PhD-FSTM-2021-57
The Faculty of Sciences, Technology and Medicine

DISSERTATION

Defence held on 19/08/2021 in Luxembourg

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN SCIENCES DE L'INGÉNIEUR

by

Manuel CASTILLO LOPEZ
Born on 23 May 1992 in Malaga (Spain)

OPTIMAL MOTION PLANNING AND CONTROL WITH SAFETY GUARANTEES FOR AERIAL ROBOTS

Dissertation defence committee:

Dr. -Ing. Holger Voos, Dissertation Supervisor.
Professor, Université du Luxembourg

Dr. -Ing. Jean-Regis Hadji-Minaglou, Chairman.
Professor, Université du Luxembourg

Dr. Miguel A. Olivares-Mendez, Vice Chairman.
Professor, Université du Luxembourg

Dr. -Ing Mohamed Darouach.
Professor, Université de Lorraine.

Dr. Maryam Kamgarpour.
Assistant Professor, University of British Columbia
Visiting Professor, École Polytechnique Fédérale de Lausanne (EPFL).

University of Luxembourg

Interdisciplinary Centre for
Security, Reliability, and Trust

Automation and Robotics Research Group

Doctoral Thesis

Optimal Motion Planning and Control with Safety Guarantees for Aerial Robots

Author:

Manuel Castillo López

Supervisors:

Prof. Dr.-Ing. Holger Voos
Prof. Dr.-Ing. Miguel A. Olivares-Mendez
Prof. Dr.-Ing. Jean-Regis Hadji-Minaglou

September 9, 2021

Declaration of Authorship

I, Manuel Castillo López, declare that this thesis titled, "Optimal Motion Planning and Control with Safety Guarantees for Aerial Robots" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

September 9, 2021, Grand Duchy of Luxembourg.

Do not let the day end without having grown a little, without being happy, without having risen your dreams. Do not let yourself be overcome by disappointment. Do not let anyone take away the right to express yourself, which is almost a duty. Do not forsake the yearning to make your life something extraordinary. Do not stop believing that words and poetry can change the world. Whatever happens, our essence is intact. We are passion-full beings. Life is a desert and an oasis. It knocks us down, it hurts us, it teaches us, it makes us protagonists of our own history. Although the wind blows against, The powerful work continues: You can make a stanza. Never stop dreaming, because in dreams you are free.

Walt Whitman – Leaves of Grass

Abstract

Autonomous aerial robots are expected to revolutionize many industries, such as construction, transportation or even space exploration. However, to target an industry where different robots and humans are meant to share the same space our algorithms need to provide safety and efficiency guarantees. Navigating autonomously in these kind of environments poses a great challenge. We may face a significant number of obstacles, and we can only estimate where they are and where they are expected to be, but not exactly. Dealing with these uncertainties is a challenging problem in most robotics applications, including motion planning and control. During the last decade, major contributions have established the theoretical basis upon which optimal motion planning and control with safety guarantees can be achieved. However, they involve a high computational cost that scales exponentially with the number of obstacles, rendering a limited domain of robotic applications. The main contribution of this thesis provides an efficient, scalable and safe approximation to this problem, allowing its application to embedded systems with fast dynamics such as aerial robots. This thesis also includes an additional contribution that allow these methods to plan longer trajectories with a minimal computational footprint, allowing to better anticipate evasive maneuvers. These contributions have been validated mathematically, in simulation and in real-time operation on aerial robots, handling uncertain dynamic obstacles such as pedestrians.

Acknowledgements

This PhD has been a truly hard and revealing experience for me. I am deeply grateful for my experience at the Automation and Robotics Research Group, lead by Prof. Holger Voos. He managed to put up with my passion, stubbornness and my unconventional ways of doing research in a gentle and nurturing manner. I carry with me many lessons learned from him.

My sincere gratitude to Prof. Miguel A. Olivares-Mendez. The first one who believed and encouraged me to pursue research. He has always given me his time and support in rough times or when organizing a barbecue. It is hard to move to a different country, but I was lucky enough to have him, his wife Lucia, and his son Marco to make me feel back home again.

Dr. Jose Luis Sanchez-Lopez. A dear and hard-working friend that always pushed me and himself towards excellence. A great deal of professional and emotional support was carried on his shoulders. We had a hell of a ride during this PhD, and it has been an honor to work on your side.

Dr. Seyed Amin Sajadi Alamdari. My mentor and dear friend. Countless discussions, lessons and emotional support came from him. He showed me the beauty of optimal control, the devil behind the details, and the core challenges in the way. This thesis wouldn't have been possible without you.

Special thanks to Prof. Maryam Kamgarpour for her valuable feedback and for welcoming me at the Automatic Control Laboratory in ETH Zürich. Her research continues to be a source of inspiration for me.

Kind regards to my colleagues and friends Maciej Zurad, Dr. Dario Cazzato, Dr. Serket Quintanar, Dr. Jann Dentler, Claudio Cimarelli and the rest of the Automation and Robotics Research Group, that lifted this experience from great to awesome.

Last but not least, I would like to thank my family, friends and girlfriend for their unconditional support and love. I am lucky to have you.

Manuel Castillo López

Contents

Abbreviations

TTC^{-1} inverse time to collision. 52–57, 74, 75, 77, 90, 92

BVP boundary-value problem. 13

FOD First Order with Delay. 36

HJB Hamilton-Jacobi-Bellman. 12, 13

MPC Model Predictive Control. 5, 32, 33, 40–42, 68, 69, 79–81, 88–90, 92

NLP nonlinear program. 13, 19, 39, 51, 61

NMPC Nonlinear Model Predictive Control. 46–48, 52

OCP Optimal Control Problem. 12–15, 20, 39, 49, 84, 87

RHC Receding Horizon Control. 32

ROS The Robot Operating System. 33, 35, 39, 52

RTI real-time iteration scheme. 39

SLAM Simultaneous Localization and Mapping. 2, 3

SQP Sequential Quadratic Programming. 39

UAV Unmanned Aerial Vehicle. 5, 31, 36, 41–43, 45–49, 51, 53, 54, 56, 58

Chapter 1

Introduction

This thesis presents different contributions to the field of optimal motion planning and control for aerial robots that allows a higher degree of autonomy and safety in navigation tasks. This chapter introduces the problem of autonomous navigation in Section 1.1, and the role optimal motion planning and control plays in such a problem. Then, Section 1.2 contextualizes and introduces the motivation of this study, outlining the significance of our study and an overview of our contributions in Section 1.3. Finally the outline of the thesis is presented in Section 1.4.

1.1 Autonomous Navigation

Walking is one of the first skills that we develop in our early lives, giving us the ability to navigate autonomously and efficiently from one location to another. This involves complex cognitive processes that, as adults, we are able to perform effortlessly without even being aware of their existence. To enable such behaviors in robots we need to account for these cognitive processes explicitly, casting them into algorithms so they can be processed by computers. In this introduction, we overview these concepts from a pseudo-human perspective, connecting these ideas to their respective field in robotics research.

The process starts with perception. The light emitted or reflected by different objects is captured by our eyes, projecting images in our retinas, encoding them as electric signals which are sent to the brain. Then, the brain decodes these signals into information about colors and depth, constructing our visual perception. In robotics we recreate such a skill using stereo cameras, which encode color and depth into a matrix called pointcloud, as shown

in Fig. 1.1. With this information, we are able to perceive the space that

Figure 1.1: Zed stereo camera and generated pointcloud.

surround us, partitioning the space into objects with different attributes and functionalities. Then, we are able to build a three-dimensional map of where these objects are, inferring our own location in such a map. Reproducing such a skill computationally in robotic systems is the task of Simultaneous Localization and Mapping (SLAM), visualized in Fig. 1.2, a thriving research topic that represents one of the biggest challenges in robotics research [Davison, 2018].

Once the model of the environment is built, the next challenge is to control our body to reach a different location in the map. To that aim, the brain must generate a sequence of electric signals over time that actuate our muscles to reach a desired posture without running into collision. To efficiently plan and execute such a motion, the brain must have a model of how these signals translate into motion. When other people are in the scene, their intentions and their future location need to be estimated to anticipate collision and efficiently re-plan our trajectory. Reproducing such a skill computationally in robotics systems is the task of optimal motion planning and control. Its real-time application on aerial robots with safety guarantees is the focus of this thesis, as visualized in Fig. 1.3.

Finally, autonomous navigation is achieved by a suitable combination of localization, mapping, optimal motion planning and control. However, it is still hard to achieve real-time performance in a resource constrained platform such as aerial robots. The robotics research community is making outstanding contributions to each of these fields at an alarming pace thanks to a combination of specialization and collaboration. In this thesis, we narrow our focus in real-time optimal motion planning and control with safety guarantees, which is a challenge on its own and a key to achieving trust in autonomous robotic applications.

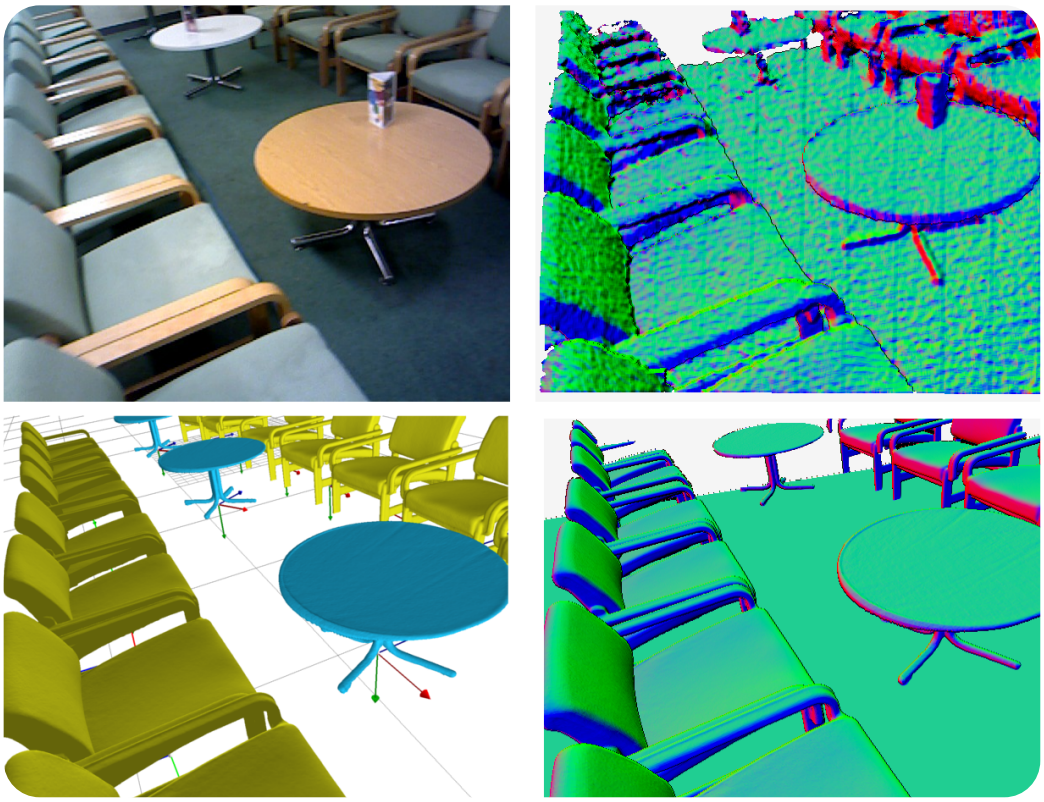


Figure 1.2: SLAM: From images to object detection, localization and mapping [Salas-Moreno et al., 2013].

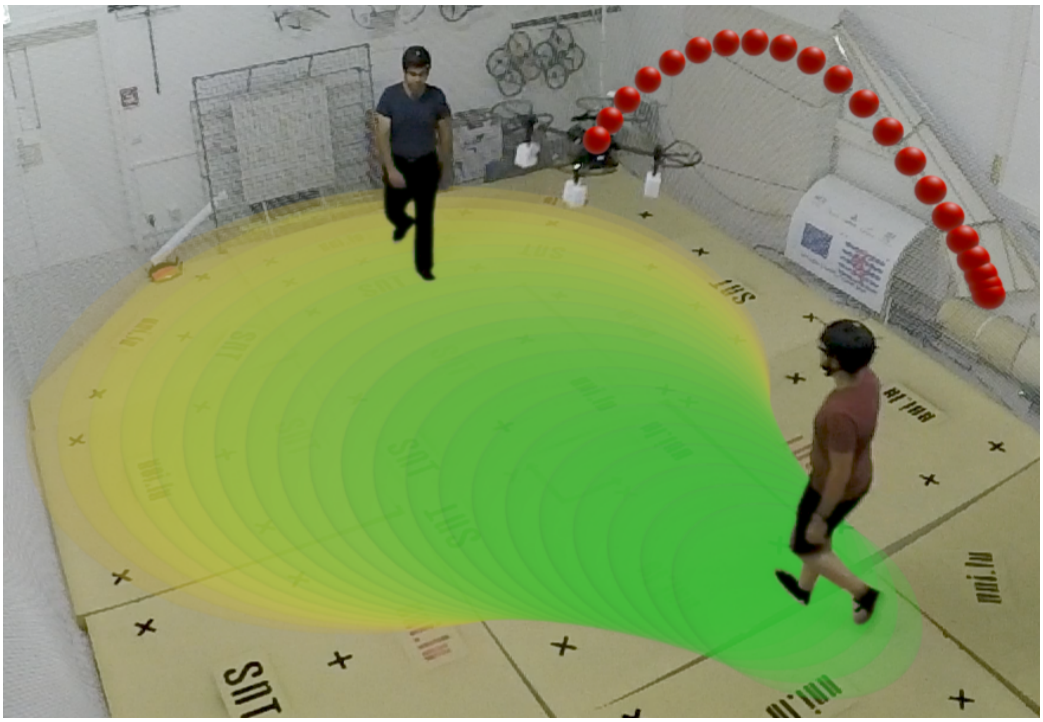


Figure 1.3: Optimal motion planning and control in dynamic environments [Castillo-Lopez et al., 2020].

1.2 Motivation and Objectives

Aerial robots, commonly known as drones or Unmanned Aerial Vehicle (UAV), are becoming more important in our society every day. What started as a recreational device for photography is now playing an important role in sensitive tasks such as high-altitude operations [Watson et al., 2020], airplane inspections [Zhu et al., 2021] or search-and-rescue operations [Dang et al., 2020]. Although human operators can be effective, they quickly become stressed and fatigued when addressing time-sensitive tasks with a limited field-of-view. This problem aggravates significantly in multi-robot and human-robot operations. As a result, increasing autonomy has become the key to fully exploit aerial robots in current and future applications, being the focus of active research [Chung et al., 2018].

From the user perspective, increasing the level of autonomy translates into higher level control commands such as "search for survivors in this area" or "transport a medical kit to this location". To enable such a technological leap, these robots need to be able to plan and execute trajectories among humans and other robots while pursuing high-level objectives such as safety [Hentzen et al., 2018] or energy consumption [Cabreira et al., 2018]. In such a setting, Model Predictive Control (MPC) is emerging as a suitable technology to address optimal motion planning and control in a unified manner, being able to generate feasible trajectories that optimize a given objective while naturally incorporating restrictions such as actuation limitations or safety constraints. However, its application to aerial robots is still the subject of active research, mainly due to the impact of the following challenges:

1. The high mass/power ratio of electric batteries places efficiency as a primary requisite for autonomy [Mulgaonkar et al., 2014].

2. Energy-efficient computers have limited computational resources, bounding the maximum complexity that can be tackled in real-time.

3. Safe navigation in dynamic environments require real-time algorithms that are able to make navigation decisions in the milisecond range.

4. Accurate motion planning involves complex models that are numerically expensive to simulate.

5. The fast dynamics and real-time requirements of aerial robots impose strong limits on the number of planning steps, limiting the prediction horizon and leading to reactive behaviors that are prone to local minima.

6. Uncertainty is present in the whole system, from measurements to predictions. Accounting for uncertainty in optimal motion planning is highly intractable. Therefore, strong assumptions need to be made for real-time applications and even then, it is hard to provide safety guarantees.

7. Uncertain dynamic obstacles such as pedestrians are difficult to predict and pose a major challenge to current map representations, which widely focus on static environments.

Thus, our objective in this thesis is to develop a real-time optimal motion planning and control approach that addresses these challenges and present different contributions on performance, autonomy and safety.

1.3 Significance of the study

This study encompasses different contributions to the field of optimal motion planning and control applied to aerial robots. An overview of these contributions follows:

1. Design and development of a model predictive control approach for real-time optimal motion planning and control in the presence of static obstacles.

2. Parametric software architecture for online re-parametrization of the model predictive control approach to different platforms and obstacle configurations.

3. Extension of the model predictive control algorithm that is able to account for obstacle dynamics and a wider variety of obstacle shapes, demonstrating its collision avoidance capabilities with multiple dynamic obstacles.

4. Mathematical proof of a novel differentiable bound for disjunctive linear-Gaussian chance-constraints, reducing conservatism and computation time when evaluating the risk of collision with dynamic obstacles.

5. A novel approach for real-time optimal motion planning and control under uncertainty. Our approach is able to provide sub-optimal trajectories with safety guarantees in environments populated with multiple dynamic obstacles such as pedestrians.

6. An infinite-horizon model predictive control approach that prioritizes near-future events while relaxing the pursuit of long-term objectives, showing considerable improvements on tracking performance and constraint satisfaction. As a result, our approach is able to generate safer trajectories that anticipate better the avoidance maneuvers on static and dynamic environments under uncertainty.

1.4 Outline of the thesis

The outline of this thesis is structured as follows: Chapter 2 provides the necessary background on optimal motion planning and control. It provides the problem statement and different alternatives to attack it, justifying the chosen one. Chapter 3 introduces the fundamental of quadrotors, its mathematical model, flying principle and a suitable architecture to perform real-time optimal motion planning and control. Chapter 4 presents our first contribution on optimal motion planning and control in static environments. It includes a real-time application on an aerial robot, describing the methodology, implementation, software architecture and experimental validation. In Chapter 5, we extend our previous work to dynamic obstacles, incorporating their prediction into the optimal control problem and different experiments demonstrating its suitability on crowded scenarios. In Chapter 6, we consider the problem of optimal motion planning and control with non-cooperative moving obstacles with uncertain localization, model and disturbances in the form of additive Gaussian noise. Here, we develop a nonlinear differentiable bound on the probability of collision with multiple obstacles to design and implement a real-time optimal motion planning and control approach with safety guarantees. The results are validated through mathematical proof, simulations and real experiments with pedestrians. In Chapter 7 we develop an infinite-horizon optimal motion planning and control to further extend the prediction horizon and improve the performance and safety of avoidance maneuvers. The results are validated through simulations and real experiments with pedestrians. Finally, in Chapter 8 we outline the conclusions of this work and promising lines of research to be addressed in the future.

Part I

Background

Chapter 2

Optimal Motion Planning and Control

Optimal motion planning and control of autonomous robots deals with the problem of planning and executing a sequence of actions that steers the robot from its initial state to a goal state. To be optimal, these actions must be chosen by minimizing a given cost function, which usually represents energy, time or path length. This chapter provides the necessary background to formulate this problem and the main strategies that can be employed to solve it.

2.1 Problem Statement

The motion of material bodies has been widely studied and accurately predicted by the field of classical mechanics [Goldstein et al., 2002]. In such a setting, the instantaneous configuration of a robotic system is described by a set of generalized coordinates $q = [q_1 \ \dots \ q_{n_q}]$. Each state corresponds to a particular point in a differentiable manifold $\mathcal{Q}$ known as the configuration space, which spans all the possible configurations that the system can assume. Then, a path is defined as a geometric curve ϕ in the configuration space rendering a continuous sequence of configurations as follows:

$$\begin{array}{rccl} \phi: & [s_0, s_f] \subset \mathbb{R} & \rightarrow & \mathcal{Q} \\ & s & \mapsto & q := \phi(s) \end{array} \tag{2.1}$$

When such a curve has a time domain, the path is called trajectory θ, mathematically defined as follows:

$$\begin{array}{rccl} \theta: & [t_0, t_f] \subset \mathbb{R}_{>0} & \rightarrow & \mathcal{Q} \\ & t & \mapsto & q := \theta(t) \end{array} \tag{2.2}$$

For convenience, we will employ the common abuse of notation $q(s)$ and $q(t)$ to refer to the path and the trajectory respectively. In general, the trajectory of any deterministic physical system is given by Hamilton's equations, which are equivalent to a set of $2n_q$ first-order ordinary differential equations [Goldstein et al., 2002]. In robotics, we explicitly separate the coordinates describing its state $x(t) \in \mathcal{X}$ and the ones representing its control inputs $u(t) \in \mathcal{U}$ to define the system dynamics as follows:

$$\dot{x}(t) := \frac{d}{dt}x(t) = f(x(t), u(t)) \tag{2.3}$$

Most robotic systems are expected to perform tasks in a workspace that is populated by physical objects, which represent obstacles to their motion. Therefore, it is convenient to define the free space $\mathcal{F}(t) \subseteq \mathcal{X}$ as the subspace of the state space for which the robot is collision-free, which is subject to change over time in the presence of moving obstacles. Thus, optimal motion planning involves finding the control trajectory $u^*(t)$ that safely drives the robot from an initial state x_I to a terminal set $\mathcal{X}_T$ while minimizing a cost functional $J(x(t), u(t))$ such as time or energy consumption. Mathematically, this problem can be encoded as an Optimal Control Problem (OCP) of the form:

$$\min_{x(\cdot),u(\cdot)} J(x(t), u(t)) = g_T(x(T)) + \int_0^T g(x(t), u(t))dt \tag{2.4a}$$

subject to:

$$x(0) = x_I \tag{2.4b}$$
$$\dot{x}(t) = f(x(t), u(t)) \tag{2.4c}$$
$$x(t) \in \mathcal{F}(t) \tag{2.4d}$$
$$u(t) \in \mathcal{U} \tag{2.4e}$$
$$x(T) \in \mathcal{X}_T \tag{2.4f}$$

In general, an analytical solution of (2.4) doesn't exist, and the OCP needs to be approximated numerically. There exist a variety of methods to numerically solve continuous time OCPs. What all approaches have in common is that at one point, the infinite-dimensional problem needs to be discretized. One family of methods first formulates what is known as the Hamilton-Jacobi-Bellman (HJB) equation, a partial differential equation for the value function, which depends on both state space and time, and then discretizes and solves it. Unfortunately, due to the "curse of dimensionality", this approach is only applicable in practice to systems with small state dimensions, or to the special case of unconstrained linear systems with quadratic costs [Rawlings and Mayne, 2009].

A second family of methods, the indirect methods, first derive optimality conditions in continuous time by algebraic manipulations that use similar expressions as the HJB equation; they typically result in the formulation of a boundary-value problem (BVP), and only discretize the resulting continuous time BVP at the very end of the procedure. One characterizes the indirect methods often as "first optimize, then discretize". A third class of methods, the direct methods, first discretizes the continuous time OCP, to convert it into a finite-dimensional optimization problem, which can be solved by tailored algorithms from the field of numerical optimization. The direct methods are often characterized as "first discretize, then optimize". In this thesis, we employ direct methods due to their ability to naturally generate the discrete-time approximation to the OCP, maintaining a closer correspondence to its real-time implementation on embedded computers.

2.2 Direct Optimal Control

In general, Direct optimal control parametrizes the continuous-time OCP (2.6) into a finite-dimensional nonlinear program (NLP), which can then be solved by tailored algorithms from the field of numerical optimization [Nocedal and Wright, 2006]. To that aim, the control trajectory $u(t)$ needs to be parametrized by a finite set of decision variables. Even though an ample range of polynomials and basis functions can be employed, we consider a piecewise constant control parametrization due to its simplicity and better correspondence to the discrete-time implementation of optimal control algorithms [Rawlings and Mayne, 2009]. To formalize it, let us divide the time horizon $[0, T]$ into N subintervals $[t_k, t_{k+1}]$ with $0 = t_0 < t_1 < \cdots < t_N = T$ and set:

$$u(t) := u(t_k) = u_k \quad t \in [t_k, t_{k+1}] \tag{2.5}$$

Therefore, the discrete-time control trajectory can be described by a finite set of parameters $U_N := \{u_0, \ldots, u_{N-1}\}$. Analogously, by defining $x_k := x(t_k)$, the discrete-time trajectory can be described by the finite set $X_N :=$

$\{x_0, \ldots, x_N\}$. As a result, the discrete-time OCP follows:

$$\min_{X_N, U_N} \; g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k) \tag{2.6a}$$

subject to:

$$x_0 = x_I \tag{2.6b}$$
$$x_{k+1} = f_k(x_k, u_k) \tag{2.6c}$$
$$x_k \in \mathcal{F}_k \tag{2.6d}$$
$$u_k \in \mathcal{U} \tag{2.6e}$$
$$x_N \in \mathcal{X}_T \tag{2.6f}$$

where $f_k(x_k, u_k)$ and $g_k(x_k, u_k)$ are numerical approximations to the system dynamics and the cost at each stage. The interplay between the numerical simulation of the dynamics and the optimization of the optimal control problem lead to two main families of methods: Shooting and direct transcription methods.

All shooting methods employ an embedded integration algorithm to compute the discrete-time dynamics $f_k(x_k, u_k)$ for $k = 0, \ldots, N-1$. In direct multiple shooting, the integration procedure is executed in parallel to obtain $f_k(x_k, u_k)$ with initial conditions given by X_N and U_N. Then, the continuity of the trajectory is ensured by the numerical optimization algorithm with the constraint (2.6c). Alternatively, the single-shooting method embeds the constraints (2.6b) and (2.6c) into the cost function, leading to an optimization problem that solely depends on x_0 and U_N. This approach highly reduces the number of variables at the expense of a sequential application of the numerical integration algorithm. Even though multiple-shooting algorithms involve a higher number of decision variables, they offer higher convergence rates at similar computation time when the sparsity structure of the problem is exploited [Albersmeyer and Diehl, 2010]. Therefore, multiple shooting remains dominant with respect to single shooting for real-time optimal control applications [Zhu and Alonso-Mora, 2019, Torrente et al., 2021].

Direct transcription methods follow the same discretization structure as multiple shooting: i.e. the state variables are kept for parallel integration. The main difference is that, instead of outsourcing the numerical simulation, the integration scheme is encoded as equality constraints of the optimization problem. Therefore they require a much finer grid with additional decision variables that store the state derivatives through intermediate steps of the integration process. Direct transcription methods often use implicit integration rules, since they offer higher orders of accuracy for the same number of state discretization variables, and come with better stability properties for stiff

systems. Probably, the most popular class of direct transcription methods are direct collocation methods. Even though they require a higher computational cost, recent efforts have significantly reduced the computational gap between multiple shooting and direct collocation methods [Quirynen et al., 2015a].

Regardless of the method employed to discretize the OCP (2.6) in time, a determinant factor is the explicit representation of the free space $\mathcal{F}_k$. Essentially there are two broad categories: discrete and continuous spaces. This choice determines the nature of the optimization problem, leading to benefits and shortcomings that are essential for a given application. Therefore, we will briefly discuss this topic to justify our choice taken in this thesis.

2.3 Planning in Discrete State Spaces

There are many situations where the state and control are naturally discrete and take a finite number of values. Such problems are often conveniently described as an acyclic graph specifying for each state x_k the possible transitions to the next states x_{k+1}, as shown in Fig. 2.3. The nodes of the graph correspond to states x_k and the arcs correspond to state-control pairs (x_k, u_k). Each arc with start node x_k correspond to a choice of a single control $u_k \in U_k(x_k)$ and has as an end node the next state $x_{k+1} = f_k(x_k, u_k)$. The cost of an arc is defined as $g_k(x_k, u_k)$. To handle the final stage, an artificial terminal node t is added. Each state x_N is connected to the terminal node t with an arc of cost $g_N(x_N)$. Note that control sequences $\{u_0, \ldots, u_{N-1}\}$ correspond to paths originating at the initial state x_0 and terminating at one of the nodes corresponding to the final stage N. If we view the cost of an arc as its length, we see that the problem is equivalent to finding the shortest path from the initial node s to the terminal node t.

These problems often arise in robotics when a discrete representation of the free space is employed, such as an occupancy grid or a roadmap, as illustrated in Fig. 2.3. Generally, they can be solved to a global optimum trough dynamic programming algorithms such as A^*. However, they need to precompute the optimal cost-to-go $J_k^*(x_k)$ for all x_k and k, defined as follows:

$$J_k^*(x_k) = \min_{u_k \in U(x_k)} \Big[g_k(x_k^*, u_k) + J_{k+1}^*(f_k(x_k, u_k)) \Big] \tag{2.7}$$

where $J_N^*(x_N) = g_N(x_N)$. As a result, this problem becomes rapidly untractable for high-dimensional state spaces [Bertsekas, 2019], known as the curse of dimensionality. Therefore, most practical approaches reduce the dimensionality of the problem by considering a small subset of the state space.

