Probabilistic Motion Planning for Multi-Robot Systems

Zhu, H.

DOI
10.4233/uuid:52038194-6cdf-4098-8723-852eb68dfd00

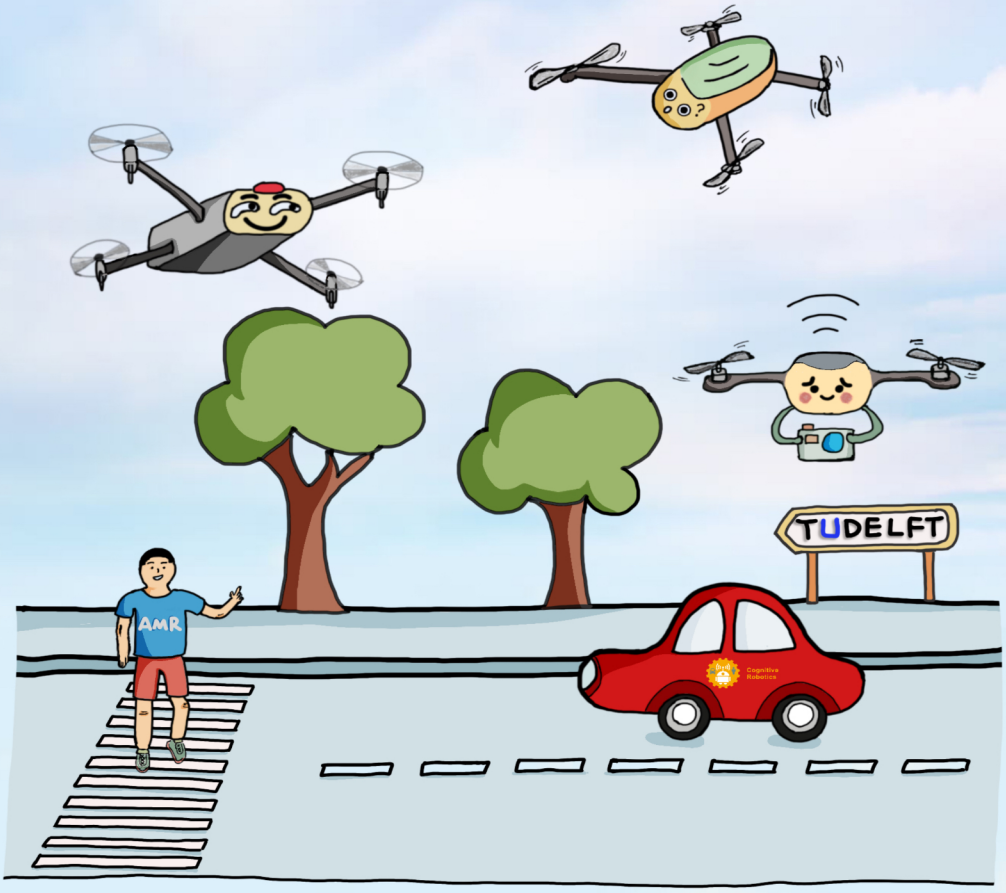
Publication date
2022

Document Version
Final published version

Citation (APA)
Zhu, H. (2022). *Probabilistic Motion Planning for Multi-Robot Systems*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:52038194-6cdf-4098-8723-852eb68dfd00

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Probabilistic Motion Planning for Multi-Robot Systems



Hai Zhu

Probabilistic Motion Planning for Multi-Robot Systems

Hai Zhu
祝海

Probabilistic Motion Planning for Multi-Robot Systems

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus Prof.dr.ir. T.H.J.J. van der Hagen
chair of the Board for Doctorates
to be defended publicly on
Thursday 27 January 2022 at 10:00 o'clock

by

Hai ZHU

Master of Engineering in Aeronautical and Astronautical Science and Technology
National University of Defense Technology, China
born in Hubei, China

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus	chairperson
Prof.dr. R. Babuska	Delft University of Technology, promotor
Dr. J. Alonso-Mora	Delft University of Technology, copromotor

Independent members:

Prof.dr.ir. T. Keviczky	Delft University of Technology
Prof.dr. G.C.H.E. de Croon	Delft University of Technology
Prof.dr. K. Alexis	Norwegian University of Science and Technology
Dr. A. Franchi	University of Twente
Dr. S. Shen	Hong Kong University of Science and Technology

The research described in this thesis was partly supported by the China Scholarship Council (CSC) grant 201703170200, the Netherlands Organization for Scientific Research (NWO) domain Applied Sciences (Veni 15916), the U.S. Office of Naval Research Global (ONRG) NICOP-Grant N62909-19-1-2027, and the Cognitive Robotics department at Delft University of Technology.

Published and distributed by: Hai Zhu
E-mail: zhuhai.frank@outlook.com

Keywords: Motion planning, collision avoidance, multi-robot systems, micro aerial vehicles, planning under uncertainty, dynamic environments
Front & Back: Hai Zhu
Printed by: ProefschriftMaken

ISBN: 978-94-6423-651-4

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

To my mother.

Contents

Summary

Planning safe motions for multi-robot systems is crucial for deploying them in real-world applications such as target tracking, environmental monitoring, and multi-view cinematography. Traditional approaches mainly solve the multi-robot motion planning problem in a deterministic manner, where the robot states and system models are perfectly known. Practically, however, many sources of uncertainty exist in real-world environments, such as noisy sensor measurements, motion disturbances, and uncertain behaviors of other decision-making agents. Reasoning about these uncertainties is of utmost importance for robust and safe navigation of multi-robot systems. To this end, this thesis aims to develop probabilistic methods for multi-robot motion planning under uncertainty.

The first main contribution of this thesis is a Chance-Constrained Nonlinear Model Predictive Control (CCNMPC) method for probabilistic multi-robot motion planning. Taking into account uncertainties in robot localization, sensing, and motion disturbances, the method explicitly considers the collision probability between each robot and obstacle and formulates a model predictive control problem with chance constraints. A tight upper bound of the collision probability is developed which makes the CCNMPC formulation tractable and solvable in real time. In addition, the CCNMPC is incorporated into multi-robot motion planning using three coordination strategies: a) centralized sequential planning, b) distributed planning in which robots communicate their future planned trajectories, and c) decentralized planning in which robots predict other robots' trajectories using the constant velocity model (CVM). Performances of the three strategies are analyzed and compared.

The CCNMPC method requires robots to know the future trajectories of other robots, either via communication or motion prediction using CVM. However, communication is not always available, and the CVM based motion prediction can lead to collisions among robots, especially in crowded environments. To achieve decentralized and communication-free multi-robot collision avoidance under uncertainty, this thesis then presents a method that relies on the introduced Buffered Uncertainty-Aware Voronoi Cell (B-UAVC). The B-UAVC defines a local safe region for each robot among other robots and obstacles, such that the collision probability between robots and obstacles is below a specified threshold if each robot's motion is constrained to be within its corresponding B-UAVC. An approach to constructing the B-UAVC is proposed, which leverages the techniques of computing a separating hyperplane between two Gaussian distributions and adding buffers for probabilistic collision avoidance. Based on B-UAVC, a set of reactive controllers are designed for single-integrator, double-integrator, and differential-drive robots, respectively; and a receding horizon planner is proposed for general nonlinear dynamical systems.

Instead of directly generating a control action for each robot to move towards its way-

point goal as in the CCNMPC and B-UAVC methods, this thesis further presents a method that can compute a safe control input by minimally modifying a given nominal controller, which may come from a high-level task-oriented planner. The method is decentralized and relies on the Chance-Constrained Safety Barrier Certificates (CC-SBC), which defines a probabilistic safe control space for each robot in a multi-robot system considering robot localization and sensing uncertainties. The CC-SBC chance constraints are reformulated into a set of deterministic quadratic constraints, based on which a quadratically constrained quadratic program (QCQP) can be formulated. By solving the QCQP, the robot can obtain a safe control action thanks to that the CC-SBC guarantees forward invariance of the robot's safety set in a probabilistic manner. Hence, the CC-SBC method can be used as a probabilistic safety filter for multi-robot systems.

While both the B-UAVC and CC-SBC methods are decentralized and communication-free, they typically lead to more conservative robot motions than the CCNMPC method with robots communicating their planner trajectories. The CCNMPC method can also be communication-free by letting each robot predict the other robots' trajectories using the constant velocity model, but it is unsafe in crowded environments. To address the issue, this thesis finally presents a novel trajectory prediction model based on Recurrent Neural Networks (RNN) that can learn multi-robot motion behaviors from demonstrated trajectories generated using a centralized motion planner. By incorporating the learned RNN-based trajectory prediction model within the MPC framework, efficient and communication-free multi-robot motion planning is achieved.

The motion planning methods developed in the thesis have been extensively evaluated and validated in simulations and real-world experiments with a team of quadrotors, showing safe navigation of robots under uncertainty.

Samenvatting

Het plannen van veilige bewegingen voor systemen met meerdere robots is cruciaal om ze in te zetten in real-world toepassingen zoals het volgen van doelen, omgevingsmonitoring en multi-view cinematografie. Traditionele benaderingen lossen het probleem van bewegingsplanning met meerdere robots voornameldie gebruik maakt van de technieken van het berekenen van een scheidend vlak tussen twee Gaussische-verdelingenijk op een deterministische manier op, waarbij de robottoestanden en systeemmodellen perfect bekend zijn. In de praktijk bestaan er echter veel bronnen van onzekerheid, zoals sensormetingen met ruis, bewegingsstoringen en onzeker gedrag van andere besluitvormers. Redeneren over deze onzekerheden is van het grootste belang voor een robuuste en veilige navigatie van multi-robotsystemen. Hiertoe heeft dit proefschrift tot doel probabilistische methoden te ontwikkelen voor multi-robot bewegingsplanning onder onzekerheid.

De eerste belangrijke bijdrage van dit proefschrift is een Chance-Constrained Nonlinear Model Predictive Control (CCNMPC) methode voor probabilistische bewegingsplanning van meerdere robots. Rekening houdend met onzekerheden in robotlokalisatie, detectie en bewegingsverstoringen, houdt de methode expliciet rekening met de botsingskans tussen elke robot en obstakel en formuleert een model voorspellend besturingsprobleem met kansbeperkingen. Er wordt een strakke bovengrens van de botsingskans ontwikkeld, waardoor de CCNMPC-formulering in realtime handelbaar en oplosbaar is. Bovendien is de CCNMPC opgenomen in de bewegingsplanning van meerdere robots met behulp van drie coördinatiestrategieën: a) gecentraliseerde sequentiële planning, b) gedistribueerde planning waarin robots hun toekomstige geplande trajecten communiceren, en c) gedecentraliseerde planning waarin robots de trajecten van andere robots voorspellen met behulp van het constante snelheidsmodel (CVM). De prestaties van de drie strategieën worden geanalyseerd en vergeleken.

De CCNMPC-methode vereist dat robots de toekomstige trajecten van andere robots kennen, hetzij via communicatie of bewegingsvoorspelling met behulp van CVM. Communicatie is echter niet altijd beschikbaar en de op CVM gebaseerde bewegingsvoorspelling kan leiden tot botsingen tussen robots, vooral in drukke omgevingen. Om gedecentraliseerde en communicatievrije multi-robot botsingsvermijding onder onzekerheid te bereiken, presenteert dit proefschrift vervolgens een methode die vertrouwt op de geïntroduceerde Buffered Uncertainty-Aware Voronoi Cell (B-UAVC). De B-UAVC definieert een lokaal veilig gebied voor elke robot tussen andere robots en obstakels, zodat de kans op botsingen tussen robots en obstakels onder een gespecificeerde drempel ligt als de beweging van elke robot wordt beperkt om binnen de overeenkomstige B-UAVC te blijven. Er wordt een benadering voorgesteld voor het construeren van de B-UAVC, die gebruik maakt van de technieken van het berekenen van een scheidend vlak tussen twee Gaussische-verdelingen en het

toevoegen van buffers voor het vermijden van probabilistische botsingen. Op basis van B-UAVC is een set reactieve controllers ontworpen voor respectievelijk single-integrator, double-integrator en differentieel aangedreven robots; en een receding horizon planner wordt voorgesteld voor algemene niet-lineaire dynamische systemen.

In plaats van direct een regelactie te genereren voor elke robot om naar zijn waypoint-doel te gaan, zoals in de CCNMPC- en B-UAVC-methoden, presenteert dit proefschrift verder een methode die een veilige besturingsinvoer kan berekenen door een bepaalde nominale controller minimaal te wijzigen, wat kan komen van een taakgerichte planner op een hirarchisch hoger niveau. De methode is gedecentraliseerd en is gebaseerd op de Chance-Constrained Safety Barrier Certificates (CC-SBC), die een probabilistische veilige controleruimte definieert voor elke robot in een systeem met meerdere robots, rekening houdend met robotlokalisatie en detectie van onzekerheden. De CC-SBC kansbeperkingen worden geherformuleerd in een set deterministische kwadratische beperkingen, op basis waarvan een kwadratisch beperkt kwadratisch programma (QCQP) kan worden geformuleerd. Door de QCQP op te lossen, kan de robot een veilige besturingsactie verkrijgen dankzij het feit dat de CC-SBC op een probabilistische manier voorwaartse invariantie van de veiligheidsset van de robot garandeert. Daarom kan de CC-SBC-methode worden gebruikt als een probabilistisch veiligheidsfilter voor systemen met meerdere robots.

Hoewel zowel de B-UAVC- als de CC-SBC-methode gedecentraliseerd en communicatievrij zijn, leiden ze doorgaans tot conservatievere robotbewegingen dan de CCNMPC-methode, waarbij robots hun plannertrajecten communiceren. De CCNMPC-methode kan ook communicatievrij zijn door elke robot de banen van de andere robots te laten voorspellen met behulp van het constante snelheidsmodel, maar het is onveilig in drukke omgevingen. Om dit probleem aan te pakken, presenteert dit proefschrift als laatst een nieuw trajectvoorspellingsmodel gebaseerd op Recurrent Neural Networks (RNN) dat bewegingsgedrag van meerdere robots kan leren van aangetoonde trajecten die zijn gegenereerd met behulp van een gecentraliseerde bewegingsplanner. Door het op RNN gebaseerde model voor trajectvoorspelling op te nemen in het MPC-paradigma, wordt een efficiënte en communicatievrije bewegingsplanning met meerdere robots bereikt.

De bewegingsplanningsmethoden die in het proefschrift zijn ontwikkeld, zijn uitgebreid geëvalueerd en gevalideerd in simulaties en praktijkexperimenten met een team van quadrotoren, die veilige navigatie van robots onder onzekerheid laten zien.

Acknowledgments

I have never had such a strong feeling that the four-year PhD study is so short before now closing this thesis. The main topic of my PhD research is planning under uncertainty. However, I do not really like too many uncertainties in my life, which may make me feel nervous. Four years ago, when stepping into this beautiful and unfamiliar country to start my PhD journey, I did not know how many challenges and uncertainties there would be in front of me. Luckily, during the four years, I have met so many people who provided their help to me to finish the journey. Now, I have not only been able to write a dissertation on planning under uncertainty, but also have leaned how to better deal with uncertainties in my life. Here, I want to express my sincere gratitude to these helpful people.

Foremost, I would like to thank my daily supervisor and co-promotor Dr. Javier Alonso-Mora for his comprehensive and professional supervision, making my four-year PhD journey exciting and enjoyable. Since I had no background in robotics when starting my doctoral research, Javier spent a lot of time helping me to learn basic knowledge and guiding me to the state of the art in the early stage with great patience. Apart from helping me to solve technical questions, he has also provided me many extremely useful advices of how to do research independently and look for collaborations in academia. I also want to thank my promotor Prof. Robert Babuska for giving me an opportunity to start my PhD in TU Delft, and for providing me continual support during the journey, in particular in preparing the final thesis and defense.

Second, I would like to thank Prof. Tamas Keviczky, Prof. Guido de Croon, Prof. Kostas Alexis, Dr. Antonio Franchi, and Dr. Shaojie Shen, for being my PhD committee members and providing me with valuable feedback to improve the thesis. I would also like to thank Prof. Roland Siegwart, Dr. Jen Jen Chung, and Dr. Nicholas Lawrance for alowing me to visit the Autonomous Systems Lab (ASL) and helping me with the research on informative path planning.

I am very grateful to my colleges currently or formerly working in the department of Cognitive Robotics (CoR). Particularly, I would like to give my thanks to Bruno Brito, who spent the most time with me in the last four years and provided me a lot of help in research and also my life in the Netherlands; to Dr. Michal Čáp for helping me to learn basics in motion planning; to Dr. Laura Ferranti for helping to revise my first conference paper; to Álvaro Serra-Gómez, Dennis Benders, Max Lodel, and Gang Chen for the time playing with drones; to Dr. Carlos Celemin, Giovanni Franzese, Rodrigo Pérez-Dattari, and Luzia Knödler for the time enjoying beers; to Maximilian Kronmüeller, Max Spahn, and Corrado Pezzato for the time cycling together in the field; to Dr. Andres Fielbaum, Dr. Xiaoshan Bai, Jelle Luijkx, Tasos Tsolakis, Bas van der Heijden, and Lasse Peters for creating a beautiful office atmosphere; to Dr. Jihong Zhu, Desong Du, Linda van der Spaa,

and Ajith Anil Meera for valuable discussions in the department. I want to thank Oscar de Groot for checking the Dutch translation of the summary of this thesis. I would also like to thank Hanneke Hustinx for her administrative support throughout my PhD journey.

I would like to thank several talented students: Jelle Juhl, Nikhil Potdar, Jiahao Lin, Francisco Martínez Claramunt, and Qi Luo. They have done very nice jobs in their master projects which also help my research.

Many thanks to my Chinese friends who have made my life in the Netherlands more colorful. Particularly, I would like to thank Sihang Qiu, Xiaohui Wang, Chenyang Ding, Guishan Wang, Qi Wang, Renfei Bu, Baozhou Zhu, He Wang, Shushuai Li, and Hongpeng Zhou for experiencing a PhD journey together; to Hanqing Liu, Yuwei Chen, Yi Wen, and Xunhui Zhang for traveling to the middle of nowhere to relax. Special thanks must go to Yujie Tang, who helped me a lot in the final stage of writing this thesis and gave me valuable suggestions in designing the cover.

I also want to thank my bachelor's and master's supervisor Prof. Yazhong Luo for leading me to start scientific research and encouraging me when I decided to study abroad.

In the end, I would like to express my deepest gratitude to my parents for their unreserved love and support.

Hai Zhu
Delft, January 2022

1

Introduction

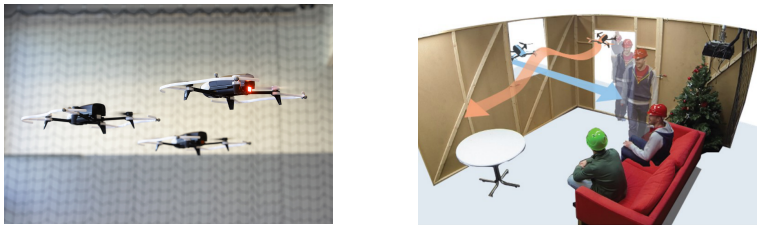
1

1.1 Motivation

Historically, deployments of multi-robot systems have been mainly restricted to static known environments due to their high costs and limited onboard capabilities, such as order picking in a warehouse [1]. Recently, with advances in onboard sensing and computing, the use of multiple robots in dynamic unknown environments is drawing significant attention from research communities and industries. Examples include teams of drones for search and rescue [2], aerial delivery [3], and multi-view cinematography [4], as shown in Fig. 1.1.

Motion planning, which plans collision-free trajectories and movements for robots to transit from their initial poses to the goal ones, is a fundamental building block in those scenarios [5]. Traditional approaches typically solve the motion planning problem in a deterministic manner where robots are assumed to have perfect knowledge of the system. Practically, however, various sources of uncertainty exist in real-world settings, especially in dynamic unknown environments. Sensor noise, for instance, is a typical source of uncertainty. Many robots now rely on cameras or LiDARs for localization and sensing [6], leading to uncertainty in robot state estimation and obstacle detection due to noisy sensory measurements. Unmodeled dynamics of the robot and disturbances are another source which results in uncertainty of the robot's motion. Moreover, in dynamic environments with other decision-making agents (e.g. humans), their uncertain behaviors are also a source of uncertainty. Taking into account these uncertainties is of great importance for robust and safe robot navigation, which makes the motion planning problem more challenging than deterministic scenarios. To overcome the challenge, this thesis studies motion planning for multi-robot systems under uncertainty.

Due to its complexity, robot motion planning is typically divided into two stages [7]: a) global motion planning that computes a full trajectory for the robot from its initial pose to the goal one; and b) local motion planning that computes a feasible local trajectory for a short duration of time. While global motion planning is mostly performed offline or at a low frequency online [8, 9], local motion planning algorithms are required to run in real time at a high frequency to deal with fast environment changes (e.g. humans). Aiming at enabling safe multi-robot systems in dynamic environments, this thesis focuses on local



(a) (b)

Figure 1.1: Illustrative examples of a team of drones deployed in real-world applications. (a) Indoor drone flying show. (b) Multi-drone automated cinematography in dynamic scenes (courtesy of [4]).

motion planning under uncertainty.

Overall, the research goal of the thesis is *to develop efficient online motion planning algorithms for multi-robot systems taking into account a variety of uncertainties potentially arising from robot localization, obstacle detection, and motion disturbances.*

1.2 Approach

The multi-robot motion planning problem addressed in this thesis can be in general formulated as follows: given current states of a team of robots and their goal locations, as well as the environment information (e.g. static and moving obstacles), the objective is to plan a local trajectory and the control input for each robot. The plan has to respect the robot's dynamics constraints, make progress towards the goal while avoiding obstacles and other robots. In deterministic scenarios, the collision avoidance constraints are usually formulated as enforcing the distance between robots and obstacles to be larger than a safe distance. However, when considering robot localization, sensing, and motion uncertainties, the states of robots and obstacles are typically described by random variables, e.g. Gaussian distributions which have infinite support. Thus, their distances are also random variables. To achieve safety in this case, this thesis formulates the collision avoidance constraints in a probabilistic way. In particular, the collision probability between each robot and other robot and obstacle is constrained to be smaller than a specified risk threshold. The resulting constraints are called *chance constraints*, and hence our general approach to dealing with uncertainty in this thesis is to develop planning algorithms that can achieve probabilistic safety by satisfying the chance constraints.

From a high-level perspective, the algorithms developed in this thesis mainly rely on the following two pillars as the foundation to solve the problem of multi-robot motion planning under uncertainty.

1. Model predictive control. The methods presented in this thesis are mostly linked to the model predictive control (MPC) framework [10]. MPC, also called receding horizon control (RHC), is a method that is used to control a dynamical system while minimizing some costs and satisfying a set of constraints. The key idea of MPC is to compute a sequence of control inputs by optimizing with a finite time horizon with multiple time steps. Within the time horizon, the system states are predicted using a known dynamical model. Only the first time-step control input in the sequence is implemented and then the optimization is performed again with an updated system state. The process is repeated online until the system reaches the desired state. The main strength of MPC is that it can anticipate future events of the system via prediction and take actions accordingly. Moreover, it can handle complex costs and constraints as well as nonlinear systems.

2. Chance-constrained optimization. Many problems considered in this thesis are formulated as an optimization problem with collision avoidance chance constraints, namely chance-constrained optimization (CCO) [11]. CCO is a natural approach to solve optimization problems under uncertainty. It is a formulation of an optimization problem that requires the probability of violating a certain constraint is below some predefined

1

risk level. Generally, CCO is very difficult to solve since the probability of violating a constraint is hard to compute. In the next chapter, we will provide a detailed overview of chance-constrained optimization in the context of robot motion planning.

1.3 Contributions and outline

To reach the overall research goal established in Section 1.1, this thesis presents the following scientific contributions in multi-robot motion planning under uncertainty:

(1) **A Chance-Constrained Nonlinear Model Predictive Control (CCNMPC) method for multi-robot motion planning.** The method allows for teams of robots to navigate in dynamic environments taking into account robot localization, sensing, and motion uncertainties. By reformulating the collision chance constraints into deterministic constraints on the mean and covariance of the robot states, the CCNMPC becomes tractable and solvable in real-time.

(2) **A method, named B-UAVC (Buffered Uncertainty-Aware Voronoi Cell), to compute a probabilistic safe region for each robot to navigate among other robots and obstacles.** While the previously presented CCNMPC method requires robots to know future trajectories of their neighbors, typically via communication, to ensure safety, the B-UAVC method is fully decentralized, communication-free and only needs each robot to have an estimation of other robots' current positions. The method can also be applied to non-holonomic robots and heterogeneous robot teams.

(3) **A method, named CC-SBC (Chance-Constrained Safety Barrier Certificates), to compute a probabilistic safe control space for each robot navigating in a multi-robot system.** Different from the previously proposed CCNMPC and B-UAVC methods which directly compute a control input for each robot, the CC-SBC method allows the robot to modify a nominal controller in a minimally invasive way while guaranteeing probabilistic safety. Hence, it can be regarded as a probabilistic safety filter.

(4) **An interaction-aware model to predict robot future trajectories in multi-robot scenarios.** While both the B-UAVC and CC-SBC methods are decentralized and communication-free, they typically lead to conservative robot motions. The CCNMPC method is more efficient but it requires each robot to know the future trajectories of other robots. Hence, a novel prediction model is developed to provide reliable trajectory predictions, which is incorporated with the MPC framework to achieve decentralized, efficient, and safe multi-robot navigation.

Fig. 1.2 shows an overview of the structure of the thesis. Chapter 2 reviews the state of the art in chance-constrained motion planning and multi-robot motion planning. Chapter 3 presents the chance-constrained nonlinear model predictive control (CCNMCP) method for collision avoidance of teams of robots in dynamic environments. Chapter 4 introduces the concept of buffered uncertainty-aware Voronoi cell (B-UAVC) and its usage in probabilistic multi-robot collision avoidance. Chapter 5 presents the CC-SBC method to compute chance-constrained safety barrier certificates for multi-robot systems under

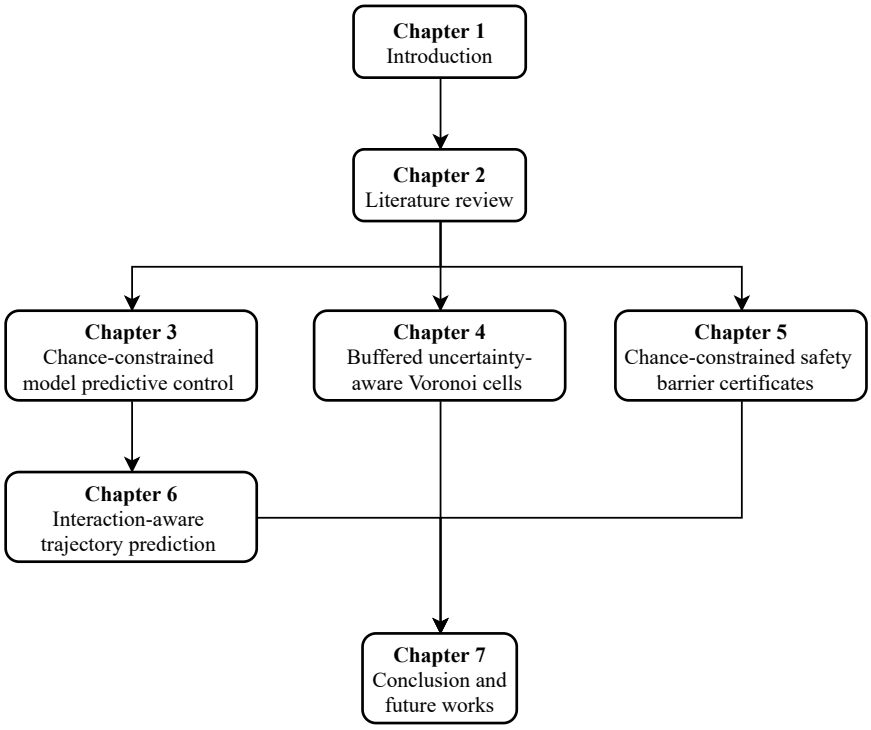


Figure 1.2: Structure of this thesis.

uncertainty which guarantee probabilistic safety among robots. Chapter 6 describes the proposed interaction-aware trajectory prediction model and applies it to decentralized multi-robot local motion planning. Finally, Chapter 7 concludes the thesis and provides recommendations for future researches.

1.4 General notations

In this section, we describe some basic notations used throughout the thesis, while notations specific to each chapter are defined within chapters. A complete list appears in Glossary of the thesis.

Scalars are denoted by italic lowercase letters, e.g. x, vectors by bold lowercase, e.g. $\mathbf{x}$, matrices by plain uppercase, e.g. M, and sets by calligraphic uppercase, e.g. $\mathcal{S}$. A superscript $\mathbf{x}^T$ or M^T denotes the transpose of a vector $\mathbf{x}$ or a matrix M. $\|\mathbf{x}\| = \sqrt{\mathbf{x}^T\mathbf{x}}$, $\|\mathbf{x}\|^2 = \mathbf{x}^T\mathbf{x}$, and $\|\mathbf{x}\|_Q^2 = \mathbf{x}^T Q\mathbf{x}$ denote the Euclidean norm, squared Euclidean norm, and weighted squared Euclidean norm of $\mathbf{x}$ respectively. $\dot{\mathbf{x}}$ denotes the derivative of $\mathbf{x}$ with respect to time t. For a random variable $\mathbf{x}$, denote by $\hat{\mathbf{x}}$ its mean with a hat $\hat{\cdot}$. The function $\Pr(\cdot)$ indicates the probability of an event and $p(\cdot)$ indicates the probability density function (PDF).

2

Literature review

In this chapter, we summarize related works of the two main research fields connected to the topic of this thesis. We start with reviewing the state of the art of chance-constrained motion planning, followed by a discussion on multi-robot local motion planning.

2

2.1 Chance-constrained motion planning

Motion planning for autonomous robots often entails planning under uncertainty, which may arise from modeling errors of the robots, sensing noise of the environment, and uncertain behaviors of other decision-making agents. Taking into account these uncertainties makes the motion planning problem harder to be solved than the deterministic counterpart in which the robot has perfect knowledge of the system [12].

Typically, there are two main ways to solve the robot motion planning problem under uncertainty. One way is in a *robust* manner [13, 14], e.g. through robust optimization [15], in which a solution is found such that it is feasible under all possible cases of the uncertain elements in the problem, namely assuming the worst case. While being able to provide safety guarantees, the approach cannot handle unbounded uncertainties with infinite support, e.g. Gaussian uncertainties. Moreover, even in scenarios with bounded uncertainties, the approach typically leads to an overly conservative solution or may be infeasible [16].

An alternative way is in a *probabilistic* or stochastic manner, e.g. via probabilistic collision checking or chance-constrained optimization, in which a solution is found such that the probability of it being feasible (safe) is above some confidence level. In other words, the probability of the solution being unsafe is below some risk level (chance constraints). However, probabilistic motion planning typically requires evaluating the robot collision probability and handling chance constraints, which are computationally difficult in practice. In the following, the state of the art in probabilistic collision checking and chance-constrained optimization is reviewed.

2.1.1 Probabilistic collision checking

Probabilistic collision checking involves computing the collision probability of a robot with its surrounding environment, given representations of the robot and environment and their associated uncertainties. It can be incorporated within both sampling-based methods [17, 18] and optimization-based methods [19, 20] to plan safe motions for the robot under uncertainty.

Trajectory and waypoint collision probability

Denote by C^k the event (collision condition) that the robot is in a collision configuration (state) where the superscript $\cdot^k$ indicates time step k. Then the collision probability of the robot at that time can be denoted by $\mathrm{CP}^k = \Pr(C^k)$, which is called the (instantaneous) waypoint collision probability. Denote by $\mathcal{T}$ a local motion plan, typically represented as a trajectory, of the robot in the future N time steps. One may want to evaluate the overall collision probability of the robot executing this motion plan $\mathrm{CP}^{\mathcal{T}}$, which is called

the trajectory collision probability.

One way to compute $\mathrm{CP}^{\mathcal{T}}$ is to determine the swapping area of the robot following the trajectory in the time interval and then computing the probability of obstacles lying within the area [21]. However, such methods are not suitable to compute the robot collision probability with moving obstacles. Another way is to represent the trajectory as a sequence of configurations and to compute $\mathrm{CP}^{\mathcal{T}}$ by reasoning about the individual waypoint collision probability of each time step $\mathrm{CP}^1, \ldots, \mathrm{CP}^N$, which are computed in advance. Depending on different assumptions on the relationships of these waypoint collision probabilities, there are three approaches to compute $\mathrm{CP}^{\mathcal{T}}$:

- **Additive approach** [20, 22]. Using Boole's inequality, an upper bound of the trajectory collision probability can be obtained by summing up the individual waypoint collision probability as

$$\mathrm{CP}^{\mathcal{T}} \approx \Pr\Big(\bigcup_{k=1}^{N} C^k\Big) \leq \sum_{k=1}^{N} \Pr(C^k) = \sum_{k=1}^{N} \mathrm{CP}^k. \tag{2.1}$$

  While providing an upper bound, the approach typically leads to an overly conservative result, particularly when the number of time steps is large. Moreover, the approximated probability may be larger than one, which is unrealistic.

- **Independent multiplicative approach** [23, 24]. The approach assumes that the waypoint collision events are mutually independent. In this case, the trajectory collision probability can be approximated as follow:

$$\begin{aligned} \mathrm{CP}^{\mathcal{T}} &\approx 1 - \Pr\Big(\bigcap \bar{C}^k\Big) = 1 - \prod_{k=1}^{N} \Pr(\bar{C}^k) \\ &= 1 - \prod_{k=1}^{N} (1 - \mathrm{CP}^k), \end{aligned} \tag{2.2}$$

  where $\bar{C}^k$ is the complement of the event C^k, in other words, the event that the robot is collision-free at time step k. In most cases, the multiplicative approach can obtain more accurate results than the additive approach. However, the approximation is still usually conservative.

- **Conditional multiplicative approach** [17, 25]. To obtain a more accurate approximation, one can replace the individual waypoint collision probability CP^k with the probability of collision of the waypoint conditional on previous waypoints being collision-free. Formally, instead of using $\mathrm{CP}^k = \Pr(C^k)$, the improved waypoint collision probability is

$$\mathrm{CP}^k = \Pr(C^k \mid \bar{C}^1 \cap \cdots \cap \bar{C}^N). \tag{2.3}$$

  While the conditional approach is less conservative than the independent one, the computation of conditional probability is typically more complex.

Next, we will discuss different methods to compute the waypoint collision probability in detail, i.e. how to compute $\mathrm{CP}^k = \Pr(C^k)$.

Sampling-based methods

Computation of the collision probability $\mathrm{CP}^k = \Pr(C^k)$ typically involves handling the form of C^k and distributions of the uncertain elements in it. Sampling-based methods can deal with an arbitrary form of C^k and uncertainty distributions since they only need to draw many samples of the random variables and then evaluate the corresponding values of C^k. In computing robot collision probability, some recent works include simple Monte Carlo sampling [26], weighted importance sampling [27], adaptive importance sampling [28], and quadrature-based sampling [20]. While sampling-based methods can give a relatively accurate estimation of the true collision probability, they may require a large number of samples, thus are computationally inefficient. Moreover, they do not provide an upper bound of the collision probability. Nevertheless, the result via simple Monte Carlo sampling with a large number of samples is often used as the ground truth collision probability for comparison with results of other methods.

Analytic and semi-analytic approximation

Instead of numerically sampling, analytic and semi-analytic approximations of the collision probability are computationally more efficient and have been widely used in robot motion planning. Denote by $\mathbf{x}^k \in \mathbb{R}^{n_x}$ the uncertain variable in the collision condition C^k, which is generally the state of the robot, following a known distribution $\mathcal{D}^k$. Denote by $\mathcal{C}^k := \{\mathbf{x}^k \in \mathbb{R}^{n_x} \mid C^k\}$ the set of collision states of the robot at time step k. Formally, the collision probability can be defined as

$$\Pr(C^k) = \int_{\mathcal{C}^k} p(\mathbf{x}^k) d\mathbf{x}^k, \tag{2.4}$$

in which $p(\mathbf{x}^k)$ is the probability density function of the distribution $\mathcal{D}^k$. Hence the collision probability is the integral of the probability density function $p(\mathbf{x}^k)$ over the collision state set $\mathcal{C}^k$. Intuitively, in case $\mathcal{C}^k$ is a closed set, one can assume that the probability density $p(\mathbf{x}^k)$ is constant over the set and then the collision probability can be computed as the following product,

$$\Pr(C^k) = p \cdot \mathrm{A}(\mathcal{C}^k), \tag{2.5}$$

where p is the assumed constant density and $\mathrm{A}(\mathcal{C}^k)$ is the area of the set. [29] considers $\mathcal{C}^k$ is a spherical set and uses the probability density of its center as the constant p, thus computing the collision probability. While [19] also considers a spherical collision set, the authors first compute the maximal probability density within $\mathcal{C}^k$ and then use it as the constant p, thus providing an upper bound of the collision probability.

