}\mathbf{r}^\top\,{}^{\mathcal{B}}\mathbf{v} & -\,{}^{\mathcal{N}}\mathbf{r}^\top[{}^{\mathcal{B}}\mathbf{v}]_\times \\ [{}^{\mathcal{B}}\mathbf{v}]_\times\,{}^{\mathcal{N}}\mathbf{r} & {}^{\mathcal{N}}\mathbf{r}^\top\,{}^{\mathcal{B}}\mathbf{v}\mathbf{I} + [{}^{\mathcal{N}}\mathbf{r}]_\times[{}^{\mathcal{B}}\mathbf{v}]_\times + [{}^{\mathcal{B}}\mathbf{v}]_\times[{}^{\mathcal{N}}\mathbf{r}]_\times\end{bmatrix},$$

which is a symmetric matrix, whose eigenvalues are between 1 and -1[1].

[1]This can be seen from the eigenvalue inequality which states that for $\|\mathbf{v}\| = 1$, $\lambda_{min} \leq \mathbf{v}^\top\mathbf{A}\mathbf{v} \leq \lambda_{max}$ where λ_{min} and λ_{max} are the minimum and maximum eigenvalues of $\mathbf{A}$.

Thus, we can write the keep out zone as the convex quadratic constraint

$$\mathbf{s}\cdot\mathbf{v} \leq \cos(\theta_{out}) \Leftrightarrow \mathbf{q}^\top\big(\mathbf{I}+\mathbf{P}({}^{\mathcal{N}}\mathbf{s},{}^{\mathcal{B}}\mathbf{v})\big)\mathbf{q} \leq 1+\cos(\theta_{out}), \tag{4.28}$$

and the keep in zone as the convex quadratic constraint

$$\mathbf{e}\cdot\mathbf{d} \geq \cos(\theta_{in}) \Leftrightarrow \mathbf{q}^\top\big(\mathbf{I}-\mathbf{P}({}^{\mathcal{N}}\mathbf{e},{}^{\mathcal{B}}\mathbf{d})\big)\mathbf{q} \leq 1-\cos(\theta_{in}). \tag{4.29}$$

Furthermore, as the function $\mathbf{P}$ yields a symmetric matrix, its eigenvectors are orthogonal [44]. Therefore, we can express $\mathbf{I}+\mathbf{P}({}^{\mathcal{N}}\mathbf{s},{}^{\mathcal{B}}\mathbf{v})$ as $\mathbf{M}^\top\mathbf{M}$ and $\mathbf{I}-\mathbf{P}({}^{\mathcal{N}}\mathbf{e},{}^{\mathcal{B}}\mathbf{d})$ as $\mathbf{N}^\top\mathbf{N}$, where $\mathbf{M}=\mathbf{D}^{\frac{1}{2}}\mathbf{V}^\top$. Here $\mathbf{D}^{\frac{1}{2}}$ represents the diagonal matrix, where the diagonal entries are the square root of the eigenvalues of $\mathbf{I}+\mathbf{P}({}^{\mathcal{N}}\mathbf{s},{}^{\mathcal{B}}\mathbf{v})$, and $\mathbf{V}$ is the matrix whose columns consist of the unit norm eigenvectors of $\mathbf{I}+\mathbf{P}({}^{\mathcal{N}}\mathbf{s},{}^{\mathcal{B}}\mathbf{v})$. The matrix $\mathbf{N}$ is defined in an analogous manner as $\mathbf{M}$, but with respect to matrix $\mathbf{I}-\mathbf{P}({}^{\mathcal{N}}\mathbf{e},{}^{\mathcal{B}}\mathbf{d})$.

With this, we define the convex state constraint set $\mathcal{X}(t)$ as

$$\mathcal{X}(t)=\Big\{(\mathbf{q}(t),\boldsymbol{\omega}(t))\in\mathbb{R}^4\times\mathbb{R}^3 : \|\mathbf{M}\mathbf{q}(t)\|\leq\sqrt{1+\cos(\theta_{out})}, \|\mathbf{N}\mathbf{q}(t)\|\leq\sqrt{1-\cos(\theta_{in})}, \|\boldsymbol{\omega}(t)\|_\infty\leq\omega_{\max}\Big\}$$

and the convex input constraint set $\mathcal{U}(t)$ as

$$\mathcal{U}(t)=\{\mathbf{u}(t)\in\mathbb{R}^3 : \|\mathbf{u}(t)\|_\infty\leq\tau_{max}\}.$$

The boundary condition functions $\mathbf{g}_{ic},\mathbf{g}_{tc} : \mathbb{R}^n\times\mathbb{R}^d\to\mathbb{R}^n$ are given by $\mathbf{g}_{ic}(\mathbf{x},\mathbf{p})=\mathbf{x}-\mathbf{x}_0$ and $\mathbf{g}_{tc}(\mathbf{x},\mathbf{p})=\mathbf{x}-\mathbf{x}_f$, where $\mathbf{x}_0=(\mathbf{q}_0,\mathbf{0})$ and $\mathbf{x}_f=(\mathbf{q}_f,\mathbf{0})$.

The running cost function $\Gamma : \mathbb{R}^n\times\mathbb{R}^m\times\mathbb{R}^d\to\mathbb{R}$ is given by $\Gamma(\mathbf{x},\mathbf{u},\mathbf{p})=\|\mathbf{u}\|^2$. The terminal cost function is the zero function, since the final state is the constant $\mathbf{x}_f$. Thus, any term that depends only on the final state does not affect the solution of the optimization problem and can be disregarded.

Note that we do not directly enforce that $\mathbf{q}(t)$ is unit norm in the optimization problem formulation, as it is implicitly enforced by the dynamics. With these definitions, the convex subproblem that the SCvx must solve is a Second Order Cone Program (SOCP) and so we can use ECOS as a solver.

After ECOS obtains the solution to the convex subproblem for the N specified nodes, SCvx obtains the continuous-time input by interpolating the discrete input values at the nodes, using a method that is user specified. We opt for the First-Order Hold (FOH) method for input interpolation, as it is the same interpolation method used in [12].

The continuous-time state trajectory solution is obtained by integrating the dynamics using the continuous-time input obtained through interpolation, between $t=0$ and $t=t_f$. If the continuous-time input is dynamically feasible, integrating it between $t_k=k/(N-1)\cdot t_f$ and $t_{k+1}=(k+1)/(N-1)\cdot t_f$, with initial condition given by the state at node k, $\mathbf{x}_k^d$, should yield a state that matches $\mathbf{x}_{k+1}^d$. If they do not match, there will be a non-zero *defect*, δ_k. The defect is a measure of dynamic feasibility, described in [2] and illustrated in Figure 4.1, where $\boldsymbol{\psi}(t_k, t, \mathbf{x}_k, \mathbf{u}, \mathbf{p})$ represents the flow map. The flow map is the function obtained by integrating the dynamics using input $\mathbf{u} : \mathbb{R} \to \mathbb{R}^m$ and parameters $\mathbf{p} : \mathbb{R}$, from t_k to t, with initial conditions $\mathbf{x}_k$.

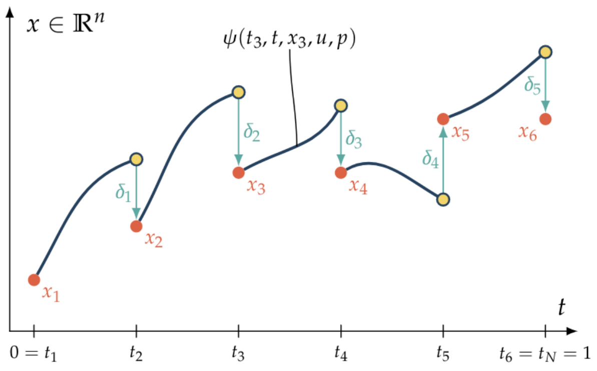


Figure 4.1: Illustration of the defect for a 1D system [2].

Depending on certain parameters, the continuous-time input and state are either accepted as a new reference or discarded, and the trust region is updated accordingly.

The parameters we used for the SCvx algorithm were the following:

$$
\begin{array}{cccc}
N = 25 & N_{sub} = 45 & \lambda = 10^2 & \varepsilon_{abs} = 10^{-5} \\
\rho_0 = 0 & \rho_1 = 0.2 & \rho_2 = 0.7 & q_{exit} = \infty \\
\beta_{sh} = 2 & \beta_{gr} = 2 & \eta_{init} = 1 & \varepsilon_{rel} = 10^{-4} \\
\eta_{lb} = 10^{-8} & \eta_{ub} = 10 & q_{tr} = \infty & feas\ tol = 10^{-3}
\end{array}
$$

Here, N represents the number of nodes used to discretize the continuous-time trajectories, N_{subs} denotes the number of nodes for Runge-Kutta Fourth-Order (RK4) integration between t_k and t_{k+1}, and λ signifies the weight for the term penalizing the use of virtual variables. The parameters ρ_0, ρ_1, ρ_2, β_{shrink}, β_{grow}, η_{init}, η_{ub}, η_{lb} pertain to the criteria for accepting a new reference trajectory and adjusting the trust region size defined by the norm q_{tr}, thereby influencing the convergence speed of the algorithm. Additionally, parameters ε_{abs}, ε_{rel}, q_{exit} and $feas\ tol$ are associated with the algorithm's stopping criteria.

For further details on these parameters, refer to [2] and the SCPToolbox documentation. The algorithm is allowed a maximum of 20 iterations before returning a solution. Furthermore, a callback function was incorporated into the RK4 integration to normalize the quaternion part of the state, preventing numerical errors from propagating.

The SCvx algorithm needs to be initialized with an initial guess for the reference trajectory, which does not need to be dynamically feasible, but does need to satisfy (4.17) and (4.18). The initial guess was given by performing spherical linear interpolation on the quaternion part of the state, from $\mathbf{q}_0$ to $\mathbf{q}_f$, and setting the angular velocity and torque to zero. Providing a better informed initial guess can result in faster convergence, however.

4.2.2 Path Following MPC Controller

Having obtained the continuous-time trajectories $\mathbf{x}^o : \mathbb{R}_0^+ \to \mathcal{S}^3 \times \mathbb{R}^3$ and ${}^{\mathcal{B}}\boldsymbol{\tau}^o : \mathbb{R}_0^+$ by using the SCvx algorithm, we need controller to follow them. With the reference trajectories, we can obtain the dynamics of the system linearized over the trajectory, which results in the Linear Time-Varying (LTV) system

$$\dot{\mathbf{x}}(t) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\,{}^{\mathcal{B}}\boldsymbol{\tau}(t) + \mathbf{z}(t),$$

where

$$\begin{aligned}
\mathbf{A}(t) &= \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\big(\mathbf{x}^o(t)\big) + \frac{\partial \mathbf{G}}{\partial \mathbf{x}}\big(\mathbf{x}^o(t)\big)\,{}^{\mathcal{B}}\boldsymbol{\tau}^o(t)\ ,\\
\mathbf{B}(t) &= \mathbf{G}\big(\mathbf{x}^o(t)\big)\ ,\\
\mathbf{z}(t) &= \mathbf{f}\big(\mathbf{x}^o(t)\big) + \mathbf{G}\big(\mathbf{x}^o(t)\big)\,{}^{\mathcal{B}}\boldsymbol{\tau}^o(t) - \mathbf{A}(t)\big(\mathbf{x}^o(t)\big) - \mathbf{B}(t)\,{}^{\mathcal{B}}\boldsymbol{\tau}^o(t).
\end{aligned}$$

Afterward, the LTV system is discretized over time intervals equal to our sampling time, Δt, employing the Zero-Order Hold (ZOH) method as detailed in [45]. This process yields a set of matrices $\{\mathbf{A}_k\}_{k=0}^T$, $\{\mathbf{B}_k\}_{k=0}^T$ and a set of vectors $\{\mathbf{z}_k\}_{k=0}^T$, where T represents the smallest positive integer satisfying $T\Delta t > t_f$, describing the discretized LTV system.