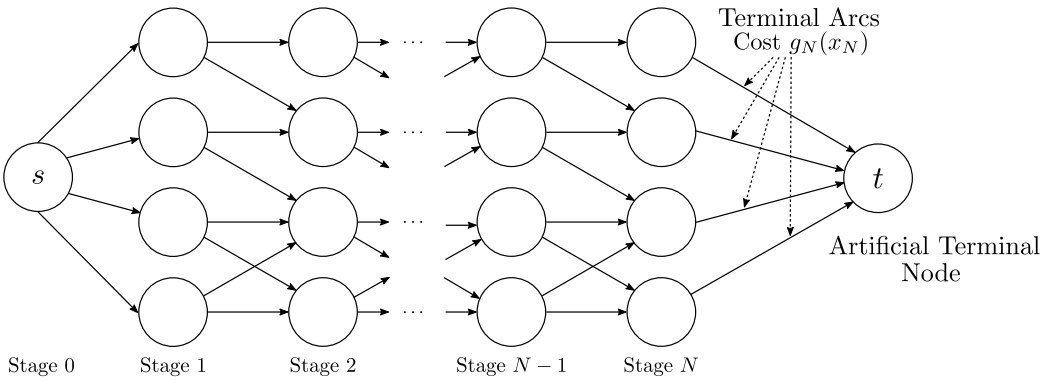


Figure 2.1: Transition graph for a deterministic finite-state system. Nodes correspond to states x_k. Arcs correspond to state-control pairs (x_k, u_k). An arc (x_k, u_k) has start and end nodes x_k and $x_{k+1} = f_k(x_k, u_k)$, respectively. We view the cost $g_k(x_k, u_k)$ of the transition as the length of this arc. The problem is equivalent to finding a shortest path from initial node x_0 to terminal node t.

For instance, [Oleynikova et al., 2018] construct a Generalized Vornoi Diagram that only consider a one-voxel thin skeleton of the free space, which allow them to perform discrete planning on cluttered spaces in the milisecond range. In [Strub and Gammell, 2020] random samples of the configuration space are employed to build a connectivity graph that reaches a specific goal.

Even though there exist real-time approaches leveraging discrete optimal planning, these map representations do not, in general, have explicit facilities for identifying and distinguishing between permanent obstacles (e.g., walls, doorways, etc.) and transient obstacles (e.g., humans, shipping packages, etc.), which remains an open challenge in robotics research [Siegwart et al., 2011].

2.4 Planning in Continuous State Spaces

The configuration space of mobile robots are naturally described in continuous spaces such as $\mathbb{R}^n$. Generally, the space occupied by obstacles is represented as a set of constraints on the free space which, in general, disrupts its convexity. The choice on the type of constraints determines the nature of the resulting optimization problem and therefore, its performance. There are two main strategies in the literature to encapsulate an obstacle's space: Using a convex polyhedron (e.g. a cuboid) [Blackmore et al., 2011, Lefkopoulos and Kamgarpour, 2019], or a smooth surface (e.g. an ellipsoid) [Castillo-Lopez

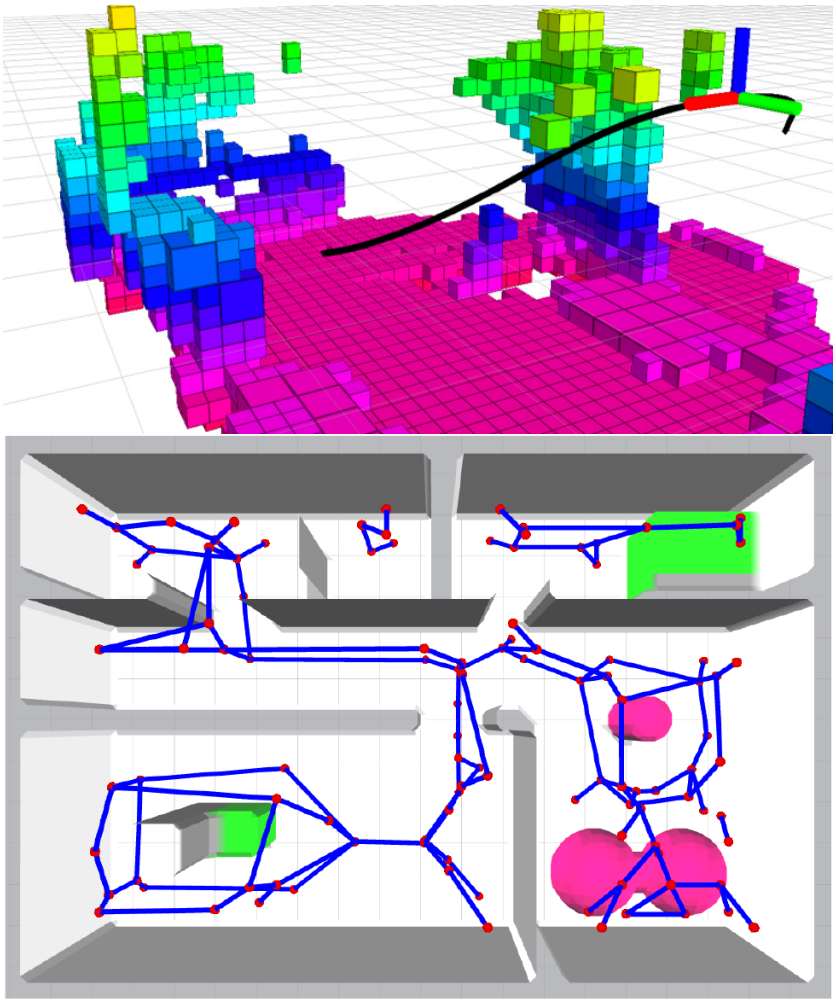


Figure 2.2: Discrete free space representations [Oleynikova et al., 2017]

et al., 2018, Kamel et al., 2017a, Zhu and Alonso-Mora, 2019].

2.4.1 Polyhedral obstacles

A polyhedral obstacle is encoded as a disjunction of linear inequality constraints. This represents logical OR relations between the infinite planes that define each face of the polyhedron. Thus, an obstacle $\mathcal{O}_k$ at stage k with N_f faces can be encoded as follows:

$$\mathcal{O}_k \iff \bigvee_{i=1}^{N_f} a_{i,k}^T x_k \leq b_i \tag{2.8}$$

where $a_i, b_i \in \mathbb{R}^{n_x}$. Thus, for N_o obstacles, the free space follows:

$$\mathcal{F}_k := \left\{ x_k \in \mathcal{X} : \bigwedge_{i=1}^{N_o} \bigvee_{j=1}^{N_f} a_{i,j,k}^T x_k \geq b_i \right\} \tag{2.9}$$

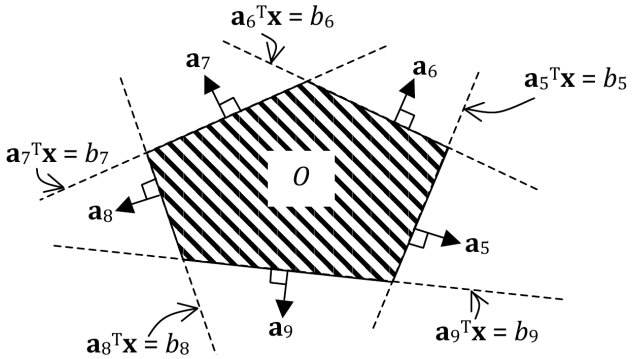


Figure 2.3: Polyhedral obstacle $\mathcal{O}$ encoded as a disjunction of linear inequality constraints [Blackmore et al., 2011].

The resulting disjunctive optimization problem can be solved to global optimality using existing branch-and-bound techniques [Balas, 2018]. This problem has a relatively high computational cost that grows exponentially with the number of obstacles and faces [Balas, 2018, Ono et al., 2013]. Even though recent efforts show promising improvements on computational efficiency, over-conservatism and probabilistic guarantees [Blackmore et al., 2011, Ono, 2012, Ono et al., 2013, Lefkopoulos and Kamgarpour, 2019], their computational cost is still too elevated to achieve real-time performance with multiple obstacles.

2.4.2 Smooth obstacles

Alternatively, an obstacle can be bounded by a single smooth surface (sphere, cylinder, ellipsoid, etc.), which can be expressed algebraically as a p-norm with metric M as follows

$$\mathcal{O}_k \iff \|x_k - a_k\|_M^p \leq 1 \tag{2.10}$$

where $p \in \mathbb{N}_{>0}$ and M is a positive definite diagonal matrix. Thus, the free space can be encoded as follows:

$$\mathcal{F}_k := \left\{ x_k \in \mathbb{R}^{n_x} : \bigwedge_{i=1}^{N_o} \|x_k - a_{i,k}\|_{M_i}^p \geq 1 \right\} \tag{2.11}$$

This results in a comparatively low-dimension NLP, which can be solved efficiently by gradient-based solvers [Houska et al., 2011]. Even though this solution cannot guarantee global optimality, its reduced computational cost makes this strategy to be widely adopted in most time-critical motion planning tasks, such as model predictive control for aerial robots [Zhu and Alonso-Mora, 2019, Castillo-Lopez et al., 2018, Kamel et al., 2017a].

2.5 Planning under Uncertainty

In practice, the behavior of real systems deviate from their mathematical model due to different sources of uncertainty [Grüne and Pannek, 2017]. For instance, the state of a robot is estimated from noisy and biased data coming from different sensors such as GPS, gyroscopes and accelerometers [Sanchez-Lopez et al., 2016]. In addition, accurately perceiving and predicting the location of dynamic obstacles like pedestrians is still the subject of active research [Rudenko et al., 2020]. Thus, enabling uncertainty awareness in optimal motion planning and control algorithms in an essential task to ensure safe operations of autonomous robots, which is the subject of active research [Quan et al., 2020].

There are two main strategies to model uncertainty in optimal motion planning and control algorithms: set-bounded models and probabilistic models. Optimal planning under set-bounded uncertainty has received a great deal of attention during the last decades due to its simplicity and computational efficiency [Jalali and Nadimi, 2006]. However, it relies on deterministic worst-case realizations of these uncertainties, leading to over-conservatism and a dramatic reduction of the free space [Blackmore et al., 2011]. In contrast, modeling uncertainty through a probabilistic framework, has shown

to adequately characterize real-world systems while overcoming the inherent over-conservatism of set-bounded uncertainty models [Mesbah, 2016].

To cast our optimal motion planning problem (2.6) into a stochastic form, additional parameters $w_k \in \mathbb{R}^{n_w}$ and $v_k \in \mathbb{R}^{n_v}$ need to be injected into the problem formulation to capture uncertainties about the system dynamics and the free space respectively. These parameters are unknown at current and future time instants, but have been suitably characterized by a known probability distribution $P_{w,k}$ and $P_{v,k}$ respectively. As a result, the expected value of the cost is minimized, while enforcing the robot to stay within the free configuration space in a probabilistic sense with confidence level $1 - \alpha$ as follows:

$$\min_{X_N, U_N} \quad \mathbb{E}\left[g_N(x_N, w_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right] \tag{2.12a}$$

subject to:

$$x_0 \sim P_{x_I} \tag{2.12b}$$

$$x_{k+1} = f_k(x_k, u_k, w_k) \tag{2.12c}$$

$$u_k \in \mathcal{U} \tag{2.12d}$$

$$\mathcal{P}\left(\bigwedge_k x_k \in \mathcal{F}_k(v_k) \right) \geq 1 - \alpha \tag{2.12e}$$

$$\mathcal{P}\left(x_N \in \mathcal{X}_T \right) \geq 1 - \alpha_T \tag{2.12f}$$

$$w_k \sim P_{w,k} \quad v_k \sim P_{v,k} \tag{2.12g}$$

Depending on the application, different challenges can be drawn around the OCP (2.12):

- The definition of the cost function (2.12a) that encodes the correct task can be challenging in practice [Finn et al., 2016].
- High model fidelity may render complex dynamics [Todorov et al., 2012], or require online model learning [Hewing et al., 2018].
- In general, the evaluation of chance constraints is untractable, especially for problems with multiple non-convex obstacles [Blackmore et al., 2011].

In this thesis we limit our scope to the real-time evaluation of chance constraints to provide a unified framework for optimal motion planning and control with safety guarantees. This problem has been the subject of active

research during the last decade [Blackmore et al., 2011,Lefkopoulos and Kamgarpour, 2021], providing numerous theoretical results validated mathematically and in simulations. However, they employ polyhedral representation of obstacles which zeroes its applicability to fast real-time applications with multiple obstacles, as discussed in the previous section. Therefore, most approaches solve a relaxed version of the problem [Kamel et al., 2017a,Zhu and Alonso-Mora, 2019,Lew et al., 2020], being robust to disturbances but failing to provide safety guarantees. The main contribution of this thesis attacks this problem by providing an efficient and scalable approximation of [Blackmore et al., 2011] that presents an adequate balance between tractability and performance for real-time optimal motion planning and control of aerial robots with multiple obstacles.

Chapter 3

Fundamentals of Quadrotors

3.1 Motor model

A quadrotor is an aerial robot driven by four identical propellers. When the aerial robot is flying, each rotor has an angular speed ω_i and produces an upwise vertical force F_i and a reactive moment M_i that can be modeled as follows [Valavanis and Vachtsevanos, 2014]:

$$F_i = k_F \omega_i^2 \qquad M_i = k_M \omega_i^2 \tag{3.1}$$

where k_F and k_M are positive real constants. The angular moment can also be expressed as $M_i = \gamma F_i$, where $\gamma = k_M / k_F$. To compensate the reactive moment produced by each propeller, they rotate in alternate directions as shown in figure 3.1.

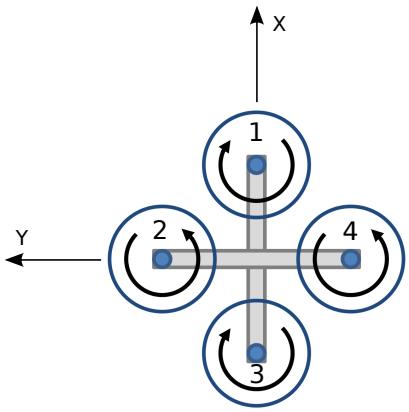


Figure 3.1: Scheme of a quadrotor propeller configuration

3.2 Flying principle

Consider a quadrotor keeping a stable position suspended in the air (hovering state). Such a state can be described by Newton's equations of motion as follows:

$$\begin{aligned} \sum F_z &= F_1 + F_2 + F_3 + F_4 - mg = 0 \\ \sum M_x &= L(F_2 - F_4) = 0 \\ \sum M_y &= L(F_3 - F_1) = 0 \\ \sum M_z &= \gamma F_1 - \gamma F_2 + \gamma F_3 - \gamma F_4 = 0 \end{aligned} \tag{3.2}$$

Therefore, a hovering state implies that every rotor has the same angular velocity and produces an up-wise force given by $F_i = mg/4$. To produce a forward movement in the x axis M_y has to be positive. To that aim, F_3 is incremented and F_1 decremented in the same proportion to maintain the same height with $M_y > 0$. The opposite modifications are needed to produce a backward movement. To produce a sidewards movement in the y axis M_x has to be positive. Then, F_2 is incremented and F_4 decremented in the same proportion to maintain the height with $M_x > 0$. The opposite modifications are needed to produce a left-sidewards movement. To change the orientation, an angular acceleration through the z axis is needed. To make a counter-clock-wise rotation, M_z has to be positive. To that aim, M_1 and M_3 need to be incremented while reducing M_2 and M_4 in the same proportion to maintain the same height with $M_z > 0$. The opposite modifications are needed to produce a clock-wise turn.

3.3 Quadrotor Kinematics

The kinematics of an aerial robot can be obtained by using four right-handed reference frames, as described in equation 3.3. The inertial frame A is settled as the canonical reference frame. E and F are intermediate frames while the body frame B is attached to the center of mass (C) of the quadrotor and aligned to its principal axes of inertia as shown in figure 3.2.

$$\begin{aligned} \{A\} &= \{a_1, a_2, a_3\} \\ \{E\} &= \{e_1, e_2, e_3\} \\ \{F\} &= \{f_1, f_2, f_3\} \\ \{B\} &= \{b_1, b_2, b_3\} \end{aligned} \tag{3.3}$$

As defined in equation 3.4, orientation of the body frame B is obtained through the rotation of the inertial frame A using the ZXY Euler convention.

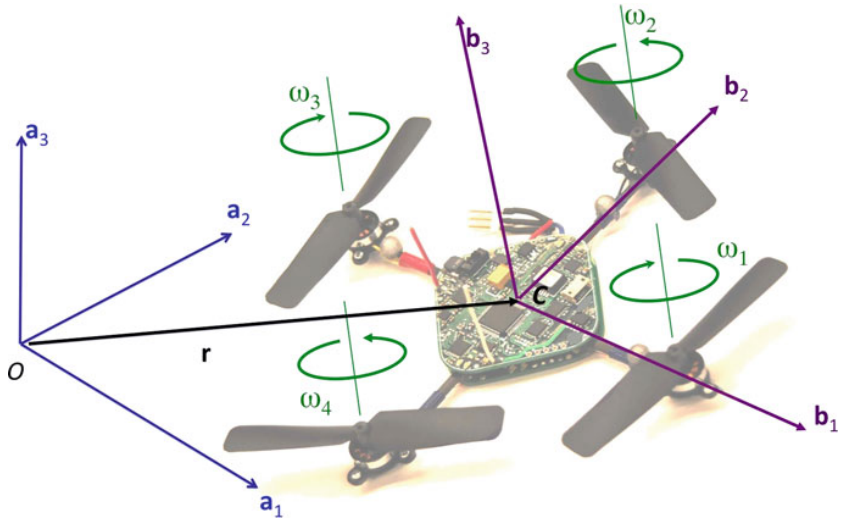


Figure 3.2: The body-fixed frame and the inertial frame provided by [Powers et al., 2015].

The three rotations are applied with yaw (ψ), roll (ϕ) and pitch (θ) angles over the axis a_3, e_1 and f_2 respectively, as shown in Fig. 3.3.

$$\{A\} \xrightarrow[R_z(\psi)]{{}^{A}R_{E}} \{E\} \xrightarrow[R_x(\phi)]{{}^{E}R_{F}} \{F\} \xrightarrow[R_y(\theta)]{{}^{F}R_{B}} \{B\} \tag{3.4}$$

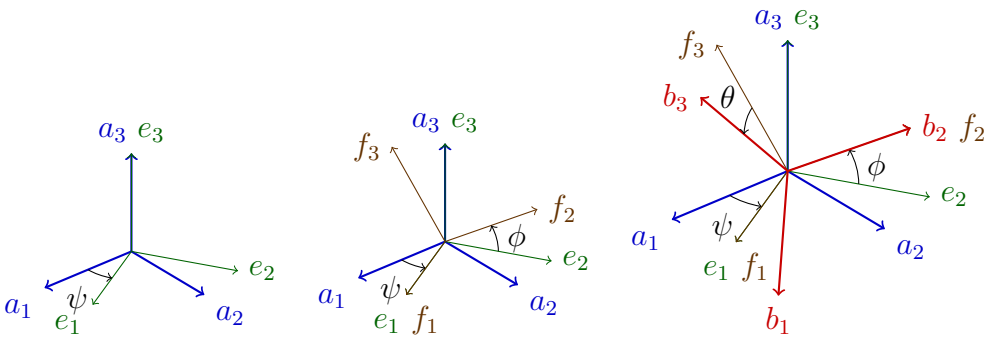


Figure 3.3: The different frames involved in the quadrotor model.

As a result, the rotation of the body frame with respect to the inertial frame ${}^{A}R_{B}$ is given by:

$${}^{A}R_{B} = {}^{A}R_{E}\,{}^{E}R_{F}\,{}^{F}R_{B} = \begin{bmatrix} c\theta c\psi - s\phi s\theta s\psi & -c\phi s\psi & s\theta c\psi + s\phi c\theta s\psi \\ c\theta s\psi + s\phi s\theta c\psi & c\phi c\psi & s\theta s\psi - s\phi c\theta c\psi \\ -c\phi s\theta & s\phi & c\phi c\theta \end{bmatrix} \tag{3.5}$$

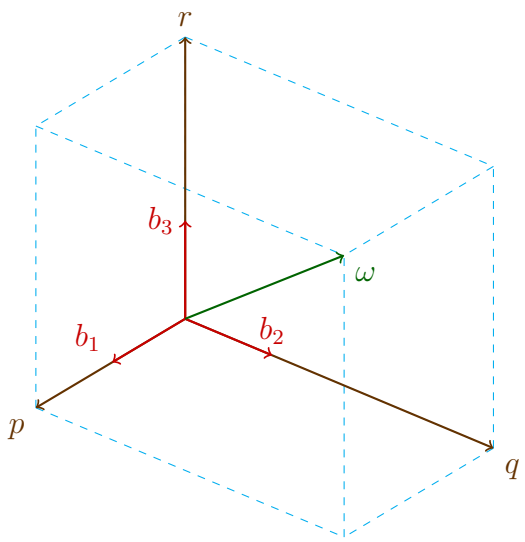


Figure 3.4: Angular velocity and its components in the body frame B

Let p, q and r be the components of the angular velocity in the body frame as shown in figure 3.4 and equation 3.6.

$$^{B}\omega = p\hat{b}_1 + q\hat{b}_2 + r\hat{b}_3 \tag{3.6}$$

As roll pitch and yaw angles are defined in different frames, its derivatives are related to $^{B}\omega$ as follows:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = {}^{B}R_E \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + {}^{B}R_A \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} c\theta & 0 & -c\phi s\theta \\ 0 & 1 & s\phi \\ s\theta & 0 & c\phi c\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{3.7}$$

which is invertible for $-\pi/2 < \phi < \pi/2$ as follows:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} c\theta & 0 & s\theta \\ s\theta t\phi & 1 & -c\theta t\phi \\ -s\theta(c\phi)^{-1} & 0 & c\theta(c\phi)^{-1} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.8}$$

3.4 Quadrotor Dynamics

Let the position of the center of mass of a quadrotor be denoted by $r \in \mathbb{R}^3$. Then, its translational motion is determined by Newton's equations of motion as follows:

$$\dot{r} = v \tag{3.9a}$$

$$\dot{v} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \frac{1}{m} {}^{A}R_{B} \begin{bmatrix} 0 \\ 0 \\ F_1 + F_2 + F_3 + F_4 \end{bmatrix} \tag{3.9b}$$

Assuming that B is rigidly attached to the quadrotor's center of mass and is aligned to its the principal axes of inertia, its inertial tensor can be represented by the diagonal matrix $I_c = diag(I_1, I_2, I_3)$. Then, its rotational motion is determined by Euler's rotation equations as follows:

$$M_{ext} = I_c {}^{B}\dot{\omega} + {}^{B}\omega \times (I_c {}^{B}\omega) \tag{3.10}$$

which leads to:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = I_c^{-1} \left(\begin{bmatrix} L(F_2 - F_4) \\ L(F_3 - F_1) \\ M_1 - M_2 + M_3 - M_4 \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times I_c \begin{bmatrix} p \\ q \\ r \end{bmatrix} \right) \tag{3.11}$$

From equations (3.9) (3.8) (3.11), the state space model of the system $\dot{x} = f(x, u)$ is trivially obtained by defining the state $x \in \mathbb{R}^{12}$ and controls $u \in \mathbb{R}^4$ as follows:

$$x = [r \ v \ \phi \ \theta \ \psi \ p \ q \ r]^T \tag{3.12}$$

$$u = \begin{bmatrix} F_1 + F_2 + F_3 + F_4 \\ L(F_2 - F_4) \\ L(F_3 - F_1) \\ M_1 - M_2 + M_3 - M_4 \end{bmatrix} \tag{3.13}$$

3.5 Quadrotor Control Architecture

Optimal motion planning and control with the full quadrotor dynamics has been shown to outperform traditional trajectory tracking methods [Bicego et al., 2020]. However, the fast real-time requirements of aerial robots leads to high-frequency sampling and short prediction horizons ($T = 1s$). As we show in Chapter 7, short prediction horizons constrains the algorithm's ability to anticipate avoidance maneuvers, leading reactive maneuvers that are prone to local minima. To target longer planning horizons, we employ a hierarchical optimal motion planning and control architecture where low-level regulation is outsourced to a commercial autopilot such as [Meier et al., 2015] that receives linear and angular velocity commands, as shown in Fig. 3.5. In this setting, the optimal motion planner is able to address higher-level optimal motion planning and control, with real-time requirements in the order of tens of miliseconds instead of few microseconds [Meier et al., 2015].

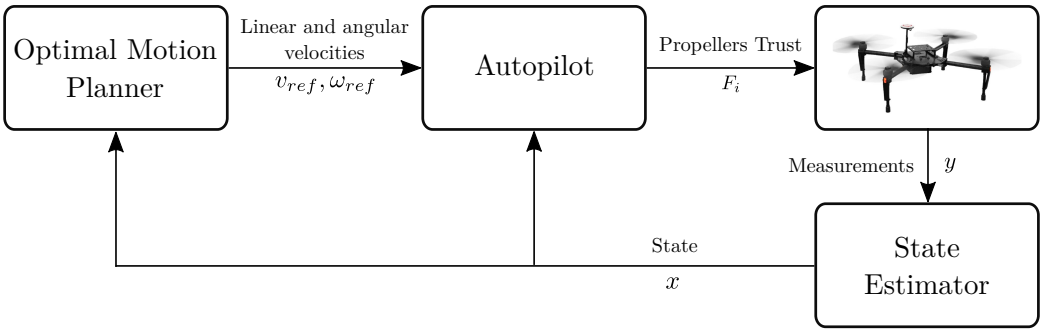


Figure 3.5: Hierarchical optimal motion planning and control architecture.

Part II

Contributions

Chapter 4

Optimal Motion Planning and Control in Static Environments.

Planning and executing collision-free trajectories in a high-dimensional workspace populated by obstacles is one of the main challenges in the quest of autonomy for aerial robots. While most approaches are able to provide feasible solutions, the energetic and computational constraints of aerial robots place a great importance to the optimality of such behaviors. To address this problem, this chapter presents a Model Predictive Control algorithm that is able to perform optimal motion planning and control in a unified manner. To that aim, obstacles are bounded by ellipsoidal surfaces that are encoded as constraints to the optimal control problem. The key benefit of our approach lies in its ability to achieve real-time performance in optimal motion planning and control in three-dimensional workspaces populated with obstacles. The key to its tractability lies in the combination of a lightweight experimental model with an state-of-the-art algorithm for embedded optimization and a parametric software architecture that minimizes its computational footprint. Experimental results in simulation and in a commercial robot show its performance in the microsecond range, demonstrating its suitability for decentralized autonomous navigation. The contributions shown in this chapter are published in our conference paper [Castillo-Lopez et al., 2017].

4.1 Introduction

Aerial Robots, also known as Unmanned Aerial Vehicles (UAVs), are now a subject of active research due to its multiple applications such as traffic monitoring, load transportation and manipulation or search-and-rescue oper-

ations [Valavanis and Vachtsevanos, 2014]. These applications often demand precise trajectories in a workspace populated by obstacles, increasing the importance of safety in autonomous navigation. Traditionally, this problem has been addressed by a hierarchical combination of planning and control, where a path planner obtains a sequence of waypoints that the control algorithm follows blindly. The mismatch between planning and control has shown to generate suboptimal and even dangerous behaviors [Shim et al., 2003]. A survey of different motion planning techniques applied to aerial robots can be found in [Goerzen et al., 2010, Hoy et al., 2015].

Unlike traditional approaches, Model Predictive Control (MPC) is able to merge motion planning and control in a single optimal control problem. It makes explicit use of the system dynamics to predict feasible state and control trajectories that drive the robot from its initial state to a given target while minimizing a cost functional such as time or energy consumption over a prediction horizon [Maciejowski, 2002]. Since the horizon is shifted forward in time at each iteration, MPC is also known as Receding Horizon Control (RHC). Its ability to naturally consider safety restrictions and actuator saturations has made MPC an attractive field of research for high-performance control of complex systems. However, it involves a high computational cost that grows exponentially with the number of prediction steps [Stellato et al., 2018].

The great domain of potential applications of aerial robots, combined with its limited energetic autonomy, places optimal trajectory generation as a main challenge and the focus of active research. Due to recent advances on computational hardware and embeded optimization algorithms [Quirynen et al., 2014], model predictive control approaches on aerial robots have emerged during the last decade. In [Abdolhosseini et al., 2013] an efficient MPC control scheme for quadrotors is presented to perform 3D path tracking. In [Alexis et al., 2016] a robust predictive flight controller with obstacle avoidance capabilities is implemented. In [Andersson et al., 2016] a Bayesian Policy Optimization with MPC is developed to provide stochastic collision avoidance for quadrotors. In [Darivianakis et al., 2014] an Hybrid Predictive Controller is designed to interact phisically in inspection operations. Nonlinear Partial Enumeration with MPC is used in [Desaraju and Michael, 2016] to develop a fast MPC controller, tested in simulations. MPC is used in [Garimella and Kobilarov, 2015] for an aerial pick-and-place application with a manipulator attached to a quadrotor. Autonomous landing on a moving platform is developed in [Vlantis et al., 2015]. To avoid its computational cost in obstacle avoidance tasks, [Zhang et al., 2016] employs a deep neural network to approximate the closed-loop policy of an MPC algorithm. In [Dentler et al., 2016] a real-time model predictive position control is imple-

mented using soft distance functions to model obstacles. Finally, in [Bouffard et al., 2012] Learning Based MPC is used to catch balls in the air and correct the ground effect.

Previous MPC approaches encapsulate obstacle space with infinite planes or spheres, which consume a large amount of safe space, making it difficult to operate in complex environments. Instead, we propose the use of ellipsoids to find optimal trajectories when the workspace is populated by obstacles with different shapes. We develop a software architecture that is able to add, remove, translate and reshape obstacles dynamically. The use of a lightweight dynamical model, coupled with active-set methods for handling multiple obstacle constraints allow us to deploy an MPC approach that is able to safely avoid multiple obstacles with a control delay in the microsecond range. Our MPC algorithm is developed and empirically validated using a Motion Capture System and a low-cost quadrotor helicopter tele-operated by a laptop computer. The resulting approach drives the aerial robot autonomously through collision-free trajectories to reach a given pose or follow a waypoint path without the need of a local re-planner. Different experiments on a commercial quadrotor validate its effectiveness in trajectory tracking and sense-and-avoid tasks.