Depending on the form of C^k and the uncertain distribution $\mathcal{D}^k$, there are different analytic approximation methods. In the following we first describe methods for different forms of C^k in the case where $\mathcal{D}^k$ is a Gaussian distribution with mean $\mu^k \in \mathbb{R}^{n_x}$ and covariance $\Sigma^k \in \mathbb{R}^{n_x \times n_x}$, i.e., $\mathbf{x}^k \sim \mathcal{N}(\mu^k, \Sigma^k)$.

- **Linear form**. The collision condition of a robot with a wall can be described as the following linear form:

$$C^k : \mathbf{a}^T \mathbf{x}^k \leq b, \tag{2.6}$$

in which $\mathbf{a} \in \mathbb{R}^{n_x}$ and $b \in \mathbb{R}$ are constants. In the case where $\mathbf{x}^k$ is a Gaussian, there is [30]

$$\Pr(C^k) = \Pr(\mathbf{a}^T\mathbf{x}^k \leq b) = \frac{1}{2} + \frac{1}{2}\text{erf}(\frac{b - \mathbf{a}^T\mu^k}{\sqrt{2\mathbf{a}^T\Sigma^k\mathbf{a}}}), \tag{2.7}$$

where $\text{erf}(\cdot) = \frac{2}{\sqrt{\pi}}\int_0^z e^{-t^2}dt$ is the standard error function.

- **Convex form**. Collision avoidance with a convex polytope obstacle represented by o_n intercepting hyperplanes can lead to a convex form of the collision condition:

$$C^k : \bigcap_{l=1}^{o_n} \mathbf{a}_l^T\mathbf{x}^k \leq b_l, \tag{2.8}$$

where $\mathbf{a}_l \in \mathbb{R}^{n_x}$ and $b_l \in \mathbb{R}$, $l = 1, \ldots, o_n$ are constants. Then the collision probability can be upper bounded as follows

$$\begin{aligned} \Pr(C^k) &= \Pr(\bigcap_{l=1}^{o_n} \mathbf{a}_l^T\mathbf{x}^k \leq b_l) \\ &\leq \min_{l \in \{1,\ldots,o_n\}} \Pr(\mathbf{a}_l^T\mathbf{x}^k \leq b_l). \end{aligned} \tag{2.9}$$

However, there is no closed form to compute the probability. Several approximation approaches have been presented recently. [31] computes an upper bound of the probability by exploiting the separability of Gaussian distributions to decompose the position uncertainty distribution into the product of 1-D Gaussian distributions. To achieve that, a linear coordination transformation technique [32] is utilized to normalize the uncertainty covariance. Considering convex polyhedron obstacles represented by triangle meshes, [33] provides an upper bound of the collision probability by applying the divergence theorem, which has also been extended to general non-convex obstacles and non-Gaussian distributions [34].

- **Quadratic form**. Collision avoidance with a spherical (ellipsoidal) obstacle can lead to a quadratic form of the collision condition:

$$C^k : (\mathbf{x}^k)^T R\mathbf{x}^k \leq 1, \tag{2.10}$$

where $R \in \mathbb{R}^{n_x \times n_x}$ is a positive definite matrix. Let $Q(\mathbf{x}^k) = (\mathbf{x}^k)^T R\mathbf{x}^k$. When $\mathbf{x}^k$ is a Gaussian random variable, $Q(\mathbf{x}^k)$ becomes a quadratic form in a multivariate Gaussian (QFMVG) [35, 36]. Hence, the collision probability $\Pr(C^k) = \Pr(Q(\mathbf{x}^k) \leq 1)$ is actually the cumulative probability function (*cdf*) of the QFMVG $Q(\mathbf{x}^k)$. However, there does not exist a closed solution for the *cdf* of a QFMVG. Several fast approximation methods with bounded errors have been developed in the statistics and communication community, including the Imhof [37] method, Series approximation [38, 39], the Liu-Tang-Zhang [40] method, and Fourier transform [41]. A disadvantage of these approaches is that they do not provide an upper bound of the probability. Some other methods have given an upper bound yet potentially very conservative approximation, such as [19, 29, 42, 43]. Recently, Wang [44] proposes a

method to compute an upper bound for the probability using Chebyshev's Inequality and sum of squares (SOS) Programming [45].

- **Polynomial form**. Let the obstacle be described in terms of a polynomial

$$\mathcal{O}(\omega) := \{\mathbf{x} \in \mathbb{R}^{n_x} : \mathcal{P}(\mathbf{x}, \omega) \leq 0\}, \tag{2.11}$$

where ω is an uncertain variable with known distribution. The collision probability $\Pr(C^k) = \Pr(\mathbf{x}^k \in \mathcal{O}(\omega))$ can be approximated with upper and lower bound using sum of squares (SOS) Programming [46–48].

Comparison of different methods

Table 2.1 gives a brief comparison of different robot collision probability computation methods. The computation time statistics are obtained by performing simulations in a standard laptop with Intel i7 CPU@2.6GHz. Practically, sampling-based approaches can be used to obtain (approximated) ground truth values of the collision probability, or incorporated with offline motion planning. Semi-analytic approaches may be incorporated with sampling-based planners to perform efficient probabilistic collision checking. Analytic approaches can be incorporated with optimization-based planners to plan probabilistic safe (chance-constrained) robot trajectories in real time. In the next section, we will discuss state-of-the-art works in chance-constrained optimization.

Table 2.1: Comparisons of robot collision probability computation methods.

Methods	Paper	Techniques	Dim.	Bound	Rob. Shape	Obs. Shape	Uncertainty	Comp. Time
Sampling	[26]	Monte Carlo sampling	2	No	polygon	polygon	Gaussian	10^3 ~ 10^4 samples
	[27]	weighted importance sampling	3	No	sphere	convex	non-Gaussian	10 ~ 10^3 samples
	[28]	adaptive importance sampling	3	No	convex	convex	non-Gaussian	~10^3 samples
	[20]	quadrature-based sampling	3	No	convex	convex	Gaussian	~10 samples
Analytic	[42]	coordinate transformation	3	No	sphere	sphere	Gaussian	< 1 ms
	[29]	probability approximation	3	No	sphere	sphere	Gaussian	< 1 ms
	[31]	box approximation	2	upper	polygon	polygon	Gaussian	< 1 ms
	[43]	chi-square approximation	3	upper	sphere	sphere	Gaussian	< 1 ms
	[49]	local linearization	3	upper	sphere	ellipsoid	Gaussian	< 1 ms
Semi-analytic	[19]	probability approximation	3	upper	sphere	sphere	Gaussian	~10 ms
	[33]	probability approximation	3	upper	non-convex	non-convex	Gaussian	~100 ms
	[34]	probability approximation	3	upper	non-convex	non-convex	non-Gaussian	~100 ms
	[50]	ϵ-shadow and bisection search	3	upper	convex	convex	Gaussian	< 1 ms

2.1.2 Chance-constrained optimization

Chance-constrained optimization (CCO) is a formulation of an optimization problem that requires that the probability of satisfying a certain constraint is above some predefined confidence level. Formally, it can be formulated as follows:

***Problem* 2.1** (Chance-constrained optimization (CCO))**.**

$$\min_{\mathbf{x}\in\mathcal{X}} \quad J(\mathbf{x}) \tag{2.12a}$$

$$\text{s.t.} \quad \Pr(\mathbf{g}(\mathbf{x},\xi) \le 0) \ge 1-\delta. \tag{2.12b}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ is the decision variable with dimension n_x, $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ the admissible space of $\mathbf{x}$, $J(\mathbf{x}) \in \mathbb{R}$ the cost function to be minimized, $\xi \in \mathbb{R}^{n_\xi}$ a random vector indicating the source of uncertainty in which n_ξ is the dimension of the random vector, $\mathbf{g}(\mathbf{x},\xi) = (g^1(\mathbf{x},\xi),\dots,g^m(\mathbf{x},\xi))^T$ the constraint vector, and δ the constraint violation probability threshold.

Joint and individual chance constraint

Recall the chance constraint in Eq. (2.12b):

$$\Pr(\mathbf{g}(\mathbf{x},\xi) \le 0) = \Pr(g^1(\mathbf{x},\xi) \le 0,\dots,g^m(\mathbf{x},\xi) \le 0) \ge 1-\delta, \tag{2.13}$$

which is called a joint chance constraint since it requires the probability of satisfying multiple constraints larger than a specified threshold. Alternatively, each of the following constraint is called an individual chance constraint:

$$\Pr(g^k(\mathbf{x},\xi) \le 0) \ge 1-\delta^k, k = 1,\dots,m, \tag{2.14}$$

where δ^k is the individual probability threshold. Joint chance constants are typically more complex and harder to deal with comparing to individual chance constraints [51]. Note that $\Pr(g^1(\mathbf{x},\xi) \le 0,\dots,g^m(\mathbf{x},\xi) \le 0) \ge 1-\delta \Longrightarrow \Pr(\bigvee_{k=1}^m g^k(\mathbf{x},\xi) \ge 0) \le \delta$, and $\Pr(g^k(\mathbf{x},\xi) \le 0) \ge 1-\delta^k \Longrightarrow \Pr(g^k(\mathbf{x},\xi) \ge 0) \le \delta^k$. A popular way to decompose a joint chance constraint to individual chance constraints is to use Boole's inequality

$$\Pr(\bigvee_{k=1}^m g^k(\mathbf{x},\xi) \ge 0) \le \sum_{k=1}^m \Pr(g^k(\mathbf{x},\xi) \ge 0) \le \delta. \tag{2.15}$$

Thus, if $\sum_{k=1}^m \delta^k \le \delta$ and the individual chance constraints in Eq. (2.14) hold, then the joint chance constraint in Eq. (2.13) also holds. However, finding those $\delta^1,\dots,\delta^m$ is usually difficult. A naive approach is to simply let $\delta^k = \frac{1}{m}\delta,\ k = 1,\dots,m$. But it typically leads to overly conservative solutions and may make the optimization infeasible. Another approach is to obtain a set of $\delta^1,\dots,\delta^m$ via optimization, which is called risk allocation [20, 52, 53].

Chance constraint reformulation

Instead of computing the probability of constraint violation, e.g. the collision probability in Section 2.1.1 directly, one can solve the CCO problem by reformulating the chance

constraint to a deterministic tractable one, thus avoiding computing probability. In the following, we discuss state-of-the-art methods in chance constraint reformulation.

Consider an individual chance constraint $\Pr(g(\mathbf{x}, \xi) \leq 0) \geq 1 - \delta$. Here we omit the superscript $\cdot^k$ for simplicity. Typically, there are two popular categories of approaches to reformulate it into a deterministic constraint, usually with the uncertain variable distribution's statistic moments (e.g. mean, covariance) as parameters. One category is built upon the linear inequality of Gaussian distributions. Consider a simple linear constraint $g(\mathbf{x}, \xi) = \xi^T \mathbf{x} - b$ where $\mathbf{x} \in \mathbb{R}^{n_x}$, $b \in \mathbb{R}$, $\xi \in \mathbb{R}^{n_\xi}$, $n_\xi = n_x$, and $\xi \sim \mathcal{N}(\mu, \Sigma)$ is a multivariate Gaussian random vector with mean $\mu \in \mathbb{R}^{n_\xi}$ and covariance $\Sigma \in \mathbb{R}^{n_\xi \times n_\xi}$. Then, there is

$$\Pr(\xi^T \mathbf{x} \leq b) = \Phi\left(\frac{b - \mu^T \mathbf{x}}{\sqrt{\mathbf{x}^T \Sigma \mathbf{x}}}\right), \tag{2.16}$$

where $\Phi(\cdot)$ is the cumulative distribution function (CDF) of a standard normal distribution. Let $\Phi^{-1}(\cdot)$ be the inverse of $\Phi(\cdot)$, then the chance constraint $\Pr(\xi^T \mathbf{x} \leq b) \geq 1 - \delta$ can be reformulated to

$$b - \mu^T \mathbf{x} \geq \Phi^{-1}(1 - \delta) \sqrt{\mathbf{x}^T \Sigma \mathbf{x}}. \tag{2.17}$$

Furthermore, assuming ξ a multivariate Gaussian, [54] considers the following linear chance constraint

$$\Pr(\mathbf{a}^T \mathbf{x} + \mathbf{b}^T \xi + \xi^T D \mathbf{x} \leq e) \geq 1 - \delta, \tag{2.18}$$

where $\mathbf{a} \in \mathbb{R}^{n_x}, \mathbf{b} \in \mathbb{R}^{n_x i}, D \in \mathbb{R}^{n_\xi \times n_x}$ and $e \in \mathbb{R}$ are constants, and reformulates it to

$$e - \mathbf{b}^T \mu - (\mathbf{a} + D^T \mu)^T \mathbf{x} \geq \Phi^{-1}(1 - \delta) \sqrt{(\mathbf{b} + D\mathbf{x})^T \Sigma (\mathbf{b} + D\mathbf{x})}. \tag{2.19}$$

It can be observed that the constraints in Eqs. (2.17) and (2.19) are second-order cone constraints for $\delta \leq 0.5$ (thus $\Phi^{-1}(1 - \delta) \geq 0$) of $\mathbf{x}$ with the mean and covariance of ξ as parameters. Built on this observation, many recent works to solve chance-constrained optimization first linearize their constraints into linear forms and then reformulate the chance constraints using the above the approach, as in [49, 55–57].

Another category of approaches is built upon concentration inequalities [58], which provide bounds on how a random variable deviates from some value (typically, its expected value). A popular way to handle the individual chance constraint $\Pr(g(\mathbf{x}, \xi) \leq 0) \geq 1 - \delta$ is by applying the Cantelli's inequality, also known as the one-tailed Chebyshev inequality as follows,

$$\Pr(g(\mathbf{x}, \xi)) \begin{cases} \leq \frac{\sigma_g^2}{\sigma_g^2 + \mu_g^2}, & \mu_g > 0 \\ \geq 1 - \frac{\sigma_g^2}{\sigma_g^2 + \mu_g^2}, & \mu_g \leq 0 \end{cases} \tag{2.20}$$

where μ_g and σ_g^2 are the mean and variance of the random variable $g(\mathbf{x}, \xi)$ (since ξ is a random vector). Hence, the chance constraint $\Pr(g(\mathbf{x}, \xi) \leq 0) \geq 1 - \delta$ can be reformulated to

$$\mu_g \leq 0, \tag{2.21a}$$

$$1 - \frac{\sigma_g^2}{\sigma_g^2 + \mu_g^2} \geq 1 - \delta. \tag{2.21b}$$

Besides the Chebyshev inequality, other concentration inequalities such as the Vysochanskij–Petunin (VP) and Gauss inequalities can also be applied to obtain a tighter bound of the probability $\Pr(g(\mathbf{x}, \xi) \leq 0)$. An advantage of using concentration inequalities to reformulate chance constraints is that it can deal with arbitrary distributions of the uncertain variable ξ, e.g. non-Gaussian distributions [59]. However, the reformulation is typically overly conservative and the transformed deterministic optimization problem may be infeasible. Moreover, giving known distributions of ξ, the statistic moments (mean and covariance) of $g(\mathbf{x}, \xi)$ are generally very difficult to be computed.

Scenario approach

The scenario approach uses a different way to reformulate chance constraints. It utilizes a set of s scenarios $\{\xi\}_{i=1}^{s}$ to approximate the CCO (2.12) and obtains the following scenario program (SP):

$$\min_{\mathbf{x}\in\mathcal{X}} \quad J(\mathbf{x}) \tag{2.22a}$$

$$\text{s.t.} \quad \mathbf{g}(\mathbf{x}, \xi_i) \leq 0, \quad i = 1, \ldots, s, \tag{2.22b}$$

where the scenarios $\{\xi\}_{i=1}^{s}$ are random samples of ξ. Define $\mathrm{v}(\mathbf{x}) = \Pr(\mathbf{g}(\mathbf{x}, \xi) \geq 0)$ as the *violation probability* of a candidate solution $\mathbf{x}$. Denote by $\mathbf{x}_s^*$ the solution of the scenario program 2.22. In case $J(\mathbf{x})$ is linear, $\mathbf{g}(\mathbf{x}, \xi)$ is convex and assuming that the optimal solution $\mathbf{x}_s^*$ exists and is unique, there is [60, 61]

$$\Pr(\mathrm{v}(\mathbf{x}_s^*) > \delta) \leq \sum_{i=0}^{n_x-1} \binom{s}{i} \delta^i (1-\delta)^{s-i}. \tag{2.23}$$

Eq. (2.23) connects the violation probability of the optimal solution $\mathbf{x}_s^*$ and the number of scenarios/samples s of the SP. According to Eq. (2.23), given a confidence level β, one can determine a minimum required number of samples s by making the right side of the equation larger than β, which leads to

$$s \geq \frac{2}{\delta}\left(n_x - 1 + \ln\left(\frac{1}{\beta}\right)\right). \tag{2.24}$$

The scenario approach was introduced in [62] with early focus on convex problems [60, 61] and recently has been extended to non-convex problems [63, 64]. The most attractive advantage of the scenario approach is that it does not require knowing the underlying distribution of ξ. Besides, if $J(\mathbf{x})$ and $\mathbf{g}(\mathbf{x}, \xi)$ are convex functions, the corresponding scenario program is convex, which can be solved efficiently. However, one drawback of the approach is observed in practice: the number of required samples s might be very large, particularly for non-convex problems [65], which hinders its application to online optimization. Efforts have been put to reduce the number of samples while keeping probabilistic guarantees by techniques such as finding support samples [62] and developing discarding algorithms [66], which has been shown effective in online motion planning under uncertainty [65].

Other approaches

Besides analytic chance constraint reformulation and the scenarios approach, some other methods in chance-constrained optimization have also been applied to robot motion planning under uncertainty. These include the distributionally robust approach [67], obstacle shadow based approach [68, 69], sampling approximation [70], and sum of squares (SOS) optimization-based approach [48, 71]. For a more detailed review on other approaches to chance-constrained optimization, the readers can refer to [51].

2.2 Multi-robot motion planning

This thesis focuses on the multi-robot local motion planning problem, which is also often referred as multi-robot collision avoidance. Given goal positions for a group of robots, which may come from a global motion planner or are specified by a user, the objective of multi-robot motion planning is to compute a local motion (trajectory/control action) for each robot that respects its kinematic and dynamical constraints, makes progress towards its goal location, and is collision-free with other robots and obstacles in the environment.

There are two ways to formulate a multi-robot motion planning problem: coupled and decoupled. Coupled methods regard multiple robots as a single robot with high-dimensional configuration space and compute motions for the group by solving one formulated problem. Albeit being able to provide theoretic guarantees on completeness and optimality, the methods do not scale with the number of robots and are computationally heavy. In contrast, decoupled methods formulate a motion planning problem for each robot individually and rely on techniques to resolve collision conflicts. Typically, algorithms for multi-robot local motion planning must be efficient to run in real-time. Hence, decoupled methods are generally preferred and more feasible in practice.

Despite formulating the problem in a coupled or decoupled way, another widely-used taxonomy is to categorize methods to solve the problem into two main groups: centralized and distributed/decentralized. Centralized methods plan trajectories for all robots on a central computer, which are then sent to the robots to execute via communication. In contrast, in distributed/decentralized methods each robot computes its own trajectory onboard and there might be communication among robots. Combining the two taxonomies, it can be observed that coupled methods can only solve the problem in a centralized way, while the decoupled can be either centralized or distributed. In the following, we focus on distributed methods for multi-robot motion planning. We first describe deterministic methods and then their extensions to probabilistic scenarios under uncertainty.

2.2.1 Deterministic multi-robot motion planning

The problem of multi-robot local motion planning in deterministic scenarios has been actively studied, where the robots' states and dynamics are precisely known. Three main bodies of research for the problem have been developed over the past years.

Reactive methods

Reactive methods compute the next action that each robot needs to take without considering robots' future actions. Early reactive methods include these artificial potential field (APF) based [72–74] which are extensions of the work for a single robot [75]. The most popular category of reactive methods are these based on the velocity obstacle (VO) paradigm. The concept of VO was first introduced for single robot collision avoidance [76], which is defined as a set of velocities of the robot that would lead to a collision with obstacles assuming they are static or moving at a constant speed. Built upon VO, the reciprocal velocity obstacle (RVO) method [77] is developed, which models robot interaction pairwise in a distributed manner and estimates future collisions as a function of their relative velocity. Based on the basic framework, RVO has been extended towards several improvements: the optimal reciprocal collision-avoidance (ORCA) method [23] casting the problem into a linear programming formulation which can be solved efficiently, the NH-ORCA method [78] considering non-holonomic constraints, the generalized RVO method [79] applying for heterogeneous teams of robots, and the ε-cooperative collision avoidance (εCCA) method [80] accounting for the cooperation of robots. Other recent reactive methods include the buffered Voronoi cell (BVC) [81] approach and control barrier functions (CBF) [82]. While these methods are computationally efficient, the robot dynamics are not fully modeled and the planning is limited by only looking one time step ahead. Hence, the resulting robot motions are typically inefficient and robots are easy to be trapped into deadlocks.

Trajectory optimization-based methods

In contrast to reactive methods that only plan one time step ahead, trajectory optimization-based methods plan a local trajectory for each robot within a short time horizon in the future. Most of these methods are built upon the model predictive control (MPC) framework, in which at each time step, each robot plans a local trajectory and corresponding control actions for a planning horizon by solving a constrained optimization problem. But the robot only executes the first planned control action. Then with time going on and at the next time step, the robot replans its trajectory with updated states and environment information. The process is repeated until the robot reaches its goal location. MPC is a very flexible and general framework. Collision avoidance is achieved by imposing inequality constraints and the robot dynamics model can be taken into account by imposing equality constraints in the formulated constrained optimization problem.

In the context of multi-robot motion planning, research focus based on the MPC framework has been given to resolving inter-robot collisions through multi-robot coordination. Early work [83] presents a decentralized MPC approach in which each robot plans its own trajectory assuming other robots are moving at a constant velocity, thus their future trajectories can be predicted and avoided. This naive coordination strategy has also been employed by many recent works [84, 85]. However, while communication among robots is not required, the planned robot trajectory is not guaranteed to be safe, in particular when the robots are moving at a high speed [49]. Another strategy is to let each robot communicate its planned trajectory with other robots in the team. Hence, each robot can update its own trajectory to be collision-free with other robots' trajectory plans, e.g. as in these sequential MPC [4, 86] and distributed MPC [87] works. Techniques in distributed

optimization have also been applied to MPC-based motion planning, the most popular one of which is the alternating direction method of multipliers (ADMM) algorithm [88]. Recent distributed MPC works for multi-robot motion planning based on ADMM include [89–92].

Besides the above coordination strategies, recently game theory has attracted increasing attention in coordinating robots in the context of trajectory optimization-based multi-robot motion planning. Game-theoretic methods typically model each robot and its neighbors with a differential game, formulating multi-robot motion planning as a problem of solving for the Nash equilibrium of the game [93–95]. These methods can be fully decentralized and communication-free while interactions among robots are considered and modeled. However, exactly solving a differential game is generally difficult and computationally heavy, which motivates developments of efficient approximation algorithms [96].

Learning-based methods

Recently, there have emerged new learning-based methods for multi-robot motion planning, such as deep imitation learning [97–99] and those that are reinforcement learning (RL) based [100–103]. These imitation learning (IL) based methods typically employ a supervised end-to-end learning framework, in which a designed neural network is trained to approximate and replace an expert planner that may be computationally expensive or requires global system information. [97] leverages the ORCA algorithm to generate a large training dataset, which is then used to train a deep neural network (DNN). In [98] the expert dataset is generated by running the conflict-based search (CBS) algorithm [104] and the designed architecture is composed of a convolutional neural network (CNN) and a graph neural network (GNN). In [99] the authors rely on a centralized global motion planner [105] to generate a demonstration dataset and then use it to learn a decentralized collision avoidance policy that can run efficiently online. Thanks to centralized features in the dataset, the learned policy is shown able to effectively avoid deadlocks in dense scenarios. Different from imitation learning, the RL-based methods formulate multi-robot motion planning as an observable Markov decision process (POMDP) and solve it using RL approaches. The most popular RL-based methods are those that are based on the deep RL framework [100–103], which can learn policies that have a long-term cumulative reward for the robots and thus are considered to be non-myopic.

Albeit being efficient, these learning-based methods are generally not able to handle hard state constraints to guarantee safety, such as hard collision avoidance constraints. To this end, many recent research efforts have been devoted to providing safety guarantees to learning-based methods by combining them with formal control-based and optimization-based methods [106]. Existing methods include synthesizing the learned policy with a safety controller [99], computing safe reachability sets [107], introducing a control barrier function (CBF) based module in the learning framework, and combining RL with MPC [108].

2.2.2 Multi-robot motion planning under uncertainty

Some of the above deterministic multi-robot motion planning approaches have been extended to probabilistic scenarios where uncertainties arising from robot localization, sens-

ing, and motion are considered. Taking into account bounded robot localization uncertainty, [109, 110] present a method called convex outline collision avoidance under localization uncertainty (COCALU) built upon the RVO paradigm. In the method, given a collision risk threshold δ, the authors first compute a convex hull in which the probability of the robot being located is greater than $1 - \delta$. Next, they calculate the Minkowski sum of the robot's footprint and the convex hull, which is then used in the RVO paradigm for collision avoidance. Such a technique to deal with uncertainty is often categorized as the bounding volume approach, in which the key idea is to replace the robot's footprint with an enlarged one that is computed based on the uncertainty information. Works employing the bounding volume approach include [111] for planar robots in which a rectangular region is computed for each robot and inter-robot collision avoidance is transformed to avoiding overlaps of those regions. Similarly, [84] presents a decentralized MPC where robot motion uncertainty is taken into account by enlarging the robots with their 3-σ confidence ellipsoids. Generally, the bounding volume approach to deal with uncertainty tends to lead to overly conservative results and may make the collision avoidance problem infeasible, particularly in dynamic dense environments [49].

Another widely-used strategy to deal with uncertainty in multi-robot motion planning is to formulate collision chance constraints and solve the problem via chance-constrained optimization. Taking into account both robot state and actuation uncertainties, [112] presents probabilistic RVO (PRVO), which defines the velocity space of the robot that ensures the probability of satisfying RVO constraints is greater than a specified confidence level. The PRVO is then approximated with a set of surrogate constraints in a closed form via chance constraints reformulation techniques. While the method is shown to outperform the bounding volume approach, it is limited to single-integrator robots. Trajectory-optimization based multi-robot motion planning has also been extended to probabilistic scenarios by formulating collision chance constraints. [49] proposes a distributed chance-constrained nonlinear MPC (CCNMPC) method to ensure the probability of inter-robot collision is below a specified threshold, in which the chance constraints are reformulated into deterministic constraints using a local linearization technique. While the method only considers uncertainties modeled as Gaussian distributions, [113] extends it to non-Gaussian cases by using a Gaussian mixture model (GMM).

Some other deterministic multi-robot motion planning methods have also been extended to probabilistic scenarios to incorporate uncertainty. Taking into account the robot measurement uncertainty of other robots, [114] introduces the probabilistic buffered Voronoi cell (PBVC) which defines a probabilistic safe region for each robot to navigate within. Since the PBVC of each robot does not have an analytic solution, they employ a sampling-based approach to approximate it. Built upon the control barrier functions (CBF) [82] theory, [115] proposes probabilistic safety barrier certificates (PrSBC) for multi-robot systems which defines the space of admissible control actions that are probabilistic safe. However, the method is only designed for single-integrator robots.

3

Chance-constrained collision avoidance for MAVs in dynamic environments

Parts of this chapter appeared in:

- H. Zhu, and J. Alonso-Mora, "Chance-constrained collision avoidance for mavs in dynamic environments," *IEEE Robotics and Automation Letters*, 4(2):776-783, Apr. 2019.
- J. Lin, H. Zhu, and J. Alonso-Mora, "Robust vision-based obstacle avoidance for micro aerial vehicles in dynamic environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2020.

3.1 Introduction

Online generation of collision-free trajectories is of utmost importance for safe navigation among other robots and in human-populated environments. In these crowded and dynamic scenarios, reasoning about the uncertainties in self-localization, in estimation of the motion of other agents, and in motion execution becomes increasingly relevant. Furthermore, tight coordination between the robots becomes essential.

In this chapter, we present a probabilistic collision avoidance method for teams of robots that accounts for robot localization and sensing uncertainties, as well as motion disturbances. We focus on micro aerial vehicles (MAVs), but the method also works for other multi-robot systems since we consider a general nonlinear dynamic model of robots. The method leverages chance-constrained nonlinear model predictive control (CCNMPC) to plan a local trajectory, which ensures that the collision probability between each robot and obstacle is below a specified threshold. We consider spherical robots and ellipsoidal dynamic obstacles and assume that the uncertainties are Gaussian distributed. By using a local linearization technique, we transform the chance constraints into deterministic constraints on the robots' states mean and covariance. Such a linearization technique was used for deterministic collision avoidance [86]. We mathematically formalize its use in the context of probability-based stochastic collision avoidance. Thus, a tractable constrained optimization problem is obtained and solved in a receding horizon fashion and online.

Furthermore, we discuss and compare three strategies for planning among other robots, a distributed approach where only the sensed velocity and position of neighboring robots are used, a distributed approach where previous plans of other robots are communicated, and a centralized approach for multi-robot coordination where a sequential planning scheme is employed.

The main contributions of this chapter are:

- An online collision avoidance method for navigation in three dimensional dynamic environments, which utilizes stochastic nonlinear model predictive control to plan safe trajectories with a specified probability of collision.
- A tighter upper bound of the collision probability with ellipsoidal obstacles, which accounts for robot localization, sensing uncertainties and disturbances.
- Incorporation of collision avoidance chance constraints into three frameworks for multi-robot motion planning (sequential, distributed with/without communication).

We evaluate our proposed method in experiments with a team of quadrotors. Fig. 3.1 shows an example of our experiments with two quadrotors avoiding two walking humans.

This chapter is structured as follows. Section 3.2 introduces preliminaries with system models. Section 3.3 describes the collision chance constraints formulation and our method to transform them into tractable deterministic constraints. Section 3.4 presents the planning approach for multi-robot coordination and gives a theoretic discussion of our method. Finally, in Section 3.5 we present and discuss experimental results of the method, followed by concluding remarks in Section 3.6.

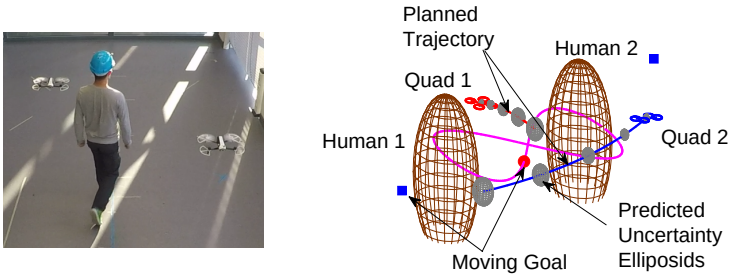


Figure 3.1: Probabilistic collision avoidance among moving obstacles. (a) A snapshot from the experiment. (b) Schematic of quadrotors, humans, trajectory plans and uncertainties.

3.2 Preliminaries

3.2.1 Robot model

Consider a multi-robot system with $n \in \mathbb{N}$ robots moving in a shared workspace $\mathcal{W} \subseteq \mathbb{R}^3$. Let $\mathcal{I} = \{1, \cdots, n\} \subset \mathbb{N}$ denote the index set of all robots in the system. We model each robot $i \in \mathcal{I}$ as an enclosing rigid sphere with radius r_i. The dynamics of robot i are described by the following stochastic nonlinear discrete-time model:

$$\mathbf{x}_i^{k+1} = \mathbf{f}_i(\mathbf{x}_i^k, \mathbf{u}_i^k) + \boldsymbol{\omega}_i^k, \quad \mathbf{x}_i^0 \sim \mathcal{N}(\hat{\mathbf{x}}_i^0, \Gamma_i^0), \tag{3.1}$$

where $\mathbf{x}_i^k = [\mathbf{p}_i^k, \mathbf{v}_i^k, \phi_i^k, \theta_i^k, \psi_i^k]^T \in \mathcal{X}_i \subset \mathbb{R}^{n_x}$ denotes the state of the robot and $\mathbf{u}_i^k \in \mathcal{U}_i \subset \mathbb{R}^{n_u}$ the control inputs at time step k. $\mathbf{p}_i^k \in \mathbb{R}^3$ and $\mathbf{v}_i^k \in \mathbb{R}^3$ are the robot position and velocity, and $\phi_i^k, \theta_i^k, \psi_i^k$ are the robot roll, pitch and yaw angles respectively. $\mathcal{X}_i$ is the state space and n_x is the state vector dimension. $\mathcal{U}_i$ is the control space and n_u is the control input vector dimension. The initial state $\mathbf{x}_i^0$ of the robot is considered as a Gaussian random variable with mean $\hat{\mathbf{x}}_i^0$ and covariance Γ_i^0, which are typically given by a state estimator (we employ an Unscented Kalman Filter (UKF)). $\mathbf{f}_i$ denotes the nonlinear dynamics. We consider uncorrelated process noise $\boldsymbol{\omega}_i^k \sim \mathcal{N}(0, Q_i^k)$ with diagonal covariance matrix Q_i^k. The Parrot Bebop2 quadrotor is employed to evaluate our method. See Appendix A for the dynamics model details.

3.2.2 Obstacle model

We also consider a number of $n_o \in \mathbb{N}_0$ obstacles populated in the environment. Let $\mathcal{I}_o = \{1, \cdots, n_o\} \subset \mathbb{N}_0$ denote the index set of all obstacles. For each obstacle $o \in \mathcal{I}_o$ at position $\mathbf{p}_o \in \mathbb{R}^3$, we model it as a non-rotating enclosing ellipsoid with semi-principal axes (a_o, b_o, c_o) and a rotation matrix R_o. Denote by $\mathbf{v}_o$ the velocity of the the obstacle. Static obstacle positions are assumed to be available for planning. For dynamic obstacles, as in [116], we assume they are continuously tracked in the environment and employ a constant velocity model (CVM) with Gaussian noise $\boldsymbol{\omega}_o(t) \sim \mathcal{N}(0, Q_o(t))$ in acceleration, i.e. $\ddot{\mathbf{p}}_o(t) = \boldsymbol{\omega}_o(t)$ for trajectory prediction. Specifically, given measured obstacle's position data, we estimate and predict their future positions and uncertainties with a linear Kalman Filter. Fig. 3.2

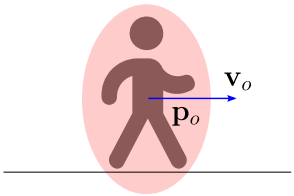


Figure 3.2: A human obstacle modeled as an enclosing ellipsoid with position $\mathbf{p}_o$ and velocity $\mathbf{v}_o$.

shows an illustration of modelling a human obstacle as an enclosing ellipsoid.

3.2.3 Collision chance constraints

The collision condition of robot i with respect to another robot j at time step k is defined as

$$C_{ij}^k := \{\mathbf{x}_i^k \mid \|\mathbf{p}_i^k - \mathbf{p}_j^k\| \leq r_i + r_j\}. \tag{3.2}$$

Collision checking between a robot i and an obstacle o requires calculating the minimum distance between a sphere and an ellipsoid, which can not be performed in closed form [117]. To this end, we approximate the obstacle with an enlarged ellipsoid and check if the robot's position is inside it. The collision condition is

$$C_{io}^k := \{\mathbf{x}_i^k \mid \|\mathbf{p}_i^k - \mathbf{p}_o^k\|_{\Omega_{io}} \leq 1\}, \tag{3.3}$$

where $\Omega_{io} = R_o^T \mathrm{diag}(1/(a_o + r_i)^2, 1/(b_o + r_i)^2, 1/(c_o + r_i)^2) R_o$.