To track the trajectory, we employ a MPC, a real-time optimal controller that at each time instance k, solves the following optimization problem:

$$\min_{\substack{\mathbf{x}_{k|k},\mathbf{x}_{k|k+1},..,\mathbf{x}_{k|k+H}\in\mathbb{R}^7,\\ \boldsymbol{\tau}_{k|k},\boldsymbol{\tau}_{k|k+1},\dots,\boldsymbol{\tau}_{k|k+H-1}\in\mathbb{R}^3}} \sum_{i=1}^{H-1} \|\mathbf{x}_{k|k+i} - \mathbf{x}^o_{k+i}\|^2 + \lambda \sum_{i=0}^{H-1} \|\boldsymbol{\tau}_{k|k+i}\|^2 + \rho\|\mathbf{x}_{k|k+H} - \mathbf{x}^o_{k+H}\|^2 \tag{4.30}$$

$$s.t.:\quad \mathbf{x}_{k|k+1+i} = \mathbf{A}_{k+i}\mathbf{x}_{k|k+i} + \mathbf{B}_{k+i}\boldsymbol{\tau}_{k|k+i}, \qquad i = 0, ..., H-1 \tag{4.31}$$

$$\mathbf{x}_{k|k} = \mathbf{x}_k \tag{4.32}$$

$$\|\mathbf{M}\mathbf{q}_{k|k+i}\| \le \sqrt{1+\cos(\theta_{out})}, \qquad i = 0, ..., H \tag{4.33}$$

$$\|\mathbf{N}\mathbf{q}_{k|k+i}\| \le \sqrt{1-\cos(\theta_{in})}, \qquad i = 0, ..., H \tag{4.34}$$

$$\|\boldsymbol{\omega}_{k|k+i}\|_\infty \le \omega_{max}, \qquad i = 1, ..., H \tag{4.35}$$

$$\|\boldsymbol{\tau}_{k|k+i}\|_\infty \le \tau_{max}, \qquad i = 1, ..., H \tag{4.36}$$

Here, $\mathbf{x}_k^o = \mathbf{x}^o(k\Delta t)$ and $\mathbf{x}_k$ represents the current state of the spacecraft. The optimization variables

$\mathbf{x}_{k|k+i} = (\mathbf{q}_{k|k+i}, \boldsymbol{\omega}_{k|k+i})$ are the states that the MPC controller "predicts" the spacecraft will have at time instance $k+i$ if it follows the sequence of inputs given by the optimization variables $\boldsymbol{\tau}_{k|k+i}$, starting at k until $k+H-1$, by using the model described in (4.31). However, these predictions are only accurate if our model is $100\,\%$ correct, and they do not account for any unmodelled factors such as disturbances[2] or nonlinearities of the true dynamics not captured by the model in (4.31). For this reason, only the first input in the sequence, ${}^{\mathcal{B}}\boldsymbol{\tau}(k\Delta t) = \boldsymbol{\tau}_{k|k}$ is applied to the spacecraft. Subsequently, in the next time instance, the MPC recalculates a new sequence of inputs and predicted states to reflect the actual evolution of the state resulting from applying $\boldsymbol{\tau}_{k|k}$.

As mentioned earlier, the trajectory produced by SCvx only guarantees safety at the nodes and not along the entire interpolated trajectory. Therefore, the second order cone constraints for the keep in and keep out zones are included in the optimization problem that the MPC solves. This allows the MPC to correct any possible infringements of the forbidden attitude zones in the interpolated trajectory. As a result, the optimization problem tackled by the MPC becomes a SOCP, presenting greater complexity compared to the QP used in a CLF-CBF-QP, described in Section 3.2.3.B.

4.3 CLF-CBF-QP Attitude Controller

Our proposed solution to the constrained attitude guidance problem consists of using the CLF-CBF-QP controller presented in Section 3.2.3.B to guide the spacecraft to $\mathbf{q}_f$ with a final angular velocity of $\mathbf{0}$.

We now wish to codify the attitude constraints (4.1) and (4.2) as a CBF. First, we consider individual CBFs for each of the constraints

$$h_{out}(\mathbf{x}) = \cos(\theta_{out}) - {}^{\mathcal{N}}\mathbf{s}^\top \mathbf{R}(\mathbf{q})\, {}^{\mathcal{B}}\mathbf{v} \;, \tag{4.37}$$

$$h_{in}(\mathbf{x}) = {}^{\mathcal{N}}\mathbf{e}^\top \mathbf{R}(\mathbf{q})\, {}^{\mathcal{B}}\mathbf{d} - \cos(\theta_{in}) \;. \tag{4.38}$$

Using these two CBFs in the CLF-CBF-QP controller is an option, but as previously mentioned in Section 3.2.4.B, the CLF-CBF-QP controller can have undesirable equilibrium points. The existence of these undesirable equilibrium points is related to the pairwise colinearity of the vectors fields $\mathbf{f}$, $\mathbf{G}\mathbf{G}^\top \nabla V$ and $\mathbf{G}\mathbf{G}^\top \nabla h$ [19], where V is a CLF and h is a CBF. It is our conjecture that adding more CBFs, and thus more vector fields of the form $\mathbf{G}\mathbf{G}^\top \nabla h$ will increase the likelihood of pairwise colinearity and thus the likelihood of having undesirable equilibrium points, though we do not present a rigorous analysis to back this claim.

Regardless of whether having less CBF constraint reduces the possibility of having undesirable equi-

[2]Which we do not consider in this work.

librium points or not, combining two or more CBFs is straightforward. Given CBFs h_i, $i = 1, ..., n$, the safe set is the set of points where all these functions are above or equal to 0

$$\mathcal{C} = \{\mathbf{x} \in \mathcal{X} : h_i(\mathbf{x}) \geq 0, i = 1, ..., n\}$$

which from basic set theory, we know that this is equivalent to

$$\mathcal{C} = \mathcal{C}_1 \cap \mathcal{C}_2 \cap ... \cap \mathcal{C}_n \tag{4.39}$$

where $\mathcal{C}_i$ are the superlevel sets of each CBF h_i. Saying that all functions h_i have to be greater than or equal to 0 at a given $x \in \mathcal{C}$ is equivalent to saying that the lowest value assumed by all of these functions at $\mathbf{x}$ has to greater than or equal to zero. So we get that (4.39) is equivalently written as

$$\mathcal{C} = \{\mathbf{x} \in \mathcal{X} : \min\{h_i(\mathbf{x}) : i = 1, .., n\} \geq 0\} = \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) \geq 0\} \tag{4.40}$$

where

$$h(\mathbf{x}) = \min\{h_i(\mathbf{x}) : i = 1, ..., n\}$$

is the combined CBF. Given this result, we choose the CBF for our constrained attitude problem to be

$$h(\mathbf{x}) = \min\left\{h_{in}(\mathbf{x}), h_{out}(\mathbf{x})\right\}, \tag{4.41}$$

which results in the safe set $\mathcal{C} = \mathcal{C}_{in} \cap \mathcal{C}_{out}$, where

$$\mathcal{C}_{in} = \{\mathbf{x} = (\mathbf{q}, \boldsymbol{\omega}) \in \mathcal{S}^3 \times \mathbb{R}^3 : h_{in}(\mathbf{x}) \geq 0\}$$

which we call the keep in zone and

$$\mathcal{C}_{out} = \{\mathbf{x} = (\mathbf{q}, \boldsymbol{\omega}) \in \mathcal{S}^3 \times \mathbb{R}^3 : h_{out}(\mathbf{x}) \geq 0\}$$

which we call the keep out zone.

Of course, the $\min$ function is not differentiable and so it is not a valid CBF. Luckily, there are several smooth approximations of the minimum function that we can use instead, letting us enforce positive invariance not of $\mathcal{C}$, but of a set that approximates it. One of the possible choice for a smooth approximation of the minimum function is the LogSumExp function [46] which is defined as

$$\mathcal{L}_\beta(x_1, ..., x_n) = \frac{1}{\beta} \log\left(\sum_{i=1}^{n} \exp(\beta x_i)\right), \tag{4.42}$$

where β is a parameter. The LogSumExp has the property that $\mathcal{L}_\beta \to \max$ as $\beta \to \infty$ and $\mathcal{L}_\beta \to \min$ as $\beta \to -\infty$. The LogSumExp with $\beta < 0$ is an under-approximation of the minimum function [46], which means that the set

$$\tilde{\mathcal{C}} = \{\mathbf{x} \in \mathcal{X} : \mathcal{L}_\beta\big(h_1(\mathbf{x}), ..., h_n(\mathbf{x})\big)\} \subseteq \mathcal{C}$$

so enforcing positive invariance of $\tilde{\mathcal{C}}$ enforces positive invariance of $\mathcal{C}$.

In light of the previous discussion, we define our CBF to be

$$h = \mathcal{L}_\beta(h_{in}, h_{out}) = \frac{1}{\beta} \log\big(\exp(\beta h_{in}) + \exp(\beta h_{out})\big)\ , \tag{4.43}$$

with appropriate $\beta < 0$.

By the chain rule, we have

$$\begin{aligned}
\frac{\partial h}{\partial \mathbf{x}} &= \left(\frac{\partial h}{\partial \mathbf{q}}, \frac{\partial h}{\partial\, {}^{\mathcal{L}}\boldsymbol{\omega}}\right) \\
&= \left(\frac{\partial \mathcal{L}_\beta}{\partial h_{in}} \frac{\partial h_{in}}{\partial \mathbf{q}} + \frac{\partial \mathcal{L}_\beta}{\partial h_{out}} \frac{\partial h_{out}}{\partial \mathbf{q}}, \mathbf{0}\right) \implies \\
\nabla h &= \begin{bmatrix} \frac{\partial \mathcal{L}_\beta}{\partial h_{in}} \nabla h_{in} + \frac{\partial \mathcal{L}_\beta}{\partial h_{out}} \nabla h_{out} \\ \mathbf{0} \end{bmatrix}\ .
\end{aligned} \tag{4.44}$$

It becomes clear then that the system with output $y(t) = h(\mathbf{x}(t))$ has relative degree of 2, since

$$L_{\mathbf{G}} h(\mathbf{x}) = \langle \nabla h(\mathbf{x}), \mathbf{G}(\mathbf{x}) \rangle = 0 \implies \dot{h}(\mathbf{x}) = L_{\mathbf{f}} h(\mathbf{x}) = \langle \nabla h(\mathbf{x}), \mathbf{f}(\mathbf{x}) \rangle,$$

$$\ddot{h} = \frac{d}{dt}\dot{h} = \frac{\partial \dot{h}}{\partial \mathbf{x}} \dot{\mathbf{x}} = \underbrace{\nabla \dot{h}^\top \mathbf{f}}_{L_{\mathbf{f}}^2 h} + \underbrace{\nabla \dot{h}^\top \mathbf{G}}_{L_{\mathbf{G}} L_{\mathbf{f}} h} {}^{\mathcal{B}}\boldsymbol{\tau}. \tag{4.45}$$

So we define the HOCBF

$$\psi(\mathbf{x}) = \dot{h}(\mathbf{x}) + \alpha_0 h(\mathbf{x}),$$

which results in the CBF constraint for the CLF-CBF-QP given by

$$\begin{aligned}
L_{\mathbf{f}}^2 h(\mathbf{x}) + L_{\mathbf{G}} L_{\mathbf{f}} h(\mathbf{x})\, {}^{\mathcal{B}}\boldsymbol{\tau} + \alpha_0 L_{\mathbf{f}} h(\mathbf{x}) \geq -\alpha_1 L_{\mathbf{f}} h(\mathbf{x}) - \alpha_1 \alpha_0 h(\mathbf{x}) \Leftrightarrow \\
-\alpha_1 L_{\mathbf{f}} h(\mathbf{x}) - \alpha_1 \alpha_0 h(\mathbf{x}) - L_{\mathbf{f}}^2 h(\mathbf{x}) - L_{\mathbf{G}} L_{\mathbf{f}} h(\mathbf{x})\, {}^{\mathcal{B}}\boldsymbol{\tau} - \alpha_0 L_{\mathbf{f}} h(\mathbf{x}) \leq 0 \Leftrightarrow \\
CBF(\mathbf{x}, {}^{\mathcal{B}}\boldsymbol{\tau}) \leq 0
\end{aligned} \tag{4.46}$$

where as a class $\mathcal{K}_\infty$ functions we choose the linear functions with constants $\alpha_0, \alpha_1 \in \mathbb{R}^+$.

Additionally, we might want to add bounds on the maximum angular velocity of the satellite. This is usually done with a norm constraint in optimization-based controllers such as MPC

$$\|\omega_{k|k+i}\|_p \leq \omega_{max}, i = 1, ..., N \ , \tag{4.47}$$

where $\omega_{max} \in \mathbb{R}^+$ and $\| \circ \|_p : \mathbb{R}^n \to \mathbb{R}$ is the p-norm, defined as

$$\begin{cases} \|\mathbf{x}\|_p = \left(\sum_{i=1}^n x_i^p\right)^{\frac{1}{p}} \text{ for } p \in \mathbb{N} \\ \\ \|\mathbf{x}\|_p = \max\{|x_i| : i = 1, ..., n\} \text{ for } p = \infty \end{cases} ,$$

where we have that the 2-norm (i.e. $p = 2$) is the Euclidean norm $\| \circ \|_2 = \| \circ \|$. Typically $p = 2$ or $p = \infty$, which results in a second order cone constraint for the 2-norm or a group of linear constraints for the infinity norm. The vectors $\omega_{k|k+i} \in \mathbb{R}^3$, $i = 1, ..., N$ are the angular velocities of the horizon of length N, starting at time instance k, which are variables of the optimization problem solved by the MPC. For the CLF-CBF-QP, we cannot enforce the limits of the angular velocity as a constraint directly, since at each time instance k, ω_k it is a constant of the optimization problem. We thus enforce the angular velocity limit as a CBF

$$h_\omega(\mathbf{x}) = \frac{1}{p}\omega_{max}^p - \frac{1}{p}\| {}^{\mathcal{B}}\boldsymbol{\omega}\|_p^p \tag{4.48}$$

where $p \in \mathbb{N} \setminus \{1\}$. The 1-norm and infinity norm are excluded since they are not differentiable and cannot be used to define a valid CBF. However, the infinity norm can be arbitrarily approximated by having p go to infinity. Of course, having p be very large can cause numerical issues, so this is an issue that needs to be accounted for.