4.2 Methodology and Equipment

In this work, we employ the commercial quadrotor Parrot® AR.Drone 2.0, which is designed to be controlled remotely through WiFi. As shown in Figure 4.2, the aerial robot operates in a flight area of [4 3 3] m limited by nets. Reflecting balls attached to the quadrotor are used to determine its position and orientation using the OptiTrack® Motion Capture System.

A taylored solver for embedded execution of our MPC algorithm is developed in C++11 with ACADO Toolkit [Houska et al., 2013]. This solver is then wrapped and interfaced to the ROS Indigo middleware [Quigley et al., 2009], running the MPC process on a Lenovo Y50-70 laptop with Intel® i7-4710HQ CPU at 2.50 GHz and 8 GiB of DDR3 1600 MHz RAM with Ubuntu 14.04. Asynchronous inter-process communications with the robot and the motion capture system are implemented through a publisher/subscriber messaging pattern. The control architecture is implemented as shown in Fig. 4.2. The laptop computer running our MPC algorithm is connected to the Motion Capture System and the robot through Ethernet and WiFi respectively. The Motion Capture System publishes the position and orientation (pose) of the robot, to which the MPC process is subscribed. Then, for each pose received, our MPC algorithm obtains and publishes the optimal control input

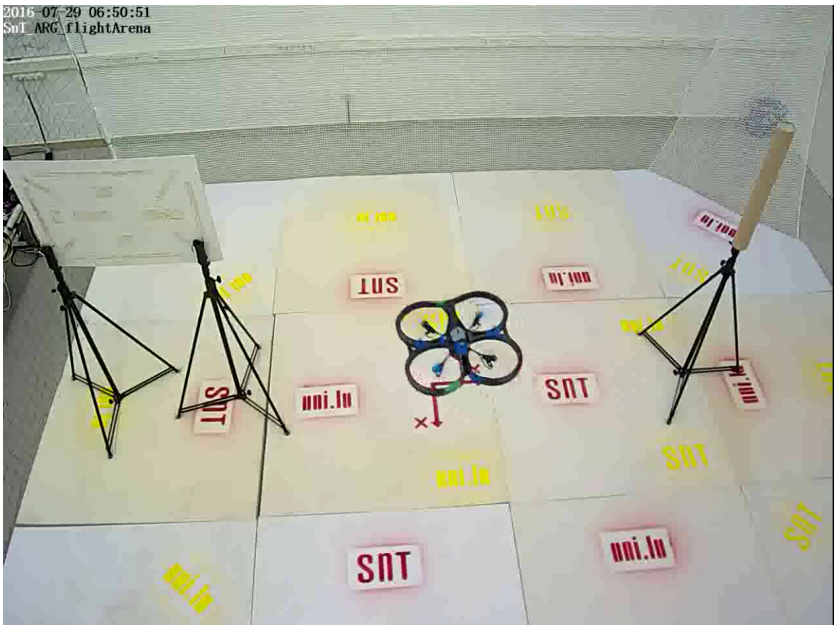


Figure 4.1: Experimental testbed composed by a flight area of [4 3 3] m limited by nets with two static obstacles and the commercial quadrotor Parrot® AR.Drone 2.0.

to the robot through the *snt_ardrone_driver* [Olivares-Mendez et al., 2014]. The MPC process is subscribed to an additional topic to which the desired pose can be published manually, or in an automated way to track a given path.

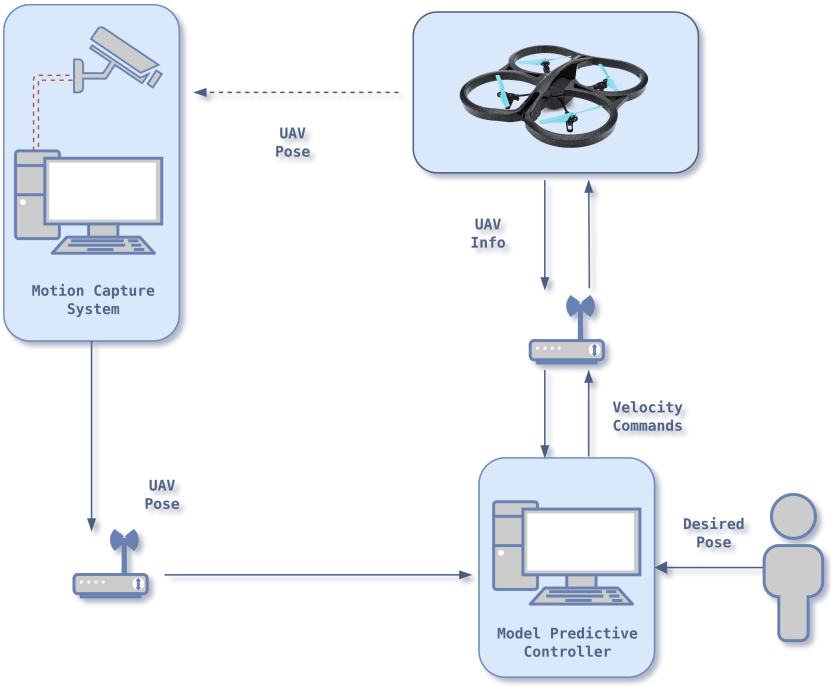


Figure 4.2: Control architecture. The Model Predictive Controller runs on a laptop connected to the Motion Capture System and the robot through Ethernet and WiFi respectively. The inter-process communications are implemented through a publisher/subscriber messaging pattern using the ROS Indigo middleware [Quigley et al., 2009]

4.3 Robot model and identification

The AR.Drone 2.0 quadrotor is equipped with an autopilot that ensures its stability while tracking the control input $u = [u_f \ u_s \ u_u \ u_h]^T$, corresponding to forward, sideward, upward and heading velocity commands with respect to the robot's hovering frame. The hovering frame is a body-fixed frame with zero roll and pitch angles, and a right-handed yaw angle ψ as shown in Fig. 4.3.

Since the autopilot control specifications of the UAV are unknown, an experimental First Order with Delay (FOD) model is proposed in equation

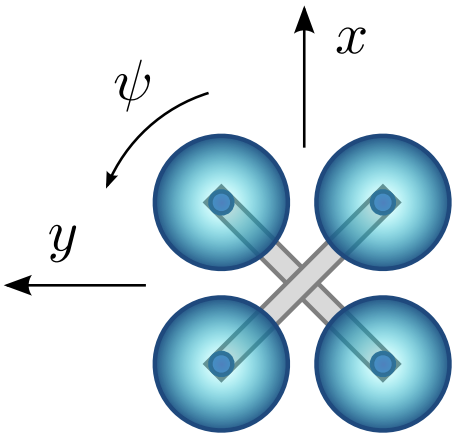


Figure 4.3: Hovering frame of the UAV

4.1 to relate each component of the input vector u to the corresponding velocity of the robot $v = [v_f\ v_s\ v_u\ v_h]^T$ in the hovering frame, where k_i and d_i and τ_i are the gain, the delay and the time constant respectively for each velocity/control pair (v_i, u_i).

$$\dot{v}_i(t) = (-v_i(t) + k_i u_i(t - d_i))/\tau_i \quad i = f, s, u, h \tag{4.1}$$

We consider the robot's state vector $x(t) = [p(t)\ \psi(t)\ v(t)]$ to be composed by its position $p(t) \in \mathbb{R}^3$ in a Cartesian world frame, the orientation of its hovering frame $\psi(t)$ with respect to the world frame, and the its velocity vector $v(t)$ in the hovering frame. Then, we render a first-order with delay state space model of the system by assuming $\psi(t) = \psi_0 := \psi(0)$ as follows:

$$\dot{x}(t) = Ax(t) + Bu(t - d) \tag{4.2}$$

where

$$A = \begin{bmatrix} & cos(\psi_0) & -sin(\psi_0) & 0 & 0 \\ & sin(\psi_0) & cos(\psi_0) & 0 & 0 \\ & 0 & 0 & 1 & 0 \\ & 0 & 0 & 0 & 1 \\ 0_{8\times4} & -\tau_f^{-1} & 0 & 0 & 0 \\ & 0 & -\tau_s^{-1} & 0 & 0 \\ & 0 & 0 & -\tau_u^{-1} & 0 \\ & 0 & 0 & 0 & -\tau_h^{-1} \end{bmatrix} \tag{4.3}$$

$$B = \begin{bmatrix} & \bar{\bar{0}}_{4\times 4} & & \\ k_f/\tau_f & 0 & 0 & 0 \\ 0 & k_s/\tau_s & 0 & 0 \\ 0 & 0 & k_u/\tau_u & 0 \\ 0 & 0 & 0 & k_h/\tau_h \end{bmatrix} \tag{4.4}$$

Then, we perform model identification by a classical step response tangent method ((4.6)). To that aim, a step signal is generated for each control input of our robot, while the motion capture system, is employed to record its corresponding velocity, as shown in Fig. 4.4. The gain k_i, time constant τ_i and delay d_i are obtain from the step response as follows:

$$k_i = \frac{v_i(\infty)}{u_i(\infty)} \qquad \tau_i = \frac{3}{2}(t_i^{63} - t_i^{28}) \qquad d_i = t_i^{63} - \tau_i \tag{4.5}$$

where $v_i(t_i^{63}) = 0.63 v_i(\infty)$ and $v_i(t_i^{28}) = 0.28 v_i(\infty)$. Thus, we provided an experimental model that is able to fairly reproduce the system dynamics at a higher level of abstraction and lower complexity, as shown in Fig. 4.4. As shown later, the reduced complexity of the model, allows our algorithm to perform real-time motion planning and control in the microsecond range.

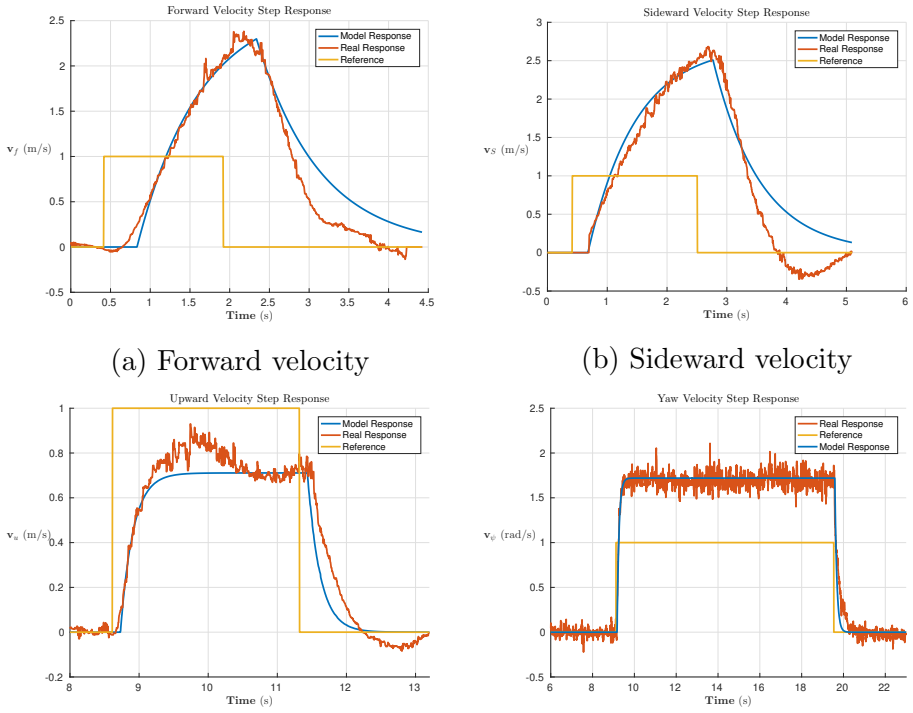
$$k = \frac{v(\infty)}{u(\infty)} \qquad \tau = \frac{3}{2}(t_{63} - t_{28}) \qquad d = t_{63} - \tau \tag{4.6}$$

	k	τ (s)	d (s)
v_f/u_f	2.7000	0.7889	0.4164
v_s/u_s	2.7000	0.7889	0.2600
v_u/u_u	0.7110	0.1815	0.1148
v_h/u_h	1.7200	0.0912	0.0483

Table 4.1: Parameters of the first-order with delay model for AR.Drone 2.0

4.4 MPC Controller Design

Making use of the state space model (4.1), an Optimal Control Problem (OCP) is defined in (4.7). The equation (4.7a) defines the objective function, where a quadratic penalty weighted by the P, Q matrices is applied to minimize the control inputs and the difference of the state with respect to a given goal x_g. The evolution of the trajectory is restricted to the system



(a) Forward velocity

(b) Sideward velocity

(c) Upward velocity

(d) Heading velocity

Figure 4.4: UAV step response (real vs modelled)

dynamics (4.7c), the actuator saturations (4.7d) and the obstacles in (4.7e), bounded by ellipsoids with center (p_x^o, p_y^o, p_z^o) and radius (r_x, r_y, r_z).

$$\min_{x(\cdot),u(\cdot)} \|x(T) - x_g\|_P^2 + \int_0^T \left(\|x(t) - x_g\|_P^2 + \|u(t)\|_Q^2\right) dt \tag{4.7a}$$

subject to:

$$x(0) = x_I \tag{4.7b}$$

$$\dot{x}(t) = Ax(t) + Bu(t-d) \tag{4.7c}$$

$$u_{min} \leq u(t) \leq u_{max} \tag{4.7d}$$

$$1 \leq \frac{(p_x(t) - p_x^o)^2}{r_x^2} + \frac{(p_y(t) - p_y^o)^2}{r_y^2} + \frac{(p_z(t) - p_z^o)^2}{r_z^2} \tag{4.7e}$$

Then, ACADO Toolkit for C++ [Quirynen et al., 2014] is used to address the OCP numerically. First, a multiple shooting discretization with piecewise constant control parametrization is employed to obtain a nonlinear program (NLP). To that aim, an explicit 4th-order Runge-Kutta integration scheme obtains the discrete-time dynamics while a Riemman sum approximates the integral cost. The resulting NLP is then addressed through a Gauss-Newton Sequential Quadratic Programming (SQP) algorithm as described in [Quirynen et al., 2014]. This algorithm approximates the NLP as a sequence of quadratic programs (QP) that can be solved to global optimum by QPOASES [Ferreau et al., 2014]. The inequality constraints are handled by active-set methods, which minimizes the computational overhead when multiple obstacle constraints are included in the OCP.

Our Model Predictive Control algorithm results when solving the OCP at each sampling time using the real-time iteration scheme (RTI) proposed by [Diehl et al., 2005]. As shown in Algorithm 1, our algorithm operates asynchronously, freeing computational resources until a message is received. When the feedback state is available the NLP is solved, applying the first control to the platform and preparing the NLP for the next step. The communication protocol is implemented in C++ with ROS Indigo in a decentralized manner using the publisher/subscriber messaging pattern. We exploited this feature to build a software architecture that allows most parameters of the OCP to be changed online. As shown in Fig. 4.5, we implemented different subscribers that allow other processes to change the weighting costs, obstacles, model parameters and control saturations, as shown in Fig. 4.5. In addition, the planned state and control trajectories are published for better

visualization and monitoring.

```
Initialization;
while true do
    Sleep until message arrives;
    if message == feedback_state then
        Update initial state x_0 = x_I;
        Solve NLP; Apply first control u_0;
        Prepare NLP for next step;
            Numerical simulation;
            Cost function evaluation;
    else
        Update corresponding parameters;
    end
end
```

Algorithm 1: MPC Algorithm

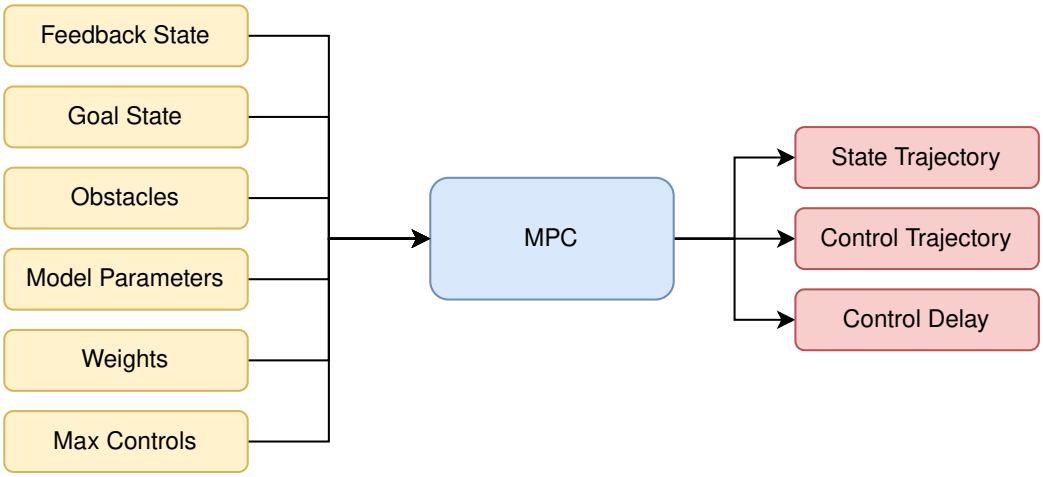


Figure 4.5: MPC's parametric implementation. Yellow boxes represent input data where red boxes correspond to the outputs provided by the MPC algorithm.

4.5 Experiments and Results

In this section we validate our approach in real experiments using the experimental testbed described in Section 4.2. We set up our MPC algorithm with the specific configuration shown in Table 4.2.

To validate the position control capabilities, step responses in (p_x, p_y, p_z, ψ) are generated as shown in Fig. 4.6. Initially, the UAV stands in $(p_x, p_y, p_z, \psi) =$

Prediction horizon	4 s
Control intervals	4
Simulation intervals	32
Integrator type	Runge-Kutta 4
Max controls (u_{min}/u_{max})	± 1 m/s
MPC sample time	10 ms
MPC control delay	0.46 ms
Weighting matrix P	$diag$(200 200 100 300 100 100 100 100)
Weighting matrix Q	$diag$(600 600 50 200)

Table 4.2: MPC controller configuration

$(0, 0, 1, 0)$ (S.I.). Then the goal state is changed alternatively for each input, showing rapid convergence and stable behavior of the MPC algorithm. The median control delay has been observed to be 0.46 ms, which validates its real-time capabilities, leaving room for increasing complexity.

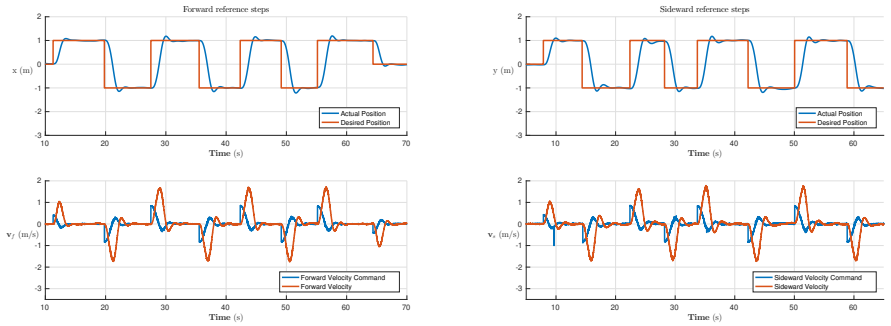
To test the path tracking and obstacle avoidance capabilities of our MPC algorithm, we define a squared waypoint path to be followed at 0.5 m/s with two ellipsoidal obstacles with the parameters shown in table 4.3. In Figure 4.8 the trajectory of the UAV and the modelled obstacles are represented precisely using the raw data from the experiment during 3 laps. For a better visualization of the experiment, the video recorded during the experiment is processed to obtain the UAV trajectory represented by its centroid in the image plane, as shown in Figure 4.7.

	Obstacle 1	Obstacle 2
(x, y, z)	$(0, -2, 1)$	$(0, 2, 2)$
(r_x, r_y, r_z)	$(1, 2, 2)$	$(1, 1, 10^4)$

Table 4.3: Position and shape of the ellipsoidal model of the obstacles in meters

4.6 Conclusions

In this chapter, we have developed a model predictive control algorithm able to perform real-time optimal motion planning and control in the presence of obstacles. Additionally, we developed a parametric software architecture that enables online reparametrization of the optimal control problem, allowing to



(a) Forward step response

(b) Sideward step response

(c) Upward step response

(d) Heading step response

Figure 4.6: Reference steps response of the UAV driven by the MPC Controller

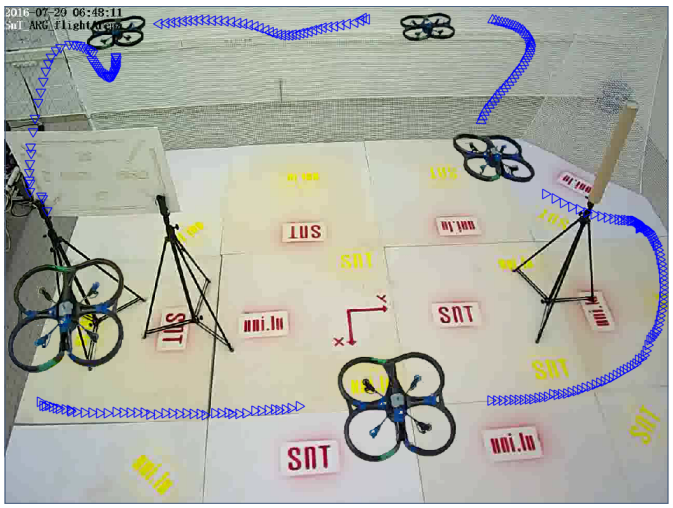


Figure 4.7: UAV trajectory when performing the path tracking with obstacles experiment

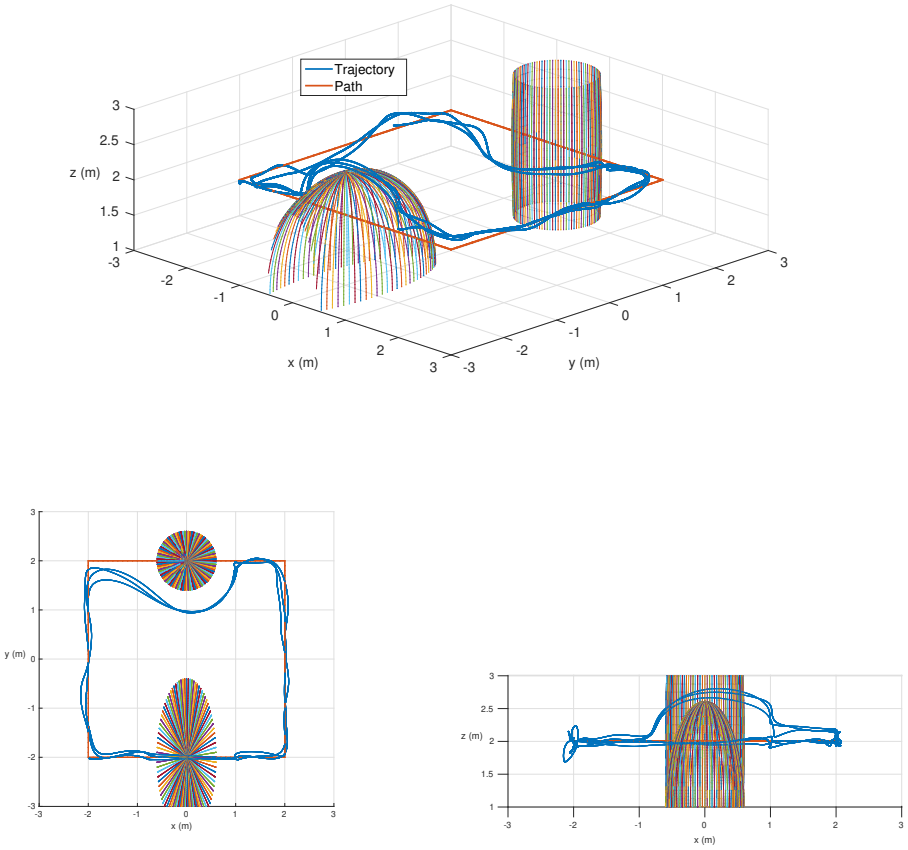


Figure 4.8: Trajectory of the UAV while performing a real path tracking with obstacles experiment. Isometric, top and front view respectively.

adapt the presented algorithm to different platforms and obstacle configurations. With an average control delay of 0.46 ms, our approach is suitable for tele-operated systems as well as embedded applications.

Chapter 5

Optimal Motion Planning and Control in Dynamic Environments.

Autonomous robot navigation in environments populated by humans and other robots is still a main challenge. In this chapter, we present a nonlinear model predictive control approach for safe navigation in workspaces populated by static and/or moving obstacles. The uniqueness of our approach lies in its ability to anticipate avoidance maneuvers by including the prediction of obstacle locations into the optimal control problem. Exploiting active-set optimization algorithms, only the obstacles that affect to the UAV's planned trajectory are accounted during optimization, which allows its application for aerial robot with multiple obstacles. We also introduce orientable ellipsoids as a more convenient encapsulation of the obstacles, which can be exploited to provide a tighter bound of the obstacle space or induce desired avoidance behaviors to avoid local minima. Finally, we present two real-time implementations based on simulation. The former demonstrates that our approach outperforms its analog formulation (without accounting for obstacles' prediction) in terms of safety and efficiency. The latter shows its capability to handle multiple dynamic obstacles. The contributions shown in this chapter are published in our conference paper [Castillo-Lopez et al., 2018].

5.1 Introduction

Unmanned aerial vehicles (UAVs), commonly known as drones, are now populating natural and industrial environments due to their multiple applications such as package delivery, traffic-surveillance or search-and-rescue oper-

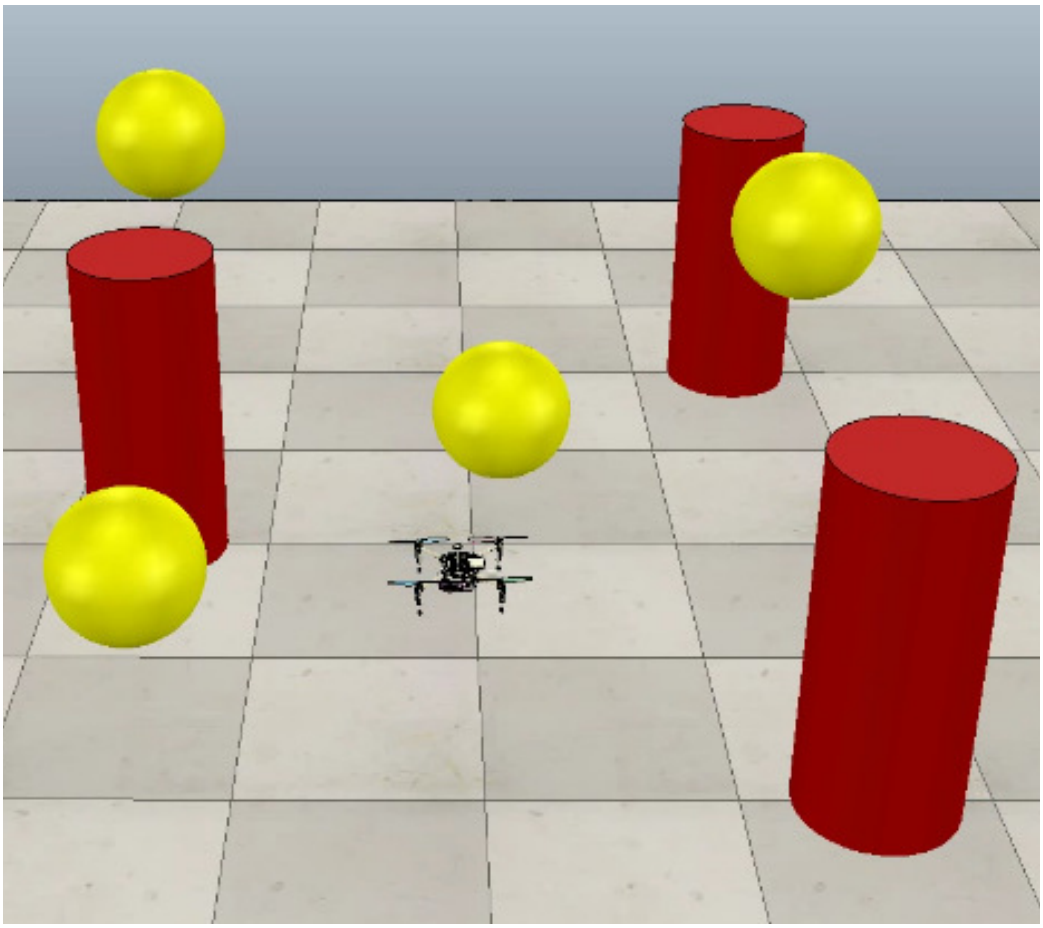


Figure 5.1: UAV crossing the street avoiding multiple dynamic obstacles. See video in https://rebrand.ly/castillo2018mpc

ations [Valavanis and Vachtsevanos, 2015]. These applications often demand the use of mobile robots in presence of humans and other robots, making autonomous navigation one of the most active topics in robotics research [Hoy et al., 2015]. This problem is usually addressed by a hierarchical combination of planning and control but, the lack of proper coordination between these agents, often leads to inefficient or dangerous situations [Shim et al., 2003]. In this chapter, we use Nonlinear Model Predictive Control (NMPC) to integrate motion planning and control in scenarios with multiple dynamic obstacles as shown in Fig. 5.1.

Most prior work on moving obstacles are based on the concept of velocity obstacles (VOs) [Fiorini and Shiller, 1998] to compute the set of velocities of the robot that will cause a collision. This technique has been used and enhanced by many authors, taking into account nonlinear dynamics [Shiller

et al., 2001], uncertainty [Fulgenzi et al., 2007], reciprocal behavior [Snape et al., 2011] or its use for computing control objectives in convex optimization [Van Den Berg et al., 2012].