Note that the positions of the robots and obstacles are random variables described by unbounded probability distributions. Hence, the collision avoidance constraints can only be satisfied in a probabilistic manner, which are formulated as chance constraints for robot i:

$$\Pr(\mathbf{x}_i^k \notin C_{ij}^k) \geq 1 - \delta_r, \quad \forall j \in \mathcal{I}, j \neq i \tag{3.4}$$

$$\Pr(\mathbf{x}_i^k \notin C_{io}^k) \geq 1 - \delta_o, \quad \forall o \in \mathcal{I}_o \tag{3.5}$$

where δ_r, δ_o are the probability thresholds for inter-robot and robot-obstacle collision respectively.

3.2.4 Problem formulation

A distributed collision avoidance problem is formulated. For each robot $i \in \mathcal{I}$, we formulate a discrete time chance-constrained optimization problem with N time steps and planning horizon $\tau = N\Delta t$, where Δt is the time step.

***Problem* 3.1** (Probabilistic collision avoidance with chance constraints)**.** *For robot i, given the position distributions $\mathbf{p}_j^{0:N}$ of other robots $j \in \mathcal{I}, j \neq i$ and position distributions $\mathbf{p}_o^{0:N}$ of*

obstacles $o \in \mathcal{I}_o$, the initial state $\hat{\mathbf{x}}_i^0$ with uncertainty covariance Γ_i^0, the goal position $\mathbf{p}_{ig}$, and the collision probability thresholds ϵ_r, ϵ_o, the objective is to compute optimal trajectories and control inputs for the robot to progress from its initial state to its goal while the collision probability with each obstacle and robot is below given thresholds. The resulting optimization problem is

$$\min_{\hat{\mathbf{x}}_i^{1:N}, \mathbf{u}_i^{0:N-1}} \quad \sum_{k=0}^{N-1} J_i^k(\hat{\mathbf{x}}_i^k, \mathbf{u}_i^k) + J_i^N(\hat{\mathbf{x}}_i^N) \tag{3.6a}$$

$$s.t. \quad \mathbf{x}_i^0 = \hat{\mathbf{x}}_i(0), \tag{3.6b}$$

$$\hat{\mathbf{x}}_i^k = \mathbf{f}_i(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}), \tag{3.6c}$$

$$\Pr(\mathbf{x}_i^k \notin \mathcal{C}_{ij}^k) \geq 1 - \delta_r, \forall j \in \mathcal{I}, j \neq i \tag{3.6d}$$

$$\Pr(\mathbf{x}_i^k \notin \mathcal{C}_{io}^k) \geq 1 - \delta_o, \forall o \in \mathcal{I}_o \tag{3.6e}$$

$$\mathbf{u}_i^{k-1} \in \mathcal{U}_i, \quad \hat{\mathbf{x}}_i^k \in \mathcal{X}_i, \tag{3.6f}$$

$$\forall k \in \{1, \dots, N\}.$$

where J_i^k denotes the cost term of the robot at time k and J_i^N denotes the terminal cost.

***Remark* 3.1.** *The positions of other robots and obstacles $\mathbf{p}_j^{0:N}$, $\mathbf{p}_o^{0:N}$ are assumed to follow Gaussian distributions.*

***Remark* 3.2.** *In Section 3.4, we describe several assumptions to obtain the predicted positions $\mathbf{p}_j^{0:N}$ of other robots.*

3.2.5 Approximate uncertainty propagation

Evaluating the chance constraints (3.6d) and (3.6e) requires calculating the uncertainty covariance at each time step, i.e. uncertainty propagation. There are many methods for uncertainty propagation for nonlinear systems, for example the polynomial chaos expansions based methods [118] and the differential algebra techniques [119, 120]. The readers can refer to [121] to get a comprehensive review. However, these methods are mostly computationally intensive and only outperform linearization methods when the propagation time is very long. In our case where the planning horizon is short, to achieve real time performance, we propagate uncertainties using a EKF-type update,

$$\Gamma_i^k = F_i^k \Gamma_i^{k-1} (F_i^k)^T + Q_i^k, \tag{3.7}$$

where Γ_i^{k-1} is the state uncertainty covariance at time $k-1$, Q_i^k is the process noise and

$$F_i^k = \left. \frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_i} \right|_{\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^k} \tag{3.8}$$

is the state transition matrix of the robot. We further denote by Σ_i^k the 3x3 covariance matrix of the position $\mathbf{p}_i^k$, extracted from Γ_i^k.

***Remark* 3.3.** *The covariance dynamics are dependent on the robot state and control inputs. Hence, it requires $\frac{N}{2}(n_x^2 + n_x)$ additional variables in the optimization Problem 3.1, which can increase the computation time greatly. In this chapter, to avoid the need of additional variables, and similar to [43], we propagate the robot uncertainties based on its last-loop trajectory and control inputs.*

***Remark* 3.4.** *If the initial state uncertainty is Gaussian, the predicted state uncertainties are Gaussian distributed when propagated using the linearized update with F_i^k computed from the last-loop trajectory and control inputs.*

3.3 Chance constraints formulation

We now present the method to address the chance constraints of Eqs. (3.6d) and (3.6e). The basic idea is to first linearize the collision conditions of Eqs. (3.2) and (3.3) to get linear chance constraints and then reformulate them into deterministic constraints on the mean and covariance of the robot states.

3.3.1 Linear chance constraints

Consider a linear chance constraint in the form $\Pr(\mathbf{a}^T\mathbf{x} \le b) \le \delta$, where $\mathbf{x} \in \mathbb{R}^{n_x}$ is a random variable, $a \in \mathbb{R}^{n_x}, b \in \mathbb{R}$ are constants and δ is the probability threshold. Assuming that $\mathbf{x}$ follows a Gaussian distribution, the chance constraint can be transformed into a deterministic constraint.

***Lemma* 3.1.** *([30]) Given a multivariate random variable $\mathbf{x} \sim \mathcal{N}(\hat{\mathbf{x}}, \Sigma)$, then*

$$\Pr(\mathbf{a}^T\mathbf{x} \le b) = \frac{1}{2} + \frac{1}{2}\mathrm{erf}\left(\frac{b - \mathbf{a}^T\hat{\mathbf{x}}}{\sqrt{2\mathbf{a}^T\Sigma\mathbf{a}}}\right),$$

where $\mathrm{erf}(\cdot)$ *is the standard error function defined as* $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}\,dt$.

***Lemma* 3.2.** *([30]) Given a multivariate random variable $\mathbf{x} \sim \mathcal{N}(\hat{\mathbf{x}}, \Sigma)$ and a probability threshold $\delta \in (0, 0.5)$, then*

$$\Pr(\mathbf{a}^T\mathbf{x} \le b) \le \delta \iff \mathbf{a}^T\hat{\mathbf{x}} - b \ge c,$$

where $c = \mathrm{erf}^{-1}(1 - 2\delta)\sqrt{2\mathbf{a}^T\Sigma\mathbf{a}}$, *and* $\mathrm{erf}^{-1}(\cdot)$ *is the inverse of* $\mathrm{erf}(\cdot)$.

Given the probability threshold δ, the corresponding error function and its inverse can be obtained by table look-up or using series approximation techniques.

3.3.2 Inter-robot collision avoidance chance constraints

We now consider the inter-robot collision avoidance constraints, Eq. (3.6d). For simplicity, we omit the superscript $\cdot^k$ in this section. Given positions and uncertainty covariances

of the two robots $\mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$, $\mathbf{p}_j \sim \mathcal{N}(\hat{\mathbf{p}}_j, \Sigma_j)$, the instantaneous collision probability of robot i with robot j is

$$\Pr(\mathbf{x}_i \in C_{ij}) = \int_{\mathbb{R}^3} I_C(\mathbf{p}_i, \mathbf{p}_j) p(\mathbf{p}_i) p(\mathbf{p}_j) d\mathbf{p}_i d\mathbf{p}_j, \tag{3.9}$$

where I_C is the indicator function

$$I_C(\mathbf{p}_i, \mathbf{p}_j) = \begin{cases} 1, & \text{if } \|\mathbf{p}_i - \mathbf{p}_j\| \le r_i + r_j; \\ 0, & \text{otherwise.} \end{cases}$$

We assume that $\mathbf{p}_i$ and $\mathbf{p}_j$ are independent Gaussian distributions, then $\mathbf{p}_i - \mathbf{p}_j$ is also a Gaussian distribution, i.e. $\mathbf{p}_i - \mathbf{p}_j \sim \mathcal{N}(\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j, \Sigma_i + \Sigma_j)$. Hence, the collision probability defined by Eq. (3.9) can be written as

$$\Pr(\mathbf{x}_i \in C_{ij}) = \int_{\|\mathbf{p}_i - \mathbf{p}_j\| \le r_i + r_j} p(\mathbf{p}_i - \mathbf{p}_j) d(\mathbf{p}_i - \mathbf{p}_j),$$

which is an integral of a multivariate Gaussian probability density function over a sphere, as illustrated in Fig. 3.3a.

However, there is no closed form to calculate the collision probability. But we can obtain an approximated upper bound by linearizing the collision condition. As shown in Fig. 3.3b, we enlarge the spherical collision region C_{ij} into a half space $\tilde{C}_{ij}$, which is defined as

$$\tilde{C}_{ij} \coloneqq \{\mathbf{x} \mid \mathbf{a}_{ij}^T(\mathbf{p}_i - \mathbf{p}_j) \le b_{ij}\},$$

where $\mathbf{a}_{ij} = (\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j)/\|\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j\|$ and $b_{ij} = r_i + r_j$.

It is apparent that $C_{ij} \subset \tilde{C}_{ij}$, thus $\Pr(\mathbf{x}_i \in C_{ij}) \le \Pr(\mathbf{x}_i \in \tilde{C}_{ij})$. Hence, following Lemma 3.1, we can obtain an upper bound of the collision probability between two robots:

$$\Pr(\mathbf{x}_i \in C_{ij}) \le \frac{1}{2} + \frac{1}{2}\mathrm{erf}\left(\frac{b_{ij} - \mathbf{a}_{ij}^T(\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j)}{\sqrt{2\mathbf{a}_{ij}^T(\Sigma_i + \Sigma_j)\mathbf{a}_{ij}}}\right). \tag{3.10}$$

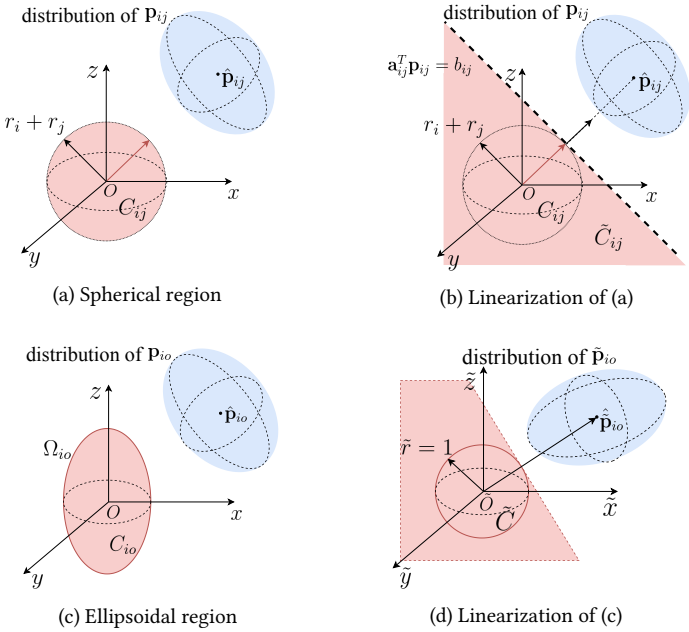
Following Lemma 3.2, the collision chance constraint of Eq. (3.6d) can be transformed into a deterministic constraint,

$$\mathbf{a}_{ij}^T(\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j) - b_{ij} \ge \mathrm{erf}^{-1}(1 - 2\delta_r)\sqrt{2\mathbf{a}_{ij}^T(\Sigma_i + \Sigma_j)\mathbf{a}_{ij}}. \tag{3.11}$$

3.3.3 Robot-obstacle collision avoidance chance constraints

For the collision avoidance constraints of Eq. (3.6e), by assuming that the positions of the robot and obstacle are independent random variables, the collision probability is

$$\Pr(\mathbf{x}_i \in C_{io}) = \int_{\|\mathbf{p}_i - \mathbf{p}_o\|_{\Omega_{io}} \le 1} p(\mathbf{p}_i - \mathbf{p}_o) d(\mathbf{p}_i - \mathbf{p}_o), \tag{3.12}$$



(a) Spherical region (b) Linearization of (a)

(c) Ellipsoidal region (d) Linearization of (c)

Figure 3.3: Chance constraints linearization. Red: collision region. Blue: confidence ellipsoid representation of the Gaussian distributed robot-robot/obstacle relative position. (a) Collision constraint with a sphere region; (b) Linearization with a half space; (c) Collision constraint with an ellipsoid region; (d) Transformation into a unit sphere region and linearization.

where the collision region C_{io} described by Ω_{io} is an ellipsoid instead of a sphere, as shown in Fig. 3.3c.

To linearize the collision condition, we first perform the affine coordinate transformation

$$W = \Omega_{io}^{\frac{1}{2}}. \tag{3.13}$$

Then the collision region is transformed into a unit sphere C_{io}^W, as illustrated in Fig. 3.3d. The robot and obstacle positions are transformed to new Gaussian distributions, i.e. $\mathbf{p}_i^W \sim \mathcal{N}(\hat{\mathbf{p}}_i^W, \Sigma_i^W)$, $\mathbf{p}_o^W \sim \mathcal{N}(\hat{\mathbf{p}}_o^W, \Sigma_o^W)$, where

$$\begin{aligned} \hat{\mathbf{p}}_i^W &= \Omega_{io}^{\frac{1}{2}} \hat{\mathbf{p}}_i, \quad \Sigma_i^W = \Omega_{io}^{\frac{1}{2}T} \Sigma_i \Omega_{io}^{\frac{1}{2}}, \\ \hat{\mathbf{p}}_o^W &= \Omega_{io}^{\frac{1}{2}} \hat{\mathbf{p}}_o, \quad \Sigma_o^W = \Omega_{io}^{\frac{1}{2}T} \Sigma_o \Omega_{io}^{\frac{1}{2}}. \end{aligned} \tag{3.14}$$

Here we use the super-script $\cdot^W$ to indicate variables in the transformed coordinate frame. In the new coordinate framework, let

$$\Pr(\mathbf{x}_i^W \in C_{io}^W) = \int_{\|\mathbf{p}_i^W - \mathbf{p}_o^W\| \leq 1} p(\mathbf{p}_i^W - \mathbf{p}_o^W) d(\mathbf{p}_i^W - \mathbf{p}_o^W),$$

then we have $\Pr(\mathbf{x}_i \in C_{io}) = \Pr(\mathbf{x}_i^W \in C_{io}^W)$.

Now, we can use the same linearization technique as for the sphere region with $\mathbf{a}_{io} = (\hat{\mathbf{p}}_i^W - \hat{\mathbf{p}}_o^W)/\|\hat{\mathbf{p}}_i^W - \hat{\mathbf{p}}_o^W\|$ and $b_{io} = 1$. The collision chance constraint of Eq. (3.6e) can thus be transformed into the following deterministic constraint:

$$\mathbf{a}_{io}^T \Omega_{io}^{\frac{1}{2}} (\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_o) - b_{io} \geq \mathrm{erf}^{-1}(1 - 2\delta_o) \cdot \sqrt{2\mathbf{a}_{io}^T \Omega_{io}^{\frac{1}{2}} (\Sigma_i + \Sigma_o) \Omega_{io}^{\frac{1}{2}T} \mathbf{a}_{io}}. \tag{3.15}$$

3.3.4 Comparison to other methods

We compare our method with several state-of-the-art collision probability approximation algorithms using a robot-obstacle proximity example. A point robot at position mean $(0.7, 0.7, 0.8)$ m with covariance diag(0.04,0.04,0.01) m^2 is close to an ellipsoid obstacle at origin with semi-principle axes $(0.6, 0.6, 2.2)$ m. See Table 3.1 for the collision probability computation results. The numerical integration result is the exact collision probability and gives a collision probability of 0.011. If we define the collision probability threshold to be $\delta = 0.03$ (thus confidence level 0.97), which corresponds to the 3σ confidence ellipsoid, then this configuration is feasible. However, when employing the enlarged bounding volume method [122], or the cube approximation [31], the configuration would be deemed infeasible. The center point PDF approximation approach [29] can give feasible checking results, but the resulting collision probability is significantly smaller than the real value, which may lead to unsafe trajectory planning. Our method thus provides a tighter bound.

Table 3.1: Comparison of collision probability algorithms.

Algorithms	Collision proba.	Comput. time (ms)	Feasible?
Numerical integral	0.011	258.665	Yes
Bounding volume [123]	1	0.011	No
Center point [29]	3.6E-18	0.016	Yes
Cube approx. [31]	0.100	0.044	No
Our method	0.017	0.011	Yes

3.4 Online local planning

We now present a tractable MPC formulation for each robot, followed by three approaches to obtain future position information of other robots and a theoretical discussion.

3.4.1 Deterministic MPC formulation

We first describe the components of the cost function presented in Eq. (3.6a), which are listed in the following.

Cost function

- *Goal navigation.* Let $\mathbf{p}_{ig}$ be the goal position of robot i, we minimize the displacement between its terminal position at the planning horizon and its goal. To this end, we define the terminal cost term,

$$J_i^N(\hat{\mathbf{x}}_i^N) = l_{i,g}\|\mathbf{p}_{ig} - \hat{\mathbf{p}}_i^N\|/\|\mathbf{p}_{ig} - \hat{\mathbf{p}}_i^0\|, \tag{3.16}$$

  where $l_{i,g}$ is the weight coefficient.

- *Control inputs cost.* The second cost term is to minimize the MAV control inputs, designed as a stage cost,

$$J_{i,u}^k(\mathbf{u}_i^k) = l_{i,u}\|\mathbf{u}_i^k\|, \quad k = \{0, 1, \dots, N-1\}, \tag{3.17}$$

  where $l_{i,u}$ is the weight coefficient.

- *Collision potential cost.* To improve flight safety, we also introduce an obstacle potential field cost. Denote by $d_{ij} = \|\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j\|_{\Omega_{ij}}$ the distance between robot i and robot/obstacle j. Two different forms of collision potential cost between i and j are tested. The first form is

$$J_{i,j,c}^k(\hat{\mathbf{x}}^k) = \begin{cases} l_{i,c}(d_{ij}^{\text{safe}} - d_{ij}), & \text{if } d_{ij} \le d_{ij}^{\text{safe}}; \\ 0, & \text{otherwise,} \end{cases} \tag{3.18}$$

  where $l_{i,c}$ is the weight coefficient and d_{ij}^{safe} is the safe potential field distance between robot i and robot/obstacle j. The other form is based on the logistic function

$$J_{i,j,c}^k(\hat{\mathbf{x}}^k) = \frac{l_{i,c}}{1 + \exp}(\lambda_{i,c}(d_{ij} - dij^{\text{safe}})), \tag{3.19}$$

  where $\lambda_{i,c}$ is a parameter defining the smoothness of the cost function. Thus, the collision potential cost for robot i at time step k is

$$J_{i,c}^k(\hat{\mathbf{x}}_i^k) = \sum_{j \in \mathcal{I} \cup \mathcal{I}_o, j \neq i} J_{i,j,c}^k(\hat{\mathbf{x}}_i^k). \tag{3.20}$$

Optimization problem

By transforming the chance constraints into the deterministic constraints presented in Section 3.3 and utilizing the above cost terms, the following tractable deterministic MPC formulation for Problem 3.1 can be derived:

$$\begin{aligned} \min_{\hat{\mathbf{x}}_i^{1:N}, \mathbf{u}_i^{0:N-1}} \quad & J_i^N(\hat{\mathbf{x}}_i^N) + \sum_{k=0}^{N-1} J_{i,u}^k(\mathbf{u}_i^k) + \sum_{k=1}^{N} J_{i,c}^k(\hat{\mathbf{x}}_i^k) \\ \text{s.t.} \quad & \mathbf{x}_i^0 = \hat{\mathbf{x}}_i(0), \\ & \hat{\mathbf{x}}_i^k = \mathbf{f}_i(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}), \\ & g_{ij}^k(\hat{\mathbf{x}}_i^k, \hat{\mathbf{p}}_j^k, \Sigma_i^k, \Sigma_j^k, \delta_r) \le 0, \\ & g_{io}^k(\hat{\mathbf{x}}_i^k, \hat{\mathbf{p}}_o^k, \Sigma_i^k, \Sigma_o^k, \delta_o) \le 0, \\ & \mathbf{u}_i^{k-1} \in \mathcal{U}_i, \quad \hat{\mathbf{x}}_i^k \in \mathcal{X}_i, \\ & \forall j \neq i \in \mathcal{I}; \forall o \in \mathcal{I}_o; \forall k \in \{1, \dots, N\}. \end{aligned} \tag{3.21}$$

where g_{ij}^k and g_{io}^k denote the deterministic constraints of Eq. (3.11) and (3.15) for probabilistic inter-robot and robot-obstacle collision avoidance respectively, and the position uncertainty covariances Σ_i^k are computed as discussed in Remark 3.3.

3.4.2 Multi-robot planning

In the CCNMPC formulation the position distribution for all other robots $j \neq i$, given by $\hat{\mathbf{p}}_j^{0:N}$ and $\Sigma_j^{0:N}$, is assumed known. Next we discuss three methods to obtain these values, but the CCNMPC formulation is general and other coordination approaches could be devised.

Constant velocity model without communication

By regarding all other robots as dynamic obstacles and employing a constant velocity model, one robot can predict other robots future behaviors based on onboard measurements. Hence, each robot can plan its own trajectory independently and without communication, which leads to a distributed planning scheme for multi-robot collision avoidance.

Given the current position and velocity distribution $\hat{\mathbf{p}}_j^0, \hat{\mathbf{v}}_j^0$ and $\Sigma_{j,pv}^0$ of robot j, we compute

$$\begin{aligned} [\hat{\mathbf{p}}_j^k, \hat{\mathbf{v}}_j^k]^T &= F_j^k [\hat{\mathbf{p}}_j^{k-1}, \hat{\mathbf{v}}_j^{k-1}]^T, \\ \Sigma_{j,pv}^k &= F_j^k \Sigma_{j,pv}^{k-1} {F_j^k}^T + Q_{j,pv}^k, \end{aligned} \tag{3.22}$$

where the state transition matrix $F_j^k = \begin{bmatrix} I_3 & \Delta t I_3 \\ O & I_3 \end{bmatrix}$, Δt is the time step for prediction, $Q_{i,pv}^k$ is the additive process noise of the model. The position uncertainty covariance is $\Sigma_j^k = \Sigma_{j,pv}^k(1:3, 1:3)$.

Sequential planning with communication

If the team of robots is centrally controlled, or a fast communication channel is available, higher coordination can be achieved by planning trajectories sequentially, i.e., each robot plans a trajectory that avoids the trajectories of all other robots and then communicates its trajectory (given by $\hat{\mathbf{p}}_i^{0:N}$ and $\Sigma_i^{0:N}$).

Denote by $\mathcal{T}_i^t = \{\hat{\mathbf{p}}_i^{0:N}, \Sigma_i^{0:N}\}|_t$ the trajectory for robot i planned at time t. At the initial time $t = 0$ robot i avoids only the plans $\mathcal{T}_j^0$ of other robots with $j < i$, in a priority scheme. In subsequent time steps, robot i plans a trajectory $\mathcal{T}_i^t$ that avoids $\mathcal{T}_j^t$ for all $j < i$ and $\mathcal{T}_j^{t-\Delta t}$ for all $j > i$.

Distributed planning with communication

Robots communicate their planned trajectories. At every time step, every robot avoids the planned trajectories of all other robots in the previous time-step. That is, at time t, robot i plans a trajectory $\mathcal{T}_i^t$ that avoids $\mathcal{T}_j^{t-\Delta t}$ for all $j \neq i \in \mathcal{I}$.

3.4.3 Theoretical discussion

Collision avoidance

Our formulation imposes, by construction, that the probability of collision with respect to each obstacle and at every stage of the plan is less or equal than δ_o under a constant velocity assumption for moving obstacles (Section 3.2.2) and a simplified propagation model (Section 3.2.5). For collision avoidance with other robots in the team, guarantees vary according to the coordination methods (and the associated assumptions) described in Section 3.4.2.

Probability of collision with any given obstacle

From Section 3.4.3, the probability of collision of robot i at time step k with respect to *any* given obstacle can be bounded by

$$\Pr(\mathbf{x}_i^k \in \bigcup_{o=1}^{n_o} C_{io}^k) \leq \sum_{o=1}^{n_o} \Pr(\mathbf{x}_i^k \in C_{io}^k) = n_o \delta_o,$$

where n_o is the number of obstacles. By choosing $\delta_o = \delta_{\text{all}}/n_o$, one may specify a joint threshold of collision δ_{all}.

Probability of collision for the planned trajectory

From Section 3.4.3, at all stages the probability of collision with any given obstacle is less or equal than the specified threshold δ_o. The probability of collision for the whole trajectory of robot i with respect to each obstacle can be bounded by

$$\Pr(\bigvee_{k=1}^{N} (\mathbf{x}_i^k \in C_{io}^k)) \leq \sum_{k=1}^{N} \Pr(\mathbf{x}_i^k \in C_{io}^k).$$

In our case this bound would be $N\delta_o$, but it is over conservative in practice. We argue that, in the context of online receding horizon planning it is beneficial to impose a probability of collision of δ_o for each individual stage - instead of for the whole trajectory - thanks to the fast re-planning and relatively small displacement between stages.

Furthermore, our formulation is consistent with a stochastic formulation of the MPC problem where the chance constraint is defined as a discounted sum of violation probabilities in the finite horizon, as proposed for example by. The rationality with this formulation is also that by penalizing violation probabilities close to the initial time and relaxing the penalty of violation probabilities in the far future, feasibility of the online optimization is enabled.

The discounted chance constraint with respect to an obstacles is defined as:

$$\sum_{k=1}^{N} (\gamma)^k \Pr(\mathbf{x}_i^k \in C_{io}^k) \leq \delta_o, \tag{3.23}$$

where $\gamma \in (0,1)$ is the discounting factor.

***Lemma* 3.3.** *Our formulation provides an upper bound in the discounted probability of collision, i.e. Eq. (3.23) is satisfied, if the discounting factor* $\gamma < 0.5$.

Proof. Our formulation guarantees that $\Pr(\mathbf{x}_i^k \in C_{io}^k) \leq \delta_o, \forall k = 1, \cdots, N$. Hence, the discounted probability of collision satisfies

$$\sum_{k=1}^{N} (\gamma)^k \Pr(\mathbf{x}_i^k \in C_{io}^k) \leq \delta_o \sum_{k=1}^{N} (\gamma)^k = \frac{\gamma(1-\gamma^N)}{1-\gamma} \delta_o.$$

Given $\gamma < 0.5$, we have $\gamma(1-\gamma^N) - (1-\gamma) = 2\gamma - 1 - \gamma^{N+1} < 0$. Thus, $\frac{\gamma(1-\gamma^N)}{1-\gamma} < 1$. Hence, $\sum_{k=1}^{N} (\gamma)^k \Pr(\mathbf{x}_i^k \in C_{io}^k) \leq \delta_o$. □

In this proof we also employ the conservative bound on the joint probability of collision. Future works should look at obtaining tighter bounds on the joint probability of collision over the whole trajectory.

Feasibility

Due to unmodeled dynamics, disturbances, or deviations from the simplifying assumptions, the optimization problem may become infeasible. In those rare situations, our approach is to command the MAVs to decelerate. Typically, the problem becomes feasible again after a small number of steps (below half a second, see Section 3.5.3).

3.5 Results

In this section we describe our implementation of the proposed method and evaluate it in experiments and simulations. We first show experimental results of applying the proposed CCNMPC to vision-based obstacle avoidance for a MAV in dynamic environments. Then we validate collision avoidance for multiple MAVs and compare the performance of our method with state-of-the-art methods. Finally, we evaluate and compare the three multi-robot coordination strategies described in Section 3.4.2 in simulations.

3.5.1 Experimental setup

Our experimental platform is the Parrot Bebop 2 quadrotor. Two different setups are tested in our experiments. In both setups, the collision probability thresholds are set to $\delta_r = 0.03$ and $\delta_o = 0.03$.

Fully onboard setup

In this setup, we use the Parrot Bebop 2 quadrotor mounted with an NVIDIA Jetson TX2 Module and an Intel RealSense Depth Camera D435i, as shown in Fig. 3.4. The Parrot Bebop 2 allows for executing control commands sent via Robot Operating System (ROS). The D435i camera is dually used for visual-inertial odometry and depth image sensing, which has a $87^\circ \times 58^\circ$ FOV and 5 m depth sensing range. The TX2 is used to perform all onboard computation and is connected with the Bebop 2 via WiFi.

We use a filtering-based stereo visual-inertial odometry algorithm, the S-MSCKF [124], for state estimation of the MAV, which runs at 15 Hz. The camera depth images are received at 60 Hz. A fast depth image-based algorithm [125] run at frame rate is employed for obstacle detection and tracking which gives obstacle ellipsoids position, velocity and size. We rely on the ACADO toolkit [126] to generate a fast C solver for our MPC, in which a sampling time of 60 ms is used and the prediction horizon is set to 1.5 s. The radius of the MAV is set as 0.4 m. The two closest detected obstacles are fed to the MPC for collision avoidance.

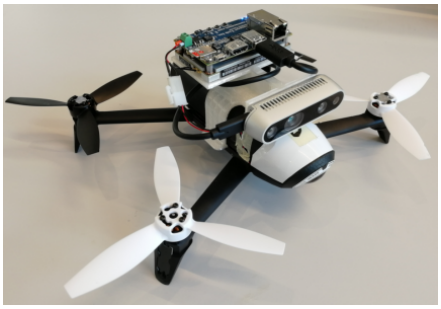


Figure 3.4: MAV used in the experiments. It is equipped with an NVIDIA Jetson TX2 Module for all on-board computation, an Intel RealSense Depth Camera D435i dually for visual-inertial odometry and depth image sensing.

Motion capture system

In this setup, perception is performed via an external motion capture system (Mocap), including robot localization and obstacle tracking. Specifically, the Mocap (OptiTrack in this thesis) is used to measure the pose of each object, including all robots and obstacles in the environment. It gives real-time stamped pose information of the objects in a predefined world frame at a frequency of 120 Hz. We assumed this to be the "real" pose of the objects.

To simulate the uncertainty considered in motion planning, we manually add Gaussian noise with a zero mean and a specified standard deviation to the obtained original Mocap pose data. Taking these noisy measurements as inputs, a Kalman filter is employed to estimate the state of each object, including its position, velocity and orientation, which are then described by Gaussian distributions with known means and covariances in the world frame. These states described by Gaussian distributions are used as inputs of our developed motion planning methods.

We use an Intel i7 CPU@2.6GHz computer for the planner and use ROS to send commands to the quadrotors. We rely on the solver Forces Pro [127] to generate optimized NMPC code. The radius of each quadrotor is set as 0.3 m. The time step used in the NMPC is $\Delta t = 0.05$ s and the total number of steps is $N = 20$.

3.5.2 Robust vision-based collision avoidance

We first test our method in the fully onboard setup with a single drone avoiding moving obstacles. The results of two typical scenarios are particularly presented here.

Scenario 1 (Flying in a confined lab space)

The MAV is required to navigate from a start point to an end point while avoiding two walking humans. Fig. 3.5 shows a snapshot of the experiment.

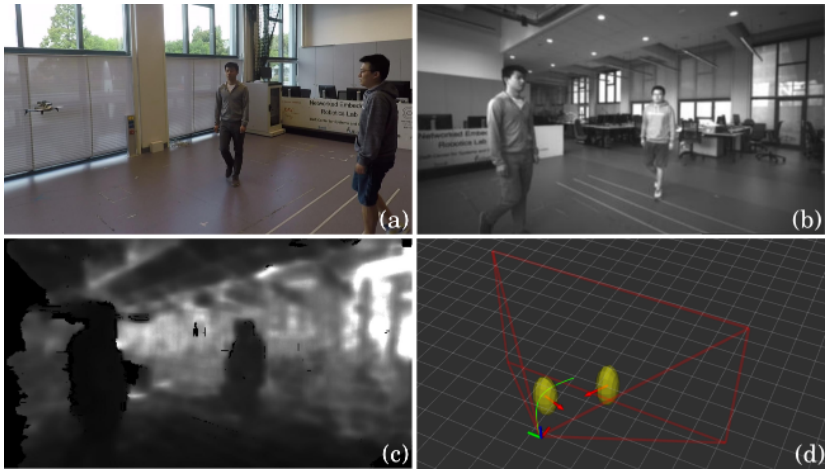
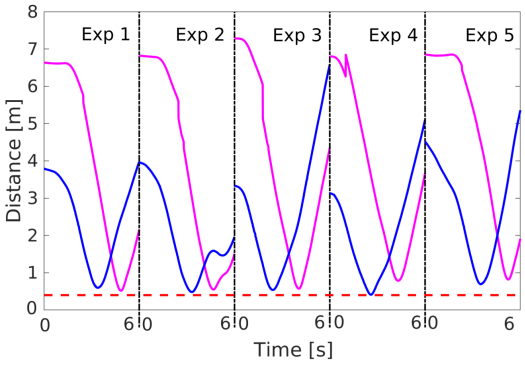
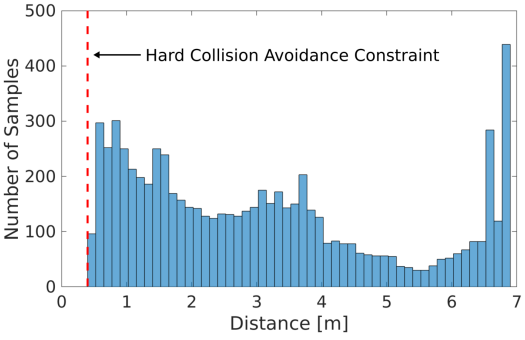


Figure 3.5: Experimental results of vision-based collision avoidance in dynamic environments with two moving humans. The MAV is equipped with a stereo camera both for visual odometry and obstacle detection. (a) A snapshot of the experiment. (b) On-board grayscale image. (c) The depth image. (d) Visualization of detected obstacles (yellow ellipsoids with red arrows indicating the velocities) and planned collision-free trajectory (green curve).

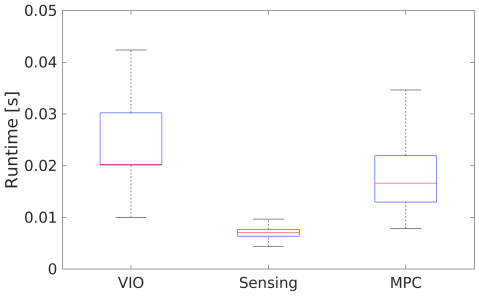
In this scenario, we performed the experiment five times and used OptiTrack to measure the position of the MAV and human obstacles which are regarded as ground truth data. Fig. 3.6a shows the measured distance between the MAV and the two moving human obstacles over time in the five experiments. The distance is computed, based on ground truth measurements, as the closest distance between the MAV's position and the obstacle ellipsoid's surface (with semi-major axis $(0.4, 0.4, 0.9)$ m). In Fig. 3.6b, we cumulate all the distance data. It can be observed that in all instances a minimum safe separation of 0.4 m was achieved and therefore collisions with the humans were avoided. A maximal speed of around 1.6 m/s of the MAV was observed in this experiment.



(a) Distance to obstacles over time.



(b) Histogram of distance.



(c) MAV on-board runtime.

Figure 3.6: Quantitative results of the experiment Scenario 1. (a) Distance between the MAV and the two moving obstacles (magenta and blue) over time during 5 experiments. (b) Histogram of all the distance data. (c) On-board runtime of the MAV state estimation (VIO), obstacle detection and tracking, and collision-free trajectory optimization (MPC).