We mentioned that it is preferable to combine CBFs as this might reduce the likelihood of having undesirable equilibrium points. However we are not combining the CBF h_ω that depends only on the angular velocity with the two CBFs, h_{in} and h_{out} that depend only on the attitude quaternion. This is because h_ω and h_{in} and h_{out} result in systems with different relative degrees. Combining them in a function $h(\mathbf{x}) = \mathcal{L}_\beta\big(h_{in}(\mathbf{x}), h_{out}(\mathbf{x}), h_\omega(\mathbf{x})\big)$ which is of relative degree 1 but in the set of states $\{(\mathbf{q}, \boldsymbol{\omega}) \in \mathcal{S}^3 \times \mathbb{R}^3 : \boldsymbol{\omega} = \mathbf{0}\}$, $L_{\mathbf{G}}h = 0$, meaning that whenever the angular velocity is zero, the CBF constraint would lose dependency in ${}^{\mathcal{B}}\boldsymbol{\tau}$, resulting in a possibly infeasible optimization problem. Therefore, we keep the CBF related to the forbidden attitude constraints separate from the CBF related to the maximum angular velocity constraint.

We now wish to add the stabilizing constraint to the CLF-CBF-QP. Given the nature of our problem, we have to slightly break away from the theory established in Section 3.1, this is due to the fact that the typical choice for a CLF, the quadratic function

$$V(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \mathbf{x}_f\|^2$$

does not work, since for any $\mathbf{x} = (\mathbf{q}, \mathbf{0})$, we get

$$\nabla V(\mathbf{x}) = (\mathbf{q} - \mathbf{q}_f, \mathbf{0}) \implies L_{\mathbf{G}} V(\mathbf{x}) = 0$$

but the system with V as output is of relative degree 1, so we cannot apply the HOCBF method from Section 3.3. Instead, we use the positive semi-definite function

$$V(\mathbf{x}) = \frac{1}{2}\|\mathbf{q} - \mathbf{q}_f\|^2 \Leftrightarrow V(\mathbf{x}) = 1 - \mathbf{q}_f^\top \mathbf{q} \tag{4.49}$$

where the quadratic function is equivalent to the cosine distances, since the quaternion is unit norm. This function is not a Lyapunov function, as it is not positive definite, so the results from 3.1 do not apply. But we assert that using (4.49) can still be used to stabilize the system. To prove our assertion, we first present LaSalle's Theorem [36].

Theorem 4.3.1 (LaSalle's Theorem)**.** *Let $\Omega \subset D$ be a compact set that is positvely invariant w.r.t. (3.13). Let $V : D \to R$ be a continuously differentiable function such that $\dot{V}(\mathbf{x}) \leq 0$ in Ω. Let E be the set of all points in Ω where $\dot{V}(\mathbf{x}) = 0$. Let M be the largest invariant set in E. Then every solution starting in Ω approaches M as $t \to \infty$.*

Note that the function V in Theorem 4.3.1 does not need to be positive definite. We can now proceed to assert the following:

Proposition 4.3.1. *The autonomous system of the form*

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{G}(\mathbf{x}(t))\mathbf{k}(\mathbf{x}(t)) = \mathbf{f}_{cl}(\mathbf{q}(t), \boldsymbol{\omega}(t)) \tag{4.50}$$

where $\mathbf{f}$ and $\mathbf{G}$ are defined as in (4.3) and $\mathbf{k} : \mathcal{X} \to \mathcal{U}$ is a control law such that

$$\dot{V}(\mathbf{x}) \leq -\lambda_0 V(\mathbf{x}) \tag{4.51}$$

and

$$\dot{h}_\omega(\mathbf{x}) \geq -\kappa h_\omega(\mathbf{x}) \tag{4.52}$$

where, $\lambda_0, \kappa \in \mathbb{R}^+$, V is given by (4.49) and h_ω is given by (4.48) is asymptotically stable to the point $(\mathbf{q}_f, \mathbf{0})$.

Proof: Take $D = \mathbb{R}^7$ and $\Omega = \mathcal{S}^3 \times U$ with $U = \{\boldsymbol{\omega} \in \mathbb{R}^3 : \|\omega\|_p \leq \omega_{max}\}$. The set Ω is a compact set, since it is the cartesian product of compact sets. Additionally, it is positively invariant, since the positive invariance of U w.r.t. (4.50) is enforced by the constraint (4.52) and

$\mathcal{S}^3$ is invariant w.r.t. (4.50) by the quaternion kinematics. We have that $V(\mathbf{x}) \geq 0, \forall \mathbf{x} \in \Omega$ so $\dot{V}(\mathbf{x}) \leq 0, \forall \mathbf{x} \in \Omega$ by constraint (4.51), which is enforced by the control law. Moreover, from (4.49) we have that $V(\mathbf{x}) = 0 \Leftrightarrow \mathbf{x} \in E = \{(\mathbf{q}, \boldsymbol{\omega}) \in \mathcal{S}^3 \times U : \mathbf{q} = \mathbf{q}_f\}$ so by constraint (4.51) we get that $\dot{V}(\mathbf{x}) = 0 \Leftrightarrow \mathbf{x} \in E$ so the set of points in Ω where $\dot{V}(\mathbf{x}) = 0$ is E.

The set of points where $\mathbf{q}(t) = \mathbf{q}_f, \forall t \in \mathbb{R}$ is the set $M = \{(\mathbf{q}, \boldsymbol{\omega}) \in \mathcal{S}^3 \times U : \mathbf{q} = \mathbf{q}_f, \mathbf{f}_{cl}(\mathbf{q}, \boldsymbol{\omega}) = \mathbf{0}\}$. Take any $\mathbf{x} = (\mathbf{q}, \boldsymbol{\omega}) \in M$, we have that $\mathbf{f}_{cl}(\mathbf{x}) = \mathbf{0} \implies 0 = -\mathbf{q}_v^\top \boldsymbol{\omega} \wedge \mathbf{0} = q_s \boldsymbol{\omega} + \mathbf{q}_v \times \boldsymbol{\omega} \Leftrightarrow \boldsymbol{\omega} = \mathbf{0}$. Furthermore, $\mathbf{x} \in M \Leftrightarrow \mathbf{q} = \mathbf{q}_f$, so M is the singleton set $M = \{(\mathbf{q}_f, \mathbf{0})\}$, which is the largest and only invariant set in E. Thus, by Theorem 4.3.1, every solution $\mathbf{x} : \mathbb{R}_0^+ \to \mathcal{S} \times \mathbb{R}^3$ of (4.50) with $\mathbf{x}(0) \in \Omega$ approaches $(\mathbf{q}_f, \mathbf{0})$ as $t \to \infty$. $\square$

Given the results of Proposition 4.3.1, we take (4.49) to be the CLF for the system[3]. Since

$$\frac{\partial V}{\partial \mathbf{x}} = \Big(\frac{\partial V}{\partial \mathbf{q}}, \mathbf{0}\Big),$$

the system with output $y(t) = V\big(\mathbf{x}(t)\big)$ has relative degree 2, so we must use the methods presented in Section 3.3 and define the HOCBF

$$\phi(\mathbf{x}) = -\dot{V}(\mathbf{x}) - \lambda_0 V(\mathbf{x})$$

which results in the CLF constraint

$$\begin{aligned} \dot{\phi} \geq -\lambda_1 \phi \Leftrightarrow \quad & (4.53)\\ \lambda_1 \dot{V} + \lambda_1 \lambda_0 V + \ddot{V} + \lambda_0 \dot{V} \leq 0 \implies \quad & (4.54)\\ \lambda_1 L_{\mathbf{f}} V(\mathbf{x}) + \lambda_1 \lambda_0 V(\mathbf{x}) + L_{\mathbf{f}}^2 V(\mathbf{x}) + L_{\mathbf{G}} L_{\mathbf{f}} V(\mathbf{x})\, {}^{\mathcal{B}}\boldsymbol{\tau} + \lambda_0 L_{\mathbf{f}} V(\mathbf{x}) \leq 0 \Leftrightarrow \quad & (4.55)\\ CLF(\mathbf{x}, {}^{\mathcal{B}}\boldsymbol{\tau}) \leq 0\ . \quad & (4.56) \end{aligned}$$

We thus have all the ingredients to formulate the CLF-CBF-QP. Following the theory in Section 3.2.3.A, we would like our controller at every time $t \in \mathbb{R}_0^+$ to apply the control torque, given by solving the optimizaton problem:

$$\begin{aligned} \min_{\boldsymbol{\tau} \in \mathbb{R}^3, \delta \in \mathbb{R}} \quad & \|\boldsymbol{\tau}\|^2 + \rho \delta^2 && (4.57)\\ s.t. \quad & CLF\big(\mathbf{x}(t), \boldsymbol{\tau}\big) + \delta \leq 0 && \text{(Attitude stabilizing CLF)}\\ & CBF\big(\mathbf{x}(t), \boldsymbol{\tau}\big) \leq 0 && \text{(Constrained attitude CBF)}\\ & \boldsymbol{\tau} - \tau_{max} \leq 0 && \text{(Torque lower bound)}\\ & -\boldsymbol{\tau} - \tau_{max} \leq 0 && \text{(Torque upper bound)}\\ & \nabla h_\omega^\top(\mathbf{x}(t)) \mathbf{f}(\mathbf{x}(t)) + \nabla h_\omega^\top(\mathbf{x}(t)) \mathbf{G}(\mathbf{x}(t)) \boldsymbol{\tau} \geq -\frac{\kappa}{p} h_\omega(\mathbf{x}(t)) && \text{(Maximum angular velocity CBF)} \end{aligned}$$

[3]Though in this case, calling it a Control LaSalle Function would be more appropriate.

However, in a practical implementation, the optimization problem is solved by a computer, which takes time to perform computations. Therefore, we cannot obtain an input for every t. Instead, we obtain an input every $t = k\Delta t$, where Δt is our sampling time and $k \in \mathbb{N}$. For $t \in [k\Delta t, (k+1)\Delta t)$, the input to the system is a ZOH of the solution obtained at $t = k\Delta t$. This makes the closed loop system a hybrid system, where the continuous time rotation dynamics are controlled by a discrete time controller.

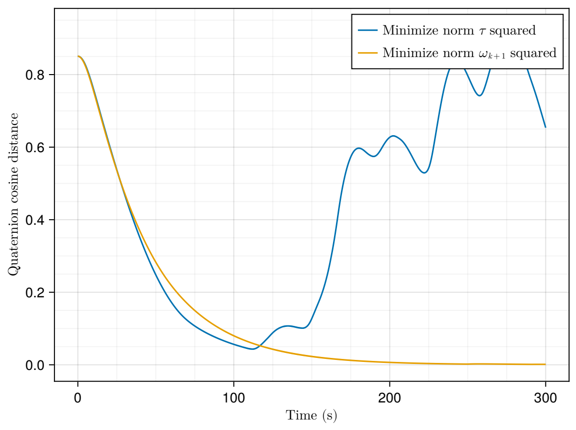
Unfortunately, our results for guaranteed stability and safety are only valid in the continuous-time case, i.e. in the limiting case where $\Delta t \to 0$. We can still experimentally guarantee safety for small enough Δt if we also make α_0 in (4.46) small enough, thus making the state stay far enough away from the unsafe set, such that the ZOH does not result in an unsafe state in the next time instance. Stability can also be obtained for small enough Δt, but only guaranteed for $\Delta t \to 0$. We found experimentally that we can obtain stability for larger sampling times by using the square of the forward Euler approximation of the angular velocity in the cost function, instead of $\|\boldsymbol{\tau}\|^2$. This is shown in Figure 4.2, which compares the results for two simulations differing only in the cost function used. Therefore, our attitude controller is given by the optimization problem:

$$
\begin{aligned}
\min_{\boldsymbol{\tau}\in\mathbb{R}^3,\delta\in\mathbb{R}} \quad & \left\| {}^{\mathcal{B}}\boldsymbol{\omega}_k + \Delta t\, {}^{\mathcal{B}}\mathbf{J}^{-1}\big({}^{\mathcal{B}}\mathbf{J}\, {}^{\mathcal{B}}\boldsymbol{\omega}_k \times {}^{\mathcal{B}}\boldsymbol{\omega}_k + \boldsymbol{\tau}\big)\right\|^2 + \rho\delta^2 && (4.58)\\
s.t. \quad & CLF\big(\mathbf{x}_k, \boldsymbol{\tau}\big) + \delta \leq 0 && \text{(Attitude stabilizing CLF)}\\
& CBF\big(\mathbf{x}_k, \boldsymbol{\tau}\big) \leq 0 && \text{(Constrained attitude CBF)}\\
& \boldsymbol{\tau} - \tau_{max} \leq 0 && \text{(Torque lower bound)}\\
& -\boldsymbol{\tau} - \tau_{max} \leq 0 && \text{(Torque upper bound)}\\
& \nabla h_\omega^\top(\mathbf{x}_k)\mathbf{f}(\mathbf{x}_k) + \nabla h_\omega^\top(\mathbf{x}_k)\mathbf{G}(\mathbf{x}_k)\boldsymbol{\tau} \geq -\frac{\kappa}{p} h_\omega(\mathbf{x}_k) && \text{(Maximum angular velocity CBF)}
\end{aligned}
$$

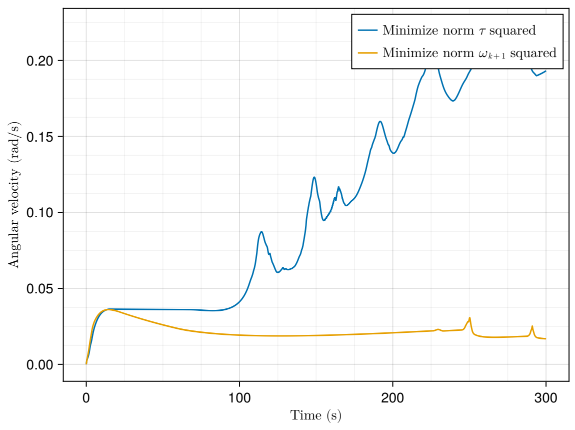
where $\mathbf{x}_k = \mathbf{x}(k\Delta t)$ is the current state.