Unlike these methods, NMPC is able to natively consider non-convex constraints such as obstacles to generate feasible trajectories, being safer and less prone to local minima than other hierarchical approaches [Shim et al., 2003]. Even though NMPC is known to be computationally expensive, recent advances on nonlinear optimization solvers [Domahidi et al., 2012, Quirynen et al., 2015b] has triggered its use in fast real-time applications such as collision avoidance for small UAVs.

Although great effort has been done in this area, dealing with multiple three-dimensional obstacles remains difficult. Recent work using NMPC often simplifies this problem by considering only the closest obstacle [Kamel et al., 2017a] or reducing the obstacles as bi-dimensional static [Garimella et al., 2017, Zhang et al., 2016] or dynamic [Andersson et al., 2016] constraints.

To provide safe navigation in cluttered dynamic scenarios we exploit active set algorithms to consider only the constraints that affect to our problem at each sample time. In this chapter, we extend our previous work [Castillo-Lopez et al., 2017] by including the dynamics of ellipsoidal obstacles without additional cost. We use parametrized soft constraints to specify the sensitiveness of the avoidance maneuvers with the guarantee of finding a locally-optimal solution even in highly constrained scenarios as shown in Fig. 5.1. Successful experiments in real-time validates our approach in scenarios with multiple dynamic obstacles, outperforming its analog formulation in terms of safety and efficiency.

This chapter is organized as follows: In section 5.2 we develop the model for a multi-rotor UAV, including the experimental identification of a real platform. Section 5.3 formulates the NMPC control policy regarding trajectory tracking and collision avoidance. Finally, we validate our approach in two realistic scenarios in section 5.4, drawing the conclusions in section 5.5.

5.2 UAV Model

A multi-rotor UAV is usually modeled using the Newton-Euler equations of a rigid body to stabilize and control the platform [Mellinger and Kumar, 2011]. However, the typical configuration of an UAV includes an autopilot that controls its stability while following velocity commands given by an external pilot (automated or human). The control input $\mathbf{u} = [u_x \; u_y \; u_z \; u_\psi]^T$ is divided in forward, sidewards, upward, and heading velocity references.

This commands are specified in a pitch/roll invariant body frame named as the hovering frame H (see Fig. 5.2). The world frame W, is defined as a standard North-East-Up fixed reference frame.

Recent work on quadrotor modeling describe it as a differentially flat system [Mellinger and Kumar, 2011], meaning that its full state can be represented by a combination of its flat outputs and their derivatives. In this work we choose the cartesian coordinates of the center of mass $\mathbf{r} = [x\ y\ z]$ and the yaw angle ψ to build the reduced state vector:

$$\mathbf{x} = [x\ y\ z\ \psi\ v_x\ v_y\ v_z\ v_\psi]^T \tag{5.1}$$

being $\mathbf{v} = [v_x\ v_y\ v_z]$ the UAV linear velocity in the hovering frame and the heading angular velocity v_ψ. Then, we propose a nonlinear model $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$ defined by the set of equations:

$$\dot{x} = v_x cos(\psi) - v_y sin(\psi) \tag{5.2a}$$
$$\dot{y} = v_x sin(\psi) + v_y cos(\psi) \tag{5.2b}$$
$$\dot{z} = v_z \tag{5.2c}$$
$$\dot{\psi} = v_\psi \tag{5.2d}$$
$$\dot{v}_i = (-v_i + k_i u_i)/\tau_i, \quad i \in \{x, y, z, \psi\} \tag{5.2e}$$

where (6.28c) models the velocity response of the UAV as a first order model of gain k_i and constant time τ_i.

In this work, we use a motion capture system[1] to perform the model identification of a DJI-M100[2] quadrotor (see Fig. 5.2). The model parameters are obtained by a classical step response tangent method [Castillo-Lopez et al., 2017] defined by:

$$k_i = \frac{v_i(\infty)}{u_i(\infty)} \qquad \tau_i = \frac{3}{2}(t_{63} - t_{28}) \tag{5.3}$$

where t_{63} and t_{28} are the times, from the start of the step, when the velocity reaches the 63% and the 28% of its final value respectively. The resulting parameters are shown in Table 5.1.

This lightweight formulation allow us to solve the UAV dynamics faster without the knowledge of its physical design, being applicable on any multirotor platform. Besides, placing the controller outside the inner control loop relaxes its real-time requirements, allowing us to increase the prediction horizon of the NMPC approach to target high-level control policies.

[1] Optitrack Motion Capture System: http://optitrack.com/
[2] DJI Matrice 100: https://www.dji.com/matrice100

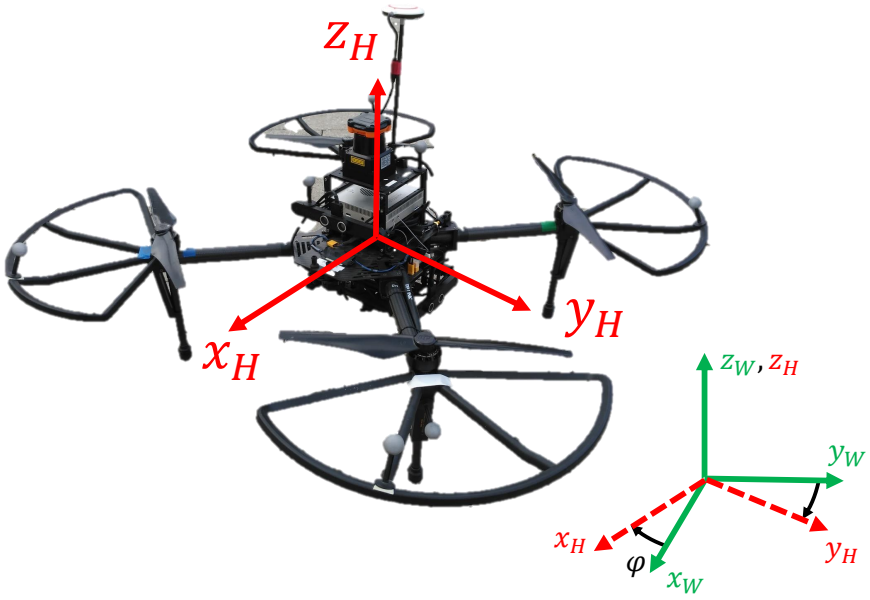


Figure 5.2: Hovering frame of the DJI-M100 platform.

Table 5.1: First order model parameters of the DJI-M100

	k_i	τ_i (s)
x	1.0000	0.8355
y	1.0000	0.7701
z	1.0000	0.5013
ψ	$\pi/180$	0.5142

5.3 Controller Design

Model predictive control obtains the control input of the system by solving an Optimal Control Problem (OCP) each sample time. That involves minimizing a given cost function over a defined prediction horizon subjected to states and input constraints. In this work, the OCP is designed to minimize the effort required for an UAV to track a desired state or trajectory while avoiding dynamic obstacles.

5.3.1 Trajectory tracking

Trajectory tracking is formulated as the cost term J^t defined in (5.4). The difference between the predicted state $\mathbf{x}_i$ and its reference $\mathbf{x}_i^*$ at each step i

is penalized over the prediction horizon N as follows:

$$J^t = \frac{1}{2}\sum_{i=0}^{N-1} \|\mathbf{x}_i - \mathbf{x}_i^*\|_P^2 + \|\mathbf{x}_N - \mathbf{x}_N^*\|_Q^2 \tag{5.4}$$

where P and Q are weighting matrices of each quadratic penalty. For stability and energy efficiency, an analog cost J^c is defined for the control inputs as:

$$J^c = \frac{1}{2}\sum_{i=0}^{N-1} \|\mathbf{u}_i\|_R^2 \tag{5.5}$$

5.3.2 Obstacle avoidance

The use of three-dimensional constraints to model obstacles in a 3D environment is often a good practice. However, the use of hard constraints to guarantee the generation of safe trajectories leads to non-feasible problems and unpredictable results, making its viable use only with few obstacles [Kamel et al., 2017a]. In this work, we propose to use parametrized soft constraints to model each ellipsoidal obstacles as follows:

$$\xi^2(\mathbf{r}_i, \mathbf{r}_i^o) + \theta^\xi s_i^\xi \geq 1 \tag{5.6}$$

where s_i^ξ is an extra control input, known as slack variable, that relaxes the constraint with sensitivity θ^ξ to guarantee feasible solutions in tight situations. The ellipsoidal term $\xi^2(\mathbf{r}_i, \mathbf{r}_i^o)$ is based on the distance function:

$$\xi(\mathbf{r}_i, \mathbf{r}_i^o) = \sqrt{(\mathbf{r}_i - \mathbf{r}_i^o)^T Q_i (\mathbf{r}_i - \mathbf{r}_i^o)} \tag{5.7}$$

where $\mathbf{r}_i^o$ is the center position of the ellipsoid in world frame at time step i. Q_i is the metric induced by the ellipsoid dimensions $M = diag(r_x^{-2}, r_y^{-2}, r_z^{-2})$ rotated by ${}^O R_W$ to the world frame as:

$$Q_i = {}^O R_W^T \; M_i \; {}^O R_W \tag{5.8}$$

This formulation generalizes the euclidean obstacle model present in the literature, allowing to move, orientate and resize the obstacles along the prediction horizon at each sample time. Thus, these parameters can be manipulated to include the dynamics of every obstacle without the need of additional variables and extra computation. In this work, we propagate the position of the obstacle at constant velocity as:

$$\mathbf{r}_i^o = \mathbf{r}_{i-1}^o + \dot{\mathbf{r}}_{i-1}^o \cdot \Delta t \tag{5.9}$$

Soft constraints has the main drawback of increasing the computational cost because of the slack variables. To mitigate this effect when dealing with multiple obstacles, we define a shared slack variable s_i^{ξ} for all the ellipsoidal obstacles, with the cost:

$$J^{\xi} = \frac{1}{2}\sum_{i=0}^{N-1} \|\mathbf{s}_i^{\xi}\|_S^2 \tag{5.10}$$

Boundary obstacles, such as walls or the floor, are also considered analogously as soft planar constraints based on its position $\mathbf{r}_i^o$ and normal vector $\mathbf{n}_i$ as shown in (5.11) and (5.12).

$$\pi(\mathbf{r}_i) + \theta^{\pi} s_i^{\pi} \geq 0 \tag{5.11}$$

$$J^{\pi} = \frac{1}{2}\sum_{i=0}^{N-1} \|\mathbf{s}_i^{\pi}\|_T^2 \tag{5.12}$$

where $\pi(\mathbf{r}_i) = \mathbf{n}_i \cdot (\mathbf{r}_i - \mathbf{r}_i^o)$

5.3.3 Optimal Control Problem

Integrating the previous definitions, the optimal control problem is formulated in the set of equations (5.13), taking the form of a discrete non-linear program (NLP).

$$\underset{X,U}{\text{minimize}} \quad J = J^t + J^c + J^{\xi} + J^{\pi} \tag{5.13a}$$

$$\begin{aligned}
\text{subject to:} \quad & \mathbf{x}_0 = \bar{x}_0 && && (5.13b)\\
& \mathbf{x}_{i+1} = \Phi_i(\mathbf{x}_i, \mathbf{u}_i) && i = 0, \ldots, N-1 && (5.13c)\\
& \xi(\mathbf{x}_i) + \theta^{\xi} s_i^{\xi} \geq 1 && i = 0, \ldots, N-1 && (5.13d)\\
& \pi(\mathbf{x}_i) + \theta^{\pi} s_i^{\pi} \geq 0 && i = 0, \ldots, N-1 && (5.13e)\\
& |\mathbf{u}_i| \leq \mathbf{u}_{max} && i = 0, \ldots, N-1 && (5.13f)
\end{aligned}$$

In (5.13a) all the cost terms are merged in a single objective function to find the best trade-off between trajectory tracking, efficiency and collision avoidance. In (5.13b) the feedback state $\bar{x}_0$ is set as the initial state in the prediction horizon. The (5.13c) introduces the discretized form of the UAV model $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$ as an equality constraint, including obstacles and maximum control inputs in (5.13d)-(5.13e) and (5.13f) respectively.

5.4 Experiments

The aim of our approach is to provide safe navigation for UAVs in complex dynamic environments. To validate it, we set a simple scenario in subsection 5.4.3 to show its performance compared with its analog approach, which uses a static obstacle formulation. Then, we set a challenging scenario in section 5.4.4 to analyze its capability to deal with multiple moving obstacles.

5.4.1 Risk evaluation

In this work, the risk of collision is evaluated by the distance to obstacle d and the inverse time to collision (TTC^{-1}) [Sajadi-Alamdari et al., 2018], which is defined as:

$$TTC^{-1} = \frac{\dot{d}}{d} \tag{5.14}$$

This rate indicates the risk of collision between two agents. Negative values correspond to a potential collision, while positive values indicate that the agents are moving away from each other. The safest situation is around zero, corresponding to high distances and small relative velocity.

5.4.2 Implementation details

To implement the optimal control problem defined in (5.13), ACADO Toolkit for C++ has been used to generate a fast explicit solver for the NMPC controller. The NMPC controller has been implemented in C++ programming language, building the communications and interfaces using the The Robot Operating System (ROS) Kinetic framework [Quigley et al., 2009]. As a simulation environment V-REP [Rohmer et al., 2013] is used to run software-in-the-loop experiments.

The implementation parameters of the NMPC algorithm are shown in Table 5.2. A long prediction horizon is chosen to promote a long-term optimal control policy, which is highly sensitive to obstacles. More sensitivity is given to ellipsoidal obstacles, which are meant to be dynamic.

An uniform weight distribution is chosen to provide a balanced trade-off between efficiency, safety and tracking. As shown in Table 5.3, orientation penalty is an exception that has been set aggressively to compensate the low gain given by the model.

Table 5.2: Model Predictive Controller implementation parameters.

Prediction horizon	4s
Discretization steps	20
Integrator type	Runge-Kutta 4
Maximum controls	1 m/s
Control Rate	20 Hz
Ellipsoidal obstacles sensitivity	0.15
Planar constraints sensitivity	0.25

Table 5.3: Model Predictive Controller weighting values.

	Weights
Position	10
Orientation	5000
Linear velocities	1
Angular velocities	1
Control inputs	10
Slack variables	10

5.4.3 Street crossing scenario

Lets consider a scenario where the goal of the UAV is to cross a street populated by humans, which must be avoided without flying over them. For that purpose, we model the pedestrians as ellipsoids with radius of $(r_x, r_y, r_z) = (0.5m, 0.6m, \infty)$. This generates an elliptical cylinder slightly slimmer in the approaching direction, which reduces the probability of reaching a local minima and promotes fluent maneuvers. The UAV is modeled as a sphere with radius of $0.5m$, applying a safety distance of $1m$. In this experiment, we consider three pedestrians moving alternatively at 1 m/s perpendicular to the robot's shortest path as shown in Fig. 5.3.

To validate the approach we compare the performance of the same controller updating the obstacles in two different ways: statically updated at each sample time and dynamically propagated based on the obstacle velocity. Fig. 5.7 shows the trajectory generated while performing the experiment with the static approach. Even though that the UAV manage to avoid all obstacles, the generated maneuvers interfere with the trajectories of pedestrians, leading to situations with greater risk and higher deviations from the optimal path. In contrast, our approach presents a safer control policy, finding optimal to avoid the obstacles without crossing their future trajectory as shown in Fig. 5.8.

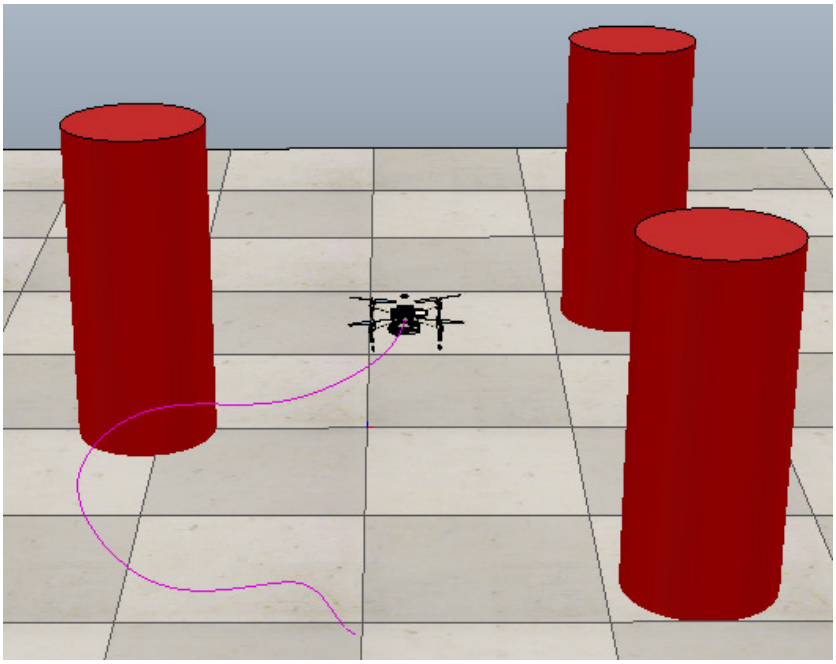


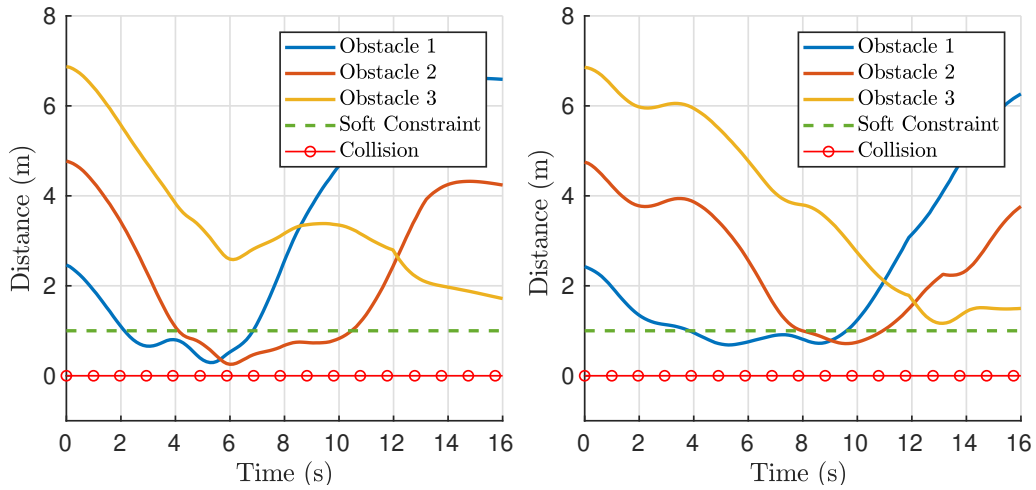
Figure 5.3: UAV avoiding human-sized dynamic obstacles. See video in https://rebrand.ly/castillo2018mpc

Even though the soft constraints are designed to be violated with high sensitivity, this situation should be minimized. In Fig. 5.4 we can see that both implementations provide a collision-free navigation. However, our approach provides safer avoidance, reducing the soft constraint violations in time and magnitude. As shown in Fig. 5.5, our approach reduces the TTC^{-1} peak values and, as a result, the magnitude of the risk. The frequency in which dangerous situations occur is shown in Fig. 5.6, where the histogram of the minimum TTC^{-1} indicates that our approach reduces the time in which the UAV is at risk of collision.

5.4.4 Multiple obstacle scenario

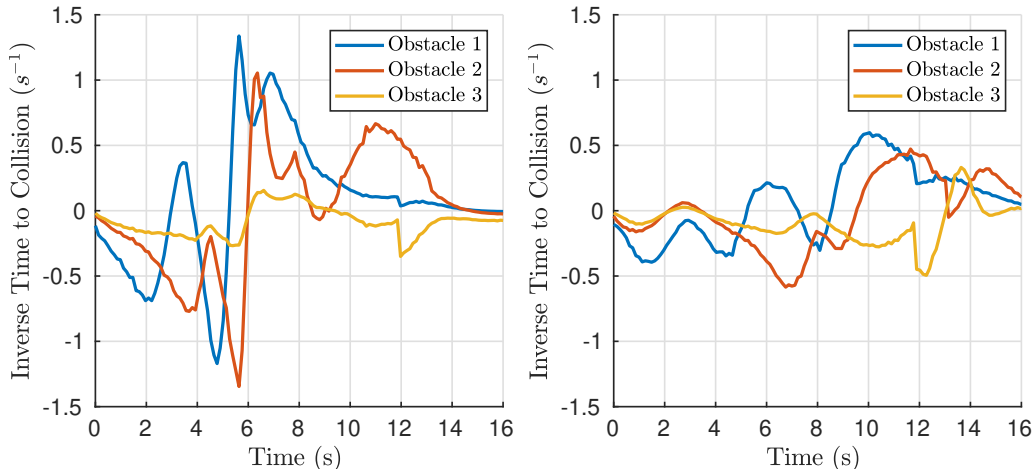
Lets consider a new scenario where the UAV has to cross a street populated not only by humans, but also other aerial robots. In this experiment, the robot is allowed to fly over other robots, but not over pedestrians. The pedestrian model of the previous experiment is used, considering other UAVs as ellipsoids with radius of $(r_x, r_y, r_z) = (0.5m, 0.6m, 0.5m)$ and a safety distance of $1m$.

In this experiment, in addition to the pedestrians of the previous sce-



(a) Static approach (b) Dynamic approach

Figure 5.4: Distance to obstacles in street crossing scenario



(a) Static approach (b) Dynamic approach

Figure 5.5: TTC^{-1} in street crossing scenario

nario, we consider four aerial robots moving alternatively at 1 m/s as shown in Fig. 5.1. Three of them move perpendicular to the robot's shortest path while the fourth is moving in diagonal direction. In Fig. 5.11 is shown that the UAV is able to avoid laterally the pedestrians while avoiding three-dimensionaly the aerial robots. Fig. 5.9 shows that our approach manage to avoid multiple collisions, with the increase of TTC^{-1} magnitude and soft constraint violations due to the complexity of the environment.

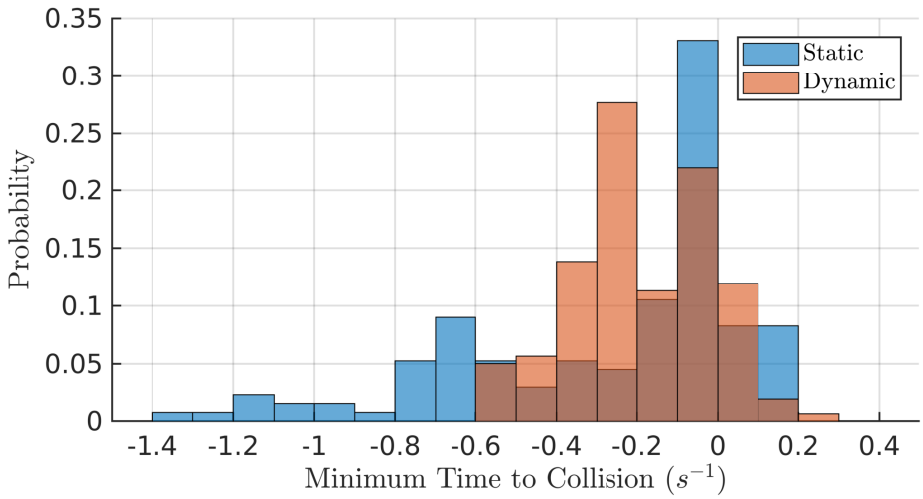


Figure 5.6: Histogram of minimum TTC^{-1} in street crossing scenario

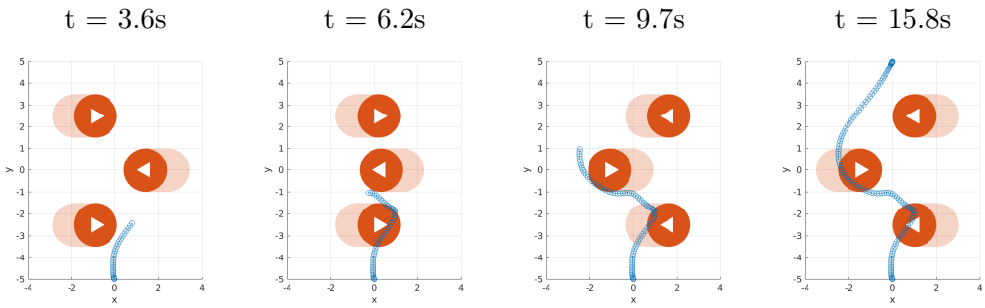


Figure 5.7: UAV trajectory in street crossing scenario. Static approach.

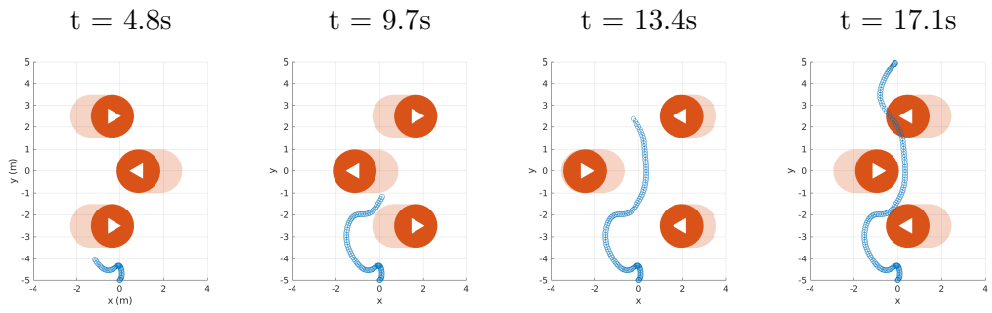
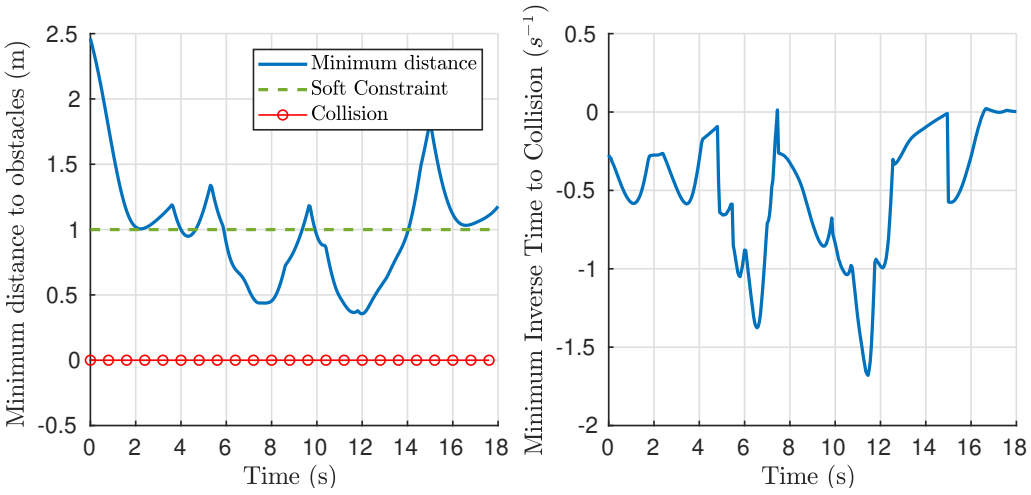


Figure 5.8: UAV trajectory in street crossing scenario. Dynamic approach

5.5 Conclusions

We presented a new model predictive control approach for three-dimensional collision avoidance in scenarios with multiple dynamic obstacles. These obstacles were modeled as orientable ellipsoids by using parametrized soft constraints, which allows a flexible obstacle definition, guaranteeing feasible solu-



(a) Minimum Distance (b) Minimum TTC^{-1}

Figure 5.9: Risk variables in the multiple obstacle scenario

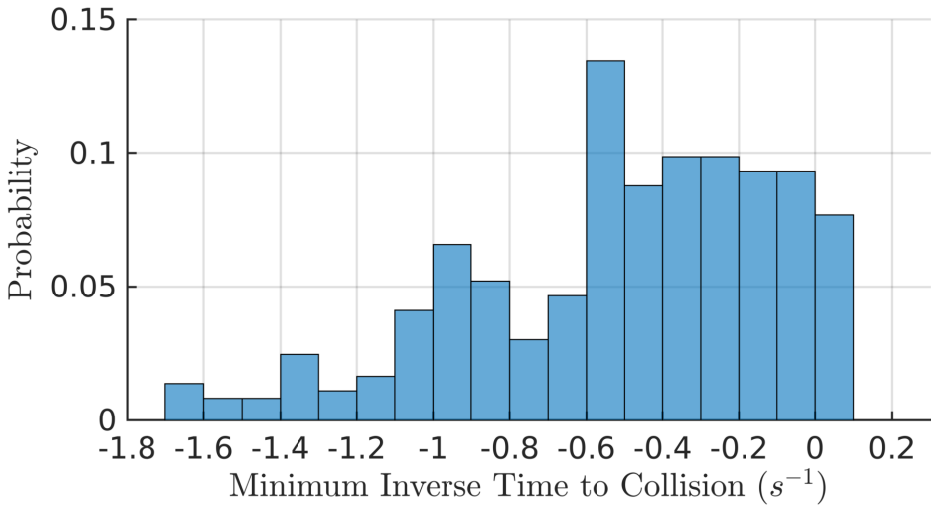


Figure 5.10: Minimum inverse time to collision histogram in multiple obstacle scenario

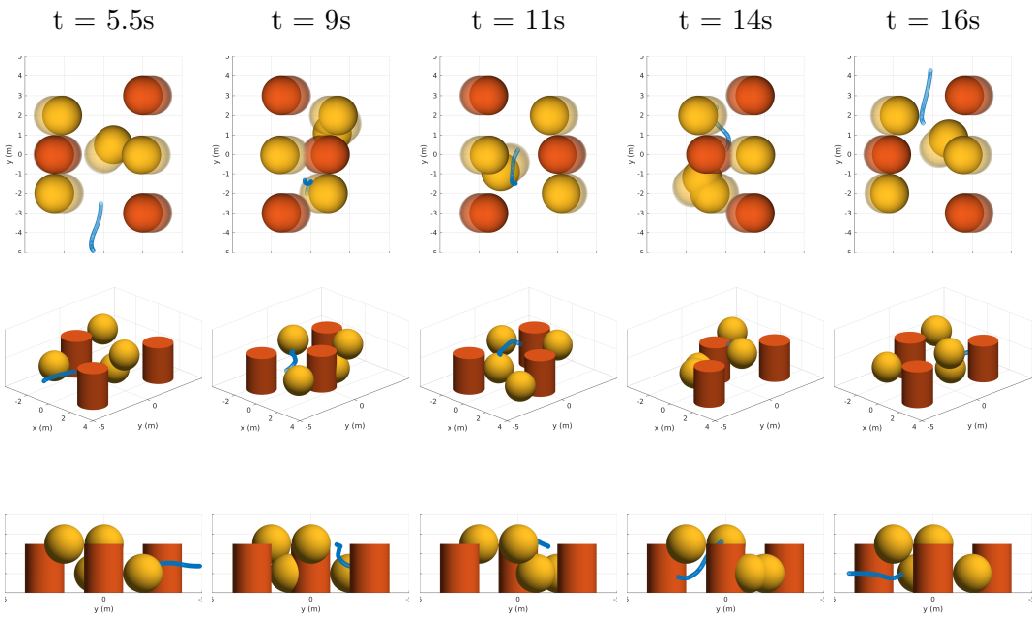


Figure 5.11: UAV trajectory in the multiple obstacle scenario

tions. With this formulation, the dynamics of each obstacle can be introduced externally without additional cost, suiting the needs of each application. In this chapter, we used a constant velocity model to test its collision avoidance performance in two real-time scenarios. The first experiment shows a considerable improvement over a static formulation, being safer and more efficient. The second experiment validates the approach for three-dimensional avoidance in a cluttered scenario with seven moving obstacles.