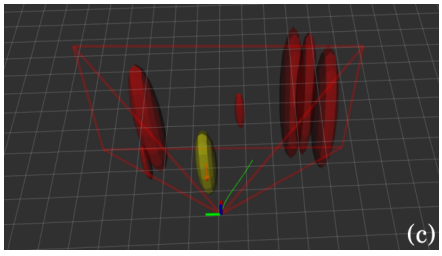
The box plots of the on-board runtime in this scenario is shown in Fig. 3.6c. For the runtime of the obstacle detection and tracking, the 75^{th} percentile is always below 8 ms, which is fast enough to be run at frame rate (60 Hz). For the runtime of the MPC framework, the 75^{th} percentile is always below 22 ms, indicating the framework can be run efficiently in real time.

Scenario 2 (Flying in a long corridor)

The MAV is flying in a long narrow corridor where there are both static and moving obstacles. Fig. 3.7 shows a snapshot taken during the experiment. A maximum speed of around 2.4 m/s was achieved by the MAV in the experiment.

(a) A snapshot during the experiment.

(b) An on-board grayscale image captured in the experiment.



(c) visualization of the corresponding obstacle detection and trajectory planning results.

Figure 3.7: Results of the experiment Scenario 2. The MAV is flying in a corridor while avoiding static and moving obstacles.

3.5.3 Collision avoidance for multiple MAVs

We then test our method with two drones avoiding walking human obstacles in the motion capture system setup. The added measurements noise is zero mean with covariance $\Sigma = \text{diag}(0.06\text{ m}, 0.06\text{ m}, 0.06\text{ m}, 0.4\text{ deg}, 0.4\text{ deg}, 0.4\text{ deg})^2$. Based on our experimental data, the average resulted state estimation error is $\|\hat{\mathbf{p}} - \mathbf{p}\| = 0.05$ m in terms of the quadrotors' position. Fig. 3.1 shows a snapshot from our experiment. In Fig. 3.8a we cumulate the distance between the two drones. They maintained a safe distance of 0.6 m over the entire run. In Fig. 3.8b we cumulate the distance between each drone and each moving human. The distance is computed as the closest distance between the quadrotor's position and the ellipsoid's surface. In all instances a minimum safe separation of 0.3 m was achieved. Close distances between robots and obstacles are observed, since they share a quite confined space. In Fig. 3.8c we show the computation time of each NMPC solver and the central sequential planning framework. The mean computation time of the NMPC solver is 14.3 ms and that of the total framework is 71.3 ms. The framework includes state estimation, uncertainty propagation, obstacles' prediction, communication and solving both NMPC problems. Among all NMPC solutions over the entire run, the percentage of infeasible solutions was 2.8% and the longest infeasible period was 9 time steps (corresponding to 0.45 s).

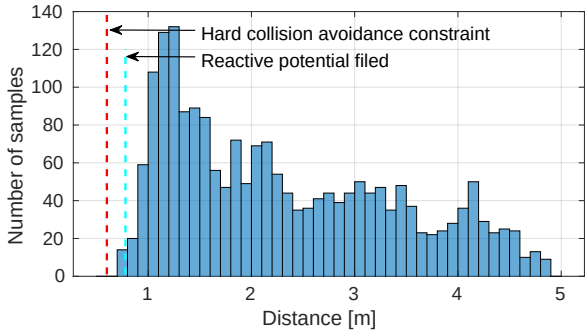
3.5.4 Trajectory safety and efficiency comparisons

In this scenario, we compare our method with a bounding volume MPC approach [123] and a deterministic MPC approach [4]. For all three methods we compute trajectories sequentially and the only difference is the way in which the uncertainties are treated. In the experiment, two quadrotors, initially at $(-1.6, 0, 1.2)$ m and $(1.6, 0, 1.2)$ m, are required to swap their positions. For each approach, we performed the experiment 50 times under three levels of measurements noise: $\frac{1}{4}\Sigma$, Σ and 4Σ. The corresponding average state estimation error for the position, i.e. $\|\hat{\mathbf{p}} - \mathbf{p}\|$, was 0.03 m, 0.05 m and 0.09 m respectively. We measured the minimum distance between the two quadrotors as a safety metric and the total trajectory length and duration as efficiency metrics.

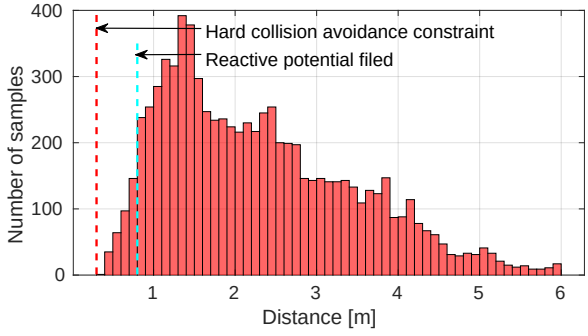
The results of the three approaches are shown in Table 3.2. Under measurements noise of Σ, the purely deterministic approach succeeded in 64% of the trials. With the larger noise level of 4Σ its performance deteriorated to a success rate of only 36%. The two probabilistic approaches succeeded in all runs. However, thanks to a tighter bound for the collision probability approximation, our method achieves the same level of safety as [123] but with more efficient collision avoidance, i.e., the trajectory length and duration are shorter. This efficiency is more apparent when the measurements noise is larger, e.g. with covariance 4Σ.

3.5.5 Comparison of multi-robot planning strategies

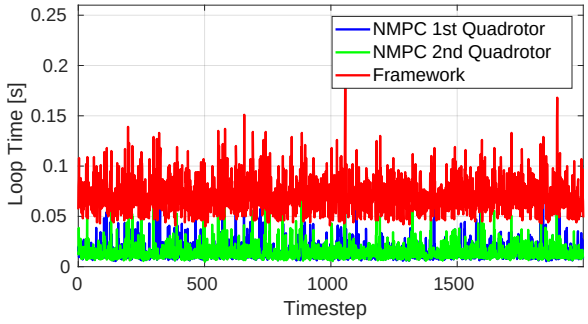
We evaluate our method in simulation with multiple quadrotors exchanging their initial positions, and compare the three multi-robot coordination strategies described in Section 3.4.2, with a noise level of Σ. Figures 3.9a-3.9c show the trajectories of six quadrotors, where



(a) Measured inter-drone distance.



(b) Distance to moving obstacles.



(c) NMPC and framework loop time.

Figure 3.8: Experimental results of two quadrotors following predefined paths while avoiding two walking humans.

the only difference is the coordination strategies. Table 3.3 shows the minimum distance among quadrotors and statistics of their trajectories. We report the average computation time and the trajectory length for all six quadrotors (minimum, maximum, mean value and standard deviation to compare cooperativeness).

Table 3.2: Trajectory safety and efficiency comparisons of planning algorithms with different levels of measurements noise. The values are computed only from successful runs. ($d_{\min}$: average minimum distance (m); l: average trajectory length (m); T: average trajectory duration (s); sr: success rate.)

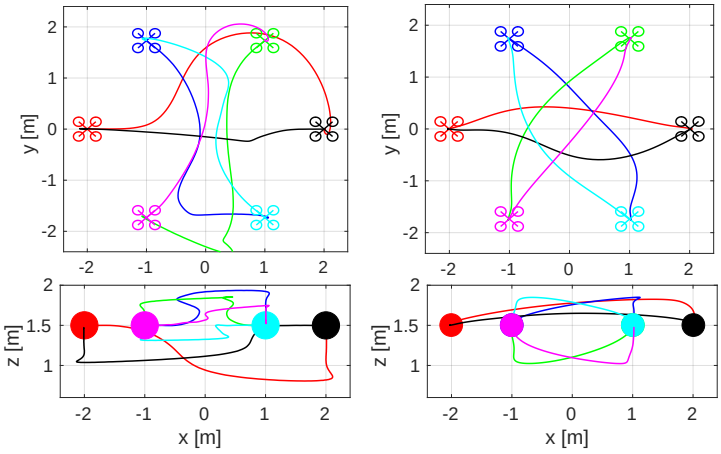
Noise level		Our method	Bounding volume[123]	Deterministic MPC[4]
$\frac{1}{4}\Sigma$	$d_{\min}$	0.74	0.74	0.63
	l	6.77	6.84	6.75
	T	2.63	2.91	2.60
	sr	100%	100%	68%
Σ	$d_{\min}$	0.81	0.87	0.64
	l	7.08	7.09	6.74
	T	2.72	2.95	2.63
	sr	100%	100%	64%
4Σ	$d_{\min}$	0.86	1.10	0.61
	l	7.21	8.18	6.88
	T	3.06	3.13	2.62
	sr	100%	100%	36%

We observe that the minimum distance when using the constant velocity model (0.56 m) is smaller than the safe distance (0.6 m). Thus, collisions happened due to the mismatch between the predicted trajectories (constant velocity) and the executed trajectories by the quadrotors. This indicates that the 97% confidence level is not enough when the constant velocity model is employed and should be increased. Instead, sequential planning (SP) and distributed planning with communication (DC) can achieve safe navigation. While SP showed better performance, it suffers from a computation burden due to its centralized scheme (the computational cost grows linearly with the number of robots). The DC approach performs well at a much lower computational cost.

Since the DC approach is scalable, in Fig. 3.9d we show the trajectories of sixteen quadrotors exchanging antipodal positions on the circle. We note that the computational time of solving the CCNMPC for each robot does increase with the number of obstacles and robots, due to the larger number of constraints. In our experiments, the average computation time of a CCNMPC planning step was 14.3 ms for two robots, 14.4 ms for four robots, 16.2 ms for six robots and 24.7 ms for sixteen robots. This indicates that the DC approach scales well with the number of robots.

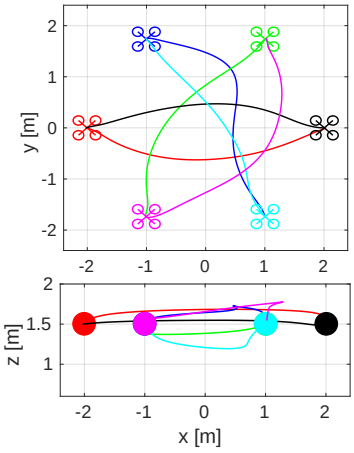
Table 3.3: Statistics for coordination strategies with six drones. CVM: constant velocity model; SP: sequential planning; DC: distributed planning with communication.

Coordination strategies	Min. dist (m)	Trajectory length (m)				Av.comp. time (ms)
		min.	max.	av.	std	
CVM	0.56	4.82	7.09	5.72	0.89	15.2
SP	0.70	4.31	4.54	4.43	0.09	115.3
DC	0.70	4.18	4.80	4.51	0.24	16.2

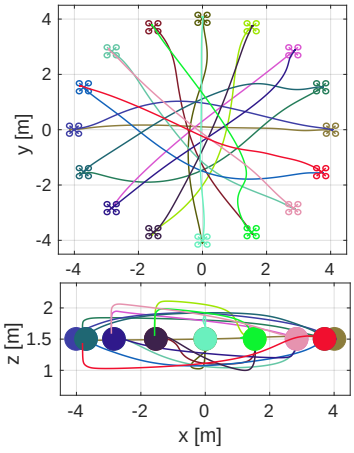


(a) Constant velocity.

(b) Sequential planning.



(c) Distributed planning.



(d) Distributed planning.

Figure 3.9: Simulation results of multiple quadrotors exchanging positions. Solid lines represent the trajectories executed by the quadrotors. The upper and lower plots show the top view (X-Y) and side view(X-Z) respectively.

3.6 Conclusion

In this chapter, we showed that robust probabilistic collision avoidance among robots and obstacles can be achieved via chance-constrained nonlinear model predictive control when the obstacles are modeled as ellipsoids. By assuming that the uncertainties are Gaussian distributed, we developed a tight bound for approximation of collision probability between each robot and obstacle. In experiments with two quadrotors, we showed that our method can generate more efficient trajectories for the robots while maintaining the same level of

safety compared with the bounding volume approach. In simulations with six quadrotors, we showed that the strategies where the planned trajectories are exchanged outperform the constant velocity model. Furthermore, while distributed planning with communication is less cooperative than sequential planning, it scales well with the number of robots.

4

Probabilistic multi-robot collision avoidance using buffered uncertainty-aware Voronoi cells

Parts of this chapter appeared in:

- H. Zhu, B. Brito, and J. Alonso-Mora, "Decentralized probabilistic multi-robot collision avoidance using buffered uncertainty-aware voronoi cells," *Autonomous Robots*, online available.
- H. Zhu, and J. Alonso-Mora, "B-uavc: Buffered uncertainty-aware voronoi cells for probabilistic multi-robot collision avoidance," in *Proceedings of the IEEE International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, Aug. 2019.

4.1 Introduction

In the previous Chapter 3, we have developed a chance-constrained collision avoidance algorithm for teams of robots via chance-constrained nonlinear model predictive control (CCNMPC) and multi-robot coordination. In the method, three coordination strategies are discussed; namely, 1) centralized sequential planning, 2) distributed planning with communication, and 3) decentralized planning without communication. The first two strategies can guarantee probabilistic collision avoidance by construction. But they require communication among robots, which is not always available in practice. While the third decentralized strategy is communication-free, experimental results show that it may lead to unsafe trajectories, particularly in crowded environments. Hence, it is necessary to develop an algorithm that is communication-free while being able to achieve probabilistic collision avoidance for multi-robot systems.

In this chapter, we present a decentralized probabilistic approach for multi-robot collision avoidance under localization and sensing uncertainty that does not rely on communication. Our approach is built on the buffered Voronoi cell (BVC) method developed by [81]. The BVC method is designed for collision avoidance among multiple single-integrator robots, where each robot only needs to know the positions of neighboring robots. We extend the method into probabilistic scenarios considering robot localization and sensing uncertainties by mathematically formalizing a buffered uncertainty-aware Voronoi cell (B-UAVC). Furthermore, we consider static obstacles with uncertain locations in the environment and apply the approach to double-integrator dynamics, differential-drive robots, and general high-order dynamical robots.

The BVC method has also been extended to probabilistic scenarios by [114]. Taking into account the robot measurement uncertainty of other robots, they present the probabilistic buffered Voronoi cell (PBVC) to assure a safety level given a collision probability threshold. However, since the PBVC of each robot does not have an analytic solution, they employ a sampling-based approach to approximate it. In contrast, our proposed B-UAVC has an explicit and analytical form, which is more efficient to be computed. Moreover, our B-UAVC can be incorporated with MPC to handle general nonlinear systems, while the PBVC method developed by [114] cannot be directly applied within the MPC framework.

Our method constructs a set of local safe regions for the robots, which decompose the workspace. Spatial decomposition is broadly used in robot motion planning. The authors of [128] proposed the IRIS (iterative regional inflation by semi-definite programming) algorithm to compute safe convex regions among obstacles given a set of seed points. The algorithm is then used for UAV path planning [129] and multi-robot formation control [125]. A simpler but more efficient iteratively inflation algorithm [130] was later presented to compute a convex polytope around a line segment among obstacles and utilizes it to construct a safe flight corridor for UAV navigation [131]. Similar safe flight corridors are constructed for trajectory planning of quadrotor swarms [105], by computing a set of max-margin separating hyperplanes between a line segment and convex polygonal obstacles. The max-margin separating hyperplanes are also used by [132] to construct a local robot-centric safe region in convex sphere worlds for sensor-based reactive navigation.

While those spatial decomposition methods have shown successful application in robot motion planning, they all assume perfect knowledge on robots and obstacles positions. In this chapter, we consider both the robot localization and obstacle position uncertainty and construct a local uncertainty-aware safe region for each robot.

The main contribution of this chapter is a decentralized and communication-free method for probabilistic multi-robot collision avoidance in cluttered environments. The method considers robot localization and sensing uncertainties and relies on the computation of buffered uncertainty-aware Voronoi cells (B-UAVC). At each time step, each robot computes its B-UAVC based on the estimated position and uncertainty covariance of itself, neighboring robots and obstacles, and plans its motion within the B-UAVC. Probabilistic collision avoidance is ensured by constraining each robot's motion to be within its corresponding B-UAVC, such that the inter-robot and robot-obstacle collision probability is below a user-specified threshold.

The remaining of this chapter is organized as follows. In Section 4.2 we present the problem statement and briefly summarize the concept of BVC. In Section 4.3 we formally introduce the buffered uncertainty-aware Voronoi cell (B-UAVC) and its construction method. We then describe how the B-UAVC is used for probabilistic multi-robot collision avoidance in Section 4.4. Simulation and experimental results are presented in Section 4.5 and Section 4.6, respectively. Finally, Section 4.7 concludes the chapter.

4.2 Preliminaries

4.2.1 Problem statement

Consider a group of n robots operating in a d-dimensional space $\mathcal{W} \subseteq \mathbb{R}^d$, where $d \in \{2,3\}$, populated with m static polygonal obstacles. For each robot $i \in \mathcal{I} = \{1,\dots,n\}$, $\mathbf{p}_i \in \mathbb{R}^d$ denotes its position, $\mathbf{v}_i = \dot{\mathbf{p}}_i$ its velocity and $\mathbf{a}_i = \dot{\mathbf{v}}_i$ its acceleration. Let $\mathcal{G} = \{\mathbf{g}_1,\dots,\mathbf{g}_n\}$ denote their goal locations. A safety radius r_s is given for all robots. We consider that the position of each robot is obtained by a state estimator and is described as a Gaussian distribution with covariance Σ_i, i.e. $\mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$. We also consider static polytope obstacles with known shapes but uncertain locations. For each obstacle $o \in \mathcal{I}_o = \{1,\dots,m\}$, denote by $\hat{\mathcal{O}}_o \subset \mathbb{R}^d$ its occupied space when located at the expected (mean) position. $\hat{\mathcal{O}}_o$ is given by a set of vertices. Hence, the space actually occupied by the obstacle can be written as $\mathcal{O}_o = \{\mathbf{x} + \mathbf{d}_o \mid \mathbf{x} \in \hat{\mathcal{O}}_o, \mathbf{d}_o \sim \mathcal{N}(0,\Sigma_o)\} \subset \mathbb{R}^d$, where $\mathbf{d}_o$ is the uncertain translation of the obstacle's position, which has a zero mean and covariance Σ_o.

A robot i in the group is collision-free with another robot j if their distance is greater than the sum of their radii, i.e. $\mathrm{dis}(\mathbf{p}_i, \mathbf{p}_j) \geq 2r_s$ and with the obstacle o if the minimum distance between the robot and the obstacles is larger than its radius, i.e. $\mathrm{dis}(\mathbf{p}_i, \mathcal{O}_o) \geq r_s$. The distance function $\mathrm{dis}(\cdot)$ between a robot with another robot or an obstacle are defined as $\mathrm{dis}(\mathbf{p}_i, \mathbf{p}_j) = \|\mathbf{p}_i - \mathbf{p}_j\|$, and $\mathrm{dis}(\mathbf{p}_i, \mathcal{O}_o) = \min_{\mathbf{p} \in \mathcal{O}_o} \|\mathbf{p}_i - \mathbf{p}\|$, respectively. Note that the robots' and obstacles' positions are random variables following Gaussian distributions, which have an infinite support. Hence, the collision-free condition can only be satisfied in a probabilistic manner, which is defined as a chance constraint as follows.

***Definition* 4.1** (Probabilistic collision-free). *A robot i at position $\mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$ is probabilistic collision-free with a robot j at position $\mathbf{p}_j \sim \mathcal{N}(\hat{\mathbf{p}}_j, \Sigma_j)$ and an obstacle o at position $\mathbf{p}_o \sim \mathcal{N}(\hat{\mathbf{p}}_o, \Sigma_o)$ if*

$$\Pr(\text{dis}(\mathbf{p}_i, \mathbf{p}_j) \geq 2r_s) \geq 1 - \delta, \quad \forall j \in \mathcal{I}, j \neq i, \tag{4.1}$$

$$\Pr(\text{dis}(\mathbf{p}_i, \mathcal{O}_o) \geq r_s) \geq 1 - \delta, \quad \forall o \in \mathcal{I}_o, \tag{4.2}$$

where δ is the collision probability threshold for inter-robot and robot-obstacle collisions.

The objective of probabilistic collision avoidance is to compute a local motion plan, $\mathbf{u}_i$, for each robot in the group, that respects its kinematic and dynamical constraints, makes progress towards its goal location, and is probabilistic collision-free with other robots as well as obstacles in the environment. In this chapter, we first consider single-integrator dynamics for the robots,

$$\dot{\mathbf{p}}_i = \mathbf{u}_i, \tag{4.3}$$

and then extend it to double integrator systems, differential-drive robots and robots with general high-order dynamics.

4.2.2 Buffered Voronoi cell

The key idea of our proposed method is to compute an uncertainty-aware collision-free region for each robot in the system, which is a major extension of the deterministic buffered Voronoi cell (BVC) method [81, 133]. In this section, we briefly describe the concept of BVC.

For a set of deterministic points $(\mathbf{p}_1, \dots, \mathbf{p}_n) \in \mathbb{R}^d$, the standard Voronoi cell (VC) of each point $i \in \mathcal{I}$ is defined as [134]

$$\mathcal{V}_i = \{\mathbf{p} \in \mathbb{R}^d \,:\, \|\mathbf{p} - \mathbf{p}_i\| \leq \|\mathbf{p} - \mathbf{p}_j\|, \forall j \neq i\}, \tag{4.4}$$

which can also be written as

$$\mathcal{V}_i = \{\mathbf{p} \in \mathbb{R}^d \,:\, \mathbf{p}_{ij}^T \mathbf{p} \leq \mathbf{p}_{ij}^T \frac{\mathbf{p}_i + \mathbf{p}_j}{2}, \forall j \neq i\}, \tag{4.5}$$

where $\mathbf{p}_{ij} = \mathbf{p}_j - \mathbf{p}_i$. It can be observed that $\mathcal{V}_i$ is the intersection of a set of hyperplanes which separate point i with any other point j in the group, as shown in Fig. 4.1a. Hence, VC can be obtained by computing the separating hyperplanes between each pair of points.

To consider the footprints of robots, a buffered Voronoi cell for each robot i is defined as follows:

$$\mathcal{V}_i^b = \{\mathbf{p} \in \mathbb{R}^d \,:\, \mathbf{p}_{ij}^T \mathbf{p} \leq \mathbf{p}_{ij}^T \frac{\mathbf{p}_i + \mathbf{p}_j}{2} - r_s \|\mathbf{p}_{ij}\|, \forall j \neq i\}, \tag{4.6}$$

which is obtained by retracting the edges of the VC with a safety distance (buffer) r_s.

In deterministic scenarios, if the robots are mutually collision-free, then the BVC of each robot is a non-empty set [81]. It is also trivial to prove that the BVCs are disjoint and if the robots are within their corresponding BVCs individually, they are collision-free

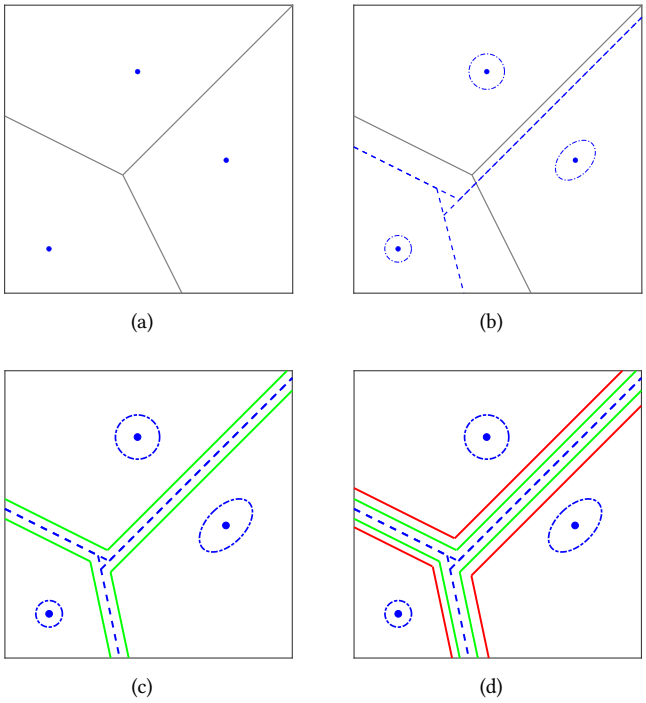


Figure 4.1: Example of buffered uncertainty-aware Voronoi cells (B-UAVC). Blue dots are robots; blue dash-dot ellipses indicate the 3-σ confidence ellipsoid of the position uncertainty. (a) Deterministic Voronoi cell (VC, the boundary in gray solid line). (b) Uncertainty-aware Voronoi cell based on the best linear separators (UAVC, the boundary in blue dashed line). (c) UAVC with robot raidus buffer (the boundary in green solid line). (d) Final B-UAVC with robot radius and collision probability buffer (the boundary in red solid line).

with each other. Using the concept of BVC, [81] proposed a control policy for a group of single-integrator robots whose control inputs are velocities. Each robot can safely and continuously navigate in its BVC, given that other robots in the system also follow the same rule. However, the guarantee does not hold for double-integrator dynamics or non-holonomic robots such as differential-drive robots.

4.2.3 Shadows of uncertain obstacles

To account for uncertain obstacles in the environment, we rely on the concept of obstacle shadows introduced by [68]. The ϵ-shadow is defined as follows:

***Definition* 4.2** (ϵ-Shadow)**.** *A set $\mathcal{S}_o \subseteq \mathbb{R}^d$ is an ϵ-shadow of an uncertain obstacle $\mathcal{O}_o$ if the probability* $\Pr(\mathcal{O}_o \subseteq \mathcal{S}_o) \geq 1 - \epsilon$.

Geometrically, an ϵ-shadow is a region that contains the uncertain obstacle with probability of at least $1 - \epsilon$, which can be non-unique. For example $\mathcal{S}_o = \mathbb{R}^d$ is an ϵ-shadow

of any uncertain obstacle. To preclude this trivial case, the maximal ϵ-shadow is defined:

***Definition* 4.3** (Maximal ϵ-shadow)**.** *A set $\mathcal{S}_o \subseteq \mathbb{R}^d$ is a maximal ϵ-shadow of an uncertain obstacle $\mathcal{O}_o$ if the probability* $\Pr(\mathcal{O}_o \subseteq \mathcal{S}_o) = 1 - \epsilon$.

The above definition ensures that if there exists a maximal ϵ-shadow $\mathcal{S}_o$ of the uncertain obstacle $\mathcal{O}_o$ that does not intersect the robot, i.e. $\mathrm{dis}(\mathbf{p}_i, \mathcal{S}_o) \geq r_s$, then the collision probability between the robot and obstacle is below ϵ, i.e. $\Pr(\mathrm{dis}(\mathbf{p}_i, \mathcal{O}_o) \geq r_s) \geq 1 - \epsilon$. Note that the maximal ϵ-shadow may also be non-unique. In this chapter, we employ the method proposed by [69] to construct such shadows. Recall that the uncertain obstacle $\mathcal{O}_o$ is related to the nominal geometry $\hat{\mathcal{O}}_o$ by $\mathcal{O}_o = \{\mathbf{x} + \mathbf{d}_o \mid \mathbf{x} \in \hat{\mathcal{O}}_o, \mathbf{d}_o \sim \mathcal{N}(0, \Sigma_o)\}$. To construct the maximal ϵ-shadow, we first define the following ellipsoidal set

$$\mathcal{D}_o = \{\mathbf{d} : \mathbf{d}^T \Sigma_o^{-1} \mathbf{d} \leq F^{-1}(1 - \epsilon)\}, \tag{4.7}$$

where $F^{-1}(\cdot)$ is the inverse of the cumulative distribution function (CDF) of the chi-squared distribution with d degrees of freedom. Next, Let

$$\mathcal{S}_o = \hat{\mathcal{O}}_o + \mathcal{D}_o = \{\mathbf{x} + \mathbf{d} \mid \mathbf{x} \in \hat{\mathcal{O}}_o, \mathbf{d} \in \mathcal{D}_o\}, \tag{4.8}$$

be the Minkowski sum of the nominal obstacle shape $\hat{\mathcal{O}}_o$ and the ellipsoidal set $\mathcal{D}_o$. Then, we have the following lemma [68] and theorem [69]:

***Lemma* 4.1.** *Let $\mathbf{d}_o \sim \mathcal{N}(0, \Sigma_o) \in \mathbb{R}^d$ and $\mathcal{D}_o = \{\mathbf{d} : \mathbf{d}^T \Sigma_o^{-1} \mathbf{d} \leq F^{-1}(1 - \epsilon)\} \subset \mathbb{R}^d$, then* $\Pr(\mathbf{d}_o \in \mathcal{D}_o) = 1 - \epsilon$.

Proof. First we can write the random variable $\mathbf{d}_o$ in an equivalent form $\mathbf{d}_o = \Sigma'_o \mathbf{d}'_o$, where $\mathbf{d}'_o \sim \mathcal{N}(0, I) \in \mathbb{R}^d$ and $\Sigma'_o \Sigma'^T_o = \Sigma_o$. Note that $\mathbf{d}'^T_o \mathbf{d}'_o$ is a chi-squared random variable with d degrees of freedom. Hence, there is

$$\Pr(\mathbf{d}'^T_o \mathbf{d}'_o \leq F^{-1}(1 - \epsilon)) = 1 - \epsilon.$$

Also note that $\Sigma_o^{-1} = (\Sigma'_o \Sigma'^T_o)^{-1} = \Sigma'^{T^{-1}}_o \Sigma'^{-1}_o$, thus $\mathbf{d}_o^T \Sigma_o^{-1} \mathbf{d}_o = \mathbf{d}'^T_o \Sigma'^T_o \Sigma'^{T^{-1}}_o \Sigma'^{-1}_o \Sigma'_o \mathbf{d}'_o = \mathbf{d}'^T_o \mathbf{d}'_o$. Hence, it follows that $\Pr(\mathbf{d}_o^T \Sigma_o^{-1} \mathbf{d}_o \leq F^{-1}(1 - \epsilon)) = 1 - \epsilon$. Thus, let $\mathcal{D}_o = \{\mathbf{d} : \mathbf{d}^T \Sigma_o^{-1} \mathbf{d} \leq F^{-1}(1 - \epsilon)\}$, there is $\Pr(\mathbf{d}_o \in \mathcal{D}_o) = 1 - \epsilon$. □

***Theorem* 4.1.** *$\mathcal{S}_o$ is a maximal ϵ-shadow of $\mathcal{O}_o$.*

Proof. We need to prove that the set $\mathcal{S}_o$ contains the set $\mathcal{O}_o$ with probability $1 - \epsilon$. It is equivalent to that for any point in $\mathcal{O}_o$, the set $\mathcal{S}_o$ contains this point with probability $1 - \epsilon$. Recall the definition of $\mathcal{O}_o$, every $\mathbf{y} \in \mathcal{O}_o$ can be written as $\mathbf{x} + \mathbf{d}_o$ with some $\mathbf{x} \in \hat{\mathcal{O}}_o$. Also note the definition $\mathcal{S}_o = \{\mathbf{x} + \mathbf{d} \mid \mathbf{x} \in \hat{\mathcal{O}}_o, \mathbf{d} \in \mathcal{D}_o\}$. Hence the probability that $\mathcal{S}_o$ contains $\mathbf{y}$ is equal to the probability that $\mathcal{D}_o$ contains $\mathbf{d}_o$. That is, $\Pr(\mathbf{y} \in \mathcal{S}_o) = \Pr(\mathbf{d}_o \in \mathcal{D}_o) = 1 - \epsilon, \forall \mathbf{y} \in \mathcal{O}_o$. Thus, $\Pr(\mathcal{O}_o \subseteq \mathcal{S}_o) = 1 - \epsilon$. $\mathcal{S}_o$ is a maximal ϵ-shadow of $\mathcal{O}_o$. □

4.3 Buffered uncertainty-aware Voronoi cells

In this section, we formally introduce the concept of buffered uncertainty-aware Voronoi cells (B-UAVC) and give its construction method.

4.3.1 Definition of B-UAVC

Our objective is to obtain a probabilistic safe region for each robot in the workspace given the robots and obstacles positions, and taking into account their uncertainties.

***Definition* 4.4** (Buffered uncertainty-aware Voronoi cell)**.** *Given a team of robots $i \in \{1,\dots,n\}$ with positions mean $\hat{\mathbf{p}}_i \in \mathbb{R}^d$ and covariance $\Sigma_i \in \mathbb{R}^{d\times d}$, and a set of convex polytope obstacles $o \in \{1,\dots,m\}$ with known shapes and locations mean $\hat{\mathbf{p}}_o \in \mathbb{R}^d$ and covariance $\Sigma_o \in \mathbb{R}^{d\times d}$, the buffered uncertainty-aware Voronoi cell (B-UAVC) of each robot is defined as a convex polytope region:*

$$\mathcal{V}_i^{u,b} = \{\mathbf{p} \in \mathbb{R}^d \,:\, \mathbf{a}_{ij}^T\mathbf{p} \le b_{ij} - \beta_{ij}, \forall j \neq i, j \in \mathcal{I}, \tag{4.9}$$

$$\text{and } \mathbf{a}_{io}^T\mathbf{p} \le b_{io} - \beta_{io}, \forall o \in \mathcal{I}_o\}, \tag{4.10}$$

such that the probabilistic collision free constraints in Definition 4.1 are satisfied.

In the above B-UAVC definition, $\mathbf{a}_{ij}, \mathbf{a}_{io} \in \mathbb{R}^d$ and $b_{ij}, b_{io} \in \mathbb{R}$ are parameters of the hyperplanes that separate the robot from other robots and obstacles, which results in a decomposition of the workspace. β_{ij} and β_{io} are additional buffer terms added to retract the decomposed space for probabilistic collision avoidance. Accordingly, we further define

$$\mathcal{V}_i^{u} = \{\mathbf{p} \in \mathbb{R}^d \,:\, \mathbf{a}_{ij}^T\mathbf{p} \le b_{ij}, \forall j \neq i, j \in \mathcal{I}, \text{ and } \mathbf{a}_{io}^T\mathbf{p} \le b_{io}, \forall o \in \mathcal{I}_o\}, \tag{4.11}$$

that does not include buffer terms to be the uncertainty-aware Voronoi cell (UAVC) of robot i.

It can be observed the UAVC and B-UAVC of robot i are the intersection of the following:

1. $n-1$ half-space hyperplanes separating robot i from robot j for all $j \neq i, j \in \mathcal{I}$;
2. m half-space hyperplanes separating robot i from obstacle o for all $o \in \mathcal{I}_o$.

In the following, we will describe how to calculate the separating hyperplanes with parameters $(\mathbf{a}_{ij}, b_{ij})$ and $(\mathbf{a}_{io}, b_{io})$ that construct the UAVC and then the corresponding buffer terms β_{ij}, β_{io} constructing the B-UAVC for probabilistic collision avoidance.

4.3.2 Inter-robot separating hyperplane

In contrast to only separating two deterministic points in Voronoi cells, we separate two uncertain robots with known positions mean and covariance. To achieve that, we rely on the concept of the best linear separator between two Gaussian distributions [135].

Given $\mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$ and $\mathbf{p}_j \sim \mathcal{N}(\hat{\mathbf{p}}_j, \Sigma_j)$, consider a linear separator $\mathbf{a}_{ij}^T\mathbf{p} = b_{ij}$ where $\mathbf{a}_{ij} \in \mathbb{R}^d$ and $b_{ij} \in \mathbb{R}$. The separator classifies the points $\mathbf{p}$ in the space into two clusters: $\mathbf{a}_{ij}^T\mathbf{p} \le b_{ij}$ to the first one while $\mathbf{a}_{ij}^T\mathbf{p} > b_{ij}$ to the second. The separator parameters $\mathbf{a}_{ij}$ and b_{ij} can be obtained by minimizing the maximal probability of misclassification.