4.4 Monte Carlo Simulations

We now present Monte Carlo (MC) simulations to validate the practical viability of using the CLF-CBF-QP controller presented in (4.58) for the constrained attitude problem.



(a) Cosine distance between current quaternion and target quaternion over time.



(b) Norm of the angular velocity over time.

Figure 4.2: Comparison of the cost functions for CLF-CBF-QP.

4.4.1 Simulation Parameters

We base the physical parameters on those presented in [12], namely, the inertia matrix in the body-fixed frame is given by

$$^{\mathcal{B}}\mathbf{J} = \begin{bmatrix} 125.734 & 0.0 & 0.0 \\ 0.0 & 216.211 & 0.0 \\ 0.0 & 0.0 & 234.055 \end{bmatrix}, \tag{4.59}$$

which is the diagonalized version of the one presented in [12]. The maximum torque is $\tau_{max} = 0.6\,\mathrm{N\,m}$ which is the maximum torque that can be applied by the reaction wheel assembly in [12] and the maximum velocity is $\omega_{max} = 5\,\mathrm{rad\,s^{-1}}$.

The controller was implemented in the Julia programming language [47]. The solver used to obtain a solution to (4.58) was ECOS [48], which is a SOCP solver, in conjunction with JuMP [49] as an interface to model the optimization problem. The Julia module ForwardDiff [50] was used to obtain the derivatives required for the optimization problem.

To increase the verisimilitude of the simulations, we consider the time it takes for the controller to perform calculations. To achieve this we consider one sampling time of delay in the control torque, i.e. the control torque calculated for state $\mathbf{x}_k$ can only be applied at $t = (k+1)\Delta t$, and if the controller has not finished calculating the control torque in the time interval $t \in [k\Delta t, (k+1)\Delta t]$, then the control torque applied at $k+1$ is $\tau_{k+1} = \mathbf{0}$[4], the maximum number of iterations that ECOS can perform was set to be 15.

The processor used to perform the simulations is an Intel© Core™i5-6200U processor that has

[4]This is, admittedly, a poor strategy. A better strategy would be to apply an intermediate solution calculated by the solver that is sub-optimal but feasible, but not having access to the internals of the solver, we will go with this approach.

$2.8\,\mathrm{GHz}$ processing speed. Space grade processors have much lower processing speeds[5], which are on the order of MHz. As such, we multiply the time it takes the solver to obtain a control solution by 100, and if that time exceeds Δt the control solution applied is zero.

We perform MC simulation for 200 random trajectories. The trajectories were generated with $\mathbf{q}_0 = (1, 0, 0, 0)$, and the target attitude, $\mathbf{q}_f$, for each run was a randomly selected point on $\mathcal{S}^3$ obtained by normalizing random draws from a 4-dimensional standard normal distribution. The random selection had a viability condition to check if the pointing vectors for the final attitude were outside the keep in/out zones, if they were not, then the final attitude is not viable and a new quaternion was randomly selected. Since $\mathbf{q}_f$ and $-\mathbf{q}_f$ describe the same final attitude, the quaternion was chosen as the one that has the minimum cosine distance to the initial quaternion, i.e. if $1+\mathbf{q}_0^\top \mathbf{q}_f < 1-\mathbf{q}_0^\top \mathbf{q}_f$ we use $-\mathbf{q}_f$ instead of $\mathbf{q}_f$, to have the generated trajectory between, $\mathbf{q}_0$ and $\mathbf{q}_f$, be the one of least distance to the final attitude. The spacecraft pointing vectors were ${}^{\mathcal{B}}\mathbf{v} = (1, 0, 0)$ and ${}^{\mathcal{B}}\mathbf{d} = (0, 1, 0)$. The vectors for the attitude keep in and keep out were ${}^{\mathcal{N}}\mathbf{e} = (1, 0, 0)$ and ${}^{\mathcal{N}}\mathbf{s} = \left(\sqrt{2}/2, 0, \sqrt{2}/2\right)$, respectively and the angles chosen were $\theta_{in} = 2\pi/3$ and $\theta_{out} = 2\pi/3$, respectively.

We determine that the slew maneuver of the satellite, that takes it from $\mathbf{q}_0$ to $\mathbf{q}_f$, should be completed at $t_f = 30\,\mathrm{min}$, the sampling time was chosen to be $\Delta t = 0.2\,\mathrm{s}$. The value of λ_0 for the CLF was chosen according to equation (3.7), where $t_{des} = t_f$, $\varepsilon_0 = 1 - \mathbf{q}_0 \cdot \mathbf{q}_f$ and $\varepsilon_{des} = 1 - \cos(0.1°)$, i.e. we expect that the angle between $\mathbf{q}(t_f)$ and $\mathbf{q}_f$ to be less than or equal to $0.1°$. The value of ρ in (4.58) should be as high as possible without causing the hessian matrix of the cost, which is $\mathbf{H} = \Delta t^2\, {}^{\mathcal{B}}\mathbf{J}^{-\top}\, {}^{\mathcal{B}}\mathbf{J}^{-1}$, to become ill-conditioned, as such we choose ρ to be the value of the smallest eigenvalue of H times 10^{14}, resulting in $\rho = 182542.5$.

As discussed in Section 3.2, the value α_0 for the HOCBF determines how close the state is allowed to get to the unsafe set defined by the superlevel set of (4.43). We want to set α_0 appropriately high, so that safety can be achieved even under the imperfect conditions of the controller being discrete and having a delay of one sampling time. However, setting α_0 too high can cause the controller to be too conservative and cause convergence to $\mathbf{q}_f$ to be much slower. We experimentally determined that $\alpha_0 = 0.1$ was a good value for our specific problem. Likewise, we want λ_1, α_1 and κ to be as high as possible but without risking the state leaving the set $\{\mathbf{x} = (\mathbf{q}, \boldsymbol{\omega}) \in \mathcal{S}^3 \times \mathbb{R}^3 : \dot{h}(\mathbf{x}) \geq -\alpha_0 h(\mathbf{x}), \dot{V}(\mathbf{x}) \leq -\lambda_0 V(\mathbf{x}), \sum_{i=1}^{3} \omega_i^{p-1} \geq -(\kappa/p) \cdot \|\boldsymbol{\omega}\|_p^p\}$. We found experimentally that $\lambda_1 = 0.01$, $\alpha_1 = 0.005$ and $\kappa = 1$ are values that allow for good performance while maintaining safety. The value of β for the LogExpSum function was chosen to be $\beta = -100$.

[5]See https://www.gaisler.com/index.php/products/processors/leon3 for an example of the specifications of space grade processors.

4.4.2 Results with no Constrained attitude CBF

First, we perform the simulation without the constrained attitude CBF from (4.58), so as to verify that the unsafe trajectories are corrected by the inclusion of the CBF.

Figure 4.3 presents a scatter plot that shows the angle between $\mathbf{q}(t_f)$ and $\mathbf{q}_f$ as a function of the angle between $\mathbf{q}(0)$ and $\mathbf{q}_f$ for each of the 200 MC runs. The desired final angle of $0.1°$ was achieved for $80\,\%$ of the MC runs, with a median final angle of $0.075°$ and a maximum angle of $0.145°$, below the acceptable limit of $0.2°$.

The fact that some runs have a final angle between the final quaternion $\mathbf{q}(t_f)$ and target quaternion $\mathbf{q}_f$ that exceeds the desired final angle of $0.1°$ is due to the time delay of one sample time introduced in the controller, which causes jittering. This can be seen in Figure 4.4(a), which plots the angle between $\mathbf{q}(t)$ and $\mathbf{q}_f$ for the final $200\,\mathrm{s}$ of the simulation, for the run that result in the maximum final angle and the run that resulted in the minimum final angle, we can see that their evolution is basically the same. Figure 4.4(b) zooms out and shows the last $800\,\mathrm{s}$ of the run, as can be seen, the controller does not seem to be able to achieve a higher precision, given the time delay and the sampling time used.

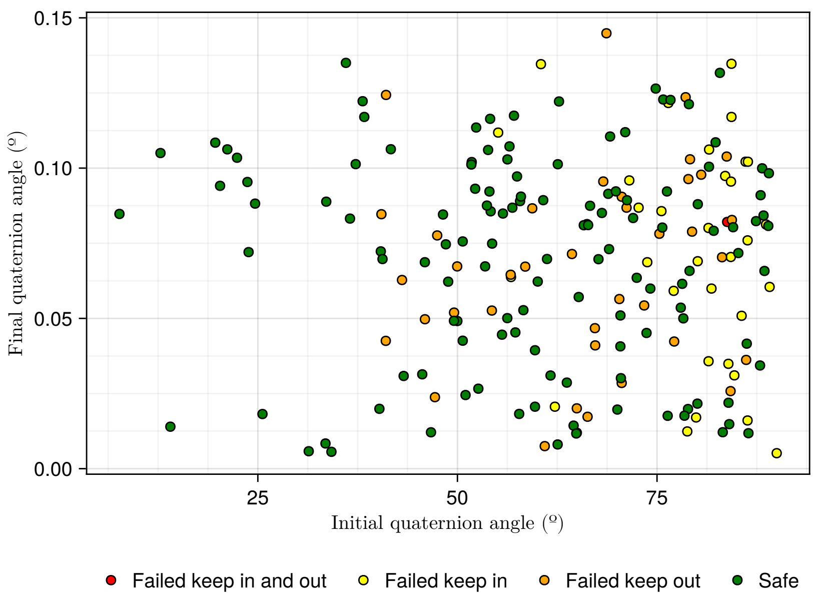
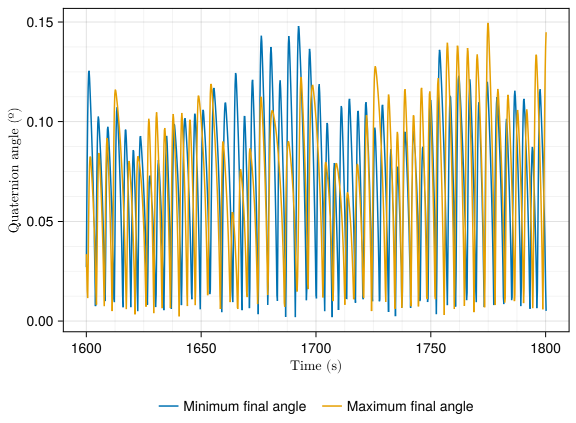


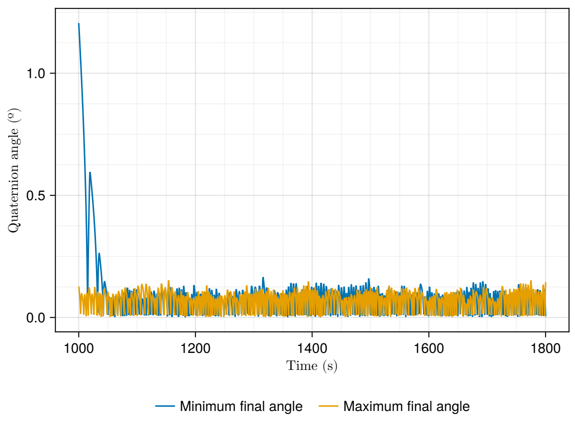
Figure 4.3: Final angle vs initial angle for the attitude controller (4.58) without the constrained attitude CBF.

In terms of safety, we have that $0.5\,\%$ of the runs breached both the keep out and keep in zone, $19\,\%$ breached only the keep out zone and $16\,\%$ breached the keep in zone.

In Figure 4.5, we show the maximum computation that ocurred for each run, with the penalty of $\times 100$ applied. As can be seen, no run took more than the sampling time of $0.2\,\mathrm{s}$ to compute the stabilizing input torque and no run had an infeasible solution.



(a) Quaternion angle evolution for last $200\,\mathrm{s}$ of the simulation.



(b) Quaternion angle evolution for last $800\,\mathrm{s}$ of the simulation.

Figure 4.4: Quaternion angle evolution.

4.4.3 Results with Constrained attitude CBF

We now consider the full attitude controller in (4.58) with the constrained attitude CBF included.

In Figure 4.7, we show the scatter plot of the final quaternion angle as a function of the initial quaternion angle for the attitude controller. As can be seen when compared to Figure 4.3, the inclusion of the constrained attitude CBF corrects unsafe trajectories, resulting in all the trajectories being safe. As a consequence, performance (in terms of achieving the desired final attitude) is degraded, since now $7.5\,\%$ of the runs have an angle between the desired attitude and final attitude that are above $0.2°$. Figure 4.8 shows the final quaternion angle as a function of the value of $h(\mathbf{q}(t_f))$, which measures how close the final attitude is to the safe set boundary, as can be seen , the runs with the attitude target closest to the boundary of the safe set are the ones that had a higher final angle.

Our chosen value of α_0 does not allow the attitude to get close to the boundary of the safe set fast enough to achieve the desired angle between $\mathbf{q}(t_f)$ and $\mathbf{q}_f$ in the desired time. This performance/safety trade-off is unavoidable. We could increase the value of α_0, but this increases the risk of the state becoming unsafe. More tuning of the parameters could allow for better results.