Chapter 6

Optimal Motion Planning and Control with Safety Guarantees.

Uncertain dynamic obstacles, such as pedestrians or vehicles, pose a major challenge for optimal robot navigation with safety guarantees. Previous work on optimal motion planning has employed two main strategies to define a safe bound on an obstacle's space: using a polyhedron or a nonlinear differentiable surface. The former approach relies on disjunctive programming, which has a relatively high computational cost that grows exponentially with the number of obstacles. The latter approach needs to be linearized locally to find a tractable evaluation of the chance constraints, which dramatically reduces the remaining free space and leads to over-conservative trajectories or even unfeasibility. In this work, we present a hybrid approach that eludes the pitfalls of both strategies while maintaining the original safety guarantees. The key idea consists in obtaining a safe differentiable approximation for the disjunctive chance constraints bounding the obstacles. The resulting nonlinear optimization problem can be efficiently solved to meet fast real-time requirements with multiple obstacles. We validate our approach through mathematical proof, simulation and real experiments with an aerial robot using nonlinear model predictive control to avoid pedestrians.

6.1 Introduction

Autonomous robots, such as self-driving cars or drones, are expected to revolutionize transportation, inspection and many other applications to come [Mohamed et al., 2018]. To fully exploit their capabilities, we need to enable their safe operation among humans and other robots while pursuing high-level

objectives such as safety [Hentzen et al., 2018] or energy consumption [Sajadi-Alamdari et al., 2019]. However, planning trajectories with obstacles whose present and future location is highly uncertain is still a difficult and computationally expensive problem [Blackmore et al., 2011]. Strong assumptions need to be made to find tractable solutions for fast real-time applications. This leads to over-conservative obstacle models that ensure collision-free operation but drastically reduce the remaining free space [Kamel et al., 2017a, Zhu and Alonso-Mora, 2019], compromising the problem's feasiblity when multiple obstacles arise. As a result, reducing conservatism in motion planning algorithms while providing safety guarantees has become a major problem and the subject of active research [Blackmore et al., 2011, Ono, 2012, Ono et al., 2013, Jha et al., 2018, Zhu and Alonso-Mora, 2019, Lefkopoulos and Kamgarpour, 2019, Ono and Williams, 2008].

In this chapter, we present a new approach to model uncertain dynamic obstacles for fast real-time motion planning applications. This method eludes the over-conservatism of existing real-time approaches while providing safety guarantees at a low computational cost. The resulting problem is modeled within the framework of disjunctive chance-constrained optimization and casted into non-linear programming, for which efficient solvers exist [Houska et al., 2011]. Thus, the main contributions of this chapter are listed as follows:

- Theoretical results on disjunctive chance constraints, providing tighter bounds on the probability of collision.
- A new real-time approach for chance-constrained motion planning in dynamic environments.
- Empirical validation through simulation and real experiments on an aerial robot to avoid pedestrians.

The rest of the chapter is organized as follows: Section 6.2 presents an overview of existing approaches for chance-constrained motion planning. We formalize the motion planning problem in Section 6.3, and present a set of preliminary results in Section 6.4 from which we build our theoretical results in Section 6.5. Finally, our approach is evaluated through a benchmark, software-in-the-loop simulations and real experiments in Section 6.6, drawing the resulting conclusions and future lines of work in Section 5.5.

6.2 Related Work

Optimal motion planning has been the subject of active research during the last decade, as surveyed in [Dadkhah and Mettler, 2012, Hoy et al., 2015]. Generally, the space occupied by obstacles is represented as a set of constraints on the free space which, in general, disrupts its convexity. The choice on the type of constraints determines the nature of the resulting optimization problem and therefore, its performance. There are two main strategies in the literature to encapsulate an obstacle's space: Using a convex polyhedron [Blackmore et al., 2011, Lefkopoulos and Kamgarpour, 2019] (e.g. a cuboid), or a single differentiable surface [Castillo-Lopez et al., 2018, Kamel et al., 2017a, Zhu and Alonso-Mora, 2019] (e.g. an ellipsoid).

A polyhedral obstacle is encoded as a disjunction of linear inequality constraints. This represents logical OR relations between the infinite planes that define each face of the polyhedron. The resulting disjunctive problem can be solved to global optimality using existing branch-and-bound techniques [Balas, 2018]. This problem has a relatively high computational cost that grows exponentially with the number of obstacles [Balas, 2018, Ono et al., 2013]. Even though recent efforts show promising improvements on computational efficiency, over-conservatism and probabilistic guarantees [Blackmore et al., 2011, Ono, 2012, Ono et al., 2013, Lefkopoulos and Kamgarpour, 2019], their computational cost is still too elevated to meet fast real-time requirements.

Alternatively, an obstacle can be bounded by a single differentiable surface (sphere, cylinder, ellipsoid, etc.) to be included as a nonlinear constraint of the optimization problem [Castillo-Lopez et al., 2018]. This results in a comparatively low-dimension nonlinear program (NLP), which can be solved efficiently by gradient-based solvers [Houska et al., 2011]. Even though this solution cannot guarantee global optimality, its reduced computational cost makes this strategy to be widely adopted in most time-critical motion planning tasks, such as model predictive control for aerial robots [Zhu and Alonso-Mora, 2019, Castillo-Lopez et al., 2018, Kamel et al., 2017a].

Accounting for uncertainty through a probabilistic framework has shown to overcome the inherent over-conservatism of set-bounded uncertainty models [Mesbah, 2016, Ono, 2012], which is essential to avoid unfeasibility in cluttered environments. However, the chosen strategy to bound the obstacles critically impacts the evaluation of the resulting chance constraints. For instance, the linear chance constraints that compose polyhedral obstacles have a closed-form deterministic equivalent for Gaussian systems [Blackmore et al., 2011]. On the other hand, nonlinear chance constraints need to be linearized [Zhu and Alonso-Mora, 2019] or approximated by sampling

methods [Blackmore et al., 2010], which leads to over-conservatism and high computational cost respectively.

This chapter proposes a hybrid solution that benefits from both strategies. First, a polyhedral obstacle formulation is exploited to provide a closed-form approximation of the disjunctive chance constraints. Then, a differential surface provides a safe bound on polyhedral obstacle regions. To meet fast real-time requirements, we restrict each polyhedral obstacle to be a cuboid (i.e. bounding box), and then obtain a tight quadratic bound analytically. As a result, we land on a nonlinear formulation that can be solved efficiently with the guarantee that the original chance constraints will be satisfied with the specified confidence level.

6.3 Problem Statement

In this work, we consider the problem of motion planning with non-cooperative moving obstacles with uncertain localization, model and disturbances in the form of additive Gaussian noise. Thus, the dynamics of a given robot and a set of N_o obstacles are described as the following stochastic, discrete-time model:

$$x_{t+1} = f(x_t, u_t) + w_t \tag{6.1a}$$

$$y^i_{t+1} = g^i(y^i_t) + v^i_t \qquad i \in \{1, \dots, N_o\} \tag{6.1b}$$

where $x_t \in \mathbb{R}^{n_x}$, $y^i_t \in \mathbb{R}^{n_y}$ and $u_t \in \mathbb{R}^{n_u}$ are the robot state, i-th obstacle state and robot inputs respectively at time $t \in \mathbb{N}$. $w_t \in \mathbb{R}^{n_w}$ and $v^i_t \in \mathbb{R}^{n_v}$ are unknown disturbances with Gaussian probability distributions; and f and g^i are (possibly nonlinear) Borel-measurable functions that describe the robot and the i-th obstacle dynamics respectively.

Let $p_t \subset x_t$ and $q^i_t \subset y^i_t$ be the $\mathbb{R}^3$ subspaces describing the position of their respective center of mass. Then, bounding boxes centered at q^i_t with semi-sizes $d^i \in \mathbb{R}^3$ can be placed such that the free configuration space $\mathcal{F}_t$ is defined as follows:

$$\mathcal{F}_t := \left\{ x_t \in \mathbb{R}^{n_x} : \bigwedge_{i=1}^{N_o} \bigvee_{j=1}^{3} |p^j_t - q^{ij}_t| \geq d^{ij} \right\} \tag{6.2}$$

where j iterates over the Cartesian coordinates of each $\mathbb{R}^3$ element. $|\cdot|$, $\bigvee$ and $\bigwedge$ denote the absolute value, the logical OR and AND respectively. Given the stochastic nature of the agents, we can define the chance constraint over

the horizon length N as follows:

$$\mathcal{P}\left(\bigwedge_{t=1}^{N} x_t \in \mathcal{F}_t\right) \geq 1-\alpha \tag{6.3}$$

which enforces the robot to stay within the free configuration space in a probabilistic sense with the confidence level $1-\alpha$. As a result, the probabilistic motion planning problem is defined as follows:

$$\min_{u_0,\ldots,u_{N-1}} \; J(u_0,\ldots,u_{N-1},x_0,\ldots,x_N) \tag{6.4a}$$

subject to:

$$x_{t+1} = f(x_t,u_t) + w_t \tag{6.4b}$$
$$y^i_{t+1} = g^i(y^i_t) + v^i_t \tag{6.4c}$$
$$w_t \sim \mathcal{N}(0,W_t) \quad v^i_t \sim \mathcal{N}(0,V^i_t) \tag{6.4d}$$
$$x_0 \sim \mathcal{N}(\hat{x}_0,\Sigma_{x,0}) \quad y_0 \sim \mathcal{N}(\hat{y}^i_0,\Sigma_{y,0}) \tag{6.4e}$$
$$x_{t+1} \in \mathbb{X}, \quad u_t \in \mathbb{U} \tag{6.4f}$$
$$\mathcal{P}\left(\bigwedge_{t} x_{t+1} \in \mathcal{F}_{t+1}\right) \geq 1-\alpha \tag{6.4g}$$

where $t \in \{0,\ldots,N-1\}$ and $i \in \{1,\ldots,N_o\}$. The cost function (6.4a) determines the objective to pursue such as energy consumption or a reference state. The stochastic model of the robot and the obstacles are included in equations (6.4b) to (6.4d). The initial states in (6.4e) are assumed to be Gaussian distributions given by a state estimation algorithm such as Kalman filtering. The equations in (6.4f) provide additional state and control constraints to be defined for a given application. Finally, the collision chance constraint is included in (6.4g) with confidence level $1-\alpha$.

The key difficulty of this problem lies on the evaluation of the non-convex chance constraint (6.4g). It requires the integration of a multivariate Gaussian distribution and the convexification of the disjunctive constraints, which is, in general, intractable [Blackmore et al., 2011]. To overcome these difficulties, we safely approximate the problem as a deterministic disjunctive program, which is then casted into a nonlinear program to be solved efficiently by existing solvers [Houska et al., 2011].

6.4 Preliminary results

For the sake of clarity, this section introduces preliminary results to support further developments in Section 6.5.

6.4.1 Chance constraints for linear-Gaussian systems

Consider a multivariate Gaussian random variable $X \sim \mathcal{N}(\mu, \Sigma)$. Then, the chance constraint

$$\mathcal{P}(a^T X + b \leq 0) \geq 1 - \alpha, \qquad a, b \in \mathbb{R}^{n_x} \tag{6.5}$$

has a deterministic equivalent of the form:

$$a^T \mu + b + \Psi^{-1}(1-\alpha)\sqrt{a^T \Sigma a} \leq 0 \tag{6.6}$$

where Ψ is the standard Gaussian cumulative distribution function defined as:

$$\Psi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} exp\left\{-\frac{t^2}{2}\right\} dt \tag{6.7}$$

6.4.2 Probability theorems

For any number of events A_i, we have:

$$\mathcal{P}\left(\bigwedge_i A_i\right) \leq \mathcal{P}(A_j) \quad \forall j \tag{6.8}$$

$$\mathcal{P}\left(\bigvee_i A_i\right) \leq \sum_i \mathcal{P}(A_i) \tag{6.9}$$

$$\mathcal{P}\left(A_i\right) = 1 - P\left(\overline{A_i}\right) \tag{6.10}$$

$$\mathcal{P}\left(\overline{\bigvee_i A_i}\right) = \mathcal{P}\left(\bigwedge_i \overline{A_i}\right) \tag{6.11}$$

$$\mathcal{P}\left(\overline{\bigwedge_i A_i}\right) = \mathcal{P}\left(\bigvee_i \overline{A_i}\right) \tag{6.12}$$

where (6.8) follows from conditional probability, (6.9) is Boole's inequality, (6.10) is the De Moivre's theorem and (6.11)-(6.12) describe De Morgan's laws.

6.4.3 Minimum volume enclosing ellipsoid of a bounding box

Consider the space outside the bounding box $\mathcal{B}$ as

$$\mathcal{B}(d) := \left\{ x \in \mathbb{R}^3 : \bigvee_{i=1}^{3} |x_i| > d_i \right\} \tag{6.13}$$

where $d \in \mathbb{R}^3_+$ and the Cartesian coordenates are iterated through the index i. This set can be safely approximated by its minimum volume enclosing ellipsoid $\mathcal{E}$, which can be computed in closed form as [John, 2014]:

$$\mathcal{E}(d) := \left\{ x \in \mathbb{R}^3 : \sum_{i=1}^{3} \left(\frac{x_i}{d_i} \right)^2 > 3 \right\} \tag{6.14}$$

6.4.4 Bounds on disjunctive chance constraints

For the sake of clarity, this section introduces two intermediate proofs to support further developments. Then, for any number of events A_i, we have:

$$\mathcal{P}\left(\bigvee_{i=1}^{N} A_i \right) \geq 1 - \alpha \Leftarrow \bigvee_{i=1}^{N} \mathcal{P}\left(A_i\right) \geq 1 - \alpha \tag{6.15}$$

Proof. By applying (6.10) and (6.11) we have:

$$\mathcal{P}\left(\bigvee_{i=1}^{N} A_i \right) \geq 1 - \alpha \Leftrightarrow \mathcal{P}\left(\bigwedge_{i=1}^{N} \overline{A}_i \right) < \alpha \tag{6.16}$$

From (6.8) follows:

$$\mathcal{P}\left(\bigwedge_{i=1}^{N} \overline{A}_i \right) < \alpha \Leftarrow \bigvee_{i=1}^{N} \mathcal{P}\left(\overline{A}_i\right) < \alpha \tag{6.17}$$

Then, Equation (6.15) is obtained after applying (6.10) and (6.12), which completes the proof. ■

Similarly, risk allocation variables $\alpha_i \in \mathbb{R}$ can be defined such that:

$$\mathcal{P}\left(\bigwedge_{i=1}^{N} A_i \right) \geq 1 - \alpha \Leftarrow \left(\bigwedge_{i=1}^{N} \mathcal{P}\left(A_i\right) \geq 1 - \alpha_i \right) \wedge \left(0 \leq \alpha_i \leq 1\right) \wedge \left(\sum_{i=1}^{N} \alpha_i \leq \alpha \right) \tag{6.18}$$

Proof. By applying (6.10) and (6.12) we have:

$$\mathcal{P}\left(\bigwedge_{i=1}^{N} A_i \right) \geq 1 - \alpha \Leftrightarrow \mathcal{P}\left(\bigvee_{i=1}^{N} \overline{A_i} \right) < \alpha \tag{6.19}$$

From (6.9) follows:

$$\mathcal{P}\left(\bigvee_{i=1}^{N}\overline{A_i}\right) < \alpha \Leftarrow \sum_{i=1}^{N}\mathcal{P}\left(\overline{A_i}\right) < \alpha \tag{6.20}$$

Which can be reformulated into

$$\mathcal{P}\left(\bigvee_{i=1}^{N}\overline{A_i}\right) < \alpha \Leftarrow \left(\bigwedge_{i=1}^{N}\mathcal{P}\left(\overline{A_i}\right) \leq \alpha_i\right) \wedge (0 \leq \alpha_i \leq 1) \wedge \left(\sum_{i=1}^{N}\alpha_i \leq \alpha\right) \tag{6.21}$$

Equation (6.18) is obtained after applying (6.10) and (6.11), which completes the proof. ■

Thus, we have an immediate result on polyhedral obstacle regions described by chance constraints of the type:

$$\mathcal{P}\left(\bigwedge_{t=1}^{N}\bigwedge_{i=1}^{N_o}\bigvee_{j=1}^{N_f}A_t^{ij}\right) \geq 1-\alpha \Leftarrow \bigwedge_{t=1}^{N}\bigwedge_{i=1}^{N_o}\bigvee_{j=1}^{N_f}\mathcal{P}\left(A_t^{ij}\right) \geq 1-\alpha_t^i \wedge \left(0 \leq \alpha_t^i \leq 1\right) \wedge \left(\sum_{t=1}^{N}\sum_{i=1}^{N_o}\alpha_t^i \leq \alpha\right) \tag{6.22}$$

where N_f is the number of faces of the i-th obstacle. By direct comparison with recent results in [Jha et al., 2018, Lefkopoulos and Kamgarpour, 2021] we can see a considerable improvement on the chance constraint bounds, since risk allocation parameters α_t^i are increased by N_f times for uniformly distributed risk allocation. Thus, less conservative obstacle bounds are obtained for the same confidence level, reducing the risk of posing unfeasible problems when multiple obstacles arise.

6.5 Nonlinear bound for collision chance constraints

This section develops the main theoretical contribution of this chapter: a safe deterministic approximation of the chance constraint (6.3) given by

$$\bigwedge_{t=1}^{N}\bigwedge_{i=1}^{N_o}\sum_{j=1}^{3}\left(\frac{\hat{p}_t^j-\hat{q}_t^{ij}}{d_t^{ij}+\Psi^{-1}(1-\alpha_t^i)\sqrt{\sigma^2(p_t^{ij})+\sigma^2(q_t^{ij})}}\right)^2\geq 3 \\ \wedge\left(0\leq\alpha_t^i\leq 1\right)\wedge\left(\sum_{t=1}^{N}\sum_{i=1}^{N_o}\alpha_t^i\leq\alpha\right) \tag{6.23}$$

where $p_t^i\sim\mathcal{N}(\hat{p}_t^i,\sigma^2(p_t^i))$ and $q_t^{ij}\sim\mathcal{N}(\hat{q}_t^{ij},\sigma^2(q_t^{ij}))$.

Proof. Let the equation (6.3) be rewritten as the disjunction:

$$\mathcal{P}\left(\bigwedge_{t=1}^{N}\bigwedge_{i=1}^{N_o}\bigvee_{j=1}^{3}\bigvee_{k=0}^{1}(-1)^k(p_t^j-q_t^{ij})+d_t^{ij}\leq 0\right)\geq 1-\alpha \tag{6.24}$$

By application of (6.22), we get:

$$\bigwedge_{t=1}^{N}\bigwedge_{i=1}^{N_o}\bigvee_{j=1}^{3}\bigvee_{k=0}^{1}\mathcal{P}\left((-1)^k(p_t^j-q_t^{ij})+d_t^{ij}\leq 0\right)\geq 1-\alpha_t^i \\ \wedge\left(0\leq\alpha_t^i\leq 1\right)\wedge\left(\sum_{t=1}^{N}\sum_{i=1}^{N_o}\alpha_t^i\leq\alpha\right) \tag{6.25}$$

Since we now have linear combinations of Gaussian variables we can apply equation (6.6) to obtain:

$$\left(\bigwedge_{t=1}^{N}\bigwedge_{i=1}^{N_o}\bigvee_{j=1}^{3}\bigvee_{k=0}^{1}(-1)^j(\hat{p}_t^j-\hat{q}_t^{ij})+d_t^{ij} \right. \\ \left. +\Psi^{-1}(1-\alpha_t^i)\sqrt{\sigma^2(p_t^j)+\sigma^2(q_t^{ij})}<0\right) \\ \wedge\left(0\leq\alpha_t^i\leq 1\right)\wedge\left(\sum_{t=1}^{N}\sum_{i=1}^{N_o}\alpha_t^i\leq\alpha\right) \tag{6.26}$$

which is equivalent to

$$\left(\bigwedge_{t=1}^{N} \bigwedge_{i=1}^{N_o} \bigvee_{j=1}^{3} |\hat{p}_t^j - \hat{q}_t^{ij}| \geq d_t^{ij} \right. \\ \left. + \Psi^{-1}(1-\alpha_t^i)\sqrt{\sigma^2(p_t^j) + \sigma^2(q_t^{ij})} \right) \\ \wedge \left(0 \leq \alpha_t^i \leq 1\right) \wedge \left(\sum_{t=1}^{N} \sum_{i=1}^{N_o} \alpha_t^i \leq \alpha \right) \quad (6.27)$$

The equation (6.27) defines a bounding box to which (6.14) can be applied to obtain (6.23) and complete the proof. ■

6.5.1 Discussion

Even though the process of obtaining an ellipsoidal approximation to each obstacle space is mathematically involved, the underlying principle is simple. As shown in Fig. 6.1, the space occupied by each obstacle is defined by a blue bounding box. Due to the different sources of uncertainty, we increase the size of the bounding box to ensure that the risk of collision remains below the user-defined level α, illustrated by the orange bounding box. Then, we obtain the minimum-volume enclosing ellipsoid to provide a smooth bounding surface while preserving the original safety guarantees.

The main benefit of employing our approximation lies in our ability to address the problem (6.4) through nonlinear programming, which critically impacts its tractability and scalability. For instance, each polyhedral obstacle requires $7N$ mixed-integer constraints and $6N$ binary variables [Lefkopoulos and Kamgarpour, 2019], while our method can be implemented with N quadratic constraints and zero additional variables. In addition, the disjunctive program has a relatively high computational cost that grows exponentially with the number of obstacles [Ono et al., 2013,Balas, 2018]. In contrast, our nonlinear program can be solved with polynomial complexity [Houska et al., 2011], being computationally efficient for large-scale problems [Biegler and Zavala, 2009].

6.6 Case Study: Robot Collision Avoidance

In this section we implement our motion planning approach (6.4) in a Model Predictive Control (MPC) fashion to provide collision-free navigation on a

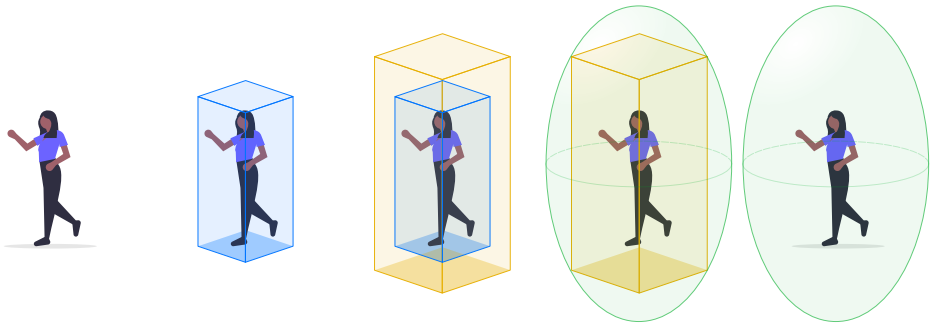


Figure 6.1: Geometric illustration of the mathematical procedure employed to derive our solution. The blue bounding box accounts for the dimensions of the obstacle. The orange bounding box renders the geometric space for which the risk of collision remains below the user-defined level α. The green minimum-volume enclosing ellipsoid is the final representation employed for optimal motion planning and control. We include our video presentation for further clarification: http://rebrand.ly/castillo_RAL2020

DJI-M100[1] quadrotor. The results of the experiments are complemented by the video demonstration https://rebrand.ly/castillo_RAL2020.

6.6.1 Robot Model

Based on the DJI SDK, the control inputs given to the quadrotor are defined as $u = [u_x \; u_y \; u_z \; u_\psi]^T$, which correspond to forward, sideward, upward, and heading velocity references, respectively based on a local frame L parallel to the ground (see [Castillo-Lopez et al., 2018] for details). Thus, the nominal system dynamics are modeled as follows:

$$\dot{p} = R(\psi)v \tag{6.28a}$$

$$\dot{v}_i = \frac{1}{\tau_i}(-v_i + k_i u_i), \quad i \in \{x, y, z\} \tag{6.28b}$$

$$\ddot{\psi} = \frac{1}{\tau_\psi}(-\dot{\psi} + k_\psi u_\psi) \tag{6.28c}$$

where $v = [v_x \; v_y \; v_z]^T$ is the linear velocity of the center of mass in the local frame and $R(\psi)$ the rotation matrix for the yaw angle ψ. k_i, k_ψ and τ_i, τ_ψ are the gain and time constants relative to each component of u respectively. Thus, the robot state is defined as $x_t = [p_t \; v_t \; \psi_t \; \dot{\psi}_t]$ and the nominal discrete dynamics $f(x_t, u_t)$ are obtained through 4-th order Runge-Kutta integration

[1]DJI Matrice 100: https://www.dji.com/matrice100

of (6.28). The nominal state prediction $\hat{x}_t$ and its covariance matrix Σ_t^x are approximated with a first-order Taylor expansion [Luo and Yang, 2017]:

$$\hat{x}_{t+1} = f(\hat{x}_t, u_t) \tag{6.29a}$$
$$\Sigma_{t+1}^x = (\nabla_x f(\hat{x}_t, u_t))\, \Sigma_t^x\, (\nabla_x f(\hat{x}_t, u_t))^T + W_t \tag{6.29b}$$

where u_t is obtained from the predicted inputs of the MPC algorithm. Even though there exists more precise uncertainty propagation methods [Luo and Yang, 2017], we use Taylor expansion for the sake of computational efficiency.

6.6.2 Obstacle Model

Obstacles are modeled with constant velocity nominal dynamics:

$$\dot{q}^i = R(\psi^i)v^i, \qquad \dot{v}^i = \ddot{\psi}^i = 0 \tag{6.30}$$

where v^i and ψ^i are the linear velocity in the body frame and yaw angle of the i-th obstacle respectively. Thus, obstacle states are defined as $y_t^i = [q_t^i\ v_t^i\ \psi_t^i\ \dot{\psi}_t^i]$ where the nominal discrete dynamics $g^i(y_t^i)$ are determined through Euler integration of (6.30). Similarly, the nominal state $\hat{y}_t^i$ and its covariance matrix $\Sigma_t^{y^i}$ are approximated with a first-order Taylor expansion

$$\hat{y}_{t+1}^i = g^i(\hat{y}_t^i) \tag{6.31a}$$
$$\Sigma_{t+1}^{y^i} = \nabla g^i(\hat{y}_t^i)\Sigma_t^{y^i} \left(\nabla g^i(\hat{y}_t^i)\right)^T + V_t^i \tag{6.31b}$$

6.6.3 Objective Function

We define the cost function in (6.4a) as:

$$J = \sum_{t=1}^{N} \left(\|x_t - x_t^r\|_P^2 + \|u_{t-1}\|_Q^2\right) \tag{6.32}$$

where x_t^r is the user-defined goal state. $\|\cdot\|_P$ and $\|\cdot\|_Q$ are the norms induced by the P and Q weighting matrices.

6.6.4 One-Horizon Benchmarks

In this section, our method is compared against three state-of-the-art approaches [Kamel et al., 2017a, Zhu and Alonso-Mora, 2019, Blackmore et al., 2011] on stochastic optimal collision avoidance for real-time systems. We design two experiments where the robot and the obstacle are placed at $p_0 = [0\ 0]$

and $q_0 = [5 \ \ -0.01]$ respectively. Uncertain obstacle's location is considered with covariance $\Sigma_q = diag(0.4\ 0.1)$. We have selected a prediction horizon of 8 seconds with $N = 40$ steps and a confidence level $1-\alpha = 0.99$ with uniform risk allocation $\alpha_t^i = \alpha/N$. For the sake of a purely chance-constrained benchmark, we have dropped the additional potential fields implemented in [Kamel et al., 2017a, Zhu and Alonso-Mora, 2019] that would have made these implementations even more conservative. For the first benchmark, the bounding box size is $d = [1\ 0.5]$ as shown in Fig. 6.2.