The misclassification probability when $\mathbf{p}$ is from the first distribution is

$$\Pr_i(\mathbf{a}_{ij}^T\mathbf{p} > b_{ij}) = \Pr_i\left(\frac{\mathbf{a}_{ij}^T\mathbf{p} - \mathbf{a}_{ij}^T\hat{\mathbf{p}}_i}{\sqrt{\mathbf{a}_{ij}^T\Sigma_i\mathbf{a}_{ij}}} > \frac{b_{ij} - \mathbf{a}_{ij}^T\hat{\mathbf{p}}_i}{\sqrt{\mathbf{a}_{ij}^T\Sigma_i\mathbf{a}_{ij}}}\right) = 1 - \Phi((b_{ij} - \mathbf{a}_{ij}^T\hat{\mathbf{p}}_i)/\sqrt{\mathbf{a}_{ij}^T\Sigma_i\mathbf{a}_{ij}}),$$

where $\Phi(\cdot)$ denotes the cumulative distribution function (CDF) of the standard normal distribution. Similarly, the misclassification probability when $\mathbf{p}$ is from the second distribution is

$$\Pr_j(\mathbf{a}_{ij}^T\mathbf{p} \leq b_{ij}) = \Pr_j\left(\frac{\mathbf{a}_{ij}^T\mathbf{p} - \mathbf{a}_{ij}^T\hat{\mathbf{p}}_j}{\sqrt{\mathbf{a}_{ij}^T\Sigma_j\mathbf{a}_{ij}}} \leq \frac{b_{ij} - \mathbf{a}_{ij}^T\hat{\mathbf{p}}_j}{\sqrt{\mathbf{a}_{ij}^T\Sigma_j\mathbf{a}_{ij}}}\right) = 1 - \Phi((\mathbf{a}_{ij}^T\hat{\mathbf{p}}_j - b_{ij})/\sqrt{\mathbf{a}_{ij}^T\Sigma_j\mathbf{a}_{ij}}).$$

The objective is to minimize the maximal value of $\Pr_i$ and $\Pr_j$, i.e.

$$(\mathbf{a}_{ij}, b_{ij}) = \arg\min_{\mathbf{a}_{ij}\in\mathbb{R}^d, b_{ij}\in\mathbb{R}} \max(\Pr_i, \Pr_j), \tag{4.12}$$

which can be solved using a fast minimax procedure. In this chapter, we employ the procedure developed by [135] to compute the best linear separator parameters $\mathbf{a}_{ij}$ and b_{ij}. A brief summary of the procedure is presented in Appendix B.

***Remark* 4.1.** *The best linear separator coincides with the separating hyperplane of Eq. (4.5) when $\Sigma_i = \Sigma_j = \sigma^2 I$. In this case, $\mathbf{a}_{ij} = \frac{2}{\sigma^2}(\hat{\mathbf{p}}_j - \hat{\mathbf{p}}_i)$ and $b_{ij} = \frac{1}{\sigma^2}(\hat{\mathbf{p}}_j - \hat{\mathbf{p}}_i)^T(\hat{\mathbf{p}}_i + \hat{\mathbf{p}}_j)$.*

***Remark* 4.2.** *$\forall i \neq j \in \mathcal{I}, \mathbf{a}_{ji} = -\mathbf{a}_{ij}, b_{ji} = -b_{ij}$. This can be obtained according to the definition of the best linear separator.*

***Remark* 4.3.** *In contrast to deterministic Voronoi cells, the UAVCs constructed from the best linear separators generally do not constitute a full tessellation of the workspace, i.e. $\bigcup_1^n \mathcal{V}_i^u \subseteq \mathcal{W}$, as shown in Fig. 4.1b.*

4.3.3 Robot-obstacle separating hyperplane

Our method to calculate the uncertainty-aware separating hyperplane between a robot and a convex polytope obstacle with uncertain location is illustrated in Fig. 4.2. Given the mean position of the robot $\hat{\mathbf{p}}_i$ and the uncertain obstacle $\mathcal{O}_o = \{\mathbf{x} + \mathbf{d}_o \mid \mathbf{x} \in \hat{\mathcal{O}}_o, \mathbf{d}_o \sim \mathcal{N}(0, \Sigma_o)\}$, we first perform a linear coordinate transformation:

$$W = (\sqrt{\Sigma_o})^{-1}. \tag{4.13}$$

Under the transformation, the robot mean position and obstacle information become

$$\hat{\mathbf{p}}_i^W = W\hat{\mathbf{p}}_i, \tag{4.14}$$

$$\hat{\mathcal{O}}_o^W = W\hat{\mathcal{O}}_o, \tag{4.15}$$

$$\mathbf{d}_o^W = W\mathbf{d}_o, \tag{4.16}$$

$$\Sigma_o^W = W\Sigma_o W^T = I^{d\times d}. \tag{4.17}$$

The transformed uncertain obstacle is then $\mathcal{O}_o^W = \{\mathbf{x}^W + \mathbf{d}_o^W \mid \mathbf{x}^W \in \hat{\mathcal{O}}_o^W, \mathbf{d}_o^W \sim \mathcal{N}(0, I)\}$. Here we use the super-script $\cdot^W$ to indicate variables in the transformed space. Note that the obstacle position uncertainty covariance is normalized to an identity matrix under the transformation, as shown in Fig. 4.2 (Top right). This coordinate transformation technique to normalize the uncertainty covariance has also been applied to other motion planning under uncertainty works [31].

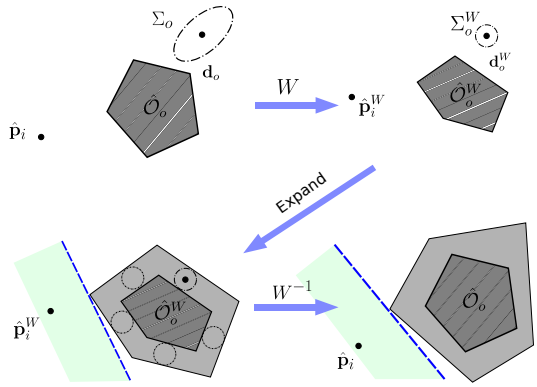


Figure 4.2: Depiction of uncertainty-aware separating hyperplane calculation between a point and an arbitrary polytope obstacle with uncertain location. (Top left) A point and a polytope obstacle with uncertain location. (Top right) Effects of the transformation W to normalize the error covariance. (Bottom left) ϵ-shadow of the transformed obstacle and the max-margin separating hyperplane in the transformation space. (Bottom right) Inverse transformation to obtain the uncertainty-aware separating hyperplane.

Then given the collision probability threshold δ, we compute a ϵ-shadow of the transformed uncertain obstacle $\mathcal{O}_o^W$ based on Eqs. (4.7)-(4.8):

$$\mathcal{D}_o^W = \{\mathbf{d}^W : \mathbf{d}^{W\,T}\mathbf{d}^W \leq F^{-1}(1-\epsilon)\}, \tag{4.18}$$

$$\mathcal{S}_o^W = \hat{\mathcal{O}}_o^W + \mathcal{D}_o^W, \tag{4.19}$$

where $\epsilon = 1 - \sqrt{1-\delta}$, making that $\Pr(\mathcal{O}_o^W \subseteq \mathcal{S}_o^W) = \sqrt{1-\delta}$.

Note that we assume $\hat{\mathcal{O}}_o$ is a convex polytope. Hence, the transformed $\hat{\mathcal{O}}_o^W$ is also a polytope. In addition, it can be observed the set $\mathcal{D}_o^W$ defined in Eq. (4.18) is a circular (sphere in 3D) set with radius $\sqrt{F^{-1}(1-\epsilon)}$. Hence, we can compute the ϵ-shadow in Eq. (4.19) of the transformed uncertain obstacle by dilating its nominal shape by the diameter of the set $\mathcal{D}_o^W$, which results in an inflated convex polytope. Note that the resulted convex polytope is slightly larger than the exact Minkowski sum $\mathcal{S}_o^W$ which has smaller round corners. This introduces some conservativeness. For simplicity, we use the same notation $\mathcal{S}_o^W$ for the resulted inflated convex polytope and thus there is $\Pr(\mathcal{O}_o^W \subseteq \mathcal{S}_o^W) > \sqrt{1-\delta}$.

Next, we separate $\hat{\mathbf{p}}_i^W$ from $\mathcal{S}_o^W$ by finding a max-margin separating hyperplane between them. Note that $\mathcal{S}_o^W$ is a bounded convex polytope that can be described by a list of

vertices $(\psi_1^W, \dots, \psi_{p_o}^W)$. Hence, finding a max-margin hyperplane between $\hat{\mathbf{p}}_i^W$ and $\mathcal{S}_o^W$ can be formulated as a support vector machine (SVM) problem [105], which can be efficiently solved using a quadratic program:

$$\begin{aligned} \min \quad & \mathbf{a}_{io}^{W\,T} \mathbf{a}_{io}^W \\ \text{s.t.} \quad & \mathbf{a}_{io}^{W\,T} \hat{\mathbf{p}}_{io}^W - b_{io}^W \leq 1, \\ & \mathbf{a}_{io}^{W\,T} \psi_k^W - b_{io}^W \geq 1, \ \ \forall\, k \in 1, \dots, p_o. \end{aligned} \tag{4.20}$$

The solution of the above quadratic program (4.20) formulates a max-margin separating hyperplane with parameters $(\mathbf{a}_{io}^W, b_{io}^W)$. We then shift it along its normal vector towards the obstacle shadow, resulting in a separating hyperplane exactly touching the shadow, as shown in Fig. 4.2 (Bottom left). Finally we perform an inverse coordinate transformation W^{-1} and obtain the uncertainty-aware separating hyperplane between the robot and obstacle in the original workspace:

$$\begin{aligned} \mathbf{a}_{io} &= W^T \mathbf{a}_{io}^W, \\ b_{io} &= b_{io}^W, \end{aligned} \tag{4.21}$$

as shown in Fig. 4.2 (Bottom right), in which the ϵ-shadow in the transformed space $\mathcal{S}_o^W$ becomes $\mathcal{S}_o$ in the original space.

***Remark* 4.4.** *The linear coordinate transformation W and its inverse W^{-1} preserves relative geometries of $\mathcal{O}_o$. That is,* $\Pr(\mathcal{O}_o \subseteq \mathcal{S}_o) = \Pr(\mathcal{O}_o^W \subseteq \mathcal{S}_o^W) > \sqrt{1-\delta}$.

4.3.4 Collision avoidance buffer and B-UAVC

In Section 4.3.2 and 4.3.3 we have described the method to compute the hyperplanes that construct the UAVC. Now we introduce two buffer terms to the UAVC, to account for the robot physical safety radius and the collision probability threshold.

Recall Eq. (4.11) that the UAVC of robot i can be written as the intersection of a set of separating hyperplanes

$$\begin{aligned} \mathcal{V}_i^u = \{\mathbf{p} \in \mathbb{R}^d \,:\, & \mathbf{a}_{ij}^T \mathbf{p} \leq b_{ij}, \forall j \neq i, j \in \mathcal{I}, \\ & \text{and } \ \mathbf{a}_{io}^T \mathbf{p} \leq b_{io}, \forall o \in \mathcal{I}_o\}. \end{aligned}$$

Let $l \in \mathcal{I}_l = \{1, \cdots, n, n+1, \cdots, n+m\}, l \neq i$ denote any other robot or obstacle, we can write the UAVC in the following form

$$\mathcal{V}_i^u = \{\mathbf{p} \in \mathbb{R}^d \,:\, \mathbf{a}_{il}^T \mathbf{p} \leq b_{il}, \forall l \in \mathcal{I}_l, l \neq i\}, \tag{4.22}$$

which combines the notations for inter-robot and robot-obstacle separating hyperplanes. Next, we will describe the computation method of probabilistic collision avoidance buffer to extend the UAVC to B-UAVC.

Robot safety radius buffer

We compute the robot safety radius buffer by shifting the boundary of the UAVC towards the robot by a distance equal to the robot's radius. Hence the corresponding buffer for the hyperplane $(\mathbf{a}_{il}, b_{il})$ is

$$\beta_i^r = r_s \|\mathbf{a}_{il}\|. \tag{4.23}$$

Figure 4.1c shows the buffered UAVC of each robot after taking into account their safety radius.

Collision probability buffer

To achieve probabilistic collision avoidance, we further compute a buffer term β_i^δ, which is defined as

$$\beta_i^\delta = \sqrt{2\mathbf{a}_{il}^T \Sigma_i \mathbf{a}_{il}} \cdot \text{erf}^{-1}(2\sqrt{1-\delta} - 1), \tag{4.24}$$

where $\text{erf}(\cdot)$ is the Gauss error function [136] defined as $\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$ and $\text{erf}^{-1}(\cdot)$ is its inverse. In this chapter, we assume the threshold satisfies $0 < \delta < 0.75$, which is reasonable in practice. Hence, $\text{erf}^{-1}(2\sqrt{1-\delta} - 1) > 0, \beta_i^\delta > 0$. This buffer can be obtained by following the proof of forthcoming Theorem 4.2 and Theorem 4.3.

Finally, the buffered uncertainty-aware Voronoi cell (B-UAVC) is obtained by combining the two buffers

$$\mathcal{V}_i^{u,b} = \{\mathbf{p} \in \mathbb{R}^d : \mathbf{a}_{il}^T \mathbf{p} \le b_{il} - \beta_i^r - \beta_i^\delta, \forall l \in \mathcal{I}_l, l \ne i\}. \tag{4.25}$$

Figure 4.1d shows the final B-UAVC of each robot in the team.

4.3.5 Properties of B-UAVC

In this subsection, we justify the design of ϵ in Eq. (4.18) when computing the shadow of uncertain obstacles, and computation of the collision probability buffer β_i^δ in Eq. (4.24) by presenting the following two theorems.

***Theorem* 4.2** (Inter-robot probabilistic collision free)**.** *$\forall \mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$ and $\mathbf{p}_j \sim \mathcal{N}(\hat{\mathbf{p}}_j, \Sigma_j)$, where $\hat{\mathbf{p}}_i \in \mathcal{V}_i^{u,b}$ and $\hat{\mathbf{p}}_j \in \mathcal{V}_j^{u,b}$, $i \ne j \in \mathcal{I}$, we have*

$$\Pr(\text{dis}(\mathbf{p}_i, \mathbf{p}_j) \ge 2r_s) \ge 1 - \delta,$$

i.e. the probability of collision between robots i and j is below the threshold δ.

Proof. We first introduce the following lemma:
***Lemma* 4.2** (Linear chance constraint [53])**.** *A multivariate random variable $\mathbf{x} \sim \mathcal{N}(\hat{\mathbf{x}}, \Sigma)$ satisfies*

$$\Pr(\mathbf{a}^T \mathbf{x} \le b) = \frac{1}{2} + \frac{1}{2}\text{erf}\left(\frac{b - \mathbf{a}^T \hat{\mathbf{x}}}{\sqrt{2\mathbf{a}^T \Sigma \mathbf{a}}}\right). \tag{4.26}$$

According to Eq. (4.25), if $\hat{\mathbf{p}}_i \in \mathcal{V}_i^{u,b}$, there is

$$\mathbf{a}_{ij}^T \hat{\mathbf{p}}_i \le b_{ij} - r_s \|\mathbf{a}_{ij}\| - \sqrt{2\mathbf{a}_{ij}^T \Sigma_i \mathbf{a}_{ij}} \cdot \text{erf}^{-1}(2\sqrt{1-\delta} - 1). \tag{4.27}$$

Applying Lemma 4.2 and substituting the above equation, we have

$$\begin{aligned}
&\Pr(\mathbf{a}_{ij}\mathbf{p}_i \le b_{ij} - r_s\|\mathbf{a}_{ij}\|) \\
&= \frac{1}{2} + \frac{1}{2}\mathrm{erf}\left(\frac{b_{ij} - r_s\|\mathbf{a}_{ij}\|) - \mathbf{a}_{ij}^T\hat{\mathbf{p}}_i}{\sqrt{2\mathbf{a}_{ij}^T\mathbf{a}_{ij}}}\right) \\
&\ge \frac{1}{2} + \frac{1}{2}\mathrm{erf}\left(\mathrm{erf}^{-1}(2\sqrt{1-\delta} - 1)\right) \\
&= \frac{1}{2} + \frac{1}{2}(2\sqrt{1-\delta} - 1) \\
&= \sqrt{1-\delta}.
\end{aligned} \tag{4.28}$$

Similarly for robot j, there is

$$\Pr(\mathbf{a}_{ji}\mathbf{p}_j \le b_{ji} - r_s\|\mathbf{a}_{ji}\|) \ge \sqrt{1-\delta}. \tag{4.29}$$

Note that $\mathbf{a}_{ij} = -\mathbf{a}_{ji}$, $b_{ij} = -b_{ji}$ (Remark 4.2). It is trivial to prove that

$$\left.\begin{aligned}\mathbf{a}_{ij}\mathbf{p}_i \le b_{ij} - r_s\|\mathbf{a}_{ij}\| \\ \mathbf{a}_{ji}\mathbf{p}_j \le b_{ji} - r_s\|\mathbf{a}_{ji}\|\end{aligned}\right\} \implies \|\mathbf{p}_i - \mathbf{p}_j\| \ge 2r_s. \tag{4.30}$$

Hence, we have

$$\begin{aligned}
&\Pr(\mathrm{dis}(\mathbf{p}_i, \mathbf{p}_j) \ge 2r_s) = \Pr(\|\mathbf{p}_i - \mathbf{p}_j\| \ge 2r_s) \\
&\ge \Pr(\mathbf{a}_{ij}\mathbf{p}_i \le b_{ij} - r_s\|\mathbf{a}_{ij}\|) \cdot \Pr(\mathbf{a}_{ji}\mathbf{p}_j \le b_{ji} - r_s\|\mathbf{a}_{ji}\|) \\
&\ge \sqrt{1-\delta} \cdot \sqrt{1-\delta} \\
&= 1 - \delta.
\end{aligned} \tag{4.31}$$

This completes the proof. □

***Theorem* 4.3** (Robot-obstacle probabilistic collision free)**.** *$\forall \mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$, where $\hat{\mathbf{p}}_i \in \mathcal{V}_i^{u,b}$, we have $\Pr(\mathrm{dis}(\mathbf{p}_i, \mathcal{O}_o) \ge r_s) \ge 1 - \delta$, i.e. the probability of collision between robot i and obstacle o is below the threshold δ.*

Proof. Similar to Eq. (4.28), we have

$$\Pr(\mathbf{a}_{io}\mathbf{p}_i \le b_{io} - r_s\|\mathbf{a}_{io}\|) \ge \sqrt{1-\delta}. \tag{4.32}$$

Based on the computation of $\mathbf{a}_{io}$ and b_{io} in Eq. (4.20)-(4.21), it is straightforward to prove that

$$\mathbf{a}_{io}\mathbf{p}_i \le b_{io} - r_s\|\mathbf{a}_{io}\| \implies \mathrm{dis}(\mathbf{p}_i, \mathcal{S}_o) \ge r_s. \tag{4.33}$$

Thus,

$$\Pr(\mathrm{dis}(\mathbf{p}_i, \mathcal{S}_o) \ge r_s) \ge \sqrt{1-\delta}. \tag{4.34}$$

If $\mathcal{O}_o \subseteq \mathcal{S}_o$ and $\mathrm{dis}(\mathbf{p}_i, \mathcal{S}_o) \ge r_s$, there is $\mathrm{dis}(\mathbf{p}_i, \mathcal{O}_o) \ge r_s$. Hence, by combining with Remark 4.4, we have

$$\begin{aligned}
\Pr(\mathrm{dis}(\mathbf{p}_i, \mathcal{O}_o) \ge r_s) &\ge \Pr(\mathcal{O}_o \subseteq \mathcal{S}_o) \cdot \Pr(\mathrm{dis}(\mathbf{p}_i, \mathcal{S}_o) \ge r_s) \\
&> \sqrt{1-\delta} \cdot \sqrt{1-\delta} \\
&= 1 - \delta,
\end{aligned} \tag{4.35}$$

which completes the proof. □

4.4 Collision avoidance using B-UAVC

In this section, we present our decentralized collision avoidance method using the B-UAVC. We start by describing a reactive feedback controller for single-integrator robots, followed by its extensions to double-integrator and non-holonomic differential-drive robots. A receding horizon planning formulation is further presented for general high-order dynamical systems. We also provide a discussion on our proposed method.

4.4.1 Reactive feedback control

Single integrator dynamics

Consider robots with single integrator dynamics $\dot{\mathbf{p}}_i = \mathbf{u}_i$, where $\mathbf{u}_i = \mathbf{v}_i$ is the control input. Similar to [81], a fast reactive feedback one-step controller can be designed to make each robot move towards its goal location $\mathbf{g}_i$, as follows:

$$\mathbf{u}_i = v_{i,\max} \cdot \frac{\mathbf{g}_i^* - \hat{\mathbf{p}}_i}{\left\|\mathbf{g}_i^* - \hat{\mathbf{p}}_i\right\|}, \tag{4.36}$$

where $v_{i,\max}$ is the robot maximal speed and

$$\mathbf{g}_i^* := \underset{\mathbf{p}\in\mathcal{V}_i^{u,b}}{\arg\min} \|\mathbf{p} - \mathbf{g}_i\|, \tag{4.37}$$

is the closest point in the robot's B-UAVC to its goal location.

The strategy used in the controller, Eq. (4.36), is also called the "move-to-projected-goal" strategy [132]. At each time step, each robot in the system first constructs its B-UAVC $\mathcal{V}_i^{u,b}$, then computes the closest point in $\mathcal{V}_i^{u,b}$ to its goal, i.e. the "projected goal", and generates a control input according to Eq. (4.36). Note that the constructed B-UAVC is a convex polytope represented by the intersection of a set of half-spaces hyperplanes. Hence, finding the closest point, Eq. (4.37), can be recast as a linearly constrained least-square problem, which can be solved efficiently using quadratic programming in polynomial time [137].

Double integrator dynamics

For single-integrator robots, the reactive controller Eq. (4.36) guarantees the robot to be always within its corresponding B-UAVC and thus probabilistic collision-free with other robots and obstacles. However, the controller may drive the robot towards to the boundary of its B-UAVC. Consider the double integrator robot which has a limited acceleration, $\ddot{\mathbf{p}}_i = \mathbf{u}_i$, where $\mathbf{u}_i = \mathbf{acc}_i$ is the control input. It might not be able to continue to stay within its B-UAVC when moving close to the boundary of the B-UAVC. Hence, to enhance safety, as illustrated in Fig. 4.3 we introduce an additional safety stopping buffer, which is defined as

$$\beta_i^s = \begin{cases} \frac{\left\|\mathbf{a}_{il}^T \mathbf{v}_i\right\|^2}{2acc_{i,\max}}, & \text{if } \mathbf{a}_{il}^T \mathbf{v}_i > 0; \\ 0, & \text{otherwise,} \end{cases} \tag{4.38}$$

where $acc_{i,\max}$ is the maximal acceleration of the robot.

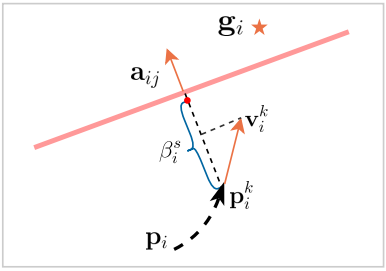


Figure 4.3: Additional buffer is added to allow robots with double integrator dynamics to have enough space to decelerate.

This additional stopping buffer heuristically leaves more space for the robot to decelerate in advance before touching the boundaries of the original B-UAVC. Hence, the updated B-UAVC in Eq. (4.25) with an additional safety stopping buffer now becomes

$$\mathcal{V}_i^{u,b} = \{\mathbf{p} \in \mathbb{R}^d \,:\, \mathbf{a}_{il}^T \mathbf{p} \leq b_{il} - \beta_i^r - \beta_i^\delta - \beta_i^s, \forall l \in \mathcal{I}_l, l \neq i\}. \tag{4.39}$$

Accordingly, the reactive feedback one-step controller for double integrator robots is as follows,

$$\mathbf{u}_i = acc_{i,\max} \cdot \frac{\mathbf{g}_i^* - \hat{\mathbf{p}}_i}{\|\mathbf{g}_i^* - \hat{\mathbf{p}}_i\|}. \tag{4.40}$$

Differential-drive robots

Consider differential-drive robots moving on a two dimensional space $\mathcal{W} \subseteq \mathbb{R}^2$, whose motions are described by

$$\begin{aligned} \dot{\hat{\mathbf{p}}}_i &= v_i \begin{bmatrix} \cos\theta_i \\ \sin\theta_i \end{bmatrix}, \\ \dot{\theta}_i &= \omega_i, \end{aligned} \tag{4.41}$$

where $\theta_i \in [-\pi, \pi)$ is the orientation of the robot, and $\mathbf{u}_i = (v_i, \omega_i)^T \in \mathbb{R}^2$ is the vector of robot control inputs in which v_i and ω_i are the linear and angular velocity, respectively. We adopt the control strategy developed by [132] and [138] and briefly describe it in the following.

As shown in Fig. 4.4, firstly, two line segments

$$L_v = \mathcal{V}_i^{u,b} \cap H_N, \tag{4.42}$$

$$L_\omega = \mathcal{V}_i^{u,b} \cap H_G, \tag{4.43}$$

are determined, in which H_N is the straight line from the robot position towards its current orientation and H_G is the straight line towards its goal location, respectively. Then the closest point in the robot's B-UAVC, $\mathbf{g}_i^*$, and in the two lines segments $\mathbf{g}_{i,v}^*$, $\mathbf{g}_{i,\omega}^*$ is computed.

Finally the control inputs of the robot are given by

$$\begin{aligned} v_i &= -k \cdot [\cos(\theta)\ \sin\theta](\hat{\mathbf{p}}_i - \mathbf{g}^*_{i,v}), \\ \omega_i &= k \cdot \operatorname{atan}\left(\frac{[-\sin(\theta)\ \cos\theta](\hat{\mathbf{p}}_i - (\mathbf{g}^*_i + \mathbf{g}^*_{i,\omega})/2)}{[\cos(\theta)\ \sin\theta](\hat{\mathbf{p}}_i - (\mathbf{g}^*_i + \mathbf{g}^*_{i,\omega})/2)} \right), \end{aligned} \tag{4.44}$$

where $k > 0$ is the fixed control gain. It is proved by [132] that if the local safe region is convex, then the robot will stay within the convex safe region under the control law of Eq. (4.44).

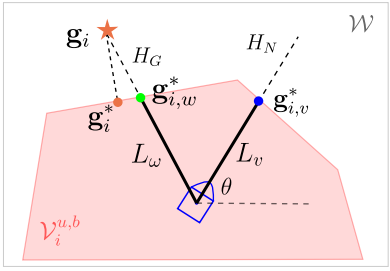


Figure 4.4: Reactive feedback control for differential-drive robots.

4.4.2 Receding horizon planning

Consider general high-order dynamical systems with, potentially nonlinear, dynamics $\mathbf{x}_i^k = \mathbf{f}_i(\mathbf{x}_i^{k-1}, \mathbf{u}_i^{k-1})$, where $\mathbf{x}_i^k \in \mathbb{R}^{n_x}$ denotes the robot state at time step k which typically includes the robot position $\mathbf{p}_i^k$ and velocity $\mathbf{v}_i^k$, and $\mathbf{u}_i^k \in \mathbb{R}^{n_u}$ the robot control input. To plan a local trajectory that respects the robot kinodynamic constraints, we formulate a constrained optimization problem with N time steps and a planning horizon $\tau = N\Delta t$, where Δt is the time step, as follows,

***Problem* 4.1** (Receding horizon trajectory planning)**.**

$$\min_{\hat{\mathbf{x}}_i^{1:N}, \mathbf{u}_i^{0:N-1}} \sum_{k=0}^{N-1} \mathbf{u}_i^k R \mathbf{u}_i^k + (\hat{\mathbf{p}}_i^N - \mathbf{g}_i^N)^T Q_N (\hat{\mathbf{p}}_i^N - \mathbf{g}_i^N)$$

$$s.t.\ \ \mathbf{x}_i^0 = \hat{\mathbf{x}}_i, \tag{4.45a}$$

$$\hat{\mathbf{x}}_i^k = \mathbf{f}_i(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}), \tag{4.45b}$$

$$\hat{\mathbf{p}}_i^k \in \mathcal{V}_i^{u,b}, \tag{4.45c}$$

$$\mathbf{u}_i^{k-1} \in \mathcal{U}_i, \tag{4.45d}$$

$$\forall i \in \mathcal{I}, \forall k \in \{1, \dots, N\}. \tag{4.45e}$$

In Problem 4.1, $\mathcal{U}_i \in \mathbb{R}^{n_u}$ is the admissible control space; $R \in \mathbb{R}^{n_u \times n_u}$, $Q_N \in \mathbb{R}^{d \times d}$ are positive semi-definite symmetric matrices. The constraint (4.45c) restrains the planned

trajectory to be within the robot's B-UAVC $\mathcal{V}_i^{u,b}$. According to the definition of $\mathcal{V}_i^{u,b}$ in Eq. (4.39), the constraint can be formulated as a set of linear inequality constraints:

$$\mathbf{a}_{il}^T \hat{\mathbf{p}}_i^k \leq b_{il} - \beta_i^r - \beta_i^\delta - \beta_i^s, \ \forall l \in \mathcal{I}_l, l \neq i. \tag{4.46}$$

At each time step, the robot first constructs its corresponding B-UAVC $\mathcal{V}_i^{u,b}$ represented by a set of linear inequalities and then solves the above receding horizon planning problem. The problem is in general a nonlinear and non-convex optimization problem due to the robot's nonlinear dynamics formulated as equality constraints $\hat{\mathbf{x}}_i^k = \mathbf{f}_i(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1})$. While a solution of the problem including the planned trajectory and control inputs is obtained, the robot only executes the first control input $\mathbf{u}_i^0$. Then with time going on and at the next time step, the robot updates its B-UAVC and solves the optimization problem again. The process is performed until the robot reaches its goal location.

***Remark* 4.5** (Probability of collision for the planned trajectory)**.** *From Theorem 4.2 and 4.3, constraint (4.45c) guarantees that at each stage within the planning horizon, the collision probability of robot i with any other robot or obstacle is below the specified threshold δ. Hence, the probability of collision for the entire planning trajectory of robot i with respect to each other robot and obstacle can be bounded by* $\Pr(\cup_{k=1}^N \hat{\mathbf{p}}_i^k \notin \mathcal{V}_i^{u,b}) \leq \sum_{k=1}^N \Pr(\hat{\mathbf{p}}_i^k \notin \mathcal{V}_i^{u,b}) = N\delta$. *Nevertheless, this bound is over conservative in practice. The real collision probability of the planned trajectory is much smaller than Nδ [28]. Hence, we impose the collision probability threshold δ for each individual stage in the context of receding horizon planning, thanks to the fast re-planning and relatively small displacement between stages [115].*

Algorithm 1 summarizes our proposed method for decentralized probabilistic multi-robot collision avoidance, in which each robot in the system first constructs its B-UAVC, and then compute control input accordingly to restrain its motion to be within the B-UAVC.

4.4.3 Discussion

Uncertainty estimation

For each robot i in the system, to construct its B-UAVC, the robot needs a) its own position estimation mean $\hat{\mathbf{p}}_i$ and uncertainty covariance Σ_i from onboard measurements via a filter, e.g. a Kalman filter, and b) to know each other robot j's position mean $\hat{\mathbf{p}}_i$ and uncertainty covariance Σ_j. In case communication is available, such position estimation information can be communicated among robots. However, in a fully decentralized system where there is no communication, each robot i will need to estimate other robot j's position mean and covariance, denoted by $\tilde{\mathbf{p}}_j$ and $\tilde{\Sigma}_j$, via its own onboard sensor measurements. In this case, we assume that robot i's estimation of robot j's position mean is the same as robot j's own estimation, i.e. $\tilde{\mathbf{p}}_j = \hat{\mathbf{p}}_j$; while robot i's estimation of the uncertainty covariance of robot j is larger than its own localization uncertainty covariance, i.e. $|\tilde{\Sigma}_j| \geq |\Sigma_i|$. This assumption is reasonable in practice since the robot generally has more accurate measurements of its own position than other robots in the environment. Then robot i computes its B-UAVC using $\hat{\mathbf{p}}_i, \Sigma_i, \tilde{\mathbf{p}}_j$, and $\tilde{\Sigma}_j$. According to the properties of the best linear separator, this assumption leads that each robot i always partitions a smaller space when computing the separating

Algorithm 1 Collision avoidance using b-uavc for each robot $i \in \mathcal{I}$ in a multi-robot team

——————— Construction of B-UAVC ———————
Obtain $\mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$ via state estimation
for Each other robot $j \in \mathcal{I}, j \neq i$ **do**
    Estimate $\mathbf{p}_j \sim \mathcal{N}(\hat{\mathbf{p}}_j, \Sigma_j)$
    Compute the best linear separator parameters $(\mathbf{a}_{ij}, b_{ij})$ via Eq. (4.12)
end for
for Each static obstacle $o \in \mathcal{I}_o$ **do**
    Estimate $\mathbf{d}_o \sim \mathcal{N}(0, \Sigma_o)$ with known $\hat{\mathcal{O}}_o$
    Compute the separating hyperplane parameters $(\mathbf{a}_{io}, b_{io})$ via Eqs. (4.13)-(4.21)
end for
for Each separating hyperplane $l \in \mathcal{I}_l, l \neq i$ **do**
    Compute the safety radius buffer via Eq. (4.23): $\beta_i^r = r_s \|\mathbf{a}_{il}\|$
    Compute the collision probability buffer via Eq. (4.24): $\beta_i^\delta = \sqrt{2\mathbf{a}_{il}^T \Sigma_i \mathbf{a}_{il}} \cdot \mathrm{erf}^{-1}(2\sqrt{1-\delta} - 1)$
    Construct the B-UAVC via Eq. (4.25)
end for
——————— Collision Avoidance Action ———————
if i is single-integrator **then**
    Compute control input via Eqs. (4.36)-(4.37)
else if i is double-integrator **then**
    Compute control input via Eqs. (4.38)-(4.40)
else if i is differential-drive **then**
    Compute control input via Eqs. (4.42)-(4.44)
else
    Compute control input by solving Problem 1
end if

hyperplane with another robot j, which results in a more conservative B-UAVC to ensure safety for robot i itself.

Empty B-UAVCs

Taking into account uncertainty, the robots being probabilistic collision-free (Definition 1), i.e., $\Pr(\|\mathbf{p}_i - \mathbf{p}_j\| \geq 2r_s) \geq 1 - \delta, \forall i, j \in \{1, \ldots, n\}, i \neq j$, does not guarantee that the defined B-UAVC $\mathcal{V}_i^{u,b}$ is non-empty. Nevertheless, the case $\mathcal{V}_i^{u,b}$ being empty is rarely observed in our simulations and experiments. We handle this situation by decelerating the robot if its B-UAVC is empty.

Deadlock resolution heuristic

Since the proposed collision avoidance method is local and decentralized, deadlocks may happen. In this chapter, we detect and resolve deadlocks in a heuristic way. Let $\|\Delta\mathbf{p}_i\|$ be the position progress between two consecutive time steps of robot i, and $\Delta\mathbf{p}_{\min}$ a predefined

minimum allowable progress distance for the robot in n_{dead} time steps. If the robot has not reached its goal and $\Sigma_{n_{\text{dead}}} \|\Delta \mathbf{p}_i\| \leq \Delta \mathbf{p}_{\min}$, we consider the robot as in a deadlock situation. For the one-step controller, each robot must be at the "projected goal" $\mathbf{g}_i^*$ when the system is in a deadlock configuration [81]. In this case, each robot chooses one of the nearby edges within its B-UAVC to move along. For receding horizon planning of high-order dynamical systems, the robot may get stuck due to a local minima of the trajectory optimization problem. In this case, we temporarily change the goal location $\mathbf{g}_i$ of each robot by clockwise rotating it along the z axis with 90°, i.e.

$$\mathbf{g}_{i,\text{temp}} = R_Z(-90^\circ)(\mathbf{g}_i - \hat{\mathbf{p}}_i) + \hat{\mathbf{p}}_i, \tag{4.47}$$

where R_Z denotes the rotation matrix for rotations around z-axis. This temporary rotation will change the objective of the trajectory optimization problem, thus helping the robot to recover from a local minima. Once the robot recovers from stuck, its goal is changed back to $\mathbf{g}_i$.

Similar to most heuristic deadlock resolutions, the solutions presented here can not guarantee that all robots will eventually reach their goals since livelocks (robots continuously repeat a sequence of behaviors that bring them from one deadlock situation to another one) may still occur.

4.5 Simulation results

We now present simulation results comparing our proposed B-UAVC method with state-of-the-art baselines as well as a performance analysis of the proposed method in a variety of scenarios.

4.5.1 Comparison to the BVC method

We first compare our proposed B-UAVC method with the BVC approach [81] that we extend in two-dimensional obstacle-free environments with single integrator robots. Both the B-UAVC and BVC methods only need robot position information to achieve collision avoidance, in contrast to the well-known reciprocal velocity obstacle (RVO) method [23] which also requires robot velocity information to be communicated or sensed. Comparison between BVC and RVO has been demonstrated by [81] in 2D scenarios, hence in this chapter we focus on comparing the proposed B-UAVC with BVC.