For the runs that did manage to be within the acceptable angle error envelope of $0.2°$, Figure 4.6 shows how long it took for the spacecraft to enter the envelope and never leave. The median time was $18.7\,\mathrm{min}$, the minimum was $12.6\,\mathrm{min}$ and the maximum time was $29.62\,\mathrm{min}$.

In Figure 4.9, we show the maximum computation time that occurred in each run of the MC simulation for the attitude controller. As can be seen, no run had a computation time that exceeded the sampling time. However, there were 4 runs for which the controller return infeasible solutions for the input control. These runs were number 41, 118, 141 and 164 and they had 4, 10, 5 and 3 instances of infeasibility,

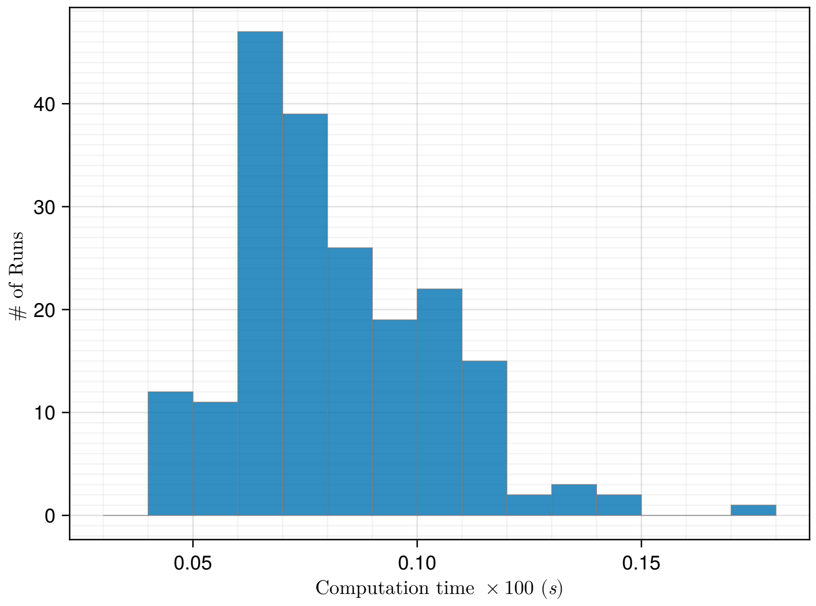


Figure 4.5: Histogram of maximum computation times.

respectively. We are unsure what caused the occurrence of infeasibility instances. All of the instances that resulted in infeasibility had a value of h above 0.096, far from the boundary of the safe set, and the maximum computation time did not exceed $0.033\,\mathrm{s}$, with the penalty applied. For the runs that had instances of infeasible solutions, these accounted for less than $0.12\,\%$ of the total sampling instances of the run, and the final angles were $0.099°$, $0.02°$, $0.02°$, $0.116°$ and $0.0496°$, respectively, so performance was not impacted.

4.4.4 SCvx results and Comparison with CLF-CBF-QP

We now present the results of the Monte Carlo simulations conducted using the MC to track trajectories generated by SCvx. The simulation parameters remain consistent with those discussed in Section 4.4.1, while the SCvx parameters are detailed in Section 4.2.1. The MPC, as defined in (4.30), employs an horizon of $H = 2$, chosen experimentally as the longest horizon for which the SOCP problem can be solved reliably in the allotted computation time. Similarly to the conditions outlined for the CBF-CLF-QP controller in Section 4.4.1, the MPC has a one sample time delay and is required to provide a feasible solution within one sampling time, otherwise no input is applied.

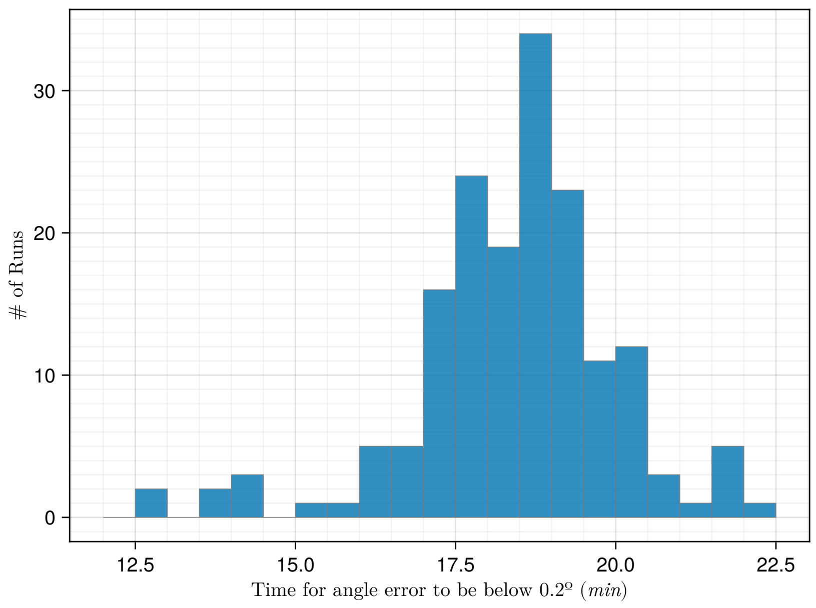


Figure 4.6: Histogram of time to enter envelope of acceptable angle error.

4.4.4.A Analysis of Generated Trajectories

First, we analyze the trajectories generated by SCvx. Figure 4.10 shows a histogram of the maximum defect for each run. The maximum defect for a run is given by

$$\delta = \max\left\{ \left\| \begin{bmatrix} \mathbf{I}_{4\times4} & \mathbf{0}_{4\times3} \\ \mathbf{0}_{3\times4} & \omega_{max}^{-1}\mathbf{I}_{3\times3} \end{bmatrix} \left(\mathbf{x}_k^d - \boldsymbol{\psi}\left(t_{k-1}, t_k, \mathbf{x}_{k-1}^d, {}^{\mathcal{B}}\boldsymbol{\tau}^o\right)\right) \right\|_\infty : k = 1, ..., N-1 \right\},$$

where $\mathbf{x}_k^d$ is the state at node k and $\boldsymbol{\psi}$ is the flow map[6], which is described in Section 4.2.1. The angular velocity part of the state vectors are divided by ω_{max} so that each element of the state has a maximum absolute value of 1.

As can be seen in Figure 4.10. The median of the maximum defects is $1.139 \cdot 10^{-6}$, which is several orders of magnitude below the dynamic feasibility tolerance of $feas\ tol = 10^{-3}$ that we specified in the SCvx parameters. However, four of the runs had maximum defects above the feasibility tolerance and two of those four runs had maximum defects on the order of 10^{-1}.

As previously mentioned, the SCvx algorithm cannot guarantee that the interpolated trajectory it returns is safe, even if it is dynamically feasible, since the state constraints are only satisfied at the nodes. On the interpolated trajectories, we sample the state at each sampling interval $\{k\Delta t\}_{k=0}^{T}$ and check whether it is in the safe set defined in (4.40). Figure 4.11 shows the histogram of the number of unsafe states that occurred in the interpolated trajectory of each run. A majority, namely $73\,\%$, of the interpolated trajectories do not violate the keep out or keep in zone, and $98.5\,\%$ have unsafe states for

[6]With the parameter vector argument excluded, since we do not use it in our problem.

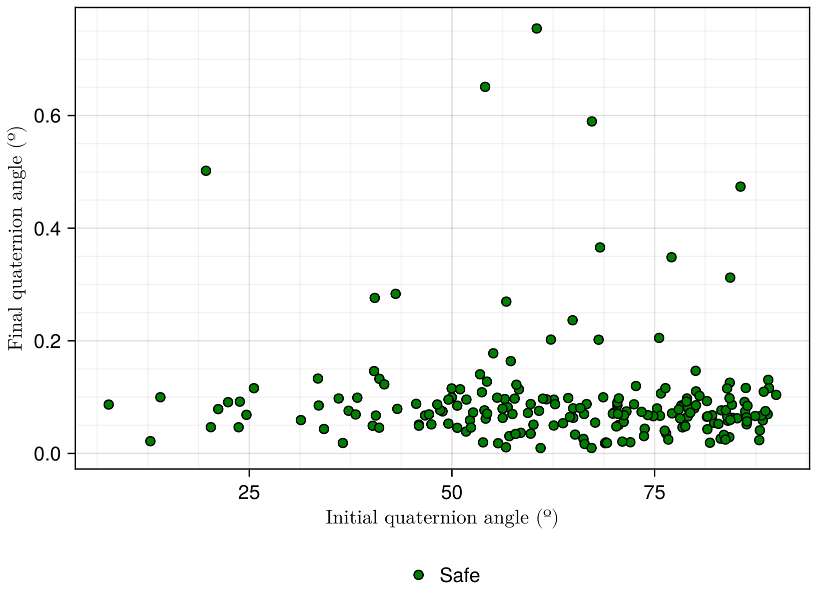


Figure 4.7: Final angle vs initial angle for the attitude controller (4.58) .

less than $10\,\%$ of the number of samples points. These $98.5\,\%$ are likely to be corrected by the MPC and result in safe maneuvers.

However, the remaining $1.5\,\%$ of the trajectories have unsafe states occurring in more than $30\,\%$ of the samples. These are the same trajectories that had the top three largest maximum defect values in Figure 4.10 and are unlikely to be rendered safe.

Figure 4.12 shows the computation times for the SCvx trajectory generation, with the $\times 100$ penalty applied. The maximum computation time was $13.5\,\mathrm{min}$, the minimum was $11.33\,\mathrm{min}$ and the median computation time was $12.52\,\mathrm{min}$.

4.4.4.B Analysis of realized maneuvers

We now analyze the realized trajectories resulting from the MPC following the reference trajectories generated by SCvx.

All except 3 of the runs performed by the MPC satisfied the the safety constraints. These runs corresponded to reference trajectories with the highest maximum defect values, as depicted in Figure 4.10. Specifically, the runs with the highest maximum defects, ranked in descending order, were: run 41, run 114, and run 64.

For the three runs that infringed the forbidden attitude zones, we consider the values of the barrier function, presented in (4.41), and the angle between the current attitude and the desired attitude, obtained in these runs. The values obtained using the SCvx+MPC method are compared those obtained

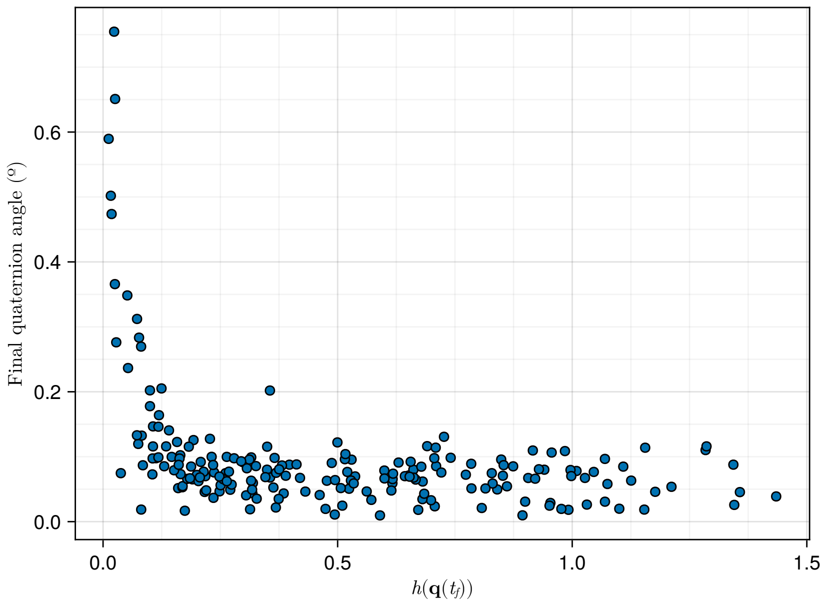


Figure 4.8: Final quaternion angle as a function of the final value of the constrained attitude barrier function.

with the CLF-CBF-QP attitude controller. These comparisons are shown in Figures 4.13, 4.14, and 4.15.

Figure 4.16 shows the difference in safety between the MPC trajectories and the CLF-CBF-QP trajectories as a histogram of values obtained by the difference between the lowest value of (4.41) obtained by the MPC trajectory of a run and the lowest value of (4.41) obtained by the CLF-CBF-QP trajectory of the same run. Negative values, mean that the MPC trajectory was closer to the boundary of the safe set than the CLF-CBF-QP controller, and hence are less safe. The three lowest values of Figure 4.16 are the unsafe MPC trajectories. Overall, for $85\,\%$ of the runs, the MPC trajectories are less safe than the CLF-CBF-QP (but still safe), for $1.5\,\%$ of the runs the MPC runs were unsafe, while the CLF-CBF-QP run was safe, and for $13.5\,\%$ of the runs, the MPC trajectories were as safe or safer than the CLF-CBF-QP trajectories.

Figure 4.17 shows the difference in angle errors, which is the difference between the final angle error obtained by the MPC and the final angle error obtained by the CLF-CBF-QP for the same run. Negative values denote runs where the final angle error for the MPC was lower than that achieved by the CLF-CBF-QP controller. Overall, in $75\,\%$ of the runs, the MPC managed to achieve a lower or equal final angle error compared to the CLF-CBF-QP controller.