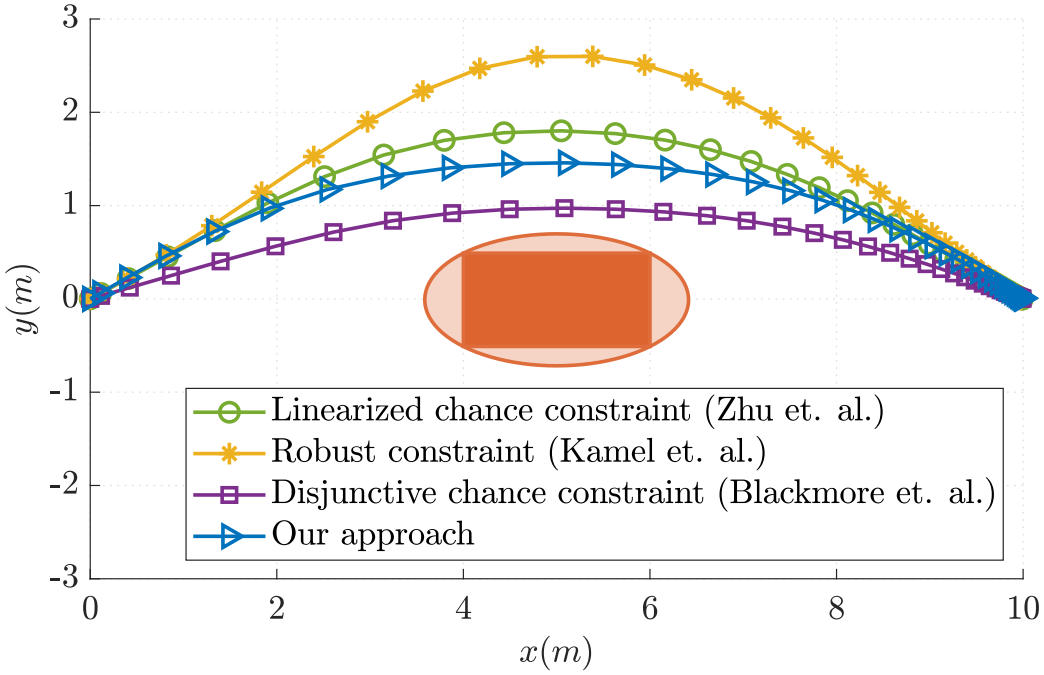


Figure 6.2: One horizon benchmark of our approach against the linearized chance constraint from [Zhu and Alonso-Mora, 2019], the robust constraint from [Kamel et al., 2017a] and the disjunctive chance constraint from [Blackmore et al., 2011].

As shown in Fig. 6.2, our approach is able to reduce conservatism with respect to existing real-time approaches while preserving the original safety guarantees from [Blackmore et al., 2011]. As presented in Table 6.1, our approach solves a conservative approximation of [Blackmore et al., 2011] over 142 times faster at the price 4% of optimality. Our computation time falls within the range of [Kamel et al., 2017a, Zhu and Alonso-Mora, 2019], which have been widely used for fast real-time motion planning and control.

Even though this benchmark has shown the benefits of our approach regarding conservatism and safety guarantees, it fails to back our claims in

Table 6.1: Relative results from the one horizon benchmark.

	Ours	Kamel *et. al.*	Zhu *et. al.*	Blackmore *et. al.*
Objective	1.0	1.0925	1.0205	0.9614
CPU time (s)	1.0	1.3198	1.2045	142.08

reducing the tendency of falling into local minima with respect to [Zhu and Alonso-Mora, 2019]. Therefore, we introduce second experiment with a larger obstacle $d = [1\ 2]$. As shown in Fig. 6.3, the linearization of the obstacle region performed by *Zhu et.al.* [Zhu and Alonso-Mora, 2019] leads to local minima, while our approach is still able to find a solution that reaches the goal. Note that the robust approach from [Kamel et al., 2017a] fails to ensure that the risk of collision remains below the desired risk level α_t^i, presenting a less conservative trajectory in this particular case. These experiments have been executed from the optimization framework CasADi [Andersson et al., 2018], being publicly available on-line to be reproduced[2].

6.6.5 Real experiment: Pedestrian collision avoidance.

The experiment consists in two pedestrians who naturally walk inside a closed area where the robot is operating. As shown in Fig. 6.4 and the complementary video, when the pedestrians intend to occupy the robot's safe space, evasive trajectories are planned and executed while tracking a reference position given by $p_t^r = [0\ 0\ 1.5]\ m$.

The experiment is conducted in a flying arena of $[4\ 3\ 3]\ m$ equipped with an Optitrack[3] motion capture system, which provides raw pose measurements of the robot and the obstacles. These poses are processed by Extended Kalman Filter (EKF) algorithms [Sanchez-Lopez et al., 2017] according to the robot (6.29) and the obstacle (6.31) models. Gaussian model disturbances in linear and angular velocities have been considered as $\sigma^2(v_t) = 0.03\ m^2/s^2$ and $\sigma^2(\dot{\psi}_t) = 0.03\ rad^2/s^2$ for the robot and the obstacles. The measurement noise on position has been identified to be $\sigma^2(p_t) = \sigma^2(q_t^i) = 2.5 \cdot 10^{-3} m^2$. The bounding boxes around the pedestrians are defined by $d_t^i = [2\ 2\ 4]\ m$ with confidence level $1 - \alpha = 0.99$ and uniform risk allocation $\alpha_t^i = \alpha / N N_o$. The real-time implementation of the problem (6.4) with $N = 20$ steps over $4s$ of prediction horizon is based on ACADO Toolkit [Houska et al., 2011] and ROS Kinetic [Quigley et al., 2009] C++ framework running on a on an Intel i7-6820HQ CPU@2.70GHz.

In this work, we include the results over 5 minutes of experiment. The

[2]Benchmark code: https://rebrand.ly/castillo_RAL2020benchmark
[3]Optitrack motion capture system https://optitrack.com/

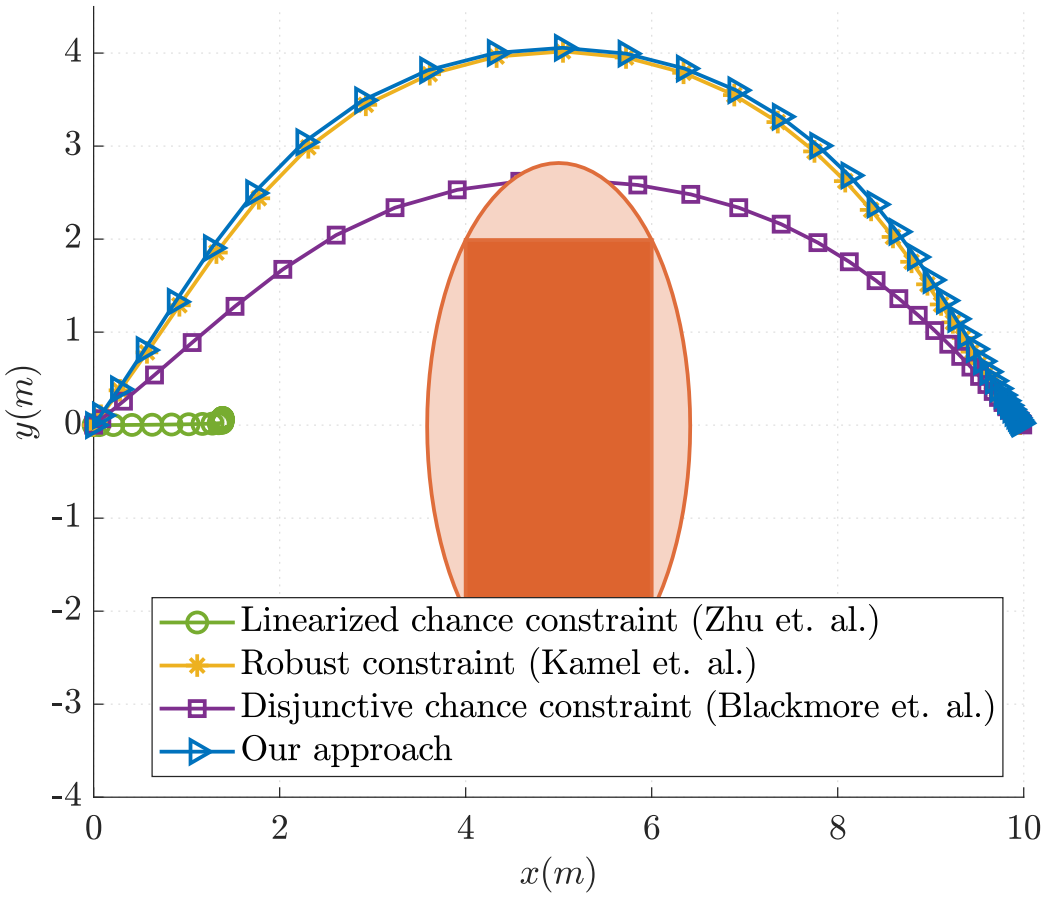


Figure 6.3: One horizon benchmark with increased obstacle dimensions $d = [1\ 2]$

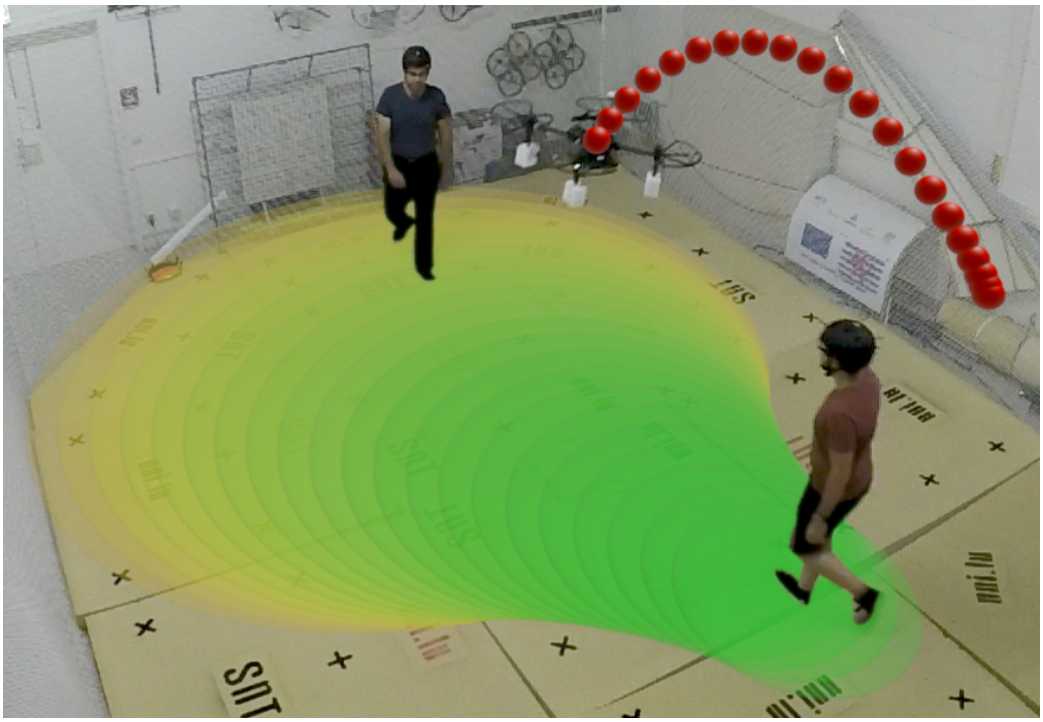


Figure 6.4: An instant of the proposed approach running on an aerial vehicle to avoid pedestrians in a cluttered environment. The planned trajectory has been rendered into the image plane as red balls. The predicted bounding ellipsoids of one pedestrian are projected into the ground as degraded-green ellipses. Video: http://rebrand.ly/castillo_RAL2020

outcome of this experiment in terms of safety are evaluated statistically through the distance to the closest obstacle d and its inverse time-to-collision $TTC^{-1} = \dot{d}/d$ [Van Der Horst and Hogema, 1993]. Large negative values of TTC^{-1} indicate high risk of collision, while values near zero correspond to safe situations [Van Der Horst and Hogema, 1993]. As shown in Fig. 6.5, the robot presented a low risk of collision, since the distance to the closest obstacle lies in the range $[1, 3]$ m with median 1.7 m and the TTC^{-1} values are concentrated around $-0.09s^{-1}$ with a minimum value of -0.4 s^{-1}. In addition, our approach presents fast real-time capabilities with a median control delay of 2.4 ms.

6.6.6 Simulation: Crowd Collision Avoidance

This experiment consists in a software-in-the-loop simulation where the robot navigates in a crowded scenario. 30 pedestrians, driven by the social force

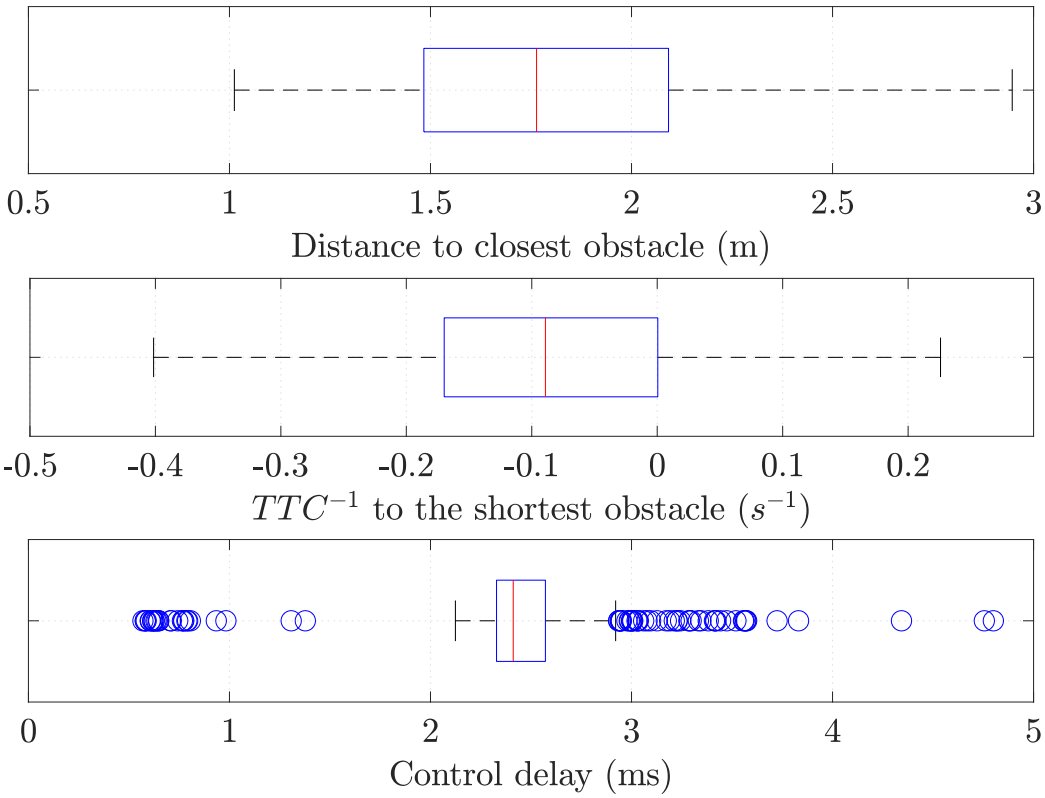


Figure 6.5: Pedestrian Collision Avoidance: Box plots for the distance to the closest obstacle, the inverse time to collision TTC^{-1}, and the control delay. The median is represented in red and the 25-75th percentiles in blue. The black whiskers represent 1.5 times the interquartile range. Outliers are plotted as blue circles

model [Helbing and Molnar, 1995][4], follow a squared path of 14 m length with a reference velocity of 1 m/s. The robot, simulated according to (6.28), is tracking the same path in opposite direction at $1.5m/s$ while avoiding the pedestrians, as shown in Fig. 6.6 and the complementary video. The simulation runs at 100 Hz with the same setup as the experiment conducted in Section 6.6.5.

In this simulation we include the results over 20 minutes of experiment. Analogously to Section 6.6.5, the outcome of this experiment is evaluated statistically through the inverse time to collision (TTC^{-1}), the distance to the closest obstacle and the control delay, as shown in Fig. 6.7. The nature of the experiment and the higher number of obstacles involves a greater risk than the previous experiment, with a median TTC^{-1} of -0.84 s^{-1}. Consequently, our algorithm shows a more conservative behavior, with a median distance to the closest obstacle of 3.22 m. Finally, the higher number of obstacles moderately increases the computation time to 4.2 ms, leaving room to scale up to more complex scenarios.

6.7 Conclusions

We presented a new real-time approach to address chance-constrained motion planning with dynamic obstacles. The obstacles are considered to have uncertain localization, model and disturbances in the form of additive Gaussian noise. We developed a closed-form differentiable bound on the probability of collision to safely approximate the disjunctive chance-constrained optimization problem as a nonlinear program. Consequently, the computational cost was reduced dramatically while maintaining the original safety guarantees, allowing its implementation in fast real-time applications. Through mathematical proof and simulations, our method has shown to reduce conservatism with respect to recent real-time approaches, remaining tractable when accounting for multiple obstacles. Finally, real-time experiments validated the presented approach using nonlinear model predictive control on an aerial robot to avoid pedestrians.

[4]Pedestrian simulator code: https://github.com/srl-freiburg/pedsim_ros

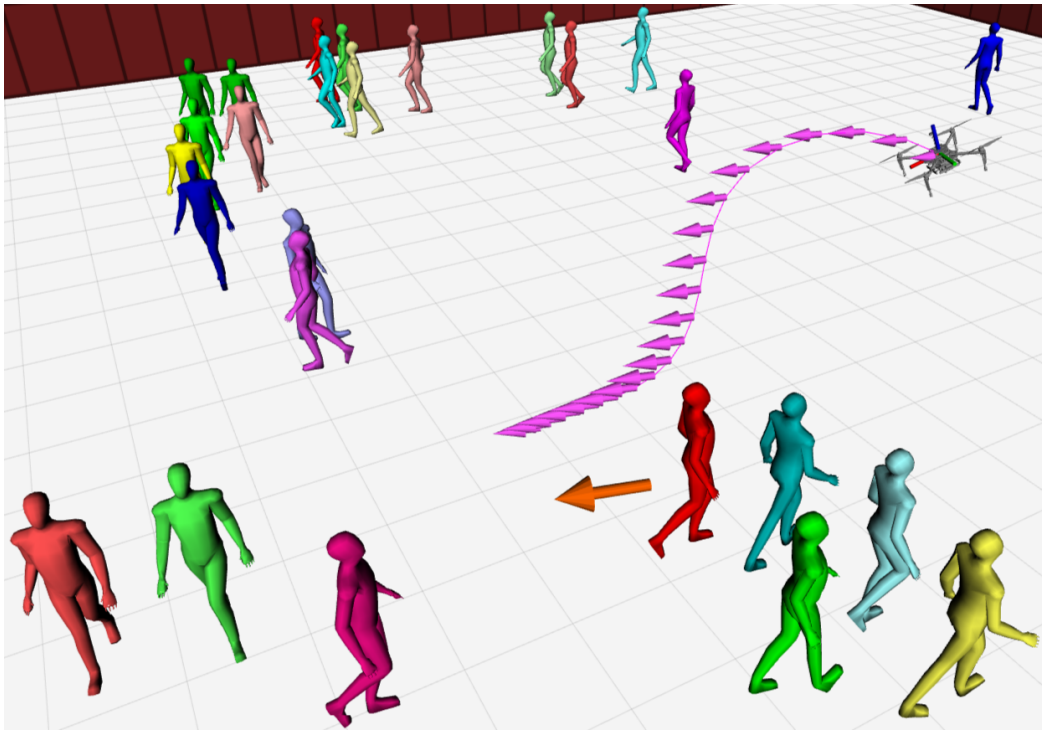


Figure 6.6: Crowd collision avoidance simulation with 30 pedestrians. The orange arrow represents the moving reference position. The robot pose and predicted trajectory are indicated by the frame and the purple arrows respectively.

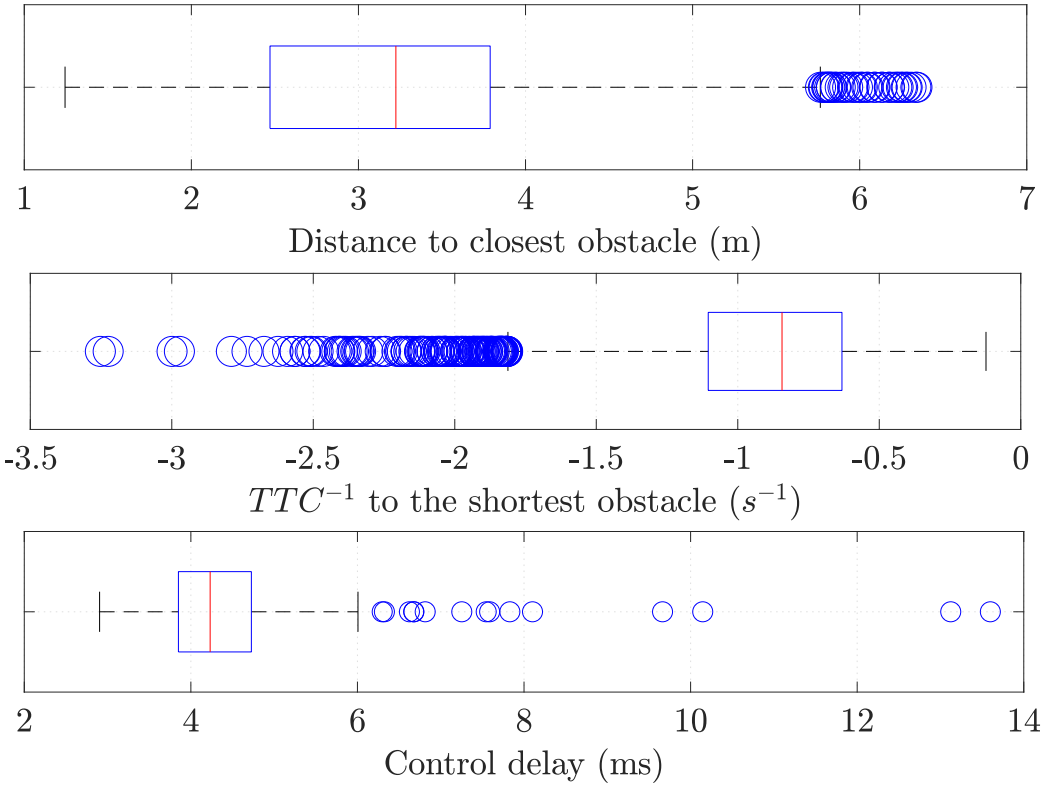


Figure 6.7: Crowd Collision Avoidance: Box plots for the distance to the closest obstacle, the inverse time to collision TTC^{-1}, and the control delay. The median is represented in red and the 25-75th percentiles in blue. The black whiskers represent 1.5 times the interquartile range. Outliers are plotted as blue circles

Chapter 7

Infinite-Horizon Optimal Motion Planning and Control.

Recent advances have settled Model Predictive Control (MPC) as a unified framework for optimal optimal motion planning and control with safety guarantees. However, the fast dynamics and the real-time requirements of autonomous robots such as self-driving cars or drones forces most MPC implementations to have short prediction horizons, reducing their capabilities to anticipate evasive maneuvers. This chapter proposes an infinite-horizon MPC approach with a non-uniform distribution of the planning steps that prioritizes near-future events while relaxing the pursuit of long-term objectives, showing considerable improvements on tracking performance and constraint satisfaction. As a result, safer trajectories are generated on static and dynamic environments under uncertainty. The proposed approach is validated empirically through simulations with static and uncertain dynamic obstacles, such as pedestrians. Additionally, a real experiment on an aerial robot with two pedestrians has been included which, as far as we know, constitutes the first real-time implementation of infinite-horizon MPC for collision avoidance on aerial robots.

7.1 Introduction

Model Predictive Control (MPC) has shown exceptional success for the high-performance control of complex systems [Mesbah, 2016, Maciejowski, 2002]. With the increasing interest of autonomous robots, such as self-driving cars or drones, the leading research is extending MPC capabilities to include obstacle avoidance in the problem formulation [Kamel et al., 2017a, Zhu and

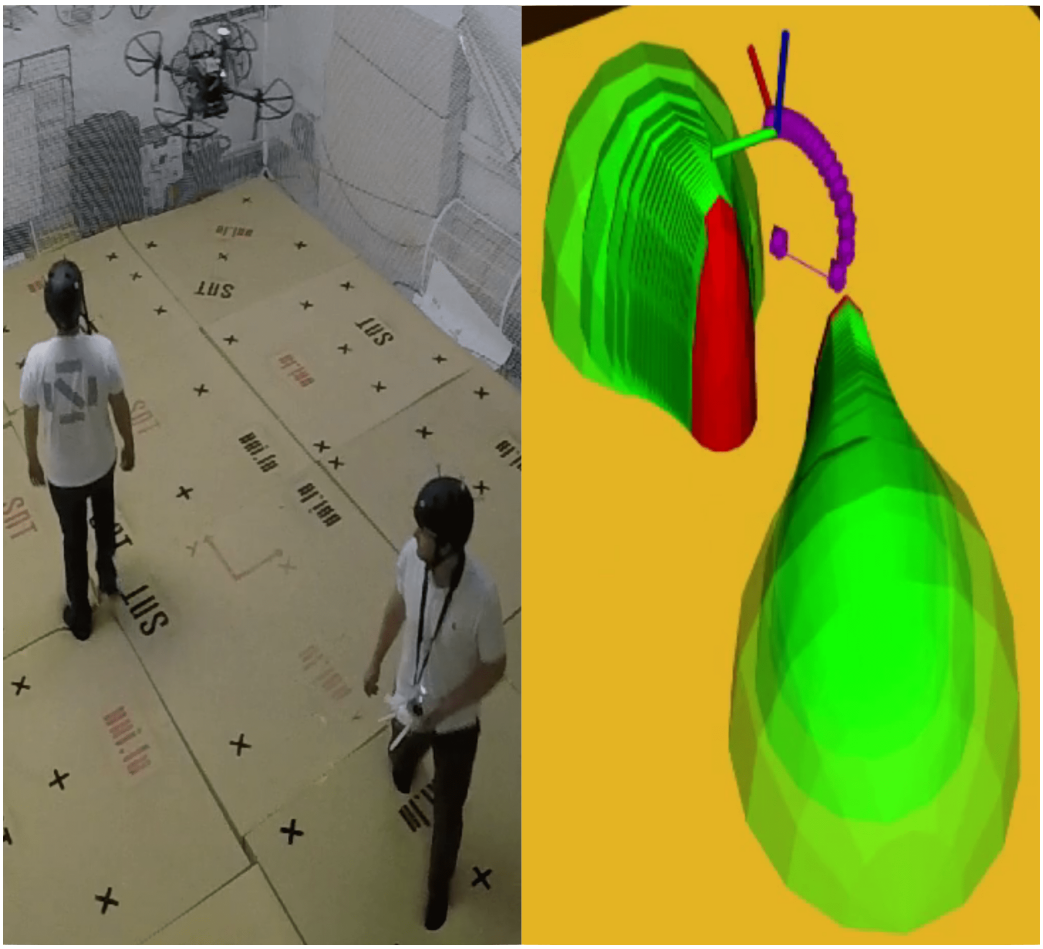


Figure 7.1: An instant of the proposed approach running on an aerial vehicle to avoid two pedestrians. The planned trajectory is shown by purple arrows. The bounding ellipsoids and their prediction are shown in red and green respectively. Video: https://rebrand.ly/castillo_ihmpc

Alonso-Mora, 2019], being able to generate sub-optimal trajectories with safety guarantees [Castillo-Lopez et al., 2020, Lew et al., 2020]. However, the fast dynamics of autonomous robots, forces most implementations to have short prediction horizons [Bicego et al., 2020, Zhu and Alonso-Mora, 2019, Kamel et al., 2017a, Falanga et al., 2018]. As a result, robots suffer a drastic reduction of their planning capabilities, showing reactive behaviors to events only when they affect the immediate future. This lack of anticipation compromises the safety and the efficiency that they were designed to pursue, requiring new approaches to address this issue.

In this chapter, we present a new infinite-horizon MPC approach that extends the planning capabilities of traditional MPC approaches with a low

computational footprint. The core approach is based on techniques borrowed from the pseudospectral optimal control literature to solve infinite-horizon optimal control problems [Fahroo and Ross, 2008, Garg et al., 2011]. Instead of truncating the prediction horizon to a finite value, the infinite horizon is mapped to a compact domain, which results in a non-uniform allocation of the planning steps. When implemented in a receding-horizon scheme, we observe significant improvements on constraint satisfaction, planning and tracking performance. We validate our approach through simulations and real experiments on collision avoidance for aerial robots.

7.2 Related Work

Model Predictive Control (MPC) is a popular technique to numerically approximate optimal control problems which, in general, cannot be solved analytically. Instead of calculating an offline feedback control law over an infinite horizon, MPC numerically addresses a finite-horizon problem in a receding horizon manner [Mayne et al., 2000]. This approach is largely employed for high-performance control of mobile robots [Kamel et al., 2017b, Di Carlo et al., 2018, Kabzan et al., 2019], for which a short prediction horizon is desirable to reduce its complexity [Morari and Lee, 1999]. However, recent contributions are exploiting the predictive nature of MPC to provide a unified framework for optimal motion planning and control with safety guarantees [Castillo-Lopez et al., 2020, Lew et al., 2020, Zhu and Alonso-Mora, 2019]. To that aim, a short horizon is no longer suitable and new methods are required to extend the planning capabilities of MPC approaches without compromising its computational tractability.