We deploy the B-UAVC and BVC in a 10×10m environment with 2, 4, 8, 16 and 32 robots forming an *antipodal circle swapping* scenario ([23]). In this scenario, the robots are initially placed on a circle (equally spaced) and their goals are located at the antipodal points of the circle. We use a circle with a radius of 4.0 m in simulation. Each robot has a radius of 0.2 m, a local sensing range of 2.0 m and a maximum allowed speed of 0.4 m/s. The goal is assumed to be reached for each robot when the distance between its center and goal location is smaller than 0.1 m. To simulate collision avoidance under uncertainty, two different levels of noise, $\Sigma_1 = \text{diag}(0.04\text{ m}, 0.04\text{ m})^2$ and $\Sigma_2 = \text{diag}(0.06\text{ m}, 0.06\text{ m})^2$, are added to the robot position measurements. Particularly, each robot's localization uncertainty

covariance is Σ_1 and its estimation of other robots' position uncertainty covariance is Σ_2. The time step used in simulation is $\Delta t = 0.1$ s.

In the basic BVC implementation, an extra 10% or 100% radius buffer is added to the robot's real physical radius to account for measurement uncertainty for comparison [114]. In the B-UAVC implementation, the collision probability threshold is set as $\delta = 0.05$. Any robot will stop moving when it arrives at its goal or is involved in a collision. Both the B-UAVC and BVC methods use the same deadlock resolution techniques proposed in this chapter. We set a maximum simulation step $K = 800$ and the collision-free robots that do not reach their goals within K steps are regarded to be in deadlocks/livelocks.

For each case (number of robots n) and each method, we run the simulation 10 times. In each single run, we evaluate the following performance metrics: (a) collision rate, (b) minimum distance among robots, (c) average travelled distance of robots, and (d) time to complete a single run. The collision rate is defined to be the ratio of robots colliding over the total number of robots. Time to complete a single run is defined to be the time when the last robot reaches its goal. Note that the metrics (2)(3)(4) are calculated for robots that successfully reach their goal locations. Finally, statistics of 10 instances under each case are presented.

The simulation results are presented in Fig. 4.5. In all runs, no deadlocks are observed. In terms of collision avoidance, both the B-UAVC approach and BVC with additional 100% robot radius achieve zero collision in all runs. The BVC with only 10% robot radius leads to collisions when the total number of robots gets larger. In particular, when there are 32 robots an average of 28% robots collide, as shown in Fig. 4.5a. While the BVC with 100% additional robot radius can also achieve zero collision rate as our proposed B-UAVC, it is more conservative and less efficient. In average, the B-UAVC saves 10.1% robot travelled distance (Fig. 4.5c) and 14.4% time for completing a single run (Fig. 4.5d) comparing to the BVC with additional 100% robot radius.

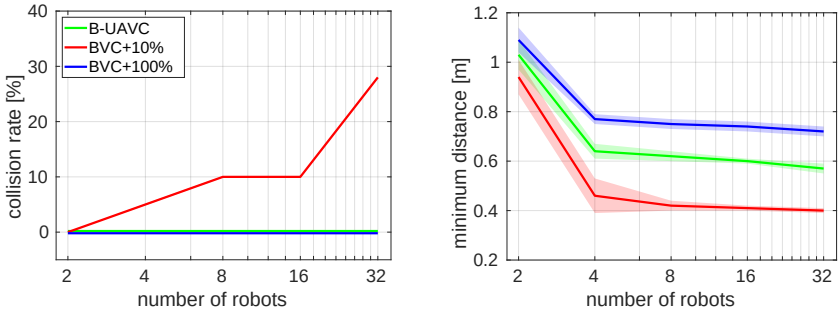
***Remark* 4.6.** *The "BVC + X%" is a heuristic way to handle uncertainty. The above simulation results show that if X is too small, then it cannot ensure safety; while if X is too large, the results will be very conservative and less efficient. So generally reasoning about individual uncertainties using the proposed B-UAVC method will perform better than determining an extra X% buffer.*

***Remark* 4.7.** *In some cases we can design such an X that it will have the same results as the B-UAVC method. Consider the case where $\Sigma_i = \Sigma_j = \sigma^2 I$. According to Remark 4.1, the best linear separator coincides with the separating hyperplane computed by the BVC method, whose parameters are denoted by $\mathbf{a}_{ij}$ and b_{ij}. The hyperplane parameters can be further normalized to make $\|\mathbf{a}_{ij}\| = 1$. In this case, our B-UAVC and the BVC have the same safety radius buffer $\beta_i^r = r_s$. Given a collision probability threshold δ, our B-UAVC further introduces another buffer to handle uncertainty*

$$\beta_i^\delta = \sqrt{2\mathbf{a}_{il}^T \Sigma_i \mathbf{a}_{il}} \cdot \mathrm{erf}^{-1}(2\sqrt{1-\delta} - 1) = \sigma\sqrt{2} \cdot \mathrm{erf}^{-1}(2\sqrt{1-\delta} - 1).$$

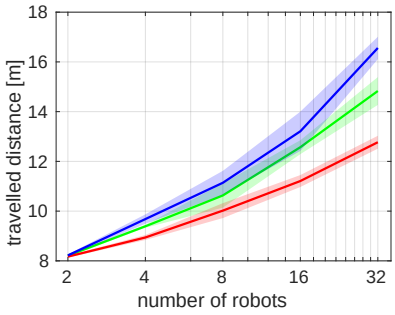
If we choose an extra safety buffer X% such that

$$X\% \cdot r_s = \sigma\sqrt{2} \cdot \mathrm{erf}^{-1}(2\sqrt{1-\delta} - 1),$$

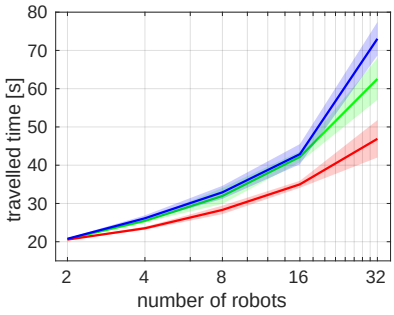


(a) Collision rate.

(b) Minimum distance.



(c) Travelled distance.



(d) Complete time.

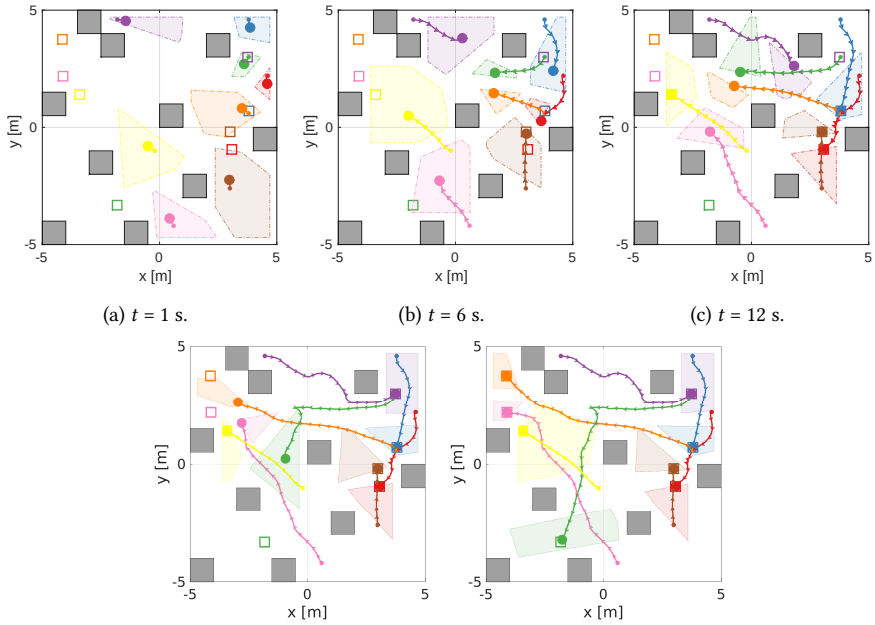
Figure 4.5: Evaluation of the antipodal circle scenario with varying numbers of single-integrator robots. The (a) collision rate, (b) minimum distance, (c) travelled distance and (d) complete time are shown. Lines denote mean values and shaded areas around the lines denote standard deviations over 10 repetitions for each scenario.

then the results of the "BVC + X%" method are the same as our B-UAVC method. However, our B-UAVC method can handle general cases where it is hard to design an X% to always achieve the same level of performance.

4.5.2 Performance analysis

We then study the effect of collision probability threshold on the performance of the proposed B-UAVC method. Similarly, we deploy the B-UAVC in a 10×10m environment with 2, 4, 8, 16 and 32 robots in obstacle-free and cluttered environments with 10% obstacle density. In the obstacle-free case for each number of robots n, 10 scenarios are randomly generated to form a challenging *asymmetric swapping* scenario [139], indicating that the environment is split into n sections around the center and each robot is initially randomly placed in one of them while required to navigate to its opposite section around the center. In the obstacle-cluttered case, 10 *random moving* scenarios are simulated for each different number of robots in which robot initial positions and goal locations are randomly generated. Fig. 4.6 shows a sample run of the scenario with 8 robots and 10 obstacles. We then run each

generated scenario 5 times given a parameter setting (collision probability threshold). The robots have the same radius and maximal speed as in Section 4.5.1. Localization noise with zero mean and covariance $\Sigma = \text{diag}(0.06\text{ m}, 0.06\text{ m})^2$ is added. For evaluation of performance, we focus on the robot collision rate, the robot deadlock rate, and the minimum distance among successful robots.



(a) $t = 1$ s. (b) $t = 6$ s. (c) $t = 12$ s.

(d) $t = 18$ s. (e) $t = 27$ s.

Figure 4.6: A sample simulation run of the random moving scenario with 8 robots and 10% obstacle density. The robot initial and goal locations are marked in circle disks and solid squares. Grey boxes are static obstacles. The B-UAVCs are shown in shaded patches with dashed boundaries.

We evaluate the performance of B-UAVC with different levels of collision probability threshold: $\delta = 0.05, 0.10, 0.20$ and 0.30. The simulation results are presented in Fig. 4.7. In the top row of the figure, we consider the collision rate among robots. The result shows that with a roughly small collision probability threshold $\delta = 0.05, 0.10, 0.20$, no collisions are observed in both obstacle-free asymmetric swapping and obstacle-cluttered random moving scenarios, indicating that the B-UAVC method maintains a high level of safety. However, when δ is set to 0.3, the collision rate among robots increase dramatically, in particular when the number of robots is large. For example, in the asymmetric swapping scenario with 32 robots, there are 68.75% robots involve in collisions in average. In the bottom row of the figure, the minimum distance among robots are compared. The result shows that with smaller threshold, the minimum distance will be a little bit larger. The reason is that robots with a smaller threshold will have more conservative behavior and have smaller B-UAVCs during navigation.

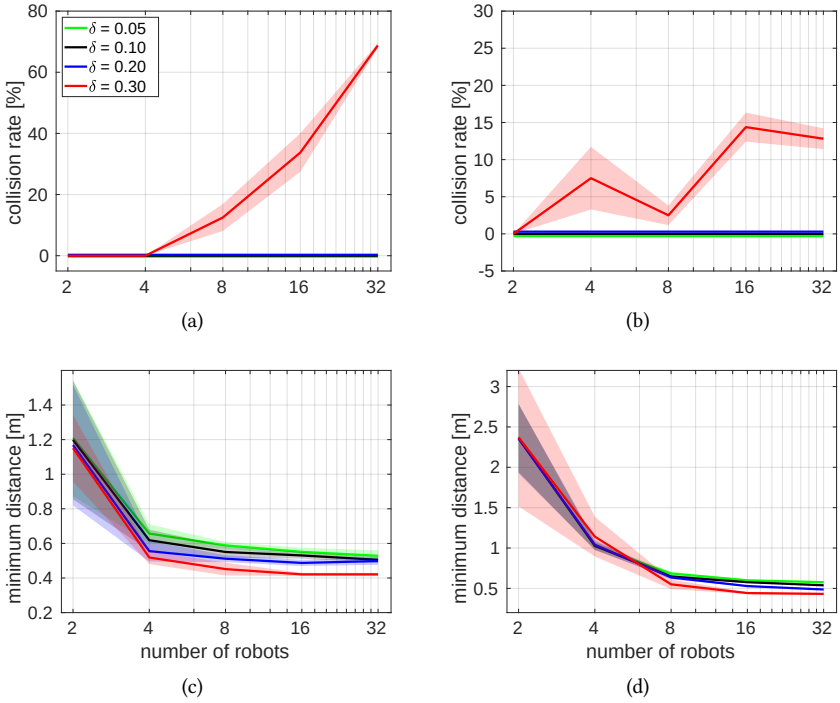


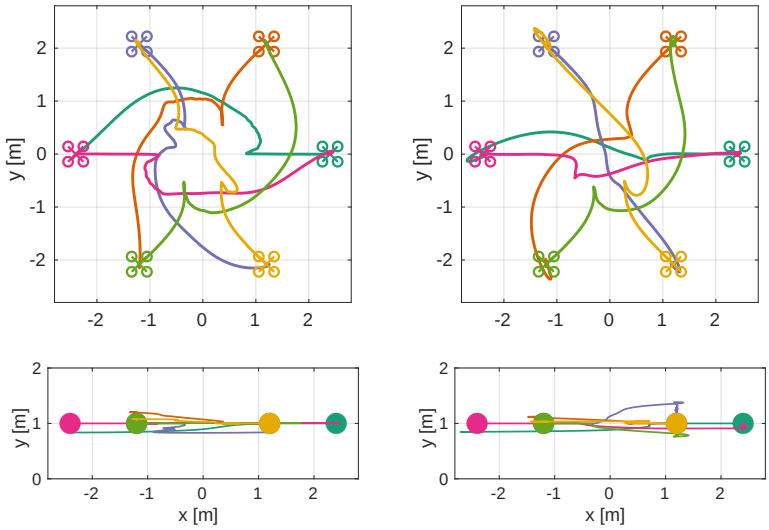
Figure 4.7: Effect of the collision probability threshold on the method performance. The (a)-(b) collision rate, and (c)-(d) minimum distance among robots are shown. The evaluation has 2, 4, 8, 16, and 32 robot cases with 10 instances each. The left column shows results of the asymmetric swapping scenario and the right column shows results of the random moving scenario with 10% obstacle density. Lines denote mean values and shaded areas around the lines denote standard deviations over 50 runs.

4.5.3 Simulations with quadrotors in 3D space

We evaluate our receding horizon planning algorithm with quadrotors in 3D space and compare our method with one of the state-of-the-art quadrotor collision avoidance methods: the chance-constrained nonlinear MPC (CCNMPC) with sequential planning [49], which requires communication of future planned trajectories among robots. For both methods, we adopt the same quadrotor dynamics model for planning. The quadrotor radius is set as $r = 0.3$ m and the collision probability threshold is set to $\delta = 0.03$. The time step is $\Delta t = 0.05$ s and the total number of steps is $N = 20$ resulting in a planing horizon of one second.

As shown in Fig. 4.8, we simulate with six quadrotors exchanging their initial positions in an obstacle-free 3D space. Each quadrotor is under localization uncertainty $\Sigma = \text{diag}(0.04\text{ m},\ 0.04\text{ m},\ 0.04\text{ m})^2$. For each method, we run the simulation 10 times and calculate the minimum distance among robots. Both our B-UAVC method and the CCNMPC method successfully navigates all robots without collision. An average minimum distance of 0.72 m is observed in our B-UAVC method, while the one of CCNMPC is 0.62 m, which

indicates our method is more conservative than the CCNMPC. However, the CCNMPC is centralized and requires robots to communicate their future planned trajectories with each other, while the B-UAVC method only needs robot positions to be shared or sensed.



(a) B-UAVC method.

(b) CCNMPC method [49].

Figure 4.8: Simulation with six quadrotors exchanging positions in 3D space. Solid lines represent executed trajectories of the robots. (a) Results of our B-UAVC method. (b) Results of the CCNMPC method [49].

4.6 Experimental validation

In this section, we describe the experimental results with a team of real robots.

4.6.1 Experimental setup

We test our proposed approach on both ground vehicles and aerial vehicles in an indoor environment of 8m (L) × 3.4m (W) × 2.5m (H). Our ground vehicle platform is the Clearpath Jackal robot and our aerial vehicle platform is the Parrot Bebop 2 quadrotor. For ground vehicles, we apply the controller designed for differential-drive robots as shown in Section 4.4.1. For quadrotors, the receding horizon trajectory planner presented in Section 4.4.2 is employed. The quadrotor dynamics model $\mathbf{f}$ in Problem 4.1 is given in Appendix A. For solving Problem 1 which is a nonlinear programming problem, we rely on the solver Forces Pro [127] to generate fast C code to solve it. Both types of robots allow executing control commands sent via ROS. The experiments are conducted in a standard laptop (Quadcore Intel i7 CPU@2.6 GHz) which connects with the robots via WiFi.

An external motion capture system (OptiTrack) is used to track the pose (position and orientation) of each robot and obstacle in the environment running in real time at 120 Hz,

which is regarded as the *real* (ground-truth) pose. To validate collision avoidance under uncertainty, we then manually add Gaussian noise to the real pose data to generate noisy measurements. Taking the noisy measurements as inputs, a standard Kalman filter running at 120 Hz is employed to estimate the states of the robots and obstacles. In all experiments, the added position measurements noise to the robots is zero mean with covariance Σ_i' = diag(0.06 m, 0.06 m, 0.06 m)^2, which results in an average estimated position uncertainty covariance Σ_i = diag(0.04 m, 0.04 m, 0.04 m)^2. The added noise to the obstacles is zero mean with covariance Σ_o' = diag(0.03 m, 0.03 m, 0.03 m)^2 and the resulted estimated position uncertainty covariance is Σ_o = diag(0.02 m, 0.02 m, 0.02 m)^2. The collision probability threshold is set as δ_r = 0.03 and δ_o = 0.03 as in previous works [49, 140].

4.6.2 Experimental results

Experiments with differential-drive robots in 2D

We first validated our proposed approach with two differential-drive robots. In the experiment, two robots are required to swap their positions while avoiding two static obstacles in the environment. The robot safety radius is set as 0.3 m. We run the experiment four times. The two robots successfully navigated to their goals while avoiding each other as well as the obstacles in all runs.

Fig. 4.9 presents the results of one run. The top row of the figure shows a series of snapshots during the experiment, while the bottom row shows the robots' travelled trajectories and their corresponding B-UAVCs. It can be seen that each robot always keeps a very safe region (B-UAVC) taking into account its localization and sensing uncertainties. In Fig. 4.10 we cumulate the distance between the two robots (Fig. 4.10a) and distance between the robots and obstacles (Fig. 4.10b) during the whole experiments. It can be seen that a minimum safe inter-robot distance of 0.6 m and a safe robot-obstacle distance of 0.3 m were maintained over all the runs.

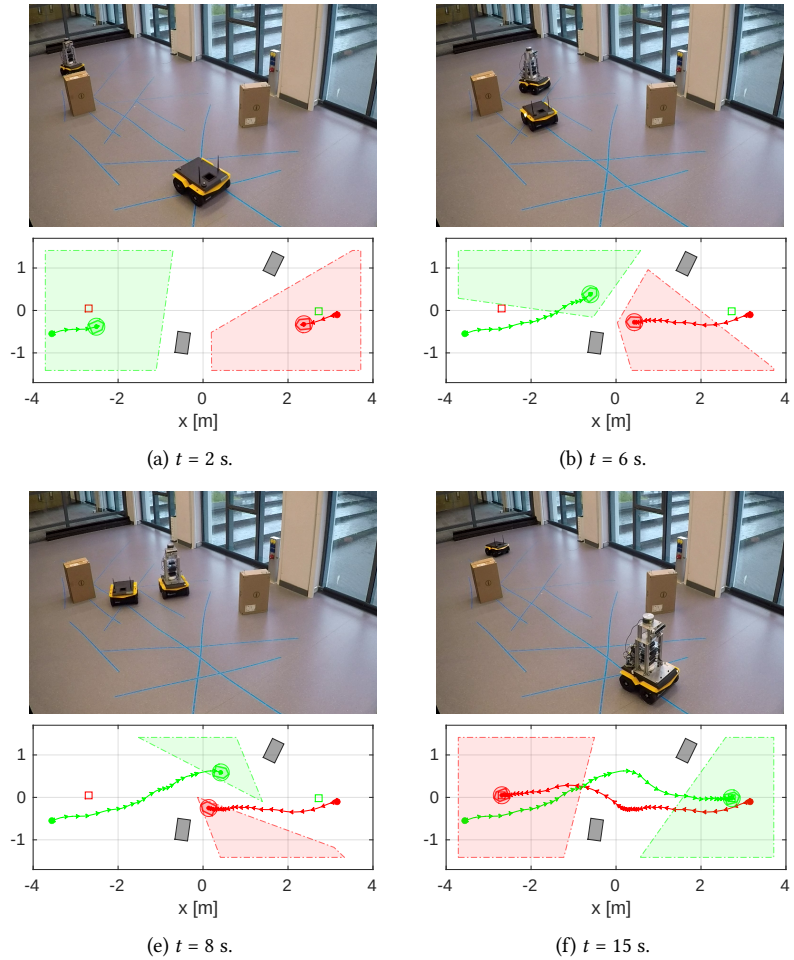
Experiments with quadrotors in 3D

We then performed experiments with a team of quadrotors in two scenarios: with and without static obstacles. The quadrotor safety radius is set as 0.3 m.

Scenario 1 Two quadrotors swap their positions while avoiding two static obstacles in the environment. We performed the swapping action four times and Fig. 4.11 presents one run of the results.

Scenario 2 Three quadrotors fly in a confined space while navigating to different goal positions. The goal locations are randomly chosen such that the quadrotors' directions from initial positions towards goals are crossing. New goals are generated after all quadrotors reach their current goals. We run the experiment for a consecutive two minutes within which the goal of each quadrotor has been changed eight times.

Fig. 4.12 presents a series of snapshots during the experiment. Fig. 4.13 cumulates the inter-quadrotor distance in the experiments of both scenarios, and the distance between quadrotors and obstacles in Scenario 1. It can be seen that a minimum safety distance of



(a) t = 2 s. (b) t = 6 s.

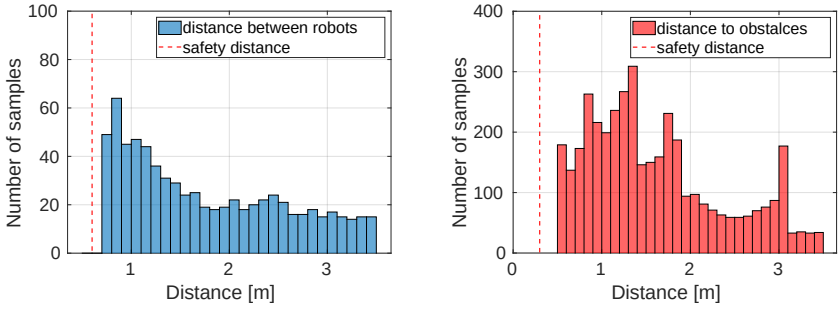
(e) t = 8 s. (f) t = 15 s.

Figure 4.9: Collision avoidance with two differential-drive robots and two static obstacles. The two robots are required to swap their positions. Top row: Snapshots of the experiment. Bottom row: Trajectories of the robots. Robot initial and goal positions are marked in circle disks and solid squares, respectively. Grey boxes are static obstacles. The B-UAVCs are shown in shaded patches with dashed boundaries.

0.6 m among quadrotors and that of 0.3 m between quadrotors and obstacles were achieved during the whole experiments.

Experiments with heterogeneous teams of robots

We further tested our approach with one ground differential-drive robot and one quadrotor to show that it can be applied to heterogeneous robot teams. In the experiment, the ground robot only considers its motion and the obstacles in 2D (the ground plane) while ignoring



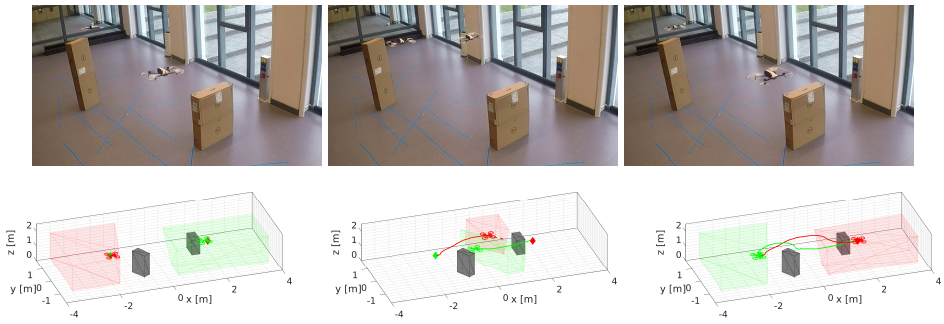
(a) Histogram of inter-robot distance.

(b) Histogram of distance between robots and obstacles.

Figure 4.10: Experimental results with two differential-drive robots.

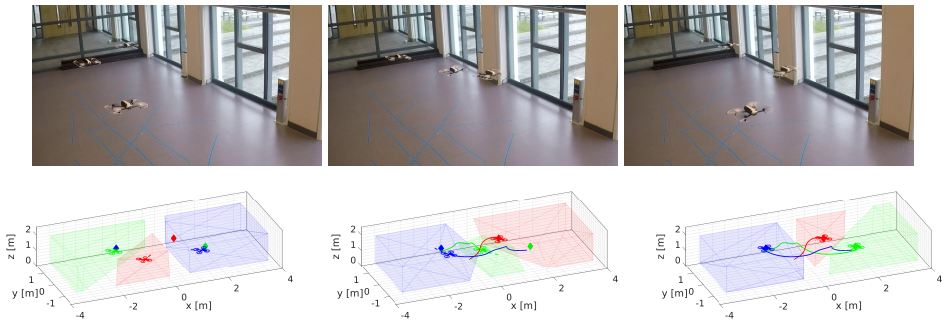
the flying quadrotor. In contrast, the quadrotor considers both itself location and the ground robot's location as well as obstacles in 3D, in which it assumes the ground vehicle has a height of 0.6 m. To this end, the B-UAVC of the ground robot is a 2D convex region while that of the quadrotor is a 3D one.

Fig. 4.14 shows the results of the experiment. It can be seen that the two robots successfully reached their goals while avoiding each other and the static obstacles. Particularly at t = 4 s, the quadrotor actively flies upward to avoid the ground robot. In Fig. 4.15 we cumulate the distance between the two robots and the distance between robots and obstacles, which show that a safe inter-robot clearance of 0.6 m and that of 0.3 m between robots and obstacles were maintained during the experiment.



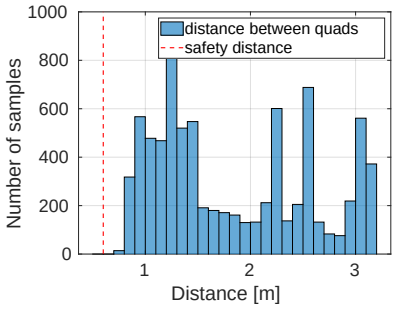
(a) t = 0.1 s. (b) t = 5 s. (c) t = 10.5 s.

Figure 4.11: Collision avoidance with two quadrotors and two static obstacles. The two quadrotors are required to swap their positions. Top row: Snapshots of the experiment. Bottom row: Trajectories of the robots. Quadrotor initial and goal positions are marked in circles and diamonds. Solid lines represent travelled trajectories and dashed lines represent planned trajectories.

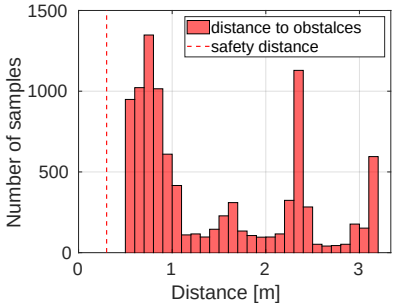


(a) t = 0.1 s. (b) t = 6 s. (c) t = 10 s.

Figure 4.12: Collision avoidance with three quadrotors in a shared workspace. Top row: Snapshots of the experiment. Bottom row: Trajectories of the robots.

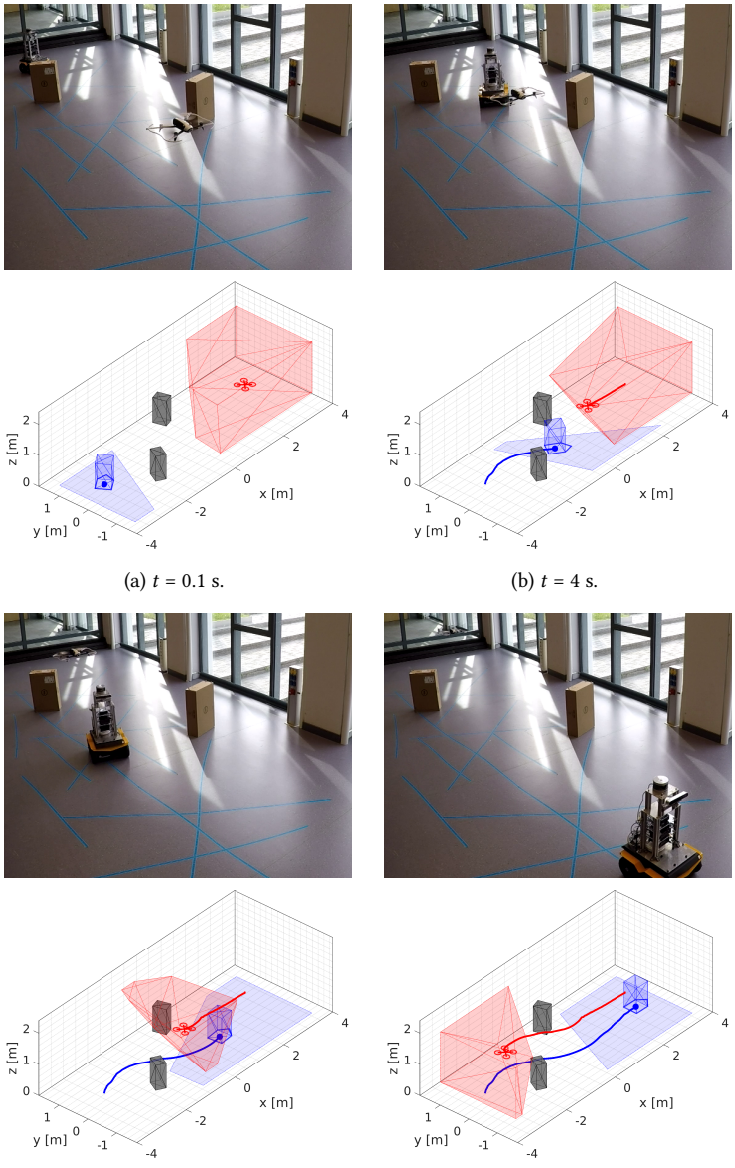


(a) Histogram of inter-robot distance.



(b) Histogram of distance between robots and obstacles.

Figure 4.13: Experimental results with two/three quadrotors with/without obstacles.



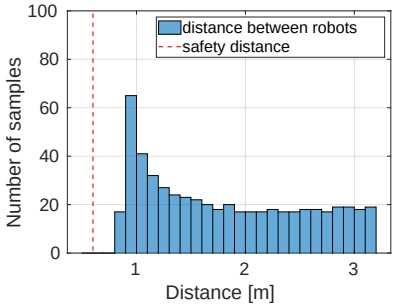
(a) t = 0.1 s.

(b) t = 4 s.

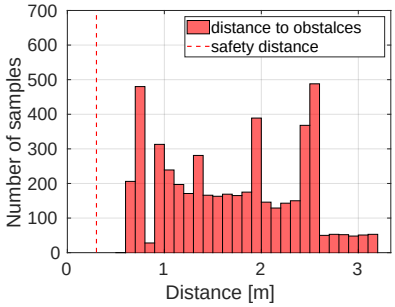
(c) t = 6 s.

(d) t = 12 s.

Figure 4.14: Collision avoidance with a heterogeneous team of a differential-drive robot and a quadrotor. Top row: Snapshots of the experiment. Bottom row: Trajectories of the robots.



(a) Histogram of inter-robot distance.



(b) Histogram of distance between robots and obstacles.

Figure 4.15: Experimental results with a ground differential-drive robot and a quadrotor.

4.7 Conclusion

In this chapter, we presented a decentralized and communication-free multi-robot collision avoidance method that accounts for robot localization and sensing uncertainties. By assuming that the uncertainties are according to Gaussian distributions, we computed a chance-constrained buffered uncertainty-aware Voronoi cell (B-UAVC) for each robot among other robots and static obstacles. The probability of collision between robots and obstacles is guaranteed to be below a specified threshold by constraining each robot's motion to be within its corresponding B-UAVC. We applied the method to single-integrator, double-integrator, differential-drive, and general high-order dynamical multi-robot systems. In comparison with the BVC method, we showed that our method achieves robust safe navigation among a large number of robots with noisy position measurements where the BVC approach will fail. In simulations with a team of quadrotors, we showed that our method achieves safer yet more conservative motions compared with the CCNMPC method, which requires robots to communicate future trajectories. We also validated our method in extensive experiments with a team of ground vehicles, quadrotors, and heterogeneous robot teams in both obstacle-free and obstacles-cluttered environments. Through simulations and experiments, two limitations of the proposed approach are also observed. The approach can achieve a high level of safety under robot localization and sensing uncertainty, however, it also leads to conservative behaviors of the robots, particularly for agile vehicles (quadrotors) in confined space. And, since the approach is local and efficient inter-robot coordination is not well investigated, deadlocks and livelocks may occur for large numbers of robots moving in complex environments.

5

Chance-constrained safety barrier certificates for multi-robot collision avoidance under uncertainty

Parts of this chapter appeared in:

- H. Zhu, and J. Alonso-Mora, "Chance-constrained safety barrier certificates for multi-robot collision avoidance under uncertainty," submitted to *2022 IEEE International Conference on Robotics and Automation (ICRA)*, under review.

5.1 Introduction

Multi-robot systems have been increasingly deployed in real-world environments to accomplish some high-level tasks, such as coverage [141], target tracking [142], and formation maintaining [125]. To achieve autonomous navigation of robots in these scenarios, a typical solution is to adopt a two-stage planning framework: a high-level task planner that outputs a sequence of task-orientated waypoints for each robot, and a low-level motion planner that plans a local motion and control input for the robot to reach these waypoints [143]. In Chapter 3 and Chapter 4, we have presented two local collision-free motion planners, namely CCNMPC and B-UAVC, which can compute safe control inputs for robots to move towards their goal waypoints. Alternatively, another two-stage planning framework is to design a primary nominal controller that outputs a task-orientated control input which may not consider collision avoidance, and then modifies this control input to ensure safety if necessary through a local collision avoidance controller. Based on safety barrier certificates (SBC), [82] designs such a collision avoidance controller that can minimally modify a given primary nominal controller for multi-robot systems. In this sense, the SBC can be regarded as a safety filter of the system.

In this chapter, we extend the SBC method [82] from deterministic scenarios to probabilistic cases considering robot measurement noise which leads to uncertainty of the robot localization and sensing of other robots. Specifically, we introduce chance-constrained safety barrier certificates (CC-SBC), which defines a probabilistic safe control space for each robot in the system. The CC-SBC is reformulated to and approximated with a set of quadratic constraints. At each time step, each robot formulates its CC-SBC according to the estimated state and uncertainty of itself and neighboring robots, based on which a quadratically constrained quadratic program (QCQP) is formulated and solved to find a safe control input that minimally modifies a given nominal controller. Probabilistic collision avoidance is guaranteed via the forward-invariance property of the safety set which is achieved by constraining the robot control input in its CC-SBC.

The remaining of this chapter is organized as follows. Section 5.2 introduces preliminaries of safety barrier certificates and moments of probability distributions. Section 5.3 presents the approach of using chance-constrained safety barrier certificates for multi-robot collision avoidance under uncertainty. Section 5.4 shows simulation results to validate the proposed approach. Finally, Section 5.5 concludes the chapter.

5.2 Preliminaries

5.2.1 Safety barrier certificates

Consider a nonlinear control-affine system:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \tag{5.1}$$

where $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^{n_x}$ and $\mathbf{u} \in \mathcal{U} \subset \mathbb{R}^{n_u}$ are the system state and control input, respectively. $\mathcal{X}$ and $\mathcal{U}$ are convex admissable state and control space of the system. $\mathbf{f} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ and $\mathbf{g} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x \times n_u}$ are locally Lipschitz continuous.