Figure 4.18 presents a scatter plot illustrating the relationship between the final quaternion angle error and the initial quaternion angle error, serving as the MPC counterpart to Figure 4.7. The scatter plot excludes unsafe trajectories, as they have a final angle above $1°$. Only $2.5\,\%$ of the runs exhibit a final angle error above the acceptable threshold of $0.2°$, indicating an improvement over the CLF-CBF-QP.

For the MPC runs that achieved the acceptable angle error, Figure 4.19 shows a histogram of the

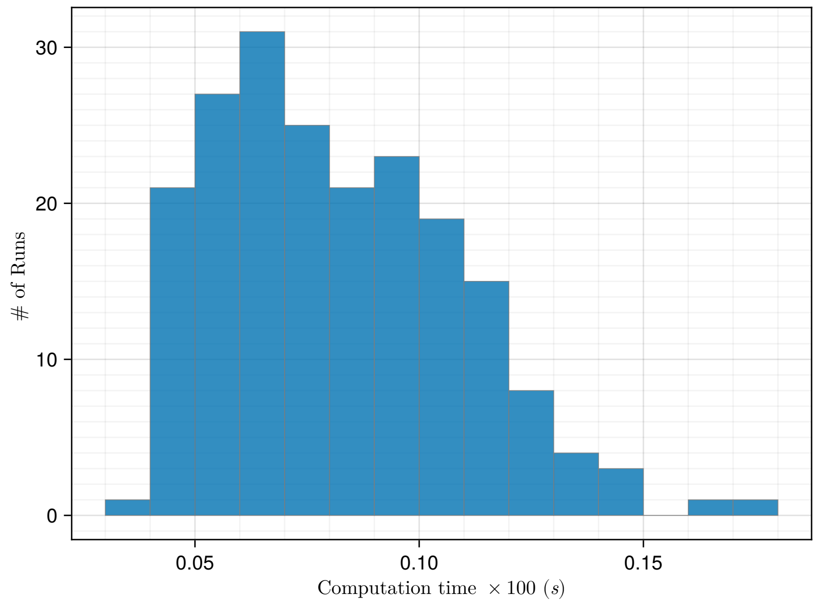


Figure 4.9: Histogram of maximum computation times.

times taken to enter the $0.2°$ envelope. Compared to Figure 4.6, it is evident that the MPC takes longer to enter the envelope. The minimum time for the MPC runs is above the maximum time for the CLF-CBF-QP runs. This difference arises because the CLF in our CLF-CBF-QP controller is designed to ensure that the cosine distance between the attitude and the desired attitude decreases exponentially. In contrast, the reference trajectory generated by SCvx only requires the final attitude to be $\mathbf{q}_f$, while minimizing energy use.

Figure 4.20 presents a histogram of the ratios between the energy used by the MPC and the CLF-CBF-QP trajectories for the same runs. The data shows that the CLF-CBF-QP controller uses at least $10^{2.19}$ times more energy than the MPC controller. In $50\,\%$ of the cases, the energy use by the CLF-CBF-QP exceeds $10^{6.18}$ times that of the MPC, with a maximum ratio of $10^{8.41}$. The higher energy consumption of the CLF-CBF-QP controller is due to its greedy optimization approach and the constant class kappa function in the CLF, which aims for exponential stabilization to the equilibrium point, making it significantly less energy-efficient than the MPC.

Figure 4.21 presents a histogram of the maximum computation times (with the $\times 100$ penalty) for each run. Comparing this with Figure 4.9, it is evident that solving a SOCP in the MPC requires significantly more computation time than solving a QP in the CLF-CBF-QP. As a result, every MPC run includes instances where the computation time exceeded the $0.2\,\mathrm{s}$ sampling time. In these instances, null torque was applied, which did not compromise safety except for the three SCvx trajectories that were not dynamically feasible. Five runs experienced instances where the MPC returned infeasible solutions, including the dynamically infeasible runs 41, 64, and 114 with 5639, 1206, and 5836 instances,

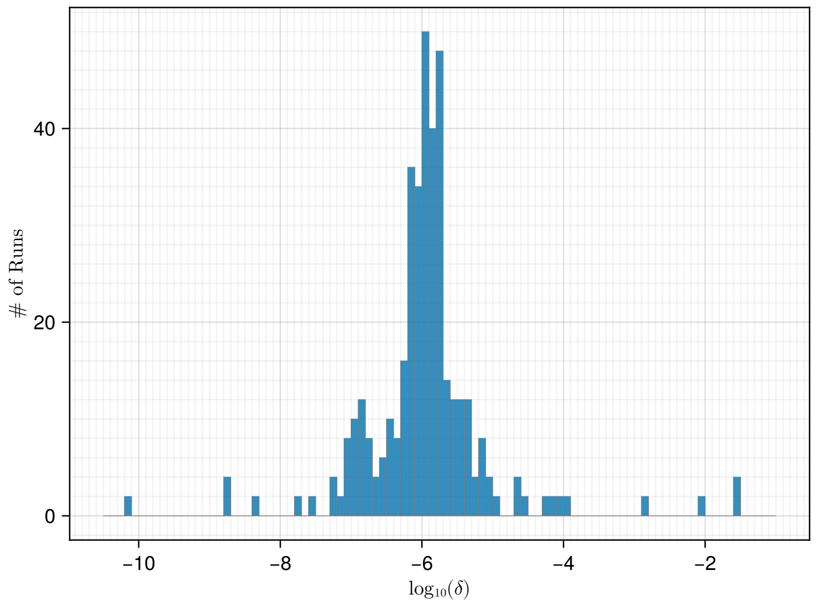


Figure 4.10: Histogram of maximum defects.

respectively, as well as the dynamically feasible runs 42 and 133 with 17 and 21 instances, respectively.

4.4.5 Summary of results

Table 4.1 summarizes the statistics for the data presented in the previous sections, while Table 4.2 shows the percentage of unsafe runs and runs with an acceptable final angle. Overall, the MPC following the trajectories generated by SCvx achieves better performance in terms of lower final angle error between the attitude quaternions and energy usage. However, there are no guarantees that SCvx will converge to a safe or dynamically feasible trajectory, compromising safety for performance. Additionally, the SCvx and MPC method is more computationally expensive. Figure 4.12 shows that generating the trajectory alone takes about one third of the time required to perform the maneuver. Furthermore, Table 4.1 indicates that the MPC controller's computation time exceeded the sampling time in every run.

	Maximum $\|\omega\|_\infty$ per run ($\mathrm{rad\,s^{-1}}$)		Maximum $\|\tau\|_\infty$ per run ($\mathrm{N\,m}$)		Final angle error ($^\circ$)		Maximum computation time per run ($\times 100$) (s)		Time to enter 0.2° envelope (min)		Energy use (J^2)		Minimum barrier value per run	
	CCQ	MPC	CCQ	MPC	CCQ	MPC	CCQ	MPC	CCQ	MPC	CCQ	MPC	CCQ	MPC
Minimum	0.00393	0.00020	0.6	7.5837×10^{-5}	0.0096	0.0045	0.0378	0.2096	12.60	27.043	966.009	1.6156×10^{-5}	0.00634	−0.49958
25th percent quantile	0.00575	0.00126	0.6	0.000658	0.0523	0.0336	0.0608	0.2321	17.75	28.86	1819.12	0.001218	0.05167	0.00159
Median	0.00696	0.00167	0.6	0.00094	0.0746	0.0469	0.0771	0.2436	18.697	28.97	2178.26	0.00238	0.10509	0.10662
75th percent quantile	0.00788	0.00202	0.6	0.00152	0.0978	0.0629	0.1002	0.2666	19.66	29.05	2494.93	0.00434	0.15892	0.15892
Maximum	0.03097	0.00738	0.6	0.58890	0.7546	35.5195	0.1741	1.9464	29.62	29.18	3449.69	27.1902	0.15892	0.15892

Table 4.1: Summary statistics for the MC results of the CLF-CBF-QP (CCQ) controller and the MPC.

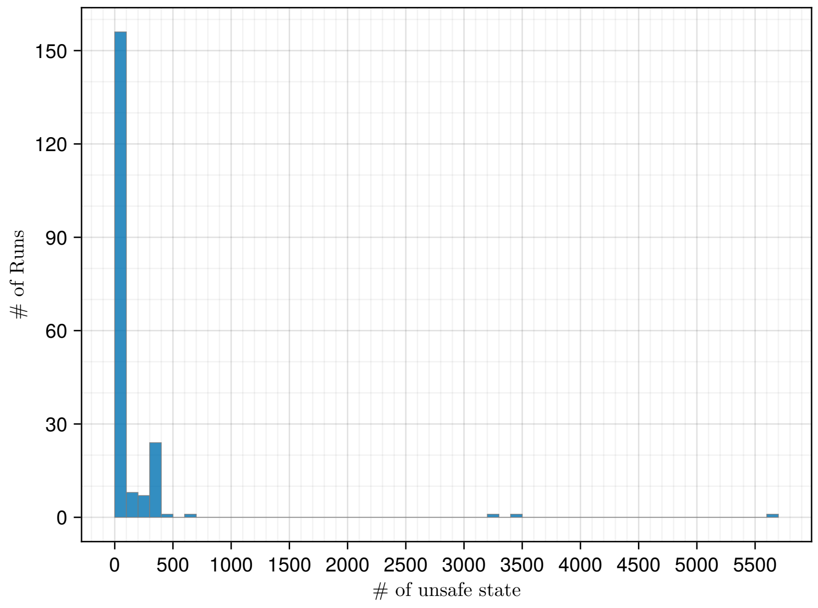


Figure 4.11: Histogram of unsafe states in interpolated trajectories.

	CCQ	**MPC**
Percentage of unsafe runs	$0\,\%$	$1.5\,\%$
Percentage of runs with final angle $\leq 0.2°$	$92.5\,\%$	$97.5\,\%$

Table 4.2: Percentages of unsafe run and runs with acceptable final angle for CLF-CBF-QP (CCQ) and MPC.

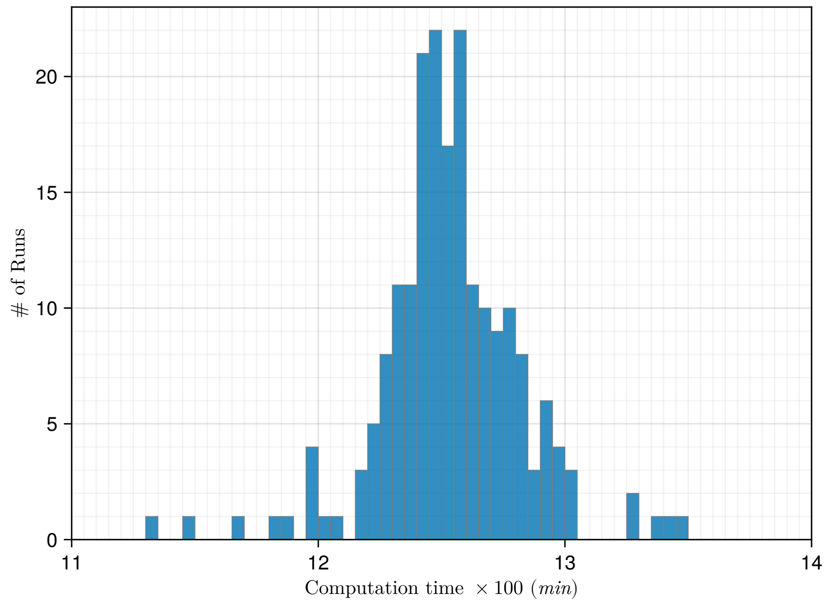
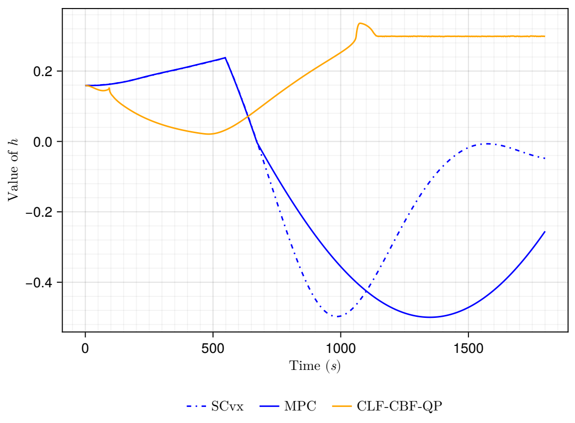
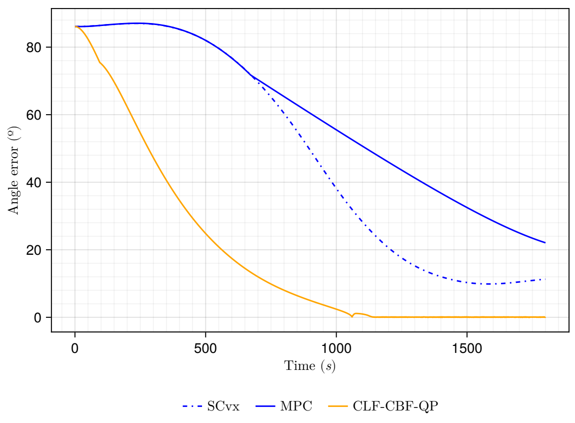


Figure 4.12: Histogram of SCvx computation times.