An alternative treatment of the horizon can be found in the pseudospectral optimal control literature [Fahroo and Ross, 2008, Ross and Karpenko, 2012, Garg et al., 2011] where, instead of truncating the horizon, the infinite horizon is mapped into a compact domain, and the transformed problem is addressed numerically through collocation methods. Different authors have exploited infinite-horizon formulations in mobile robotics to obtain guarantees on closed-loop stability and recursive feasibility [Mayne et al., 2000]. For instance, [Erez et al., 2012] developed an infinite-horizon MPC approach for contact-based robot control. In [Mehrez et al., 2020], they employ infinite-horizon value approximation to provide closed-loop stability of holonomic ground robots. In [Muehlebach et al., 2017], they developed an infinite-horizon MPC approach to stabilize a rocket-like aerial platform. Even though different contributions have exploited the benefits of infinite-horizon formulations for regulation purposes, none of them have explored their benefits in

motion planning applications such as obstacle avoidance. In this chapter, we attack this problem by developing an infinite-horizon MPC approach for real-time collision avoidance in aerial robots.

7.3 Infinite-Horizon Optimal Control Problem

Let us consider the following Infinite-Horizon Optimal Control Problem:

$$\min_{x(t),u(t)} \int_0^\infty g(x(t),u(t))dt \tag{7.1a}$$

subject to:

$$x(0) = \hat{x}_0 \tag{7.1b}$$

$$\dot{x}(t) = f(x(t),u(t)) \tag{7.1c}$$

$$0 \geq h(x(t),u(t)) \tag{7.1d}$$

where $x(t) \in \mathbb{R}^{n_x}$ and $u(t) \in \mathbb{R}^{n_u}$ are the state and control trajectories over the prediction horizon $t \in \mathbb{R}^+$. The optimal state $x^*(t)$ and control $u^*(t)$ trajectories are obtained by minimizing the cost functional (7.1a) while satisfying initial conditions (7.1b), system dynamics (7.1c) and path constraints (7.1d). g, f and h are possibly nonlinear differentiable functions.

To map the infinite horizon to a finite one, arbitrary domain transformations have been proposed in the literature [Fahroo and Ross, 2008,Garg et al., 2011]. In this work, we aim to provide a model-based domain transformation, where the domain transformation is induced by the system dynamics. To that aim, we assume that the system is linear or exponentially stable, which is reasonable for aerial robots, since modern control approaches provide exponential stability while tracking a reference state [Lee et al., 2010,Gamagedara et al., 2019]. Therefore, the error metric of our system $r(t) \in \mathbb{R}$ will present an exponential decay with a time constant $\lambda \in \mathbb{R}^+$ as shown in Fig. 7.2. Mathematically, the error metric is expressed as follows:

$$r(t) = r(0)exp\left\{-\frac{t}{\lambda}\right\} \tag{7.2}$$

Then, with $\alpha \in \mathbb{N}^+$, we propose the following domain transformation

$$\tau = 1 - exp\left\{-\frac{t}{\alpha\lambda}\right\} \iff t = \alpha\lambda log\left(\frac{1}{1-\tau}\right) \tag{7.3}$$

which maps the time variable $t \in [0,\infty)$ into $\tau \in [0,1)$ by adding the tuning parameter $\alpha \in \mathbb{N}^+$ as a multiplier to the time constant of the system.

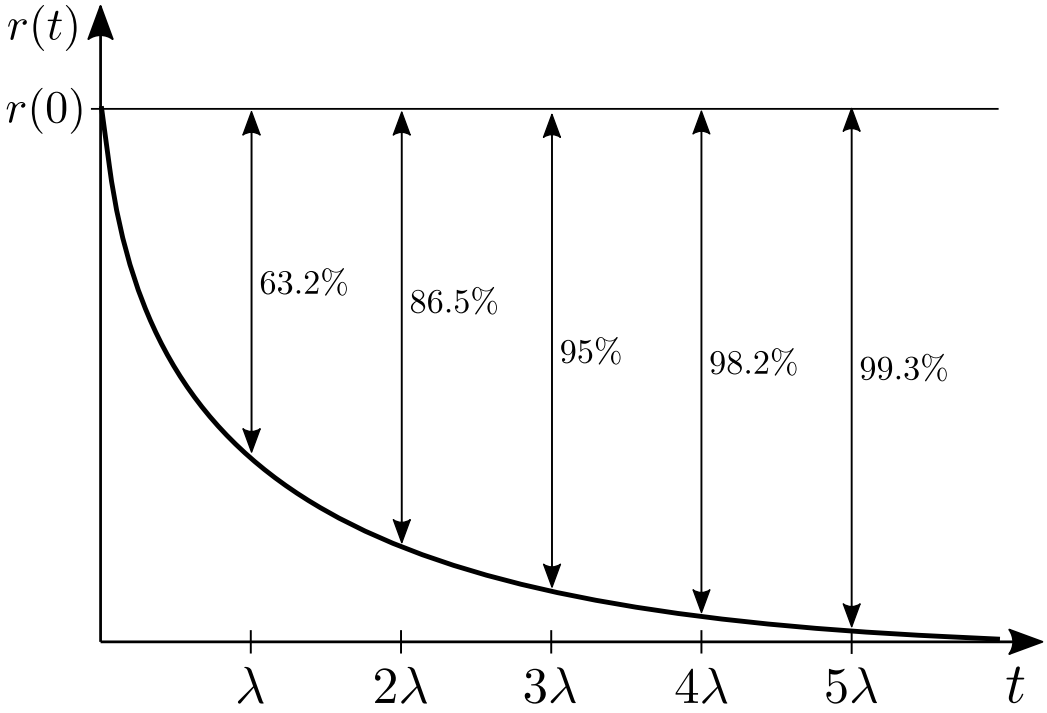


Figure 7.2: Exponential decay of the error dynamics $r(t)$ with initial value $r(0)$ and time constant λ.

When applying the domain transformation to the error metric, we obtain a polynomial expression as follows:

$$r(\tau) = r(0)(1-\tau)^{\alpha} \tag{7.4}$$

Thus, by uniform sampling of the time variable $\tau_k = k/N$, the parameter α can be tuned such that

$$\frac{d^{\alpha+1}}{d\tau^{\alpha+1}} r(\tau) = 0 \Rightarrow \frac{d^{\alpha}}{d\tau^{\alpha}} r(\tau_{k+1}) = \frac{d^{\alpha}}{d\tau^{\alpha}} r(\tau_k) \tag{7.5}$$

for $k = 0, 1, \ldots, N-1$. Due to the singularity at $\tau = 1$, we can only approach it in the sense of a limit, rendering a numerically finite horizon [Fahroo and Ross, 2008]. Thus, the last time sample is defined as $\tau_N = 1 - \varepsilon$, where ε need to be chosen sufficiently small to nullify the cost of the OCP at the end of the horizon, and preserve the infinite-horizon equivalence [Mayne et al., 2000].

The use of the proposed discretization already leads to improvements in numerical simulation. Fig. 7.3 shows the result of applying the proposed time grid ($\alpha = 4$) to simulate the unitary impulse response of the error metric ($\lambda = 1$) by Euler integration. One can see that, for a given number of discretization steps ($N = 40$), the proposed grid significantly reduces the integration error, providing a finer discretization grid during the transient response of the system.

As shown in [Garg et al., 2011, Fahroo and Ross, 2008], we can apply the domain transformation (7.3) to the OCP (7.1), to obtain the equivalent finite-horizon OCP as follows:

$$\min_{x(\tau),u(\tau)} \int_0^1 \phi(\tau) g(x(\tau), u(\tau)) d\tau \tag{7.6a}$$

subject to:

$$x(0) = \hat{x}_0 \tag{7.6b}$$

$$\dot{x}(\tau) = f(x(\tau), u(\tau))\phi(\tau) \tag{7.6c}$$

$$0 \geq h(x(\tau), u(\tau)) \tag{7.6d}$$

where

$$\phi(\tau) := \frac{dt}{d\tau} = \frac{\alpha\lambda}{1-\tau} \tag{7.7}$$

To address the OCP (7.6) numerically, we perform a multiple shooting transcription of the OCP over the horizon $[0, 1-\epsilon]$, which results in the

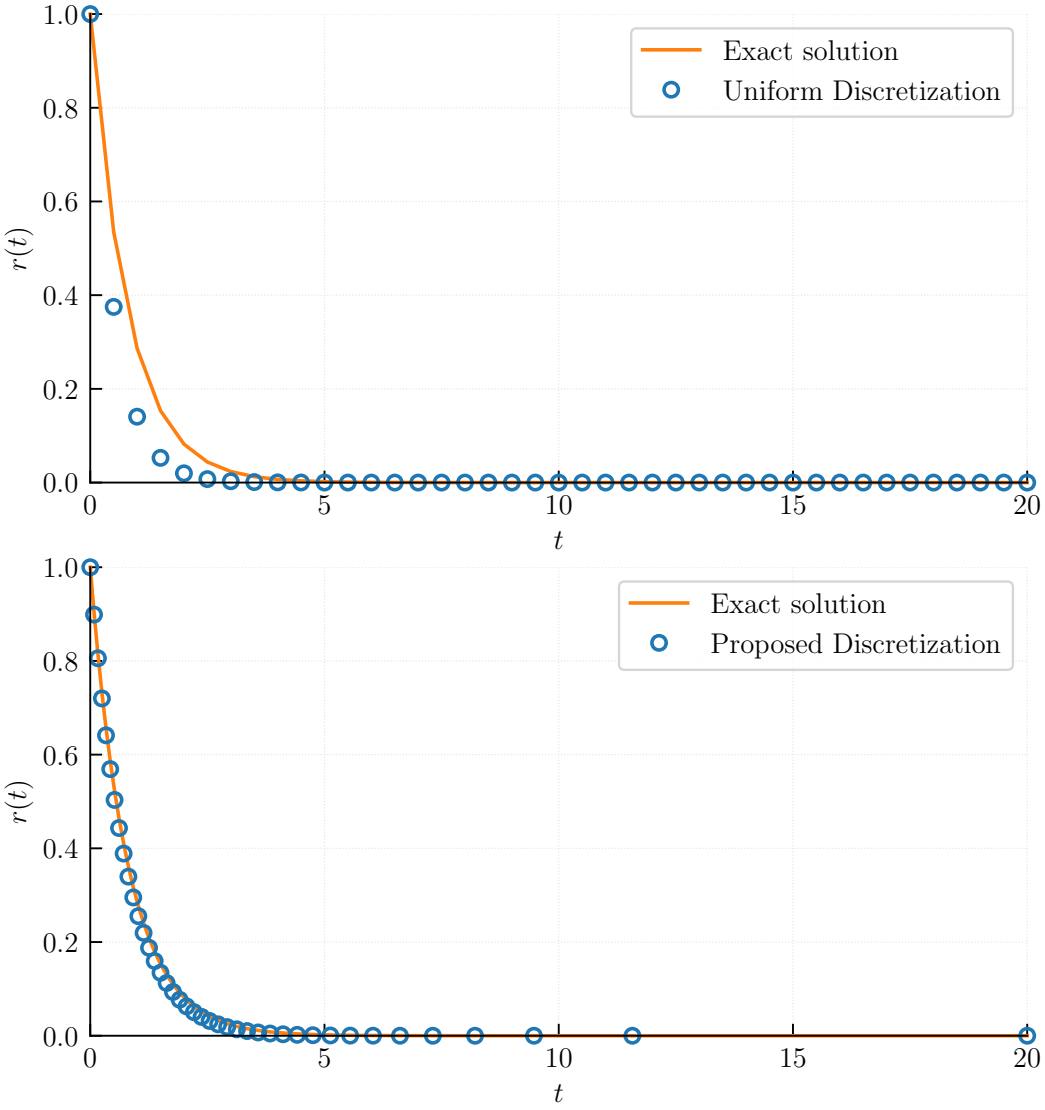


Figure 7.3: Uniform discretization grid (above) vs proposed discretization grid with $\alpha = 4$ (below), both with 40 discretization steps.

nonlinear program:

$$\min_{X,U}\ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k) \tag{7.8a}$$

subject to:

$$x_0 = \hat{x}_0 \tag{7.8b}$$

$$x_{k+1} = f_k(x_k, u_k) \tag{7.8c}$$

$$0 \geq h(x_k, u_k) \qquad k = 0, \ldots, N-1 \tag{7.8d}$$

where $g_k(x_k, u_k)$ denotes the Riemman sum approximation of (7.6a) and $f_k(x_k, u_k)$ represents the simulation of the nonlinear dynamics (7.6c) over each shooting interval, employing a collocation integration scheme to ensure global convergence [Quirynen et al., 2015a]. Note that, to maintain the infinite-horizon equivalence for the computational domain $[0, 1-\varepsilon]$, ε must be sufficiently small to nullify the stage cost at the end of the horizon $g_N(x_N)$ [Mayne et al., 2000].

Since the evaluation of $\phi(\tau)$ near the end of the horizon may lead to numerical instability during optimization, we employed a piecewise constant approximation of $\phi(\tau)$ given by:

$$\phi(\tau) \approx \phi_k \qquad \tau_k \leq \tau < \tau_{k+1} \tag{7.9}$$

where we ϕ_k is the average value of $\phi(\tau)$ on each shooting interval.

$$\begin{aligned} \phi_k :&= \frac{1}{\tau_{k+1} - \tau_k} \int_{\tau_k}^{\tau_{k+1}} \phi(\tau) d\tau \\ &= \frac{\alpha\lambda}{\tau_{k+1} - \tau_k} log\left(\frac{1-\tau_k}{1-\tau_{k+1}}\right) \qquad k = 0, \ldots, N-1 \end{aligned} \tag{7.10}$$

7.4 Implementation for Collision Avoidance in Aerial Robots

In this section, we implement the proposed methodology to design a Model Predictive Control approach for stochastic collision avoidance on aerial robots. We build upon our previous approach [Castillo-Lopez et al., 2020] to generate evasive trajectories with safety guarantees.

7.4.1 Robot model

Based on the SDK provided by DJI-M100[1] aerial robot, the control inputs given to the quadrotor are defined as $u = [u_x\ u_y\ u_z\ u_\omega]^T$, which corresponds to forward, sideward, upward, and heading velocity references respectively based on a local frame L parallel to the ground (see [Castillo-Lopez et al., 2018] for details). Thus, we can define the nominal system dynamics as follows:

$$\dot{p} = R(\psi)v \tag{7.11a}$$
$$\dot{\psi} = \omega \tag{7.11b}$$
$$\dot{v}_i = (-v_i + k_i u_i)/\delta_i, \quad i \in \{x, y, z\} \tag{7.11c}$$
$$\dot{\omega} = (-\omega + k_\omega u_\omega)/\delta_\omega \tag{7.11d}$$

where $p = [p_x\ p_y\ p_z]^T$ is the robot's position in the world frame, $v = [v_x\ v_y\ v_z]^T$ is its linear velocity in the local frame and $R(\psi)$ is the rotation matrix for the yaw angle ψ. k_i, k_ω and δ_i, δ_ω are the gain and time constants relative to each component of u respectively. Thus, the robot state is defined as $x_k = [p_k\ v_k\ \psi_k\ \omega_k]$ and the discrete dynamics $f_k(x_k, u_k)$ are obtained through 4-th order Gauss-Legendre integration of (7.11).

7.4.2 Obstacle Model

Obstacles are modeled with constant velocity nominal dynamics:

$$\dot{q}^i = R(\psi^i)v^i, \qquad \dot{v}^i = \ddot{\psi}^i = 0 \tag{7.12}$$

where q^i, v^i and ψ^i are the position in the world frame, linear velocity in the body frame and yaw angle of the i-th obstacle respectively. Thus, obstacle states are defined as $y^i = [q^i\ v^i\ \psi^i\ \dot{\psi}^i]$ where the nominal discrete dynamics are determined through Euler integration of (7.12).

7.4.3 Obstacle Chance Constraints

According to [Castillo-Lopez et al., 2020], if we consider the robot and obstacle models with additive Gaussian noise, including the constraint (7.13) in the OCP guarantees that the probability of collision with N_o obstacles is below a determined risk level β

$$\sum_j \left(\frac{p_j - q_j^i}{d_j^i + \Psi^{-1}(1 - \beta/NN_o)\sqrt{\sigma^2(p_j) + \sigma^2(q_j^i)}} \right)^2 \geq 3 \tag{7.13}$$

[1] DJI Matrice 100: https://www.dji.com/matrice100

where j iterates over the Cartesian coordinates $\{x, y, z\}$ of each $\mathbb{R}^3$ element and the i-th obstacle is bounded by a cuboid with semi-sizes $d^i \in \mathbb{R}^3$. $\Psi(\cdot)$ is the standard Gaussian cumulative distribution function and $\sigma^2(p)$ and $\sigma^2(q^i)$ correspond to the position covariance of the robot and the i-th obstacle respectively. The nominal position and orientation of the obstacles are propagated along the prediction horizon according to (7.12). The covariance of the robot and the obstacles are propagated by first-order Taylor expansion, as shown in [Castillo-Lopez et al., 2020].

7.4.4 Objective Function

We define the cost function as:

$$g_k(x_k, u_k) = \sum_{k=0}^{N-1} \left(\|x_{k+1} - x^r_{k+1}\|^2_P + \|u_k\|^2_Q \right) \tag{7.14a}$$

$$g_N(x_N) = \|x_N - x^r_N\|^2_P \tag{7.14b}$$

where x^r_k is the user-defined goal state. $\|\cdot\|_P$ and $\|\cdot\|_Q$ are the norms induced by the weighting matrices P and Q.

7.5 Experimental Validation

In this section we perform a series of simulations and experiments that validate our approach and shows its benefits regarding constraint satisfaction, planning and tracking performance. Since [Castillo-Lopez et al., 2020] is already benchmarked with state-of-the-art MPC approaches for collision avoidance, we compare our proposed treatment of the horizon against two finite-horizon schemes, all of them with the same optimal control problem formulation, as described in the previous section. Using $N = 20$ discretization steps, our approach is set with $t_N = 200$ s of prediction horizon. Then, one finite-horizon scheme is set with horizon 1.64 s to match the smallest sample time of the proposed scheme. The second implementation corresponds to [Castillo-Lopez et al., 2020], which includes a longer horizon of $4s$. The real-time implementation is based on ACADO Toolkit [Quirynen et al., 2015b] and ROS melodic [Quigley et al., 2009] C++ framework running on an Intel i7-6820HQ CPU@2.70GHz. The results of the experiments are complemented by the video demonstration https://rebrand.ly/castillo_ihmpc.

7.5.1 Simulation: Static Obstacle Avoidance

In this experiment, we define a scenario with 30 static cylindrical obstacles with 1 m radius plus an additional meter to account for the robot size as shown in Fig. 7.4. From a steady state at the coordinates $(0, 0, 1)$ m the aerial robot has to navigate 40 m in the x-axis direction and avoid the obstacles.

In Fig. 7.4 and Table 7.1 we can see the different results generated by each MPC scheme. On the one hand, the long horizon scheme (4s) from [Castillo-Lopez et al., 2020] anticipates well the avoidance maneuvers, but the larger separation between time samples leads to over 66% deeper constraint violations. On the other hand, the finer discretization grid of the short horizon scheme (1.64s) considerably reduces the constraint violations at the price of reactive trajectories that present 65% larger deviations with respect to a straight path. Unlike these finite-horizon schemes, our approach is able to perform efficient maneuvers with moderate constraint violations (4.7% with respect to the diameter of the cylindrical obstacle) with a control delay of 8.4 ms, remaining suitable for real-time application on embedded platforms.

Table 7.1: Relative results from the static obstacle avoidance simulation. It includes the median control delay (MCD), the median constraint violation (MCV) and the root-mean-square error (RMSE) with respect to the reference path (straight line).

	MCD	MCV	RMSE
Infinite Horizon (200s)	1.0	1.0	1.0
Finite Horizon (1.64s)	0.9721	1.1191	1.6500
Finite Horizon (4s)	0.9972	1.6679	0.9429

7.5.2 Simulation: Dynamic Obstacle Avoidance

In this experiment we run a simulation over 10 minutes where the robot navigates in a crowded environment under uncertainty. 30 pedestrians, driven by the social force model [Helbing and Molnar, 1995], follow a squared path of 14 m length with a reference velocity of 1 m/s. The robot, simulated according to (7.11), is tracking the same path in opposite direction at 1.5 m/s while avoiding the pedestrians, as shown in Fig. 7.5 and the complementary video. The simulation runs at 100 Hz, which provides raw pose measurements of the robot and the obstacles. These poses are processed by Extended Kalman Filter (EKF) algorithms [Sanchez-Lopez et al., 2017] according to the robot (7.11) and the obstacle (7.12) models. Gaussian model disturbances in linear and angular velocities have been considered as $\sigma^2(v) = 0.03\ m^2/s^2$ and

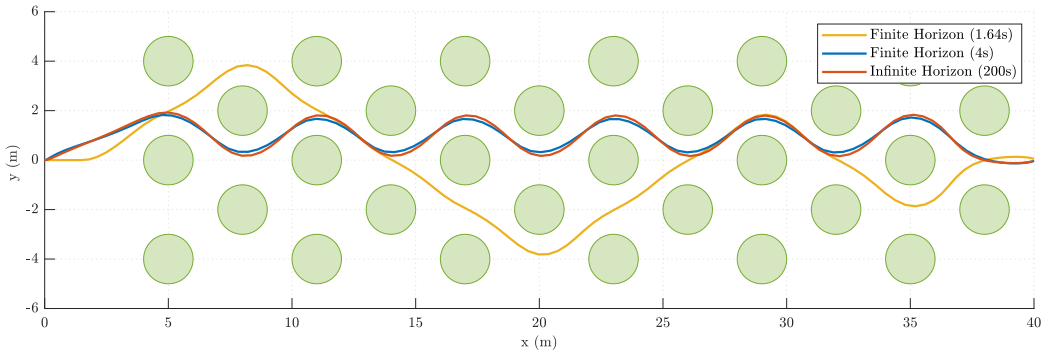


Figure 7.4: Trajectory of the aerial robot during the static collision avoidance simulation running the MPC approach proposed by [Castillo-Lopez et al., 2020] with three different treatments of the horizon. The proposed infinite-horizon approach (200s), the one employed in [Castillo-Lopez et al., 2020] (4s), and a prediction horizon with the same resolution as the first step of the infinite-horizon scheme (1.64s).

$\sigma^2(\omega^i) = 0.03\ rad^2/s^2$ for the robot and the obstacles. The measurement noise on position has been considered to be $\sigma^2(p) = \sigma^2(q^i) = 2.5 \cdot 10^{-3}\ m^2$. The bounding boxes around the pedestrians are defined by $d^i = [2\ 2\ 4]\ m$ with confidence level $1 - \beta = 0.99$.

The outcome of this experiment in terms of safety are evaluated statistically through the distance to the closest obstacle d and its inverse time-to-collision $TTC^{-1} = \dot{d}/d$ [Van Der Horst and Hogema, 1993]. Large negative values of TTC^{-1} indicate high risk of collision, while values near zero correspond to safe situations [Van Der Horst and Hogema, 1993]. As shown in Table 7.2, our approach outperforms both MPC schemes in terms of safety, with median distance to the closest obstacle of 3.46 m, median TTC^{-1} of $-0.72\ s^{-1}$ and median control delay of 13.73 ms. Therefore, our approach remains suitable for real-time collision avoidance with multiple obstacles. Note that further research is needed for long-term pedestrian prediction and uncertainty propagation, which directly impacts the planning performance of our algorithm.

7.5.3 Real experiment: Pedestrian collision avoidance.

The experiment consists in two pedestrians who naturally walk inside a closed area where the robot is operating. As shown in Fig. 7.1 and the complementary video, when the pedestrians intend to occupy the robot's safe space, evasive trajectories are planned and executed while tracking a reference position given by $p^r = [0\ 0\ 1.5]\ m$.

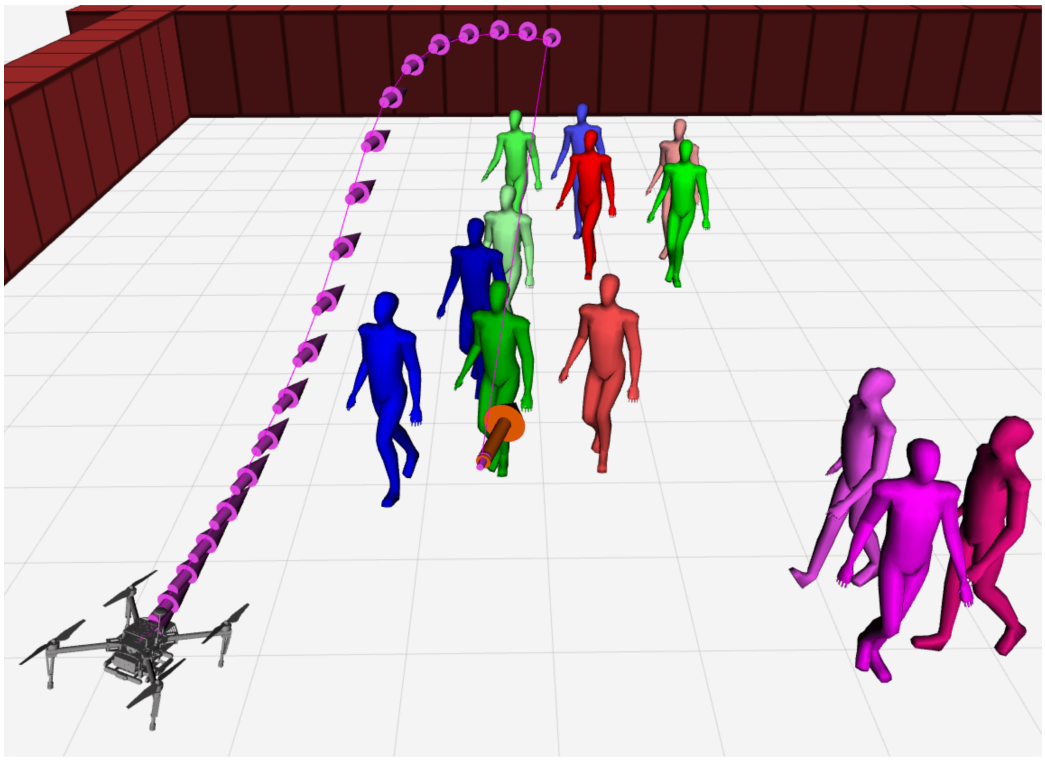


Figure 7.5: Dynamic obstacle avoidance simulation with 30 pedestrians. The orange arrow represents the moving reference position. The predicted trajectory of the robot is indicated by the purple arrows.

Table 7.2: Relative results from the obstacle avoidance benchmark. Includes the median control delay (MCD), the median inverse time to collision ($MTTC^{-1}$) and the median distance to the closest obstacle (MDCO).

	MCD	$MTTC^{-1}$	MDCO
Infinite Horizon (200s)	1.0	1.0	1.0
Finite Horizon (1.64s)	0.8004	1.1667	0.8769
Finite Horizon (4s)	1.0160	1.0278	0.9682

The experiment is conducted in a flying arena of [4 3 3] m equipped with an Optitrack[2] motion capture system, which provides raw pose measurements of the robot and the obstacles. We employ the same experimental setup to process the pose measurements as in Section 7.5.2. Analogously, the outcome of the experiment is evaluated statistically through the inverse time to collision (TTC^{-1}), the distance to the closest obstacle and the control delay. As shown in Fig. 7.6, the robot presented a relatively low risk of collision, since the distance to the closest obstacle lies in the range $[0.64, 2.6]$ m with median 1.33 m and the TTC^{-1} values are concentrated around -0.15 s^{-1} with a minimum value of -0.8 s^{-1}. In addition, our approach presents fast real-time capabilities with a median control delay of 4.52 ms.

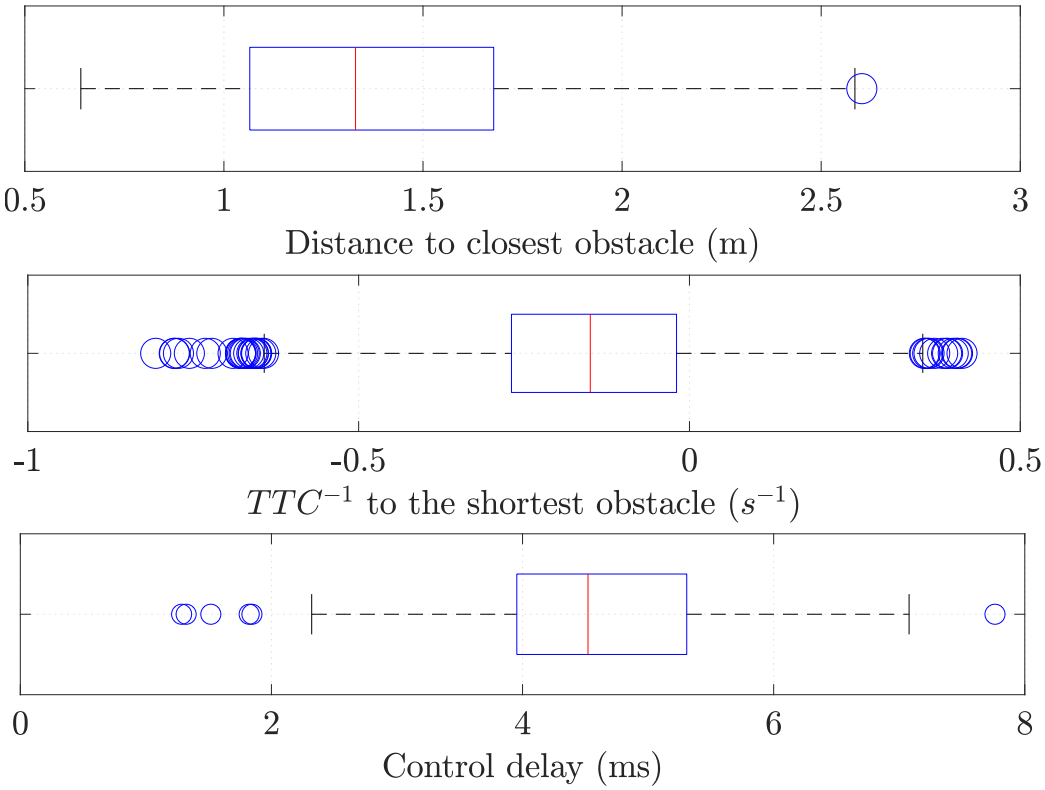


Figure 7.6: Pedestrian Collision Avoidance: Box plots for the distance to the closest obstacle, the inverse time to collision TTC^{-1}, and the control delay. The median is represented in red and the 25-75th percentiles in blue. The black whiskers represent 1.5 times the interquartile range. Outliers are plotted as blue circles.