Let $\mathcal{C} \subset \mathbb{R}^{n_x}$ be a safe set of the system state, defined as the superlevel set of a continuously differentiable function $h : \mathbb{R}^{n_x} \to \mathbb{R}$:

$$\mathcal{C} := \{\mathbf{x} \in \mathbb{R}^{n_x} \mid h(\mathbf{x}) \geq 0\}. \tag{5.2}$$

The system (5.1) is ensured to be safe with respect to $\mathcal{C}$ if $\mathcal{C}$ is *forward invariant*, i.e. for any $\mathbf{x}(0) \in \mathcal{C}$, $\mathbf{x}(t)$ remains in $\mathcal{C}$ for all $t \geq 0$. Forward invariance of the safe set $\mathcal{C}$ can be achieved using control barrier functions [144]. Formally,

***Definition* 5.1.** *The function h in Eq. (5.2) is a control barrier function (CBF) if there exists an extended class $\mathcal{K}_\infty$ function κ such that*

$$\sup_{\mathbf{u} \in \mathcal{U}} \mathrm{CBC}(\mathbf{x}, \mathbf{u}) \geq 0,$$

where

$$\mathrm{CBC}(\mathbf{x}, \mathbf{u}) = L_f h(\mathbf{x}) + L_g h(\mathbf{x})\mathbf{u} + \kappa(h(\mathbf{x})),$$

is the control barrier condition (CBC).

***Theorem* 5.1** ([145])**.** *If h is a CBF on $\mathcal{C}$, then any Lipschitz continuous controller satisfying Eq. (5.1) renders the system (5.1) safe with respect to $\mathcal{C}$.*

The safety barrier certificates (SBC) define a safe admissable control space for the system as follows:

$$\mathcal{S}(\mathbf{x}) := \{\mathbf{u} \in \mathcal{U} \mid \mathrm{CBC}(\mathbf{x}, \mathbf{u}) \geq 0\}. \tag{5.3}$$

Note that the condition $\mathrm{CBC}(\mathbf{x}, \mathbf{u}) \geq 0$ can be written as

$$-L_g h(\mathbf{x})\mathbf{u} \leq L_f h(\mathbf{x}) + \kappa(h(\mathbf{x})), \tag{5.4}$$

which is affine in $\mathbf{u}$. That is, $\mathcal{S}(\mathbf{x})$ can be represented by a set of linear inequality constraints.

To apply the above safety barrier certificates for safety-critical control, one can assume that there is a nominal controller $\mathbf{u}^*$ and formulate a quadratic program (QP) to find a safety-critical minimally invasive controller:

$$\begin{aligned} \mathbf{u}(\mathbf{x}) = \underset{\mathbf{u} \in \mathcal{U}}{\arg\min} \quad & \frac{1}{2}\|\mathbf{u} - \mathbf{u}^*\|^2 \qquad \textbf{(SBC-QP)} \\ \text{s.t.} \quad & \mathrm{CBC}(\mathbf{x}, \mathbf{u}) \geq 0. \end{aligned} \tag{5.5}$$

The QP formulation in Eq. (5.5) can be regarded as a safety filter. In this chapter, we employ the above idea of safety barrier certificates and extend it to probabilistic scenarios to account for system state uncertainty.

5.2.2 Moments of distributions

In statistics, the moments of a probabilistic distribution are quantitative measures of the distribution's shape. For example, the first moment is the expected value and the second central moment is the covariance. Formally, for a multivariate random variable

$\mathbf{x} = [x^1, \dots, x^{n_x}] \in \mathbb{R}^{n_x}$ with probability density function (PDF) $p(\mathbf{x})$, its m-th moment about zero is denoted by and defined as follows:

$$\mathbb{E}[x^1 x^2 \cdots x^m] = \int_{\infty} x^{k_1} x^{k_2} \cdots x^{k_m} p(\mathbf{x}) d\mathbf{x}.$$

Moments of a distribution can also be computed using its characteristic function (CF) defined as $\Phi(\mathbf{x}) = \mathbb{E}[e^{\mathrm{i}\mathbf{x}^T \mathbf{x}}], \mathrm{i} = \sqrt{-1}$, as follows:

$$\mathbb{E}[x^1 x^2 \cdots x^m] = \mathrm{i}^{-m} \left. \frac{\partial^m \Phi(\mathbf{x})}{\partial x^{k_1} x^{k_2} \cdots x^{k_m}} \right|_{\mathbf{x}=0}.$$

The up to forth moments about zero of a multivariate Gaussian distribution $\mathbf{x} \sim \mathcal{N}(\mu, \Sigma)$, where $\mu \in \mathbb{R}^{n_x}, \Sigma \in \mathbb{R}^{n_x \times n_x}$ are the mean vector and covariance matrix, are as follows,

$$\mathbb{E}[x^i] = \mu^i, \tag{5.6a}$$
$$\mathbb{E}[x^i x^j] = \mu^i \mu^j + \Sigma^{ij}, \tag{5.6b}$$
$$\mathbb{E}[x^i x^j x^k] = \mu^i \mu^j \mu^k + \mu^i \Sigma^{jk} + \mu^j \Sigma^{ik} + \mu^k \Sigma^{ij}, \tag{5.6c}$$
$$\mathbb{E}[x^i x^j x^k x^l] = \mu^i \mu^j \mu^k \mu^l, \tag{5.6d}$$
$$+ \mu^i \mu^j \Sigma^{kl} + \mu^i \mu^k \Sigma^{jl} + \mu^i \mu^l \Sigma^{jk} + \mu^j \mu^k \Sigma^{il} + \mu^j \mu^l \Sigma^{ik} + \mu^k \mu^l \Sigma^{ij}, \tag{5.6e}$$
$$+ \Sigma^{ij} \Sigma^{kl} + \Sigma^{ik} \Sigma^{jl} + \Sigma^{il} \Sigma^{jk}, \tag{5.6f}$$
$$\forall i, j, k, l \in \{1, \dots, n_x\}. \tag{5.6g}$$

Furthermore, let $\Delta\mathbf{x} = \mathbf{x} - \mu$, then we have $\Delta\mathbf{x} \sim \mathcal{N}(0, \Sigma)$, and its up to forth moments about zero are

$$\mathbb{E}[\Delta x^i] = 0, \tag{5.7a}$$
$$\mathbb{E}[\Delta x^i \Delta x^j] = \Sigma^{ij}, \tag{5.7b}$$
$$\mathbb{E}[\Delta x^i \Delta x^j \Delta x^k] = 0, \tag{5.7c}$$
$$\mathbb{E}[\Delta x^i \Delta x^j \Delta x^k \Delta x^l] = \Sigma^{ij} \Sigma^{kl} + \Sigma^{ik} \Sigma^{jl} + \Sigma^{il} \Sigma^{jk}. \tag{5.7d}$$

5.2.3 Problem formulation

Similar to Chapter 4, we consider a group of n mobile robots moving in a d-dimensional space $\mathcal{W} \subseteq \mathbb{R}^d$, where $d \in \{2, 3\}$. The robot dynamics are described in the form of Eq. (5.1). For each robot $i \in \mathcal{I} = \{1, \dots, n\}$, denote by $\mathbf{x}_i \in \mathbb{R}^{n_x}$ and $\mathbf{u}_i \in \mathbb{R}^{n_u}$ its state and control input. The state $\mathbf{x}_i$ typically may contain the robot's position $\mathbf{p}_i \in \mathbb{R}^d$ and velocity $\mathbf{v}_i \in \mathbb{R}^d$. We use the sub-script i to indicate variables corresponding to the robot i. To achieve safe navigation, all robots in the group are required to keep a safety distance d_s from each other, i.e. $\|\mathbf{p}_i - \mathbf{p}_j\| \geq d_s, \ \forall i \neq j \in \mathcal{I}$.

We consider that the state of each robot is obtained by a state estimator, e.g. a Kalman filter, and is described as a Gaussian distribution with covariance Σ_i, i.e. $\mathbf{x}_i \sim \mathcal{N}(\hat{\mathbf{x}}_i, \Sigma_i)$.

Taking the state estimation uncertainty into account, the collision-free condition for each pair of robots can only be satisfied in a probabilistic manner, which is defined as the following chance constraint,

$$\Pr(\|\mathbf{p}_i - \mathbf{p}_j\| \geq d_s) \geq 1 - \delta, \quad \forall i, j \in \mathcal{I}, i \neq j. \tag{5.8}$$

The objective of probabilistic multi-robot collision avoidance is to find a control action $\mathbf{u}_i$ for each robot in the group such that the probability of collision among robots is below a specified threshold, as shown in Eq. (5.8).

5.3 Approach

In this section, we present our probabilistic multi-robot collision avoidance method by introducing chance-constrained safety barrier certificates (CC-SBC).

5.3.1 SBC for multi-robot systems

Let C_{ij} be the pairwise safe set between robot i and j defined by

$$C_{ij} = \{\mathbf{x}_{ij} \in \mathbb{R}^{n_x} \mid h_{ij}(\mathbf{x}_{ij}) \geq 0\},$$

where $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ is the relative state of the two robots. $h_{ij}(\mathbf{x}_{ij})$ is the level set function of the set C_{ij} as well as the candidate CBF used to ensure the forward invariance of C_{ij}, which should lead to

$$h_{ij}(\mathbf{x}_{ij}) \geq 0 \implies \|\mathbf{p}_i - \mathbf{p}_j\| \geq d_s, \tag{5.9}$$

for inter-robot collision avoidance. Here we use a general form to denote the function $h_{ij}(\mathbf{x}_{ij})$. Detailed formulations of $h_{ij}(\mathbf{x}_{ij})$ for robots with particular dynamics are discussed later.

Denote by $\mathrm{CBC}_{ij}(\mathbf{x}_{ij}, \mathbf{u}_{ij})$ the pairwise control barrier condition (CBC) of robot i and j [82]. Then Eq. (5.4) becomes

$$-\frac{\partial h(\mathbf{x}_{ij})}{\partial \mathbf{x}_{ij}}\mathbf{g}(\mathbf{x}_{ij})\mathbf{u}_{ij} \leq \frac{\partial h(\mathbf{x}_{ij})}{\partial \mathbf{x}_{ij}}\mathbf{f}(\mathbf{x}_{ij}) + \kappa(h(\mathbf{x}_{ij})).$$

Where $\mathbf{u}_{ij} = \mathbf{u}_i - \mathbf{u}_j \in \mathbb{R}^{n_u}$. Let

$$\begin{aligned} A_{ij}(\mathbf{x}_{ij}) &= -\frac{\partial h(\mathbf{x}_{ij})}{\partial \mathbf{x}_{ij}}\mathbf{g}(\mathbf{x}_{ij}), \\ b_{ij}(\mathbf{x}_{ij}) &= \frac{\partial h(\mathbf{x}_{ij})}{\partial \mathbf{x}_{ij}}\mathbf{f}(\mathbf{x}_{ij}) + \kappa(h(\mathbf{x}_{ij})), \end{aligned} \tag{5.10}$$

then there is

$$A_{ij}(\mathbf{x}_{ij})\mathbf{u}_{ij} \leq b_{ij}(\mathbf{x}_{ij}).$$

The above pairwise safety barrier constraint between robot i and j can be distributed to each robot as

$$A_{ij}(\mathbf{x}_{ij})\mathbf{u}_i \leq \frac{1}{2}b_{ij}(\mathbf{x}_{ij}),$$
$$-A_{ij}(\mathbf{x}_{ij})\mathbf{u}_j \leq \frac{1}{2}b_{ij}(\mathbf{x}_{ij}).$$

Hence, for each robot i in the system, its safety barrier certificates (SBC) can be defined as

$$\mathcal{S}_i(\mathbf{x}_{1:n}) := \{\mathbf{u}_i \in \mathcal{U} \mid A_{ij}(\mathbf{x}_{ij})\mathbf{u}_i \leq \frac{1}{2}b_{ij}(\mathbf{x}_{ij}),\ \forall j \neq i \in \mathcal{I}\}, \tag{5.11}$$

where $\mathbf{x}_{1:n} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ is the joint state of the multi-robot system.

5.3.2 Chance-constrained SBC

5

According to Theorem 5.1, for any pair of robots i and j which are initially safe, i.e. $\mathbf{x}_{ij} \in \mathcal{C}_{ij}$ at current time, by enforcing the control input of each robot to be within its SBC, the evolved relative state $\mathbf{x}_{ij}$ is guaranteed to be safe, i.e. we have $\mathbf{u}_i \in \mathcal{S}_i(\mathbf{x}_{1:n}), \mathbf{u}_j \in \mathcal{S}_j(\mathbf{x}_{1:n}) \Longrightarrow \mathbf{x}_{ij} \in \mathcal{C}_{ij}$. Hence, in probabilistic scenarios where the state $\mathbf{x}_{1:n}$ estimation uncertainty is considered, it is straightforward to show that $\Pr(\mathbf{u}_i \in \mathcal{S}_i(\mathbf{x}_{1:n}), \mathbf{u}_j \in \mathcal{S}_j(\mathbf{x}_{1:n})) \leq \Pr(\mathbf{x}_{ij} \in \mathcal{C}_{ij})$. Consequently, also according to Eq. (5.9), we can derive the following probabilistic safe condition,

$$\begin{aligned}&\Pr(\mathbf{u}_i \in \mathcal{S}_i(\mathbf{x}_{1:n}), \mathbf{u}_j \in \mathcal{S}_j(\mathbf{x}_{1:n})) \geq 1-\delta\\ \Longrightarrow\ &\Pr(\mathbf{x}_{ij} \in \mathcal{C}_{ij}) \geq 1-\delta\\ \Longrightarrow\ &\Pr(\|\mathbf{p}_i - \mathbf{p}_j\| \geq d_s) \geq 1-\delta.\end{aligned}$$

We can further decompose the chance constraint $\Pr(\mathbf{u}_i \in \mathcal{S}_i(\mathbf{x}_{1:n}), \mathbf{u}_j \in \mathcal{S}_j(\mathbf{x}_{1:n})) \geq 1-\delta$ to each robot as $\Pr(\mathbf{u}_i \in \mathcal{S}_i(\mathbf{x}_{1:n})) \geq \sqrt{1-\delta}$ and $\Pr(\mathbf{u}_j \in \mathcal{S}_j(\mathbf{x}_{1:n})) \geq \sqrt{1-\delta}$, taking into account that the two robots are controlled independently in a decentralized manner. Recall that $\mathcal{S}_i(\mathbf{x}_{1:n})$ in Eq. (5.11) is defined by the condition $A_{ij}(\mathbf{x}_{ij})\mathbf{u}_i \leq \frac{1}{2}b_{ij}(\mathbf{x}_{ij})$, so $\Pr(\mathbf{u}_i \in \mathcal{S}_i(\mathbf{x}_{1:n})) = \Pr(A_{ij}(\mathbf{x}_{ij})\mathbf{u}_i \leq \frac{1}{2}b_{ij}(\mathbf{x}_{ij}))$, and we can then formally define the following chance-constrained safety barrier certificates (CC-SBC)

$$\mathcal{S}_i^{\delta}(\mathbf{x}_{1:n}) := \{\mathbf{u}_i \in \mathcal{U} \mid \Pr(A_{ij}(\mathbf{x}_{ij})\mathbf{u}_i \leq \frac{1}{2}b_{ij}(\mathbf{x}_{ij})) \geq \sqrt{1-\delta},\ \forall j \neq i \in \mathcal{I}\}, \tag{5.12}$$

which hence characterizes a probabilistically safe control space for each robot in the system. Namely, by enforcing $\mathbf{u}_i \in \mathcal{S}_i^{\delta}(\mathbf{x}_{1:n})$, $\forall i \in \mathcal{I}$, the system is probabilistically safe corresponding to Eq. (5.8).

***Remark* 5.1.** *The set $\mathcal{S}_i^{\delta}(\mathbf{x}_{1:n})$ is in general non-convex.*

***Remark* 5.2.** *Our definition of chance-constrained safety barrier certificates (CC-SBC) is different from the probabilistic safety barrier certificates (PrSBC) defined by [115], which is defined as a convex set directly and is only designed for single-integrator dynamics, while our definition is for general nonlinear control-affine systems.*

5.3.3 Quadratically constrained quadratic program

Similar to Eq. (5.5), we can then formulate a chance-constrained optimization (CCO) problem to find a probabilistic safety-critical controller for each robot i in the system as follow:

$$\begin{aligned} \mathbf{u}_i(\mathbf{x}_{1:n}) = \underset{\mathbf{u}_i \in \mathcal{U}}{\arg\min} \ & \frac{1}{2}\left\|\mathbf{u}_i - \mathbf{u}_i^*\right\|^2 \quad \textbf{(CC-SBC-CCO)} \\ \text{s.t. } & \Pr(A_{ij}(\mathbf{x}_{ij})\mathbf{u}_i \le \frac{1}{2} b_{ij}(\mathbf{x}_{ij})) \ge \sqrt{1-\delta},\ \forall j \ne i \in \mathcal{I}. \end{aligned} \tag{5.13}$$

where $\mathbf{u}_i^*$ is a nominal controller of the robot.

The CCO (5.13) is hard to solve due to the chance constraints $\Pr(A_{ij}(\mathbf{x}_{ij})\mathbf{u}_i \le \frac{1}{2} b_{ij}(\mathbf{x}_{ij})) \ge \sqrt{1-\delta}$. Without loss of generality and for simplicity, we omit the sub-script $\cdot_{ij}$ and $\mathbf{x}_{ij}$ in the equations in this sub-section, and let $A = A_{ij}(\mathbf{x}_{ij}), b = \frac{1}{2} b_{ij}(\mathbf{x}_{ij})$. Further let $\lambda = b - A\mathbf{u}_i$, then the chance constraint can be written as

$$\Pr(A\mathbf{u}_i \le b) = \Pr(\lambda \ge 0) \ge \sqrt{1-\delta}. \tag{5.14}$$

To deal with the above chance constraint, we first apply the Cantelli's inequality, which is also known as the one-tailed Chebyshev inequality [58]:

$$\Pr(\lambda \ge 0) \begin{cases} \le \frac{\sigma_\lambda^2}{\sigma_\lambda^2 + \mu_\lambda^2}, & \text{if } \mu_\lambda < 0 \\ \ge 1 - \frac{\sigma_\lambda^2}{\sigma_\lambda^2 + \mu_\lambda^2}, & \text{if } \mu_\lambda \ge 0 \end{cases} \tag{5.15}$$

where μ_λ and σ_λ^2 are the mean and variance of $\lambda(\mathbf{x}_{ij})$ respectively. Intuitively, we need the expectation (mean) of $\lambda(\mathbf{x}_{ij})$ to be larger than zero. Hence, the chance constraint Eq. (5.14) can be reformulated as

$$\mu_\lambda \ge 0, \tag{5.16a}$$

$$\sqrt{1-\delta}\sigma_\lambda^2 - (1 - \sqrt{1-\delta})\mu_\lambda^2 \le 0, \tag{5.16b}$$

which can be proved to be quadratic in the following theorem.

***Theorem* 5.2.** *The constraints Eq. (5.16a) and (5.16b) are quadratic with respect to* $\mathbf{u}_i$.

Proof. Note that $\lambda = b - A\mathbf{u}_i = b - \mathbf{u}_i^T A^T$. Hence we have

$$\begin{aligned} \mu_\lambda &= \mathbb{E}[b] - \mathbb{E}[A]\mathbf{u}_i, \\ \sigma_\lambda^2 &= \mathbb{V}[b] + \mathbf{u}_i^T \mathbb{V}[A^T]\mathbf{u}_i + 2(\mathbb{E}[b]\mathbb{E}[A] - \mathbb{E}[bA])\mathbf{u}_i. \end{aligned}$$

Then, Eq. (5.16a) and (5.16b) can be written as,

$$A'\mathbf{u}_i + b' \le 0,$$

$$\frac{1}{2}\mathbf{u}_i^T P \mathbf{u}_i + \mathbf{q}^T \mathbf{u}_i + r \le 0,$$

which are quadratic with respect to $\mathbf{u}_i$, where

$$\begin{aligned} A' &= \mathbb{E}[A],\ b' = -\mathbb{E}[b], \\ P &= 2\sqrt{1-\delta}\mathbb{V}[A^T] - 2(1-\sqrt{1-\delta})\mathbb{E}[A]\mathbb{E}[A]^T, \\ q &= 2\mathbb{E}[b]\mathbb{E}[A] - 2\sqrt{1-\delta}\mathbb{E}[bA], \\ r &= \sqrt{1-\delta}\mathbb{V}[b] - (1-\sqrt{1-\delta})\mathbb{E}[b]^2. \end{aligned} \tag{5.17}$$

This completes the proof. □

The CCO (5.13) can then be reformulated to the following quadratically constrained quadratic program (QCQP):

$$\begin{aligned} \mathbf{u}_i(\mathbf{x}_{1:n}) = \underset{\mathbf{u}_i \in \mathcal{U}}{\arg\min}\ & \frac{1}{2}\left\|\mathbf{u}_i - \mathbf{u}_i^*\right\|^2 \qquad \textbf{(CC-SBC-QCQP)} \\ \text{s.t. } & A'\mathbf{u}_i + b' \le 0, \\ & \frac{1}{2}\mathbf{u}_i^T P \mathbf{u}_i + \mathbf{q}^T \mathbf{u}_i + r \le 0. \end{aligned} \tag{5.18}$$

in which the parameters of the quadratic constraints, as shown in Eq. (5.17), can be computed by evaluating $\mathbb{E}[A]$, $\mathbb{E}[b]$, $\mathbb{V}[A^T]$, $\mathbb{V}[b]$ and $\mathbb{E}[bA]$, given the uncertainty information of the relative state $\mathbf{x}_{ij}$. In the following, we present a general method to compute these statistical moments.

Denote by $\xi(\mathbf{x}) : \mathbb{R}^{n_x} \longrightarrow \mathbb{R}^{n_\xi}$ a function of $\mathbf{x}$, that is, it can be b with $n_\xi = 1$, or A and bA with $n_\xi = n_u$. We start with the special case where $\xi(\mathbf{x})$ is a polynomial function of $\mathbf{x}$.

***Lemma* 5.1.** *If $\xi(\mathbf{x})$ is a polynomial of $\mathbf{x}$ with degree n, then its m-th moment can be computed as the weighted sum of up to mn-th moments of $\mathbf{x}$.*

Proof. Let $\mathbf{x} = [x^1, x^2, \ldots, x^{n_x}]^T$ and $\xi(\mathbf{x}) = [\xi^1, \xi^2, \ldots, \xi^{n_\xi}]^T$. Since $\xi(\mathbf{x})$ is a polynomial of $\mathbf{x}$ with degree n, for its k-th element ξ^k, there exists a multi-index $\mathcal{A} = (\alpha^1, \ldots, \alpha^{n_x}) \in \mathbb{N}_0^{n_x}$ with $\alpha^1 + \cdots + \alpha^{n_x} \le n$, and coefficients $C_\mathcal{A} = \{c_\alpha \in \mathbb{R} : \alpha \in \mathcal{A}\}$ such that

$$\xi^k = \sum_{\alpha \in \mathcal{A}} c_\alpha \mathbf{x}^\alpha.$$

Furthermore, the product of polynomials is also a polynomial. Hence, for the product of m elements of $\xi(\mathbf{x})$, that is $\xi^{k_1} \cdots \xi^{k_m}$, there exists a corresponding multi-index $\mathcal{A}_m = (\alpha_m^1, \ldots, \alpha_m^{n_x}) \in \mathbb{N}_0^{n_x}$ with $\alpha_m^1 + \cdots + \alpha_m^{n_x} \le mn$, and coefficients $C_{\mathcal{A}_m} = \{c_{\alpha_m} \in \mathbb{R} : \alpha_m \in \mathcal{A}_m\}$ such that

$$\xi^{k_1} \cdots \xi^{k_m} = \sum_{\alpha_m \in \mathcal{A}_m} c_{\alpha_m} \mathbf{x}^{\alpha_m}.$$

By applying the linearity of expectation, we can have the m-th moment of $\xi(\mathbf{x})$ as follows

$$\mathbb{E}[\xi^{k_1} \cdots \xi^{k_m}] = \sum_{\alpha_m \in \mathcal{A}_m} c_{\alpha_m} \mathbb{E}[\mathbf{x}^{\alpha_m}]. \tag{5.19}$$

Note that $\mathbb{E}[\mathbf{x}^{\alpha_m}] = \mathbb{E}[\prod_{k=1}^{n_x}(x^k)^{\alpha_m^k}]$ which represents the up to mn-th moment of $\mathbf{x}$. Hence, according to Eq. (5.19) the m-th moment $\xi(\mathbf{x})$ is a weighted sum of up to mn-th moments of $\mathbf{x}$. This completes the proof. □

In the case where $\xi(\mathbf{x})$ is not a polynomial of $\mathbf{x}$, we can first find a polynomial approximation of it by applying the Taylor's theorem. Particularly, we fit a quadratic polynomial of each k-th element $\xi^k(\mathbf{x})$ at the mean of $\mathbf{x}$, i.e. $\hat{\mathbf{x}}$, as follows:

$$\xi^k(\mathbf{x}) = \xi^k(\hat{\mathbf{x}}) + J_k(\hat{\mathbf{x}})\Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}^T H_k(\hat{\mathbf{x}})\Delta\mathbf{x},$$

where $\Delta\mathbf{x} = \mathbf{x} - \hat{\mathbf{x}}$, $J_k(\hat{\mathbf{x}})$ is the gradient (Jacobian) of ξ^k at $\hat{\mathbf{x}}$, and $H_k(\hat{\mathbf{x}})$ is the Hessian matrix at $\hat{\mathbf{x}}$.

Thus, according to Lemma 5.1, the m-th moments of $\xi(\mathbf{x})$ can be computed as weighted sums of up to $2m$-th moments of $\Delta\mathbf{x}$. Particularly, the first and second moments of $\xi(\mathbf{x})$ are as follows,

$$\begin{aligned}
\mathbb{E}[\xi^k] &= \xi^k(\hat{\mathbf{x}}) + \frac{1}{2}\sum_{l_2=1}^{n_x}\sum_{l_1=1}^{n_x} H_k^{l_1 l_2}(\hat{\mathbf{x}})\mathbb{E}[\Delta\mathbf{x}^{l_1}\Delta\mathbf{x}^{l_2}],\\
\mathbb{E}[\xi^{k_1}\xi^{k_2}] &= \xi^{k_1}(\hat{\mathbf{x}})\xi^{k_2}(\hat{\mathbf{x}}) + \sum_{l_2=1}^{n_x}\sum_{l_1=1}^{n_x}(J_{k_1}^{l_1}J_{k_2}^{l_2} + \frac{1}{2}\xi^{k_1}(\hat{\mathbf{x}})H_{k_2}^{l_1 l_2}(\hat{\mathbf{x}}) + \frac{1}{2}\xi^{k_2}(\hat{\mathbf{x}})H_{k_1}^{l_1 l_2}(\hat{\mathbf{x}}))\mathbb{E}[\Delta\mathbf{x}^{l_1}\Delta\mathbf{x}^{l_2}]\\
&\quad + \frac{1}{2}\sum_{l_3=1}^{n_x}\sum_{l_2=1}^{n_x}\sum_{l_1=1}^{n_x}(J_{k_1}^{l_1}(\hat{\mathbf{x}})H_{k_2}^{l_1 l_2}(\hat{\mathbf{x}}) + J_{k_2}^{l_1}(\hat{\mathbf{x}})H_{k_1}^{l_1 l_2}(\hat{\mathbf{x}}))\mathbb{E}[\Delta\mathbf{x}^{l_1}\Delta\mathbf{x}^{l_2}\Delta\mathbf{x}^{l_3}]\\
&\quad + \frac{1}{4}\sum_{l_4=1}^{n_x}\sum_{l_3=1}^{n_x}\sum_{l_2=1}^{n_x}\sum_{l_1=1}^{n_x} H_{k_1}^{l_1 l_2}H_{k_2}^{l_3 l_4}(\hat{\mathbf{x}})\mathbb{E}[\Delta\mathbf{x}^{l_1}\Delta\mathbf{x}^{l_2}\Delta\mathbf{x}^{l_3}\Delta\mathbf{x}^{l_4}].
\end{aligned} \tag{5.20}$$

5.4 Results

In this section, we evaluate the proposed CC-SBC method in a simulated multi-robot system, and present the simulation results.

5.4.1 Derivation of quadratic constraints

We consider multi-robot systems with single-integrator dynamics, in which the robot dynamics are $\dot{\mathbf{p}}_i = \mathbf{u}_i$, $\forall i \in \mathcal{I}$. Hence, corresponding to Eq. (5.1), we have $\mathbf{x}_i = \mathbf{p}_i \in \mathbb{R}^d$, $\mathbf{u}_i = \mathbf{v}_i \in \mathbb{R}^d$, $\mathbf{f} = \mathbf{0}^d$, $\mathbf{g} = I^{d\times d}$. The level set function to define safety sets of single-integrator systems is [115]

$$h(\mathbf{x}_{ij}) = \|\mathbf{p}_i - \mathbf{p}_j\|^2 - d_s^2. \tag{5.21}$$

Thus corresponding to Eq. (5.10), there are

$$\begin{aligned}
A_{ij}(\mathbf{x}_{ij}) &= -2\|\mathbf{p}_i - \mathbf{p}_j\|^T,\\
b_{ij}(\mathbf{x}_{ij}) &= \gamma(\|\mathbf{p}_i - \mathbf{p}_j\|^2 - d_s^2).
\end{aligned} \tag{5.22}$$

Here we adopt the particular function $\kappa(h(\mathbf{x}_{ij})) = \gamma h(\mathbf{x}_{ij})$ with $\gamma > 0$.

Consider the robots' states (position) are estimated and described by Gaussian distributions, i.e. $\mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$, $\forall i \in \mathcal{I}$. Then, there is $\mathbf{x}_{ij} = \mathbf{p}_i - \mathbf{p}_j \sim \mathcal{N}(\hat{\mathbf{p}}_{ij}, \Sigma_{ij})$ where $\hat{\mathbf{p}}_{ij} = \hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j$ and $\Sigma_{ij} = \Sigma_i + \Sigma_j$. By applying the approach developed in Section 5.3.3, we can obtain the parameters of the quadratic constraints for probabilistic safe-critical control of the system, in an exact closed form as follows:

$$\begin{aligned} A' &= -2\hat{\mathbf{p}}_{ij}^T, \\ b' &= \frac{1}{2}\gamma(\hat{\mathbf{p}}_{ij}^T\hat{\mathbf{p}}_{ij} - d_s^2 + \mathrm{tr}(\Sigma_{ij})). \end{aligned} \tag{5.23}$$

where $\mathrm{tr}(\cdot)$ indicates the trace of a matrix. The k-th row and k'-th column of the $P \in \mathbb{R}^{d\times d}$ is

$$P^{kk'} = 8[\sqrt{1-\delta}\Sigma_{ij}^{kk'} - (1-\sqrt{1-\delta})\hat{\mathbf{p}}_{ij}^k\hat{\mathbf{p}}_{ij}^{k'}]. \tag{5.24}$$

The k-the element of $\mathbf{q} \in \mathbb{R}^d$ is

$$\begin{aligned} \mathbf{q}^k = &\frac{1}{2}\gamma[2(\sqrt{1-\delta}\hat{\mathbf{p}}_{ij}^T\Sigma_{ij}^{k\cdot} \\ &-(1-\sqrt{1-\delta})\hat{\mathbf{p}}_{ij}^k(\hat{\mathbf{p}}_{ij}^T\hat{\mathbf{p}}_{ij} + \mathrm{tr}(\Sigma_{ij}) - d_s^2)]. \end{aligned} \tag{5.25}$$

And r is given by

$$r = \frac{1}{4}\gamma^2[\sqrt{1-\delta}r' - (1-\sqrt{1-\delta})(\hat{\mathbf{p}}_{ij}^T\hat{\mathbf{p}}_{ij} + \mathrm{tr}(\Sigma_{ij}) - d_s^2)^2], \tag{5.26}$$

where

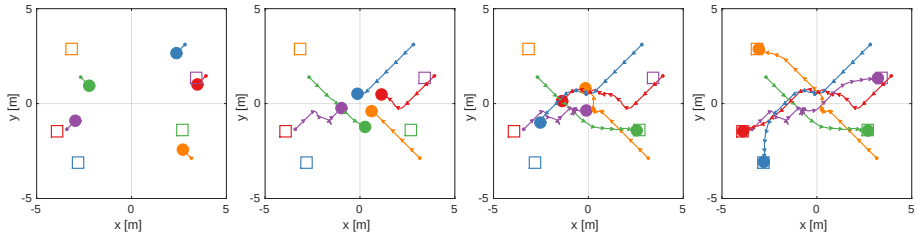
$$r' = 2\Sigma_{ij}^{kl}(2\hat{\mathbf{p}}_{ij}^k\hat{\mathbf{p}}_{ij}^l + 2\Sigma_{ij}^{kl}),$$

with the Einstein summation convention employed $\forall k, l = 1, \ldots, d$.

5.4.2 Simulation results

We test the method with a team of $n = 5$ planar robots navigating in a shared workspace. The challenging *asymmetric swap* scenario [139, 146] is considered, in which the robots are initially distributed around the perimeter of a circle with some random offsets to break symmetry. The goal positions $\mathbf{p}_{i,\mathrm{goal}}$ are also placed around the same circle with random offsets. Each robot can measure the positions of itself and other robots with Gaussian noise, and uses a Kalman filter for estimation. The robots have a maximum velocity of 0.1 m/s and uses a simple PD controller $\mathbf{u}_i^* = -k_1(\mathbf{p}_i - \mathbf{p}_{i,\mathrm{goal}})$ as the nominal controller where the gain $k_1 = 1.5$. The inter-robot safe distance is $d_s = 0.6$ m and the collision probability threshold is set as $\delta = 0.05$. The simulation time step is $\Delta t = 0.05$ s, and the simulated measurement noise is zero mean with covariance $\mathrm{diag}(0.1\text{ m}, 0.1\text{ m})^2$, which leads to an average state estimation error of 0.06 m in the simulations.

We run the simulation 100 times and compare our CC-SBC method with the basic SBC method [82]. Simulation results show that collision happens in all runs with the basic SBC due to the uncertainty in robot state estimation. Instead, our CC-SBC successfully



(a) time step: 10. (b) time step: 60. (c) time step: 120. (d) time step: 200.

Figure 5.1: A sample simulation run of the asymmetric swap scenario with 5 robots. The robot initial and goal locations are marked in circle disks and solid squares.

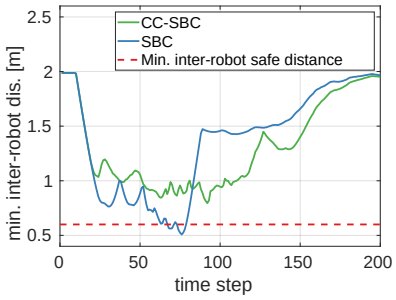


Figure 5.2: History of minimum inter-robot distance among robots.

navigates the robots without collision in every run. Fig. 5.1 shows a series of snapshots of one run of the simulations. Fig. 5.2 shows the minimum inter-robot distance during the run. It can be observed that the minimum distance among robots always stays larger than the safe distance (0.6 m) using our CC-SBC method, while the SBC method leads to collisions.

5.5 Conclusion

In this chapter, we presented a decentralized multi-robot collision avoidance method that accounts for uncertainty in robot state estimation. We formally introduced Chance-Constrained Safety Barrier Certificates (CC-SBC) which defines a probabilistic safe control space for each robot in the system. The CC-SBC chance constraints are reformulated to a set of quadratic constraints of the robot state estimation mean and covariance, which are then used to formulate a quadratically constrained quadratic program (QCQP). By solving the QCQP, the robot can obtain a safe control action that minimally modifies a given nominal controller. In simulations with a team of mobile robots, we showed that our method achieves safe navigation in challenging scenarios under robot state estimation uncertainty, and the inter-robot collision probability is always below a specified threshold thanks that the CC-SBC guarantees forward invariance of the robot's safety set in a probabilistic manner.

6

Learning interaction-aware trajectory predictions for multi-robot motion planning

Parts of this chapter appeared in:

- H. Zhu, F.M. Claramunt, B. Brito, and J. Alonso-Mora, "Learning interaction-aware trajectory predictions for decentralized multi-robot motion planning in dynamic environments," *IEEE Robotics and Automation Letters*, 6(2):2256-2263, Apr. 2021.