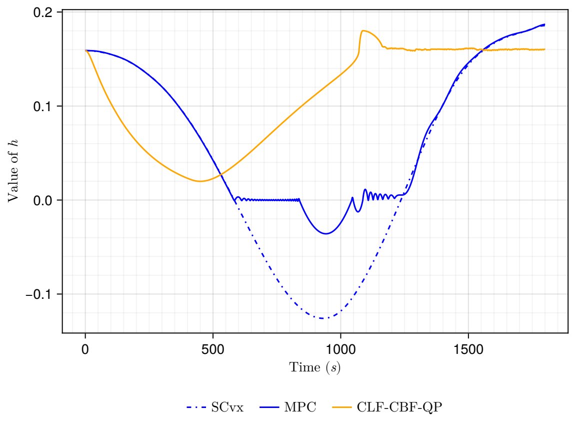


(a) Quaternion angle evolution for last $200\,\mathrm{s}$ of the simulation.

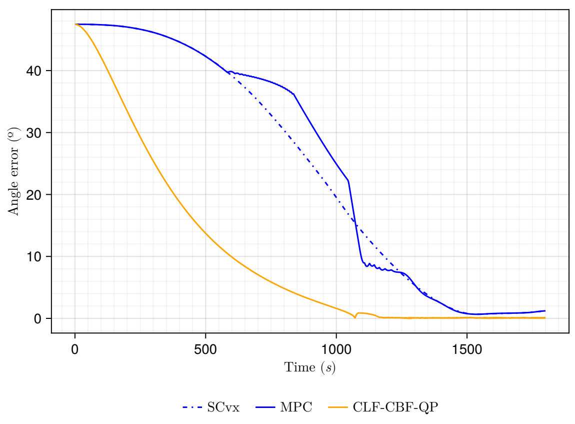


(b) Quaternion angle evolution for last $800\,\mathrm{s}$ of the simulation.

Figure 4.13: Comparison of barrier and angle error values for run 41.

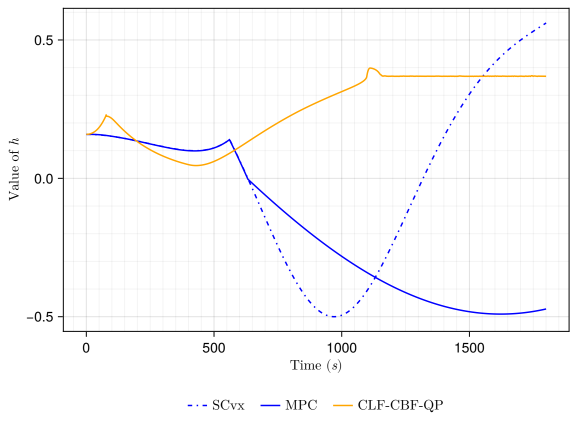


(a) Barrier values.

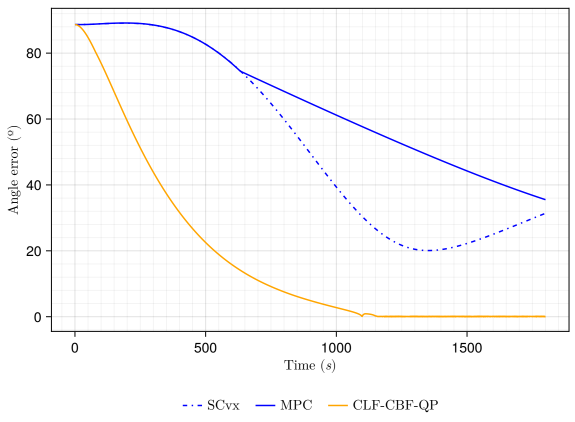


(b) Angle error values.

Figure 4.14: Comparison of barrier and angle error values for run 64.



(a) Barrier values.



(b) Angle error values.

Figure 4.15: Comparison of barrier and angle error values for run 114.

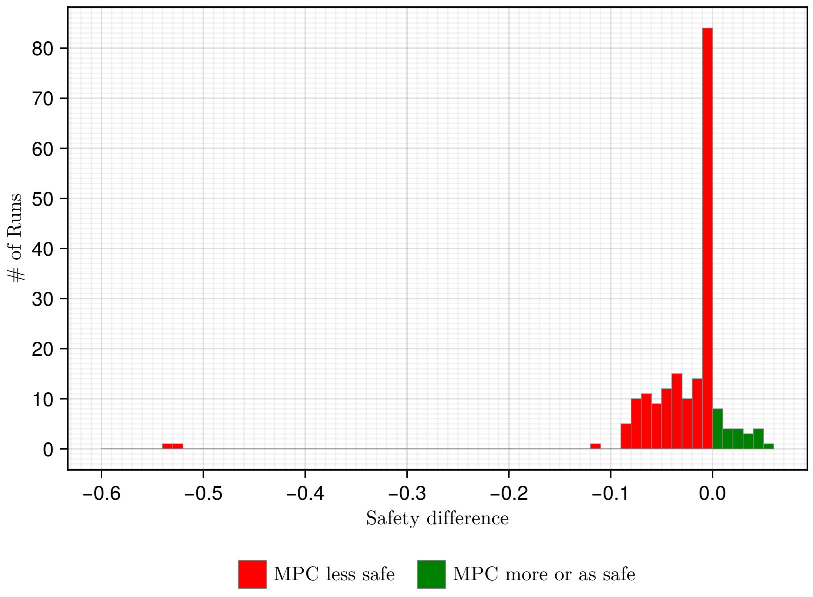


Figure 4.16: Histogram of safety differences.

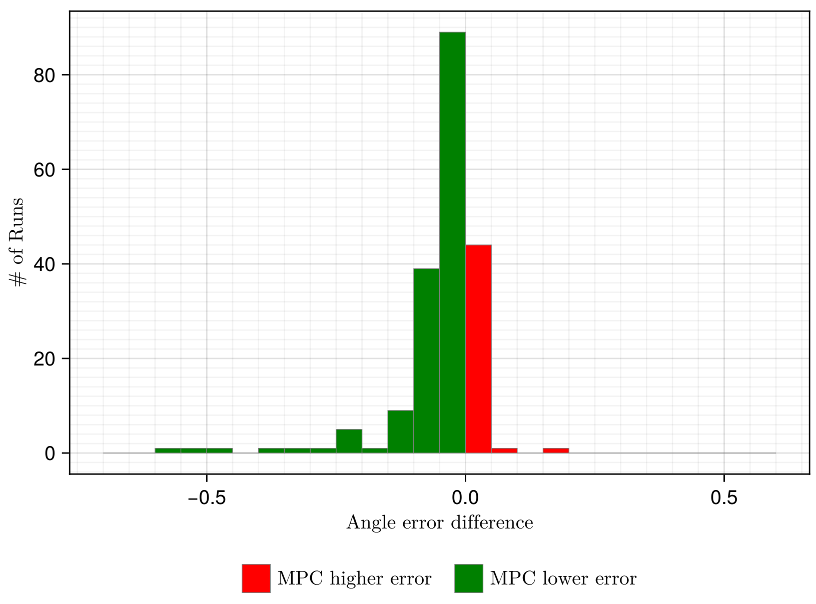


Figure 4.17: Histogram of angle error differences.

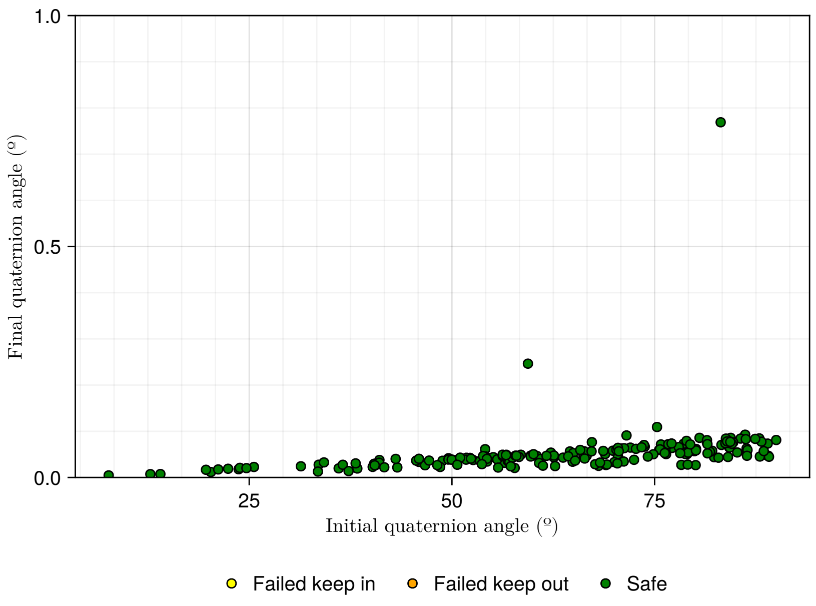


Figure 4.18: Final angle vs initial angle for MPC.

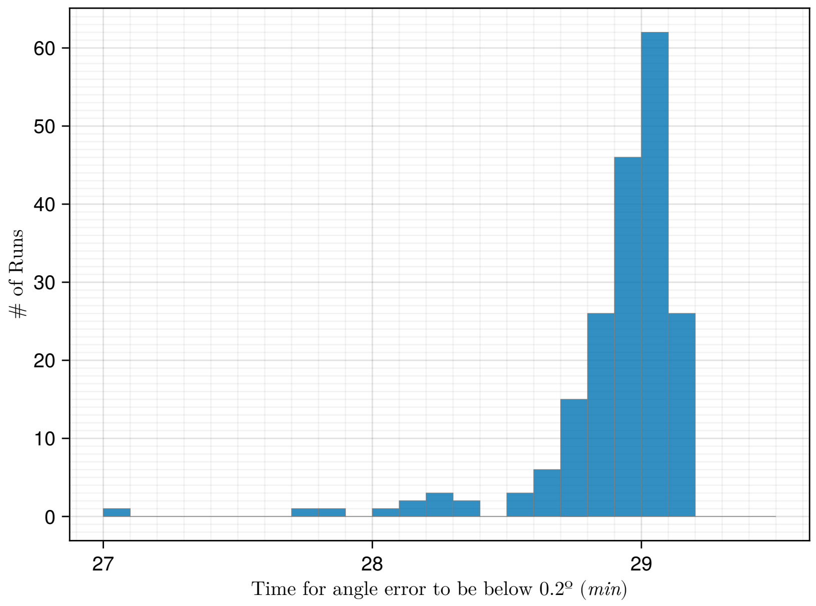


Figure 4.19: Histogram of time to enter envelope of acceptable angle error.

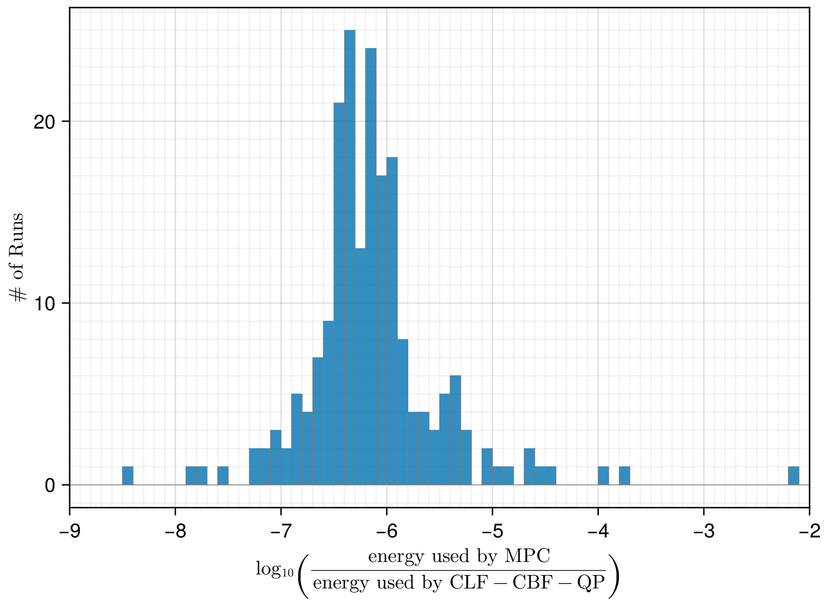


Figure 4.20: Histogram of energy ratios between MPC and CLF-CBF-QP.

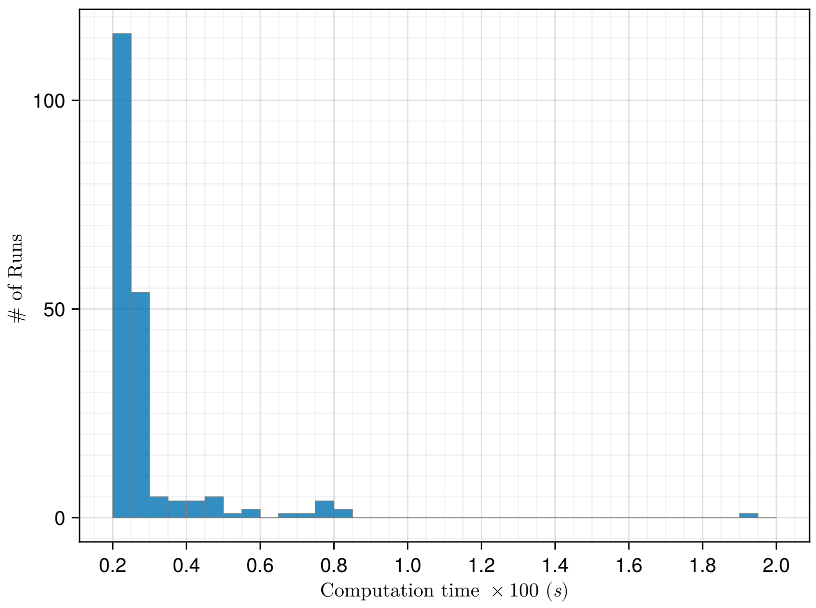


Figure 4.21: Histogram of maximum computation times for MPC.