[2]Optitrack motion capture system https://optitrack.com/

7.6 Conclusions

We presented a new infinite-horizon Model Predictive Control approach for stochastic collision avoidance with static and/or dynamic obstacles. As far as we know, this constitutes the first real-time implementation of infinite-horizon MPC for collision avoidance on aerial robots. Our approach presents a non-uniform allocation of the planning steps that prioritizes near-future events while relaxing the pursuit of long-term objectives. As a result, significant improvements on constraint satisfaction, planning performance and safety have been observed during real-time simulations with static obstacles. However, minor improvements have been observed for pedestrian avoidance. The main reason behind this issue lies in our ability to produce reliable long-term pedestrian predictions, which remains a challenge and the subject of future work.

Chapter 8

Conclusions and Future Work.

This thesis provided different contributions to the state of the art towards optimal motion planning and control with safety guarantees for aerial robots. To that aim, we pushed the boundaries of Model Predictive Control beyond regulation purposes by adding multiple obstacles in the problem formulation. Unifying planning and control pipelines led to different contributions to the existing literature, listed as follows:

- Efficient and flexible obstacle representations. We included orientable ellipsoids as an alternative representation to spheres, which was the main approach proposed in the existing literature. This resulted in a reduction of the obstacle space that allowed real-time optimal motion planning and control in crowded environments.

- Inclusion of obstacle dynamics. Modeling obstacle dynamics with additional decision variables in the optimization problem did not allow real-time execution, so we represented each obstacle by a sequence of ellipsoids that spanned their expected occupied space. This approach proved to outperform previous approaches while remaining tractable and scalable in real applications.

- Real-time optimal motion planning and control with safety guarantees. This problem was already addressed in the literature. However, the existing formulations never reached real-time application on aerial robots due to their high computational cost and poor scalability. We provided an efficient and scalable approximation to this problem while preserving the original safety guarantees, which allowed its application in real experiments with multiple dynamic obstacles such as pedestrians.

- Infinite-Horizon optimal motion planning and control. In practice, the computational limitations of aerial robots led to algorithms with short prediction horizons and poor planning capabilities. We built upon techniques from the pseudospectral optimal control literature, to propose a new MPC approach with a non-uniform distribution of the planning steps that allowed to increase the prediction horizon with minimal computational footprint. The results have shown considerable improvements on tracking performance, safety and constraint satisfaction with static obstacles, while providing minor improvements when applied to uncertain dynamic obstacles such as pedestrians. The challenge of accurately predicting these pedestrians was influencing negatively to the performance to our algorithm, requiring further research.

These contributions have been materialized into three publications [Castillo-Lopez et al., 2017, Castillo-Lopez et al., 2018, Castillo-Lopez et al., 2020] plus Chapter 7, which remains to be published. Additionally, we have participated in three publications for global optimal planning [Sanchez-Lopez et al., 2018,Sanchez-Lopez et al., 2020a] and situational awareness [Sanchez-Lopez et al., 2020b].

Different future lines of research can be considered from this stage, which are listed as follows:

- Modeling: Combining physics-driven with data-driven models has shown to boost performance on regulation problems [Torrente et al., 2021, Hewing et al., 2018]. We consider to be a promising line of research to apply those methods to optimal motion planning and control with safety guarantees.

- Reducing conservatism: Different techniques such as closed-loop chance constraint satisfaction [Hewing et al., 2020], disturbance feedback [Sessa et al., 2018] and the use of multi-modal Gaussian distributions [Ahn et al., 2021] has shown promising results to further reduce conservatism while maintaining safety guarantees.

- Uncertainty propagation: Propagating uncertainty remains one of the biggest challenges for optimal motion planning and control with safety guarantees, since it quickly leads to large reductions of the free space [Mesbah, 2016].

- Improving pedestrian prediction: Accurately predicting pedestrians remains difficult. However, exploiting information about the environment [Kooij et al., 2014] and their expected goals [Rehder and Kloeden, 2015] shows promising results that could be exploited by this work.

- Including perception objectives: Actively accounting for perception capabilities has shown to improve performance for regulation purposes [Falanga et al., 2018, Jacquet and Franchi, 2020]. It is expected to improve safety, but remains open for future work.

Bibliography

[Abdolhosseini et al., 2013] Abdolhosseini, M., Zhang, Y., and Rabbath, C. A. (2013). An efficient model predictive control scheme for an unmanned quadrotor helicopter. *Journal of intelligent & robotic systems*, pages 1–12.

[Ahn et al., 2021] Ahn, H., Chen, C., Mitchell, I. M., and Kamgarpour, M. (2021). Safe motion planning against multimodal distributions based on a scenario approach. *IEEE Control Systems Letters*.

[Albersmeyer and Diehl, 2010] Albersmeyer, J. and Diehl, M. (2010). The lifted newton method and its application in optimization. *SIAM Journal on Optimization*, 20(3):1655–1684.

[Alexis et al., 2016] Alexis, K., Papachristos, C., Siegwart, R., and Tzes, A. (2016). Robust model predictive flight control of unmanned rotorcrafts. *Journal of Intelligent & Robotic Systems*, 81(3-4):443–469.

[Andersson et al., 2018] Andersson, J. A. E. et al. (2018). CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*.

[Andersson et al., 2016] Andersson, O., Wzorek, M., Rudol, P., and Doherty, P. (2016). Model-predictive control with stochastic collision avoidance using bayesian policy optimization. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 4597–4604. IEEE.

[Balas, 2018] Balas, E. (2018). *Disjunctive Programming*. Springer.

[Bertsekas, 2019] Bertsekas, D. P. (2019). *Reinforcement learning and optimal control*. Athena Scientific Belmont, MA.

[Bicego et al., 2020] Bicego, D., Mazzetto, J., Carli, R., Farina, M., and Franchi, A. (2020). Nonlinear model predictive control with enhanced actuator model for multi-rotor aerial vehicles with generic designs. *Journal of Intelligent & Robotic Systems*, 100(3):1213–1247.

[Biegler and Zavala, 2009] Biegler, L. T. and Zavala, V. M. (2009). Large-scale nonlinear programming using ipopt: An integrating framework for enterprise-wide dynamic optimization. *Computers & Chemical Engineering*.

[Blackmore et al., 2011] Blackmore, L. et al. (2011). Chance-constrained optimal path planning with obstacles. *IEEE Transactions on Robotics*.

[Blackmore et al., 2010] Blackmore, L., Ono, M., Bektassov, A., and Williams, B. C. (2010). A probabilistic particle-control approximation of chance-constrained stochastic predictive control. *IEEE transactions on Robotics*.

[Bouffard et al., 2012] Bouffard, P., Aswani, A., and Tomlin, C. (2012). Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 279–284. IEEE.

[Cabreira et al., 2018] Cabreira, T. M., Di Franco, C., Ferreira, P. R., and Buttazzo, G. C. (2018). Energy-aware spiral coverage path planning for uav photogrammetric applications. *IEEE Robotics and Automation Letters*, 3(4):3662–3668.

[Castillo-Lopez et al., 2017] Castillo-Lopez, M. et al. (2017). Evasive maneuvering for uavs: An mpc approach. In *Iberian Robotics conference*, pages 829–840. Springer.

[Castillo-Lopez et al., 2018] Castillo-Lopez, M. et al. (2018). Model predictive control for aerial collision avoidance in dynamic environments. In *2018 26th Mediterranean Conference on Control and Automation (MED)*. IEEE.

[Castillo-Lopez et al., 2020] Castillo-Lopez, M., Ludivig, P., Sajadi-Alamdari, S. A., Sanchez-Lopez, J. L., Olivares-Mendez, M. A., and Voos, H. (2020). A real-time approach for chance-constrained motion planning with dynamic obstacles. *IEEE Robotics and Automation Letters*, 5(2):3620–3625.

[Chung et al., 2018] Chung, S.-J., Paranjape, A. A., Dames, P., Shen, S., and Kumar, V. (2018). A survey on aerial swarm robotics. *IEEE Transactions on Robotics*, 34(4):837–855.

[Dadkhah and Mettler, 2012] Dadkhah, N. and Mettler, B. (2012). Survey of motion planning literature in the presence of uncertainty: Considerations for uav guidance. *Journal of Intelligent & Robotic Systems*, 65(1-4):233–246.

[Dang et al., 2020] Dang, T., Mascarich, F., Khattak, S., Nguyen, H., Nguyen, H., Hirsh, S., Reinhart, R., Papachristos, C., and Alexis, K. (2020). Autonomous search for underground mine rescue using aerial robots. In *2020 IEEE Aerospace Conference*, pages 1–8. IEEE.

[Darivianakis et al., 2014] Darivianakis, G., Alexis, K., Burri, M., and Siegwart, R. (2014). Hybrid predictive control for aerial robotic physical interaction towards inspection operations. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 53–58. IEEE.

[Davison, 2018] Davison, A. J. (2018). Futuremapping: The computational structure of spatial ai systems. *arXiv preprint arXiv:1803.11288*.

[Dentler et al., 2016] Dentler, J., Kannan, S., Mendez, M. A. O., and Voos, H. (2016). A real-time model predictive position control with collision avoidance for commercial low-cost quadrotors. In *Control Applications (CCA), 2016 IEEE Conference on*, pages 519–525. IEEE.

[Desaraju and Michael, 2016] Desaraju, V. R. and Michael, N. (2016). Fast nonlinear model predictive control via partial enumeration. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 1243–1248. IEEE.

[Di Carlo et al., 2018] Di Carlo, J., Wensing, P. M., Katz, B., Bledt, G., and Kim, S. (2018). Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9. IEEE.

[Diehl et al., 2005] Diehl, M., Bock, H. G., and Schlöder, J. P. (2005). A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on control and optimization*, 43(5):1714–1736.

[Domahidi et al., 2012] Domahidi, A., Zgraggen, A. U., Zeilinger, M. N., Morari, M., and Jones, C. N. (2012). Efficient interior point methods for multistage problems arising in receding horizon control. In *Decision and*

Control (CDC), 2012 IEEE 51st Annual Conference on, pages 668–674. IEEE.

[Erez et al., 2012] Erez, T., Tassa, Y., and Todorov, E. (2012). Infinite-horizon model predictive control for periodic tasks with contacts. *Robotics: Science and systems VII*, page 73.

[Fahroo and Ross, 2008] Fahroo, F. and Ross, I. M. (2008). Pseudospectral methods for infinite-horizon nonlinear optimal control problems. *Journal of Guidance, Control, and Dynamics*, 31(4):927–936.

[Falanga et al., 2018] Falanga, D., Foehn, P., Lu, P., and Scaramuzza, D. (2018). Pampc: Perception-aware model predictive control for quadrotors. *arXiv preprint arXiv:1804.04811*.

[Ferreau et al., 2014] Ferreau, H. J., Kirches, C., Potschka, A., Bock, H. G., and Diehl, M. (2014). qpoases: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6(4):327–363.

[Finn et al., 2016] Finn, C., Levine, S., and Abbeel, P. (2016). Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pages 49–58. PMLR.

[Fiorini and Shiller, 1998] Fiorini, P. and Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research*, 17(7):760–772.

[Fulgenzi et al., 2007] Fulgenzi, C., Spalanzani, A., and Laugier, C. (2007). Dynamic obstacle avoidance in uncertain environment combining pvos and occupancy grid. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 1610–1616. IEEE.

[Gamagedara et al., 2019] Gamagedara, K., Bisheban, M., Kaufman, E., and Lee, T. (2019). Geometric controls of a quadrotor uav with decoupled yaw control. In *2019 American Control Conference (ACC)*, pages 3285–3290. IEEE.

[Garg et al., 2011] Garg, D., Hager, W. W., and Rao, A. V. (2011). Pseudospectral methods for solving infinite-horizon optimal control problems. *Automatica*, 47(4):829–837.

[Garimella and Kobilarov, 2015] Garimella, G. and Kobilarov, M. (2015). Towards model-predictive control for aerial pick-and-place. In *Robotics*

and Automation (ICRA), 2015 IEEE International Conference on, pages 4692–4697. IEEE.

[Garimella et al., 2017] Garimella, G., Sheckells, M., and Kobilarov, M. (2017). Robust obstacle avoidance for aerial platforms using adaptive model predictive control. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 5876–5882. IEEE.

[Goerzen et al., 2010] Goerzen, C., Kong, Z., and Mettler, B. (2010). A survey of motion planning algorithms from the perspective of autonomous uav guidance. *Journal of Intelligent and Robotic Systems*, 57(1-4):65–100.

[Goldstein et al., 2002] Goldstein, H., Poole, C., and Safko, J. (2002). Classical mechanics.

[Grüne and Pannek, 2017] Grüne, L. and Pannek, J. (2017). Nonlinear model predictive control. In *Nonlinear Model Predictive Control*, pages 45–69. Springer.

[Helbing and Molnar, 1995] Helbing, D. and Molnar, P. (1995). Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282.

[Hentzen et al., 2018] Hentzen, D. et al. (2018). On maximizing safety in stochastic aircraft trajectory planning with uncertain thunderstorm development. *Aerospace Science and Technology*, 79:543–553.

[Hewing et al., 2018] Hewing, L., Liniger, A., and Zeilinger, M. N. (2018). Cautious nmpc with gaussian process dynamics for autonomous miniature race cars. In *2018 European Control Conference (ECC)*, pages 1341–1348. IEEE.

[Hewing et al., 2020] Hewing, L., Zeilinger, M. N., et al. (2020). Data-driven distributed stochastic model predictive control with closed-loop chance constraint satisfaction. *arXiv preprint arXiv:2004.02907*.

[Houska et al., 2011] Houska, B. et al. (2011). ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization. *Optimal Control Applications and Methods*.

[Houska et al., 2013] Houska, B., Ferreau, H., Vukov, M., and Quirynen, R. (2009–2013). ACADO Toolkit User's Manual. http://acado.github.io.

[Hoy et al., 2015] Hoy, M., Matveev, A. S., and Savkin, A. V. (2015). Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey. *Robotica*, 33(3):463–497.

[Jacquet and Franchi, 2020] Jacquet, M. and Franchi, A. (2020). Motor and perception constrained nmpc for torque-controlled generic aerial vehicles. *IEEE Robotics and Automation Letters*, 6(2):518–525.

[Jalali and Nadimi, 2006] Jalali, A. A. and Nadimi, V. (2006). A survey on robust model predictive control from 1999-2006. In *2006 International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA'06)*, pages 207–207. IEEE.

[Jha et al., 2018] Jha, S., Raman, V., Sadigh, D., and Seshia, S. A. (2018). Safe autonomy under perception uncertainty using chance-constrained temporal logic. *Journal of Automated Reasoning*, 60(1):43–62.

[John, 2014] John, F. (2014). Extremum problems with inequalities as subsidiary conditions. In *Traces and emergence of nonlinear programming*. Springer.

[Kabzan et al., 2019] Kabzan, J., Hewing, L., Liniger, A., and Zeilinger, M. N. (2019). Learning-based model predictive control for autonomous racing. *IEEE Robotics and Automation Letters*, 4(4):3363–3370.

[Kamel et al., 2017a] Kamel, M., Alonso-Mora, J., Siegwart, R., and Nieto, J. (2017a). Robust collision avoidance for multiple micro aerial vehicles using nonlinear model predictive control. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE.

[Kamel et al., 2017b] Kamel, M., Stastny, T., Alexis, K., and Siegwart, R. (2017b). Model predictive control for trajectory tracking of unmanned aerial vehicles using robot operating system. In *Robot operating system (ROS)*, pages 3–39. Springer.

[Kooij et al., 2014] Kooij, J. F. P., Schneider, N., Flohr, F., and Gavrila, D. M. (2014). Context-based pedestrian path prediction. In *European Conference on Computer Vision*, pages 618–633. Springer.

[Lee et al., 2010] Lee, T., Leok, M., and McClamroch, N. H. (2010). Geometric tracking control of a quadrotor uav on se (3). In *49th IEEE conference on decision and control (CDC)*, pages 5420–5425. IEEE.

[Lefkopoulos and Kamgarpour, 2019] Lefkopoulos, V. and Kamgarpour, M. (2019). Using uncertainty data in chance-constrained trajectory planning. In *18th European Control Conference (ECC)*, pages 2264–2269. IEEE.

[Lefkopoulos and Kamgarpour, 2021] Lefkopoulos, V. and Kamgarpour, M. (2021). Trajectory planning under environmental uncertainty with finite-sample safety guarantees. *Automatica*, 131:109754.

[Lew et al., 2020] Lew, T., Bonalli, R., and Pavone, M. (2020). Chance-constrained sequential convex programming for robust trajectory optimization. In *2020 European Control Conference (ECC)*, pages 1871–1878. IEEE.

[Luo and Yang, 2017] Luo, Y.-z. and Yang, Z. (2017). A review of uncertainty propagation in orbital mechanics. *Progress in Aerospace Sciences*, 89:23–39.

[Maciejowski, 2002] Maciejowski, J. M. (2002). *Predictive control: with constraints*. Pearson education.

[Mayne et al., 2000] Mayne, D. Q., Rawlings, J. B., Rao, C. V., and Scokaert, P. O. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814.

[Mehrez et al., 2020] Mehrez, M. W., Worthmann, K., Cenerini, J. P., Osman, M., Melek, W. W., and Jeon, S. (2020). Model predictive control without terminal constraints or costs for holonomic mobile robots. *Robotics and Autonomous Systems*, 127:103468.

[Meier et al., 2015] Meier, L., Honegger, D., and Pollefeys, M. (2015). Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 6235–6240. IEEE.

[Mellinger and Kumar, 2011] Mellinger, D. and Kumar, V. (2011). Minimum snap trajectory generation and control for quadrotors. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2520–2525. IEEE.

[Mesbah, 2016] Mesbah, A. (2016). Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems*.

[Mohamed et al., 2018] Mohamed, N. et al. (2018). Unmanned aerial vehicles applications in future smart cities. *Technological Forecasting and Social Change*.

[Morari and Lee, 1999] Morari, M. and Lee, J. H. (1999). Model predictive control: past, present and future. *Computers & Chemical Engineering*, 23(4-5):667–682.

[Muehlebach et al., 2017] Muehlebach, M., Sferrazza, C., and D'Andrea, R. (2017). Implementation of a parametrized infinite-horizon model predictive control scheme with stability guarantees. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2723–2730. IEEE.

[Mulgaonkar et al., 2014] Mulgaonkar, Y., Whitzer, M., Morgan, B., Kroninger, C. M., Harrington, A. M., and Kumar, V. (2014). Power and weight considerations in small, agile quadrotors. In *Micro-and Nanotechnology Sensors, Systems, and Applications VI*, volume 9083, page 90831Q. International Society for Optics and Photonics.

[Nocedal and Wright, 2006] Nocedal, J. and Wright, S. (2006). *Numerical optimization*. Springer Science & Business Media.

[Oleynikova et al., 2017] Oleynikova, H., Taylor, Z., Fehr, M., Siegwart, R., and Nieto, J. (2017). Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1366–1373. IEEE.

[Oleynikova et al., 2018] Oleynikova, H., Taylor, Z., Siegwart, R., and Nieto, J. (2018). Sparse 3d topological graphs for micro-aerial vehicle planning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9. IEEE.

[Olivares-Mendez et al., 2014] Olivares-Mendez, M. A., Kannan, S., and Voos, H. (2014). Setting up a testbed for uav vision based control using v-rep & ros: A case study on aerial visual inspection. In *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, pages 447–458. IEEE.

[Ono, 2012] Ono, M. (2012). Closed-loop chance-constrained mpc with probabilistic resolvability. In *2012 IEEE 51st IEEE conference on decision and control (CDC)*, pages 2611–2618. IEEE.

[Ono and Williams, 2008] Ono, M. and Williams, B. C. (2008). An efficient motion planning algorithm for stochastic dynamic systems with constraints on probability of failure. In *AAAI*, pages 1376–1382.

[Ono et al., 2013] Ono, M., Williams, B. C., and Blackmore, L. (2013). Probabilistic planning for continuous dynamic systems under bounded risk. *Journal of Artificial Intelligence Research*, 46:511–577.

[Powers et al., 2015] Powers, C., Mellinger, D., and Kumar, V. (2015). Quadrotor kinematics and dynamics. In *Handbook of Unmanned Aerial Vehicles*, pages 307–328. Springer.

[Quan et al., 2020] Quan, L., Han, L., Zhou, B., Shen, S., and Gao, F. (2020). Survey of uav motion planning. *IET Cyber-systems and Robotics*, 2(1):14–21.

[Quigley et al., 2009] Quigley, M. et al. (2009). Ros: an open-source robot operating system. In *ICRA workshop on open source software*. Kobe, Japan.

[Quirynen et al., 2015a] Quirynen, R., Gros, S., and Diehl, M. (2015a). Lifted implicit integrators for direct optimal control. In *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pages 3212–3217. IEEE.

[Quirynen et al., 2014] Quirynen, R., Vukov, M., Zanon, M., and Diehl, M. (2014). Autogenerating Microsecond Solvers for Nonlinear MPC: a Tutorial Using ACADO Integrators. *Optimal Control Applications and Methods*.

[Quirynen et al., 2015b] Quirynen, R., Vukov, M., Zanon, M., and Diehl, M. (2015b). Autogenerating microsecond solvers for nonlinear mpc: a tutorial using acado integrators. *Optimal Control Applications and Methods*, 36(5):685–704.

[Rawlings and Mayne, 2009] Rawlings, J. B. and Mayne, D. Q. (2009). *Model predictive control: Theory and design*. Nob Hill Pub.

[Rehder and Kloeden, 2015] Rehder, E. and Kloeden, H. (2015). Goal-directed pedestrian prediction. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 50–58.

[Rohmer et al., 2013] Rohmer, E., Singh, S. P., and Freese, M. (2013). V-rep: A versatile and scalable robot simulation framework. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1321–1326. IEEE.

[Ross and Karpenko, 2012] Ross, I. M. and Karpenko, M. (2012). A review of pseudospectral optimal control: From theory to flight. *Annual Reviews in Control*, 36(2):182–197.

[Rudenko et al., 2020] Rudenko, A., Palmieri, L., Herman, M., Kitani, K. M., Gavrila, D. M., and Arras, K. O. (2020). Human motion trajectory prediction: A survey. *The International Journal of Robotics Research*, 39(8):895–935.

[Sajadi-Alamdari et al., 2018] Sajadi-Alamdari, S. A., Voos, H., and Darouach, M. (2018). Stochastic optimum energy management for advanced transportation network. In *Control in Transportation Systems (CTS), 2018 IFAC Symposium on*. IFAC.

[Sajadi-Alamdari et al., 2019] Sajadi-Alamdari, S. A., Voos, H., and Darouach, M. (2019). Nonlinear model predictive control for ecological driver assistance systems in electric vehicles. *Robotics and Autonomous Systems*.

[Salas-Moreno et al., 2013] Salas-Moreno, R. F., Newcombe, R. A., Strasdat, H., Kelly, P. H., and Davison, A. J. (2013). Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1352–1359.

[Sanchez-Lopez et al., 2020a] Sanchez-Lopez, J. L., Castillo-Lopez, M., Olivares-Mendez, M. A., and Voos, H. (2020a). Trajectory tracking for aerial robots: an optimization-based planning and control approach. *Journal of Intelligent & Robotic Systems*, 100(2):531–574.

[Sanchez-Lopez et al., 2020b] Sanchez-Lopez, J. L., Castillo-Lopez, M., and Voos, H. (2020b). Semantic situation awareness of ellipse shapes via deep learning for multirotor aerial robots with a 2d lidar. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1014–1023. IEEE.

[Sanchez-Lopez et al., 2017] Sanchez-Lopez, J. L. et al. (2017). Visual marker based multi-sensor fusion state estimation. *IFAC-PapersOnLine*.

[Sanchez-Lopez et al., 2016] Sanchez-Lopez, J. L., Fernández, R. A. S., Bavle, H., Sampedro, C., Molina, M., Pestana, J., and Campoy, P. (2016). Aerostack: An architecture and open-source software framework for aerial robotics. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 332–341. IEEE.

[Sanchez-Lopez et al., 2018] Sanchez-Lopez, J. L., Olivares-Mendez, M. A., Castillo-Lopez, M., and Voos, H. (2018). Towards trajectory planning from a given path for multirotor aerial robots trajectory tracking. In *2018*

International Conference on Unmanned Aircraft Systems (ICUAS), pages 1342–1351. IEEE.

[Sessa et al., 2018] Sessa, P. G., Frick, D., Wood, T. A., and Kamgarpour, M. (2018). From uncertainty data to robust policies for temporal logic planning. In *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control (part of CPS Week)*, pages 157–166.

[Shiller et al., 2001] Shiller, Z., Large, F., and Sekhavat, S. (2001). Motion planning in dynamic environments: Obstacles moving along arbitrary trajectories. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 3716–3721. IEEE.

[Shim et al., 2003] Shim, D. H., Kim, H. J., and Sastry, S. (2003). Decentralized nonlinear model predictive control of multiple flying robots. In *Decision and control, 2003. Proceedings. 42nd IEEE conference on*, volume 4, pages 3621–3626. IEEE.

[Siegwart et al., 2011] Siegwart, R., Nourbakhsh, I. R., and Scaramuzza, D. (2011). *Introduction to autonomous mobile robots*. MIT press.

[Snape et al., 2011] Snape, J., Van Den Berg, J., Guy, S. J., and Manocha, D. (2011). The hybrid reciprocal velocity obstacle. *IEEE Transactions on Robotics*, 27(4):696–706.

[Stellato et al., 2018] Stellato, B., Banjac, G., Goulart, P., Bemporad, A., and Boyd, S. (2018). Osqp: An operator splitting solver for quadratic programs. In *2018 UKACC 12th international conference on control (CONTROL)*, pages 339–339. IEEE.

[Strub and Gammell, 2020] Strub, M. P. and Gammell, J. D. (2020). Advanced bit*(abit*): Sampling-based planning with advanced graph-search techniques. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 130–136. IEEE.

[Todorov et al., 2012] Todorov, E., Erez, T., and Tassa, Y. (2012). Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE.

[Torrente et al., 2021] Torrente, G., Kaufmann, E., Föhn, P., and Scaramuzza, D. (2021). Data-driven mpc for quadrotors. *IEEE Robotics and Automation Letters*, 6(2):3769–3776.

[Valavanis and Vachtsevanos, 2014] Valavanis, K. P. and Vachtsevanos, G. J. (2014). *Handbook of unmanned aerial vehicles, Section XX: UAV Applications*. Springer Publishing Company, Incorporated.

[Valavanis and Vachtsevanos, 2015] Valavanis, K. P. and Vachtsevanos, G. J. (2015). Uav applications: introduction. In *Handbook of Unmanned Aerial Vehicles*, pages 2639–2641. Springer.

[Van Den Berg et al., 2012] Van Den Berg, J., Wilkie, D., Guy, S. J., Niethammer, M., and Manocha, D. (2012). Lqg-obstacles: Feedback control with collision avoidance for mobile robots with motion and sensing uncertainty. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 346–353. IEEE.

[Van Der Horst and Hogema, 1993] Van Der Horst, R. and Hogema, J. (1993). Time-to-collision and collision avoidance systems.

[Vlantis et al., 2015] Vlantis, P., Marantos, P., Bechlioulis, C. P., and Kyriakopoulos, K. J. (2015). Quadrotor landing on an inclined platform of a moving ground vehicle. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 2202–2207. IEEE.

[Watson et al., 2020] Watson, R. J., Pierce, S. G., Kamel, M., Zhang, D., MacLeod, C. N., Dobie, G., Bolton, G., Dawood, T., and Nieto, J. (2020). Deployment of contact-based ultrasonic thickness measurements using over-actuated uavs. In *European Workshop on Structural Health Monitoring*, pages 683–694. Springer.

[Zhang et al., 2016] Zhang, T., Kahn, G., Levine, S., and Abbeel, P. (2016). Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 528–535. IEEE.

[Zhu and Alonso-Mora, 2019] Zhu, H. and Alonso-Mora, J. (2019). Chance-constrained collision avoidance for mavs in dynamic environments. *IEEE Robotics and Automation Letters*, 4(2):776–783.

[Zhu et al., 2021] Zhu, H., Chung, J. J., Lawrance, N. R., Siegwart, R., and Alonso-Mora, J. (2021). Online informative path planning for active information gathering of a 3d surface. *arXiv preprint arXiv:2103.09556*.