6.1 Introduction

In previous chapters, we have shown that while the proposed B-UAVC and CC-SBC methods (Chapter 4 and Chapter 5) are decentralized and communication-free, they typically lead to more conservative robot motions than the MPC based method (Chapter 3). Albeit more efficient, the MPC-based method requires each robot to know the future motion predictions of other robots. These motion predictions can be obtained among robots by sharing their future planned trajectories with each other via communication. However, such communication may not be available nor reliable in practice. Alternatively, the robot can employ a constant velocity model to predict other robots' trajectories. Even though communication among robots is not required in this case, the planned robot motions may not be safe, particularly in crowded environments, as shown in Chapter 3.

In this chapter, to achieve efficient and decentralized multi-robot motion planning, we propose an interaction- and obstacle-aware trajectory prediction model and combine it with the model predictive control (MPC) framework. Fig. 6.1 gives an overview of the proposed method. In particular, we first generate a demonstration dataset consisting of robot trajectories using a multi-robot collision avoidance simulator developed in Chapter 3. It utilizes a centralized sequential MPC for local motion planning in which inter-robot communication is employed. Next, we formulate the robot trajectory prediction problem as a sequence modeling task and hence design a model based on recurrent neural networks (RNN). By training the model using the generated dataset, it learns to imitate the centralized sequential MPC and thus can predict the planning behaviors of the robots. Finally, by combining the trajectory prediction model with the MPC framework, multi-robot local motion planning is achieved in a decentralized manner.

The main contributions of this chapter are:

- A RNN-based interaction- and obstacle-aware model that is able to provide robot trajectory predictions in a multi-robot scenario.
- Incorporation of the trajectory prediction model with MPC to achieve decentralized multi-robot local motion planning in dynamic environments.

We show that our designed model can make accurate trajectory predictions, thanks to which the proposed decentralized multi-robot motion planner can achieve a comparable level of performance to the centralized planner while being communication-free. We also validate our approach with a team of quadrotors in real-world experiments.

This chapter is structured as follows. Section 6.2 describes related work in motion prediction of decision-making agents. Section 6.3 introduces preliminaries with system models. Section 6.4 presents the approach of robot trajectory prediction and its application in decentralized multi-robot motion planning. Section 6.5 shows simulation and experimental results. Finally, Section 6.6 concludes the chapter.

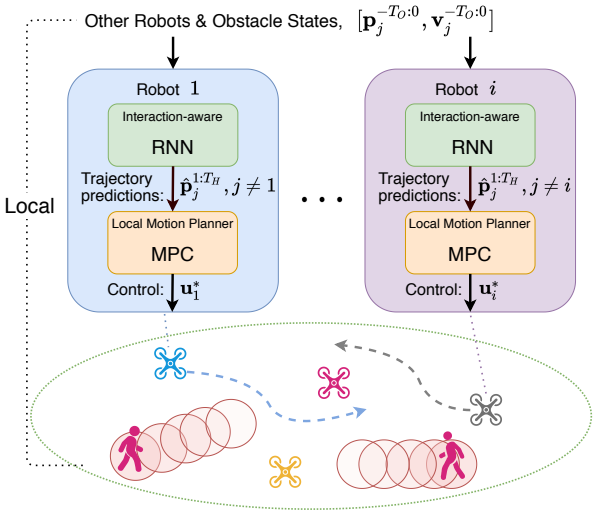


Figure 6.1: The proposed decentralized communication-free motion planner that relies on a RNN-based model for interaction-aware trajectory prediction and a MPC for local motion planning.

6

6.2 Related work

Our proposed approach decouples motion prediction and trajectory planning to achieve decentralized and communication-free collision avoidance. Such a decoupling is also seen in [147, 148], where the motion prediction of humans are used to plan a safe trajectory for the ego robot. Motion prediction for decision-making agents has drawn significant research efforts over the past years, with most works focusing on human trajectory prediction [149]. Early works on motion prediction are typically model-based such as the renowned social force-based method [150] which models pedestrian behaviors through the use of attractive and repulsive potentials. The model is later generalized and adapted to modeling traffic car behaviors [151]. While these methods are computationally efficient, the prediction accuracy is quite low. There have also been several notable attempts to utilize game theory to model interacting decision-making agents and predict their future trajectories [152, 153], in which the agents are assumed to play a non-cooperative game and their trajectory predictions can be obtained from computing the Nash equilibria of the game. A more sophisticated approach is presented in [94], where game theory is combined with a psychology concept called Social Value Orientation (SVO) in order to quantify autonomous cars' degree of selfishness or altruism when predicting their trajectories. While interaction-aware trajectory predictions can be obtained, these methods are limited to specific road scenarios and cannot be directly applied to general multi-robot systems.

The class of approaches that have achieved state-of-the-art performance in trajectory prediction problems are the learning-based methods. Some of these include inverse reinforcement learning (IRL) [154], recurrent neural networks (RNN) [155, 156], variational

autoencoders [157], generative adversarial networks (GAN) [158] that provide predicted human trajectories in two-dimensional (2D) environments, Gaussian mixture regression (GMR) [159] and Gaussian process regression (GPR) [160] that can predict human actions in three-dimensional (3D) workspaces. Our approach of predicting the trajectories of other robots is based on previous works on human motion prediction since both can be formulated as a sequence modelling problem. In particular, our prediction model is based on RNN, inspired by the works in [161] for interaction-aware pedestrian motion prediction in which static obstacles are considered and represented using a grid map. We adapt the model to predict robots trajectories in multi-robot scenarios with moving obstacles described by their positions and velocities, and further apply the model to decentralized multi-robot motion planning by incorporating it within MPC.

6.3 Preliminaries

6.3.1 Robot and obstacle model

Following [49], we consider a team of n robots moving in a shared workspace $\mathcal{W} \subseteq \mathbb{R}^3$, where each robot $i \in \mathcal{I} = \{1, 2, \dots, n\} \subset \mathbb{N}$ is modeled as an enclosing sphere with radius r. The robots follow the same dynamical model that is described by a discrete-time equation as follows,

$$\mathbf{x}_i^{k+1} = \mathbf{f}(\mathbf{x}_i^k, \mathbf{u}_i^k), \quad \mathbf{x}_i^0 = \mathbf{x}_i(0), \tag{6.1}$$

where $\mathbf{x}_i^k \in \mathcal{X} \subset \mathbb{R}^{n_x}$ denotes the state of the robot, typically including its position $\mathbf{p}_i^k$ and velocity $\mathbf{v}_i^k$, and $\mathbf{u}_i^k \in \mathcal{U} \subset \mathbb{R}^{n_u}$ the control inputs at time k. Without loss of generality, $k = 0$ indicates the current time. $\mathcal{X}$ and $\mathcal{U}$ are the admissible state space and control space, respectively. $\mathbf{x}_i(0)$ is the current state of robot i. In addition, moving obstacles for example pedestrians in the environment are considered. For each obstacle $o \in \mathcal{I}_o = \{1, 2, \dots, n_o\} \subset \mathbb{N}$ at position $\mathbf{p}_o \in \mathbb{R}^3$, we model it as an upright non-rotating enclosing *ellipsoid* centered at $\mathbf{p}_o$ with semi-principal axes (a_o, b_o, c_o). In this chapter, we assume that each robot can observe the states (positions and velocities) of all other robots and moving obstacles and keep their history information.

6.3.2 Multi-robot collision avoidance

Multi-robot local motion planning is considered in this chapter, in which the goal is to achieve real-time collision-free navigation for multiple robots. Each robot has a given goal location $\mathbf{g}_i$, which generally comes from some high-level path planner [105] or is specified by the user. Any pair of robots i and j from the group are mutually collision-free if $\left\|\mathbf{p}_i^k - \mathbf{p}_j^k\right\| \geq 2r, \forall i \neq j \in \mathcal{I}, k = 0, 1, \dots$. Regarding robot-obstacle collision avoidance, we approximate the obstacle with an enlarged ellipsoid and check if the robot's position is inside it. Hence, the robot i is collision-free with the obstacle o at time step k if $\left\|\mathbf{p}_i^k - \mathbf{p}_o^k\right\|_\Omega \geq 1$, where $\Omega = \mathrm{diag}(1/(a_o + r)^2, 1/(b_o + r)^2, 1/(c_o + r)^2)$.

The objective is to compute a local motion $\mathbf{u}_i^k$ for each robot in the group, that respects its dynamic constraints, makes progress towards its goal location $\mathbf{g}_i$ and is collision-free

with other robots in the group as well as moving obstacles within a planning time horizon.

6.3.3 Model predictive control

The multi-robot collision avoidance problem can be solved using model predictive control by formulating a receding horizon constrained optimization problem. For each robot $i \in \mathcal{I}$, a discrete-time constrained optimization formulation with N time steps and planning horizon $N\Delta t$, where Δt is the sampling time, is derived as follows,

$$\min_{\substack{\mathbf{x}_i^{0:N}, \mathbf{u}_i^{0:N-1}, \\ s^{0:N}}} \quad \sum_{k=0}^{N-1} J_i^k(\mathbf{x}_i^k, \mathbf{u}_i^k, s^k) + J_i^N(\mathbf{x}_i^N, \mathbf{g}_i, s^N) \tag{6.2a}$$

$$\text{s.t.} \quad \mathbf{x}_i^0 = \mathbf{x}_i(0), \tag{6.2b}$$

$$\mathbf{x}_i^k = \mathbf{f}(\mathbf{x}_i^{k-1}, \mathbf{u}_i^{k-1}), \tag{6.2c}$$

$$\left\| \mathbf{p}_i^k - \mathbf{p}_j^k \right\| \geq 2r - s^k, \tag{6.2d}$$

$$\left\| \mathbf{p}_i^k - \mathbf{p}_o^k \right\|_\Omega \geq 1 - s^k, \tag{6.2e}$$

$$s^k \geq 0, \mathbf{u}_i^{k-1} \in \mathcal{U}, \mathbf{x}_i^k \in \mathcal{X}, \tag{6.2f}$$

$$\forall j \neq i \in \mathcal{I}; \forall o \in \mathcal{I}_o; \forall k \in \{1, \dots, N\}, \tag{6.2g}$$

where $J_i^k(\mathbf{x}_i^k, \mathbf{u}_i^k, s^k)$ and $J_i^N(\mathbf{x}_i^N, \mathbf{g}_i, s^N)$ are the stage and terminal costs respectively [49], and s is the slack variable. At each time step, each robot in the team solves online the constrained optimization problem (6.2) and then executes the first step control inputs, in a receding horizon fashion.

Note that for each robot to solve the optimization problem (6.2), it has to know the future trajectories of other robots and moving obstacles, as shown in Eq. (6.2d) and Eq. (6.2e). For moving obstacles (pedestrians), we assume their motions follow a constant velocity model (CVM) in the short planning horizon and predict their future trajectories accordingly. This assumption is reasonable since CVM can achieve state-of-the-art performance when used for pedestrian motion prediction [162]. For robots' future trajectories, denote by $\mathcal{T}_i^0 = \{\mathbf{p}_i^{1:N}\}$ the robot i's current planned trajectory. Further denote by $\hat{\mathcal{T}}_{i,j}^0 = \{\mathbf{p}_j^{1:N}\}$ the trajectory of robot $j \in \mathcal{I}, j \neq i$ that robot i assumes and uses in solving the problem (6.2), where the hat $\hat{\cdot}$ indicates that it is robot i's estimation of the other robot's trajectory.

Typically, there are two ways for robot i to obtain the future trajectory of the other robot j. The first way is via communication: all robots in the team communicate their planned trajectories to each other at each time step. It can be implemented using a centralized sequential planning framework as in [49], that is, $\hat{\mathcal{T}}_{i,j}^0 = \mathcal{T}_j^0$. Although this method guarantees collision avoidance by construction, it does not scale well with a large number of robots. Moreover, communication is not always available and reliable in practice.

The other way is without communication. Hence, robot i has to predict another robot j's future trajectory based on its observation of the environment:

$$\hat{\mathcal{T}}_{i,j}^0 = \textbf{prediction}(\mathcal{H}_i^0), \tag{6.3}$$

where $\mathcal{H}_i^0$ is the information that robot i can acquire until current time from its observation. Previous works [49, 84] use a constant velocity model to perform the prediction only based on the other robot's current state, that is, $\mathcal{H}_i^0 = \mathbf{x}_j^0$. However, such a prediction can be inaccurate and may lead to unsafe trajectory planning [49]. In this chapter, we will develop an interaction- and obstacle-aware model for the trajectory prediction taking into account surrounding environment information of the robot to model the interaction and environment constraints.

6.4 Approach

In this section, we present our interaction and obstacle-aware trajectory prediction method and incorporate it with the MPC framework to achieve decentralized multi-robot collision avoidance in dynamic environments.

6.4.1 Trajectory prediction problem formulation

As shown in Eq. (6.3), robot $i \in \mathcal{I}$ needs to predict the future trajectories of other robots $j \neq i \in \mathcal{I}$ to plan its safe motion. Hereafter, we refer to the robot i as the ego robot and the robot j as the *query robot* that is indicated by the sub-script $\cdot_q$. In addition, we use the sub-script $\cdot_{-q}$ to indicate the collection of all the other robots except for the query robot.

We aim to address the problem of finding a trajectory prediction model for the query robot q that gives a sequence of its future positions $\mathbf{p}_q^{1:T_H}$ in a multi-robot scenario. Here $T_H \geq N$ is the prediction horizon that should not be smaller than the local motion planning horizon. As has been shown in previous trajectory prediction works [156, 161], we will instead work with sequences of velocities $\mathbf{v}_q^{1:T_H}$ for prediction to avoid overfitting when based on position sequences, and numerically integrate them afterwards starting from the query robot's current position $\mathbf{p}_q^0$.

Denote by $\mathbf{v}_q^{-T_O:0}$ the past sequence of velocities of the query robot within an observation time $T_O \geq 1$. Denote by $\mathbf{p}_{-q,r}^{-T_O:0}$ and $\mathbf{v}_{-q,r}^{-T_O:0}$ the past relative positions and velocities of other robots with respect to the query robot. Further denote by $\mathbf{p}_{\mathcal{I}_o,r}^0$ and $\mathbf{v}_{\mathcal{I}_o,r}^0$ the current relative positions and velocities of the moving obstacles $o \in \mathcal{I}_o$ with respect to the query robot. By observing history states of the query robot and its surrounding other robots as well as moving obstacles, we want to find an interaction- and obstacle-aware model $\mathbf{h}_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$:

$$\mathbf{v}_q^{1:T_H} = \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{v}_q^{-T_O:0}, \mathbf{p}_{-q,r}^{-T_O:0}, \mathbf{v}_{-q,r}^{-T_O:0}, \mathbf{p}_{\mathcal{I}_o,r}^0, \mathbf{v}_{\mathcal{I}_o,r}^0), \tag{6.4}$$

that outputs a prediction of the query robot's future states.

6.4.2 Demonstration data generation

We use a simulation dataset to train our designed network model. The dataset is generated using demonstrations from a multi-robot collision avoidance simulator [49] which employs a centralized sequential planner to solve the problem (6.2). This involves each robot solving

a MPC problem sequentially and communicates its planned trajectory to other robots to avoid. Note that the planner differs from the prioritized planning approach since each robot has to avoid all other robots and hence it shows cooperation among robots.

Specifically, we create a three-dimensional environment in which a team of robots and moving obstacles are simulated. In the simulation, each robot navigates to a randomly generated goal position, which is changed dynamically to a new location after being reached. The generated robots' goal positions are ensured to be collision-free with each other and the obstacles. Moving obstacles are simulated in the environment by randomly specifying an initial position and velocity (with speed between 0.5 m/s and 1.2 m/s) to each of them and then make them move at a constant velocity. Once any obstacle moves out of the environment, a new initial position and velocity will be set to it. Moreover, we add small Gaussian noise to the velocities of the moving obstacles in simulation. We perform the simulation for N_{sim} time steps and record the positions and velocities of all robots and obstacles at each time step. After running the simulation, for each time step k and robot q, we retrieve its future sequence of velocities and observation of the past states of the system from the recorded data. Hence, our dataset is as follows

$$\mathcal{D} = \{(\mathcal{H}_q^k, \mathbf{v}_q^{k+1:k+T_H}) \mid \forall q \in \mathcal{I}, \forall k \in \{1, \dots, N_{\text{sim}-T_H}\}\}, \tag{6.5}$$

where the observation information $\mathcal{H}_q^k$ is

$$\mathcal{H}_i^k = \{\mathbf{v}_q^{k-T_O:k}, \mathbf{p}_{-q,r}^{k-T_O:k}, \mathbf{v}_{-q,r}^{k-T_O:k}, \mathbf{p}_{\mathcal{I}_o,r}^k, \mathbf{v}_{\mathcal{I}_o,r}^k\}. \tag{6.6}$$

6.4.3 Interaction- and obstacle-aware model

We now present our recurrent neural network (RNN) model for interaction- and obstacle-aware trajectory prediction, as shown in Fig. 6.2. The model first creates a joint representation of three input channels: the query robot's history state, information of other interacting robots and moving obstacles, via a query robot state encoder and an environment encoder module. Then a decoder module is adopted to output a predicted trajectory of the query robot. The recurrent layers in the model are of the LSTM type [163] that has been shown able to learn time dependencies over a long period of time. Next, we describe the three main modules of the model in detail.

Query robot state encoder

It consists of a recurrent layer that produces a flat encoding $\mathbf{z}_q^0$ from the history velocities of the query robot $\mathbf{v}_q^{-T_O:0}$. This layer learns a dynamical model of the query robot, so that the network can leverage it to obtain better predictions.

Environment encoder module

It includes $n-1$ parallel recurrent layers with shared weights to encode the sequences of past relative positions and velocities of other robots with respect to the query robot $(\mathbf{p}_{-q,r}^{-T_O:0}, \mathbf{v}_{-q,r}^{-T_O:0})$ into a set $\mathcal{Z}_{-q,r}^0$, and n_o parallel dense layers with shared weights that encode the current relative positions and velocities of moving obstacles with respect to

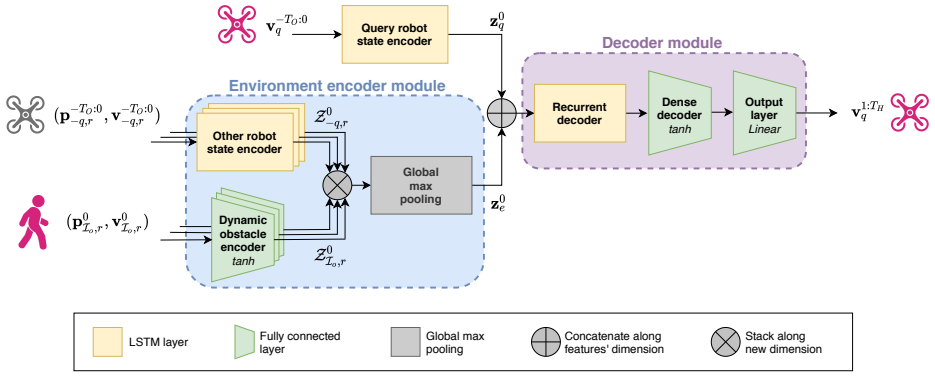


Figure 6.2: Network architecture of the interaction- and obstacle-aware model. Three channels of information are taken as inputs: the query robot's past velocities $\mathbf{v}_q^{-T_O:0}$, past relative states of other robots $(\mathbf{p}_{-q,r}^{-T_O:0}, \mathbf{v}_{-q,r}^{-T_O:0})$ and current relative states of obstacles $(\mathbf{p}_{\mathcal{I}_o,r}^0, \mathbf{v}_{\mathcal{I}_o,r}^0)$. A joint representation of the inputs is created through a query robot encoder and an environment encoder. A decoder is adopted to output a sequence of velocities $\mathbf{v}_q^{1:T_H}$ predicted for the query robot's future trajectory.

6

the query robot $(\mathbf{p}_{\mathcal{I}_o,r}^0, \mathbf{v}_{\mathcal{I}_o,r}^0)$ into a set $\mathcal{Z}_{\mathcal{I}_o,r}^0$. The encodings from both of these sets, which are made to have the same length, are then stacked together and followed by a global max pooling operation executed along the new data axis. Thus, this module can capture the interaction of the query robot with a variable number of other robots and obstacles in the environment and encode it into a single flat vector $\mathbf{z}_e^0$. This framework also makes it possible to account for potentially different types of agents and obstacles by training their own set of encoders and stacking them with the rest of intermediate encodings.

Decoder module

It takes in the concatenation of $\mathbf{z}_q^0$ with $\mathbf{z}_e^0$ and passes it through a recurrent decoder followed by a dense decoder and an output layer that finally generates a sequence of predicted future velocities $\mathbf{v}_q^{1:T_H}$ for the query robot over the prediction horizon.

6.4.4 Model training

Using the generated demonstration data in Section 6.4.2, the designed model is trained end-to-end using back-propagation through time (BTTP) [164] with a fixed truncation depth t_{trunc}. We learn the trajectory prediction model by minimizing the following loss function,

$$L(\mathbf{v}_q^{1:T_H}, \boldsymbol{\theta}) = \frac{1}{T_H} \sum_{k=1}^{T_H} \left\| \mathbf{v}_q^k - \mathbf{v}_{q,\text{true}}^k \right\|^2 + \lambda \cdot l(\boldsymbol{\theta}), \tag{6.7}$$

where $\mathbf{v}_{q,\text{true}}^k$ is the ground truth velocity from the demonstration dataset, $l(\boldsymbol{\theta})$ represents the regularization terms and λ is the regularization factor. In our model, the $L2$ regularization

method is adopted.

6.4.5 Decentralized multi-robot motion planning

Having the trained trajectory prediction model, we can incorporate it with the MPC framework and solve the problem (6.2) in a decentralized manner. As shown in Fig. 6.1, in a multi-robot navigation scenario, each robot first performs inference with the trained neutral network to predict the future trajectories of its neighboring robots and then plans a collision-free trajectory accordingly. Hence, decentralized multi-robot motion planning in dynamic environments is achieved. To be able to perform the inference, each robot needs to measure its own state as well as its neighbors', and keep a history memory of the information for a time horizon T_O. In addition, the robot also needs to measure the current states of moving obstacles in the environment.

6.5 Results

We now present results of simulation comparing the proposed approach with other methods and real-world experiments with quadrotors.

6.5.1 Implementation details

To generate the dataset, we use an existing MATLAB multi-robot collision avoidance simulator[1] developed in Chapter 3 and simulate $N_{\text{sim}} = 10^5$ time steps in a $10 \times 10 \times 3$ m environment with 10 robots and 10 moving obstacles. The robot we simulate is the Parrot Bebop 2 quadrotor with a radius set as 0.4 m. Ellipsoids representing the moving obstacles have semi-axes $(0.4, 0.4, 0.9)$ m. The sampling time and MPC planning horizon length are $\Delta t = 0.05$ s and $N = 20$, respectively. We employ the same dynamics model and cost functions in the MPC problem (2) of our previous work [49]. The Forces Pro solver [127] is used to solve the MPC problem. We set $T_O = 20$ and $T_H = 20$ the horizon length for robot past states observation and trajectory prediction. We further generate another test dataset by running the simulator in six different scenarios for 2×10^4 time steps for each one of them. All computations are performed in a commodity computer with an Intel i7 CPU@2.60GHz and an NVIDIA GTX 1060 GPU.

The designed learning network is implemented in Python using TensorFlow 2. All layers in the network have 64 neurons except for the recurrent decoder that has 128 neurons and the output layer that has 3 neurons. While the activation function of the output layer is linear, all other layers in the network use a hyperbolic activation function. The regularization factor used during model training is $\lambda = 0.01$.

6.5.2 Trajectory prediction evaluation

We first evaluate our trajectory prediction model on a test dataset that has not been used for training nor validation. The dataset includes different test scenarios: an open environment with 4, 10, and 20 quadrotors, and with 10 moving obstacles. We compare our interaction-

[1] Code: https://github.com/tud-amr/mrca-mav

aware RNN-based model to three alternative methods: a) the constant velocity model (CVM) that is widely used in decentralized multi-robot motion planning; b) a simple RNN model that only considers the query robot's past states for trajectory prediction while ignoring its surrounding environment (this allows us to highlight the interaction awareness of our designed model); and c) an open-loop MPC planner assuming that the goal, robot model and constraints are known.

In Fig. 6.3 we present quantitative results of the prediction error with respect to ground truth in the test dataset. Recall that ground truths are the recorded robot traveled trajectories computed with the centralized sequential MPC (closed-loop). As expected, the prediction error of the open-loop MPC has the smallest prediction error among the methods since it was used for data generation and has perfect knowledge about the goal locations of all robots, which are not available for prediction in our proposed RNN-based model. Our proposed model can still achieve accurate trajectory predictions and significantly outperforms the CVM method across all scenarios. Moreover, compared to the simple RNN model, our interaction-aware approach achieves more accurate trajectory predictions, particularly in cluttered scenarios where interactions among robots are more frequent, as shown in Fig. (6.3b)-(6.3f). Furthermore, to evaluate the generalization capability of the learned network, we perform simulations in the scenarios (e) and (f) with 20 quadrotors which are beyond our training dataset. The results show that the proposed model still performs well on trajectory prediction in the two scenarios. Corresponding to the six scenarios in the figure, the average computation times to perform motion prediction using our RNN-based model for all other robots are (a) 28 ms, (b) 28 ms, (c) 28 ms, (d) 30 ms, (e) 30 ms, and (f) 32 ms, respectively. This shows that the computation time for motion prediction does not increase much with a larger number of robots.

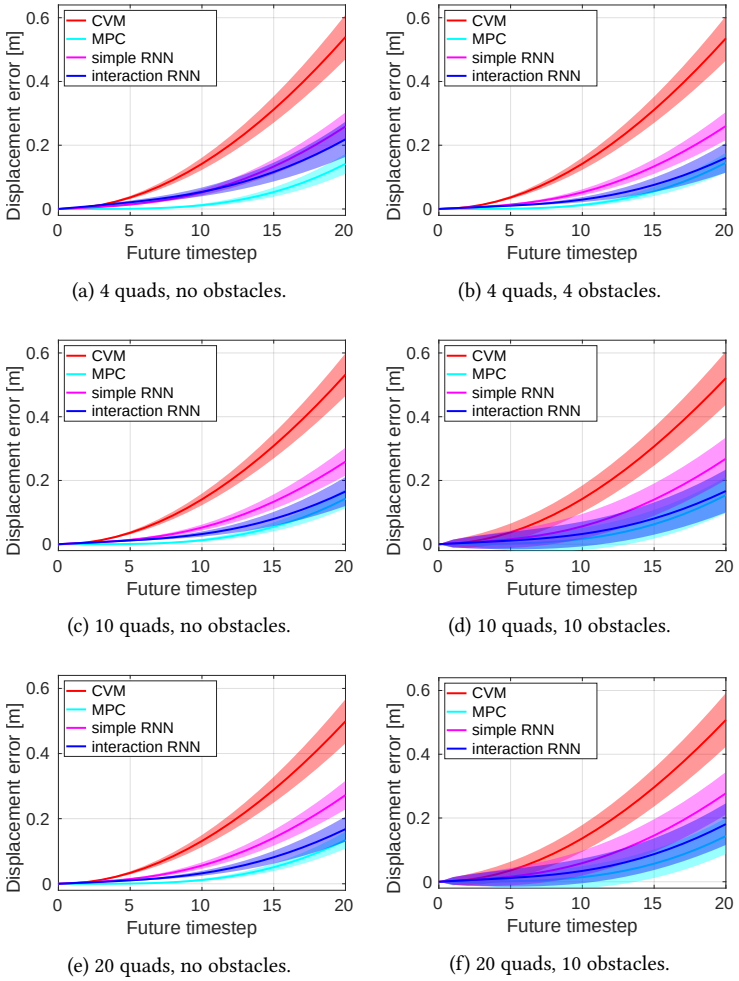
6.5.3 Decentralized motion planning

We then evaluate performance of the proposed decentralized planner that incorporates the learned trajectory prediction model.

Comparisons to other methods

We compare our method to the centralized sequential planning method [49] with full communication among robots and the decentralized planning method [84] that uses the constant velocity model (CVM) for trajectory prediction to analyze whether more accurate trajectory forecasts of our RNN-based model lead to better planning performance. Besides, another decentralized method, the buffered Voronoi cell (BVC) [81], which guarantees collision avoidance is also implemented for comparison.

Six quadrotors flying in four types of scenarios that represents different levels of difficulty [139] are considered. Moreover, in order to avoid potential bias results, each scenario includes 50 instances where the robots have different starting and goal locations. The four scenarios are: *1) symmetric swap*, in which the robots initially located at the vertices of a virtual horizontal regular hexagon are required to exchange their positions; *2) asymmetric swap*, which differs from the previous scenario in that the hexagons are irregular, thus leading to more challenging collision-avoidance problems; *3) pair-wise swap*,



(a) 4 quads, no obstacles.

(b) 4 quads, 4 obstacles.

(c) 10 quads, no obstacles.

(d) 10 quads, 10 obstacles.

(e) 20 quads, no obstacles.

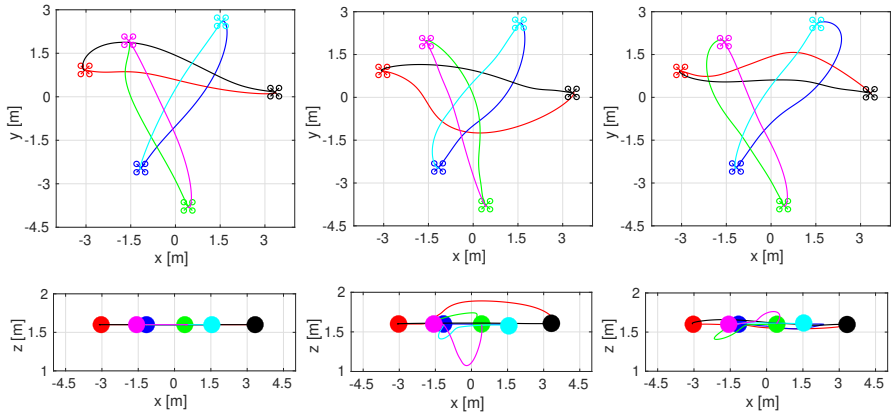
(f) 20 quads, 10 obstacles.

Figure 6.3: Performance results of our proposed interaction-aware RNN model for trajectory prediction compared to the baselines. The solid lines represent the average errors along the prediction horizon and the filled patches around them are 30% of the standard deviation. The sampling period is 50 ms and the prediction horizon has 20 timesteps.

in which the robots are placed at random starting positions and assigned to three pairs within which the two robots need to swap their positions; and *4) random moving*, in which each robot moves from a random starting position to a random goal in the environment.

Qualitatively, Fig. 6.4 shows the sample trajectory trails of the six quadrotors for one instance from the asymmetric swap scenario. It can be seen that our RNN-based decentralized planner achieves results that are closer to the centralized sequential planner than the CVM-based planner. To quantitatively evaluate the performance of different

motion planners, we consider a wide range of metrics: the number of instances that lead to collisions within the entire 50 runs, the average trajectory length and trajectory duration of the team of robots, and the overall robot average speed during the whole simulation. The last three metrics are only computed for those successful runs. Table 6.1 summaries the simulation results. It can be seen that our RNN-based planner significantly outperforms the planner using the CVM for trajectory prediction in terms of safety, in particular in the challenging asymmetric swap scenario. In addition, our planner also achieves consistently smaller trajectory lengths and durations compared to the CVM-based planner in all scenarios. Compared to the BVC method, our proposed approach achieves significantly shorter trajectory durations, particularly in the (a)symmetric swapping scenarios, which shows superiority of the MPC framework over the reactive BVC method. Finally, compared to the centralized sequential planner with full communication, our planner can achieve a comparable level of performance in terms of safety and trajectory efficiency while being decentralized and communication-free. However, three instances out of 50 in the challenging asymmetric swap scenario is still observed with collisions using the RNN-based method, indicating that in few rare cases, highly-accurate trajectory predictions of other robots, for example obtained via communication, are necessary to ensure safety. In the simulation, on average the computation time of the proposed decentralized MPC planner with the learned predictor is 36.3 ms, which is smaller than that of the centralized sequential planner which plans trajectories for all six robots (43.9 ms). Besides, our decentralized approach is communication-free.



(a) Cen. Comm. (b) Decen. CVM. (c) Decen. RNN.

Figure 6.4: Simulation results of six quadrotors exchanging positions in the asymmetric swap scenario. Solid lines represent the trajectories. The upper and lower plots show the top view (X-Y) and side view(X-Z), respectively.

Table 6.1: Performance comparison of different multi-robot motion planners (centralized with communication [49], decentralized buffered Voronoi cell (BVC) method [81], decentralized with constant velocity model (CVM) [84] and decentralized with our RNN-based model) across the four different types of scenarios (symmetric swap, asymmetric swap, pair-wise swap and random moving). Each scenario includes 50 running instances.

Scenario	Motion Planner	Num. of coll. instances	Trajectory length (m)			Trajectory duration (s)			Average speed (m/s)
			Min.	Average	Max.	Min.	Average	Max.	
Symmetric swap	Cen. Comm.	0	5.46	7.22±0.85	8.96	5.15	5.62±0.24	6.05	1.27±0.05
	Decen. (BVC)	**0**	5.75	7.63±0.87	9.63	10.00	12.21±1.25	14.80	0.63±0.03
	Decen. (CVM)	4	5.43	7.45±0.88	9.55	4.95	6.02±0.55	7.30	1.23±0.05
	Decen. (RNN)	**0**	5.41	**7.35±0.91**	10.60	4.75	**5.59±0.40**	9.70	**1.30±0.05**
Asymmetric swap	Cen. Comm.	0	5.08	6.77±0.80	9.02	4.65	5.17±0.30	5.85	1.30±0.05
	Decen. (BVC)	**0**	5.31	7.25±0.87	9.29	9.70	11.34±1.00	13.90	0.65±0.03
	Decen. (CVM)	15	5.32	7.76±1.89	18.06	5.05	5.96±0.51	7.30	1.28±0.04
	Decen. (RNN)	3	5.16	**7.14±1.10**	12.60	4.80	**5.48±0.42**	6.85	**1.29±0.05**
Pair-wise swap	Cen. Comm.	0	1.64	5.10±1.87	9.92	3.50	4.76±0.44	5.85	1.06±0.12
	Decen. (BVC)	**0**	1.83	5.25±1.93	10.13	5.20	7.58±1.66	13.20	0.67±0.07
	Decen. (CVM)	3	1.74	5.54±2.53	17.42	3.60	5.00±0.65	5.75	**1.06±0.10**
	Decen. (RNN)	**0**	1.70	**4.94±2.02**	9.94	3.40	**4.83±0.45**	5.95	1.01±0.14
Random moving	Cen. Comm.	0	0.39	4.66±1.94	8.63	3.50	7.72±0.40	5.50	0.98±0.13
	Decen. (BVC)	**0**	0.39	4.81±2.00	8.84	4.70	7.13±1.30	10.10	0.69±0.09
	Decen. (CVM)	**0**	0.39	4.82±2.06	9.53	3.60	4.89±0.57	6.35	**0.98±0.12**
	Decen. (RNN)	**0**	0.39	**4.36±2.11**	9.19	3.85	**4.76±0.43**	5.95	0.91±0.11

Effect of non-MPC robots on performance

Our proposed decentralized approach assumes that all robots interact and adopt the same motion planning strategy, namely MPC-based trajectory optimization with the learned motion prediction model. We now evaluate the performance of our approach in a mixture scenario where some robots employ the BVC method [81] for collision avoidance. We simulate 50 instances with six quadrotors in the symmetric swap scenario of Section V-C-1. Table 6.2 presents the simulation results. When there is only one BVC robot, no collisions are observed. However, when more BVC robots are in the team, particularly when half of the robots (3) are BVC-based, collisions will happen due to incorrect motion predictions of them by other MPC robots. This indicates that the assumption that the robots interact with the same planning strategy is necessary to ensure safety.

Table 6.2: Simulation results of six quadrotors in the symmetric swap scenario where a varying number of BVC-based robots are in the team. 50 running instances are simulated.

Num. of BVC robots	0	1	2	3	4
Num. of coll. instan.	0	0	2	4	1
Ave. traj. time (s)	5.59	6.82	8.517	9.63	10.27

6.5.4 Experimental validation