5

Conclusion

Contents

5.1 Concluding remarks

The main goal of this thesis was to demonstrate how state-of-the-art, real-time optimization-based control methods, specifically CLFs and CBFs, which are widely studied in academia but largely absent in industry, can be applied to solve practical problems in the aerospace sector. We focused on the constrained attitude problem, which is critical for a wide range of space missions [14] [15] [12] [11].

Currently, attitude representations in the aerospace industry primarily use quaternions. However, the mathematical foundation of the quaternion representation is often overlooked in engineering textbooks. A thorough understanding of this mathematics is essential to grasp the subtle differences between various quaternion conventions and their interrelations. To address this, Chapter 2 provides a concise theory of attitude representations, focusing on quaternions, starting from first principles. This chapter aims to serve as a valuable reference for engineers.

Similarly, the theoretical foundations of CLFs and CBFs are often presented in academic papers in a manner that is not easily accessible to most engineers. To bridge this academia-industry gap, Section 3 derives CLFs from the fundamental principles of Lyapunov functions and CBFs from Nagumo's Theorem. This elementary approach, besides being accessible, has the advantage that the guarantees of stability for CLFs and of safety for CBFs are proven by construction. It is our hope that this theoretical presentation will lead to the wider adoption of these methods in industry.

In Chapter 4, we formulated the constrained attitude problem and reviewed state-of-the-art methods based on SCvx and MPC, which are used for constrained attitude control in SOC-i and were proposed for the Comet Intercept mission [11] [12]. Our proposed solution, detailed in Section 4.3, extends the theory of CLFs by demonstrating that a function V that is only positive semidefinite can still be used as a CLF and proven stable using LaSalle's Theorem.

Finally, Monte Carlo simulations of our proposed CLF-CBF-QP controller were presented in Section 4.4, thus providing experimental validation for our claims that CLF and CBF based methods are provably stable, safe and computationally efficient. In Section 4.4.4, we compared our CLF-CBF-QP based approach to the constrained attitude problem with the SCvx and MPC based approaches. We concluded that while our CLF-CBF-QP controller is more reliable in terms of safety, it incurs a trade-off of lower performance in terms of final error and energy usage.

5.2 Future work

We have shown that the CLF-CBF-QP is an attractive solution to the constrained attitude reorientation problem, but some work remains to be done before its full potential as a reliable atittude controller can be achieved. Namely, we have yet to prove that the proposed CLF-CBF-QP controller does not have undesirable stable equilibria, as Reis et al. [37] have shown to be possible. Although the MC simulations did not result in any undesirable equilibria, to fully satisfy the stability guarantees for space applications we must prove that these undesirable stable equilibrium points either: do not exist; that they exist but the set of initial conditions that converges to these points is of measure zero; or otherwise, implement the proposed fix by Reis et al.

A possible avenue of future work worth pursuing, that we did not have time to approach in this thesis, consists of complementing SCvx with the CLF-CBF-QP controller. As we saw in Section 4.2.1, the SCvx algorithm requires an initial guess and using an uninformed initial guess based on state interpolation resulted in trajectories which failed to converge to a dynamically feasible and safe trajectory. The quality of the trajectory generated by SCvx and the speed of convergence depends on the initial guess, so if one were to use the CLF-CBF-QP to first generate a safe trajectory as an initial guess, it is possible that SCvx could quickly obtain a refined solution that is safe, dynamically feasible and more energy efficient than the one obtained using just the CLF-CBF-QP.

Bibliography

[1] F. Blanchini and S. Miani, *Set-theoretic methods in control*, ser. Systems & control. Birkhäuser [Springer, distributor], 2008.

[2] D. Malyuta, T. P. Reynolds, M. Szmuk, T. Lew, R. Bonalli, M. Pavone, and B. Açıkmeşe, "Convex optimization for trajectory generation: A tutorial on generating dynamically feasible trajectories reliably and efficiently," *IEEE Control Systems Magazine*, vol. 42, no. 5, pp. 40–113, 2022.

[3] D. Malyuta, Y. Yu, P. Elango, and B. Açıkmeşe, "Advances in trajectory optimization for space vehicle control," *Annual Reviews in Control*, vol. 52, pp. 282–315, Jan 2021.

[4] P. Lu, "Introducing computational guidance and control," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 193–193, 2017.

[5] E. Minisci, A. Riccardi, and M. Vasile, "A preface to the special issue on optimization in space engineering," *Optimization and Engineering*, vol. 24, no. 1, pp. 93–97, Mar 2023.

[6] M. Szmuk, T. P. Reynolds, and B. Açıkmeşe, "Successive convexification for real-time six-degree-of-freedom powered descent guidance with state-triggered constraints," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 8, pp. 1399–1413, 2020.

[7] X. Liu, P. Lu, and B. Pan, "Survey of convex optimization for aerospace applications," *Astrodynamics*, vol. 1, no. 1, pp. 23–40, Sep 2017.

[8] L. Blackmore, "Autonomous precision landing of space rockets," *The Brirdge*, vol. 46, pp. 15–20, 01 2016.

[9] M. Leomanni, R. Quartullo, G. Bianchini, A. Garulli, and A. Giannitrapani, "Variable-horizon guidance for autonomous rendezvous and docking to a tumbling target," *Journal of Guidance, Control, and Dynamics*, vol. 45, no. 5, pp. 846–858, 2022.

[10] T. Guffanti and S. D'Amico, "Robust passively safe spacecraft swarming via closed-form and optimization-based control approaches," in *2022 American Control Conference (ACC)*, 2022, pp. 416–423.

[11] T. P. Reynolds, C. L. Kelly, C. Morgan, A. Grebovic, J. Erickson, H. Brown, W. C. Pope, J. Casamayor, K. Kearsley, G. Caylak, K. E. Fisher, C. Wutzke, K. Ashton, J. Rosenthal, D. Tormey, E. Freneau, G. Giddings, H. E. Horata, A. Dighde, S. Parakh, J. Zhen, J. C. Purpura, D. B. Pratt, and A. Hunt, “Soc-i: A cubesat demonstration of optimization-based real-time constrained attitude control,” in *2021 IEEE Aerospace Conference (50100)*, 2021, pp. 1–18.

[12] V. Preda, A. Hyslop, and S. Bennani, “Optimal science-time reorientation policy for the comet interceptor flyby via sequential convex programming,” *CEAS Space Journal*, vol. 14, no. 1, pp. 173–186, Jan 2022.

[13] P. Tsiotras and M. Mesbahi, “Toward an algorithmic control theory,” *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 194–196, 2017.

[14] G. Singh, G. Macala, E. Wong, R. Rasmussen, G. Singh, G. Macala, E. Wong, and R. Rasmussen, “A constraint monitor algorithm for the cassini spacecraft,” in *Guidance, Navigation, and Control Conference*, 1997.

[15] A. Stern and D. Grinspoon, *Chasing New Horizons: Inside the Epic First Mission to Pluto*, first edition ed. Picador, 2018.

[16] J. D. Koenig, “A novel attitude guidance algorithm for exclusion zone avoidance,” in *2009 IEEE Aerospace conference*, 2009, pp. 1–10.

[17] E. D. Sontag, “A ‘universal’ construction of artstein’s theorem on nonlinear stabilization,” *Systems & Control Letters*, vol. 13, no. 2, pp. 117–123, 1989.

[18] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs for safety critical systems,” *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2017.

[19] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control barrier functions: Theory and applications,” in *2019 18th European Control Conference (ECC)*, 2019, pp. 3420–3431.

[20] H. Sommer, I. Gilitschenski, M. Bloesch, S. Weiss, R. Siegwart, and J. Nieto, “Why and how to avoid the flipped quaternion multiplication,” *Aerospace*, vol. 5, no. 3, 2018.

[21] D. Yazell, *Origins of the Unusual Space Shuttle Quaternion Definition*, ser. Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, Jan 2009.

[22] S. L. Altmann, *Rotations, Quaternions, and Double Groups*, ser. Oxford Science Publications. Clarendon Pr, 1986.

[23] M. D. Shuster, “Survey of attitude representations,” *Journal of the Astronautical Sciences*, vol. 41, no. 4, pp. 439–517, Oct. 1993.

[24] J. Sola, “Quaternion kinematics for the error-state kalman filter,” 2017.

[25] B. O’Neill, *Elementary Differential Geometry*, rev. 2nd ed ed. Elsevier Academic Press, 2006.

[26] J. Stillwell, *The Four Pillars of Geometry*, ser. Undergraduate Texts in Mathematics. Springer, 2005.

[27] S. Axler, *Linear Algebra Done Right*. Springer, 2014.

[28] H. Goldstein, J. L. Safko, and C. P. Poole, *Classical Mechanics*, third edition, pearson new international edition ed. Pearson, 2014.

[29] J. M. Lee, *Introduction to Topological Manifolds*, 2nd ed., ser. Graduate Texts in Mathematics. Springer, 2011.

[30] T. A. Garrity, *All the Math You Missed: But Need to Know for Graduate School*, second edition ed. Cambridge University Press, 2021.

[31] K. Tapp, *Matrix Groups for Undergraduates*, second edition ed., ser. Student Mathematical Library. American Mathematical Society, 2016, no. Volume 79.

[32] J. Stuelpnagel, “On the parametrization of the three-dimensional rotation group,” *SIAM Review*, vol. 6, no. 4, pp. 422–430, 2024/06/20/ 1964, full publication date: Oct., 1964.

[33] R. P. Bob Palais and S. Rodi, “A disorienting look at euler’s theorem on the axis of a rotation,” *The American Mathematical Monthly*, vol. 116, no. 10, pp. 892–909, 2009.

[34] S. L. Altmann, “Hamilton, rodrigues, and the quaternion scandal,” *Mathematics Magazine*, vol. 62, no. 5, pp. 291–308, 1989.

[35] B. Paul, “On the composition of finite rotations,” *The American Mathematical Monthly*, vol. 70, no. 8, pp. 859–862, 1963.

[36] H. K. Khalil, *Nonlinear systems*, 3rd ed. Prentice Hall, 2002.

[37] M. F. Reis, A. P. Aguiar, and P. Tabuada, “Control barrier function-based quadratic programs introduce undesirable asymptotically stable equilibria,” *IEEE Control Systems Letters*, vol. 5, no. 2, pp. 731–736, 2021.

[38] A. J. Taylor, P. Ong, T. G. Molnar, and A. D. Ames, “Safe backstepping with control barrier functions,” in *2022 IEEE 61st Conference on Decision and Control (CDC)*, 2022, pp. 5775–5782.

[39] G. Wu and K. Sreenath, “Safety-critical and constrained geometric control synthesis using control lyapunov and control barrier functions for systems evolving on manifolds,” in *2015 American Control Conference (ACC)*, 2015, pp. 2038–2044.

[40] Q. Nguyen and K. Sreenath, “Exponential control barrier functions for enforcing high relative-degree safety-critical constraints,” in *2016 American Control Conference (ACC)*, 2016, pp. 322–328.

[41] W. Xiao and C. Belta, “High-order control barrier functions,” *IEEE Transactions on Automatic Control*, vol. 67, no. 7, pp. 3655–3662, 2022.

[42] Y. Mao, M. Szmuk, X. Xu, and B. Açikmese, “Successive convexification: A superlinearly convergent algorithm for non-convex optimal control problems,” *arXiv preprint arXiv:1804.06539*, 2018.

[43] Y. Mao, M. Szmuk, and B. Açıkmeşe, “Successive convexification of non-convex optimal control problems and its convergence properties,” in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 3636–3641.

[44] D. C. Lay, *Linear Algebra and Its Applications*, 4th ed. Addison-Wesley, 2012.

[45] D. Malyuta, T. Reynolds, M. Szmuk, M. Mesbahi, B. Acikmese, and J. M. Carson, *Discretization Performance and Accuracy Analysis for the Rocket Powered Descent Guidance Problem*, ser. AIAA SciTech Forum. American Institute of Aeronautics and Astronautics, Jan 2019.

[46] T. G. Molnar and A. D. Ames, “Composing control barrier functions for complex safety specifications,” *IEEE Control Systems Letters*, vol. 7, pp. 3615–3620, 2023.

[47] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, “Julia: A fresh approach to numerical computing,” *SIAM review*, vol. 59, no. 1, pp. 65–98, 2017.

[48] A. Domahidi, E. Chu, and S. Boyd, “ECOS: An SOCP solver for embedded systems,” in *European Control Conference (ECC)*, 2013, pp. 3071–3076.

[49] M. Lubin, O. Dowson, J. Dias Garcia, J. Huchette, B. Legat, and J. P. Vielma, “JuMP 1.0: Recent improvements to a modeling language for mathematical optimization,” *Mathematical Programming Computation*, 2023.

[50] J. Revels, M. Lubin, and T. Papamarkou, “Forward-mode automatic differentiation in Julia,” *arXiv:1607.07892 [cs.MS]*, 2016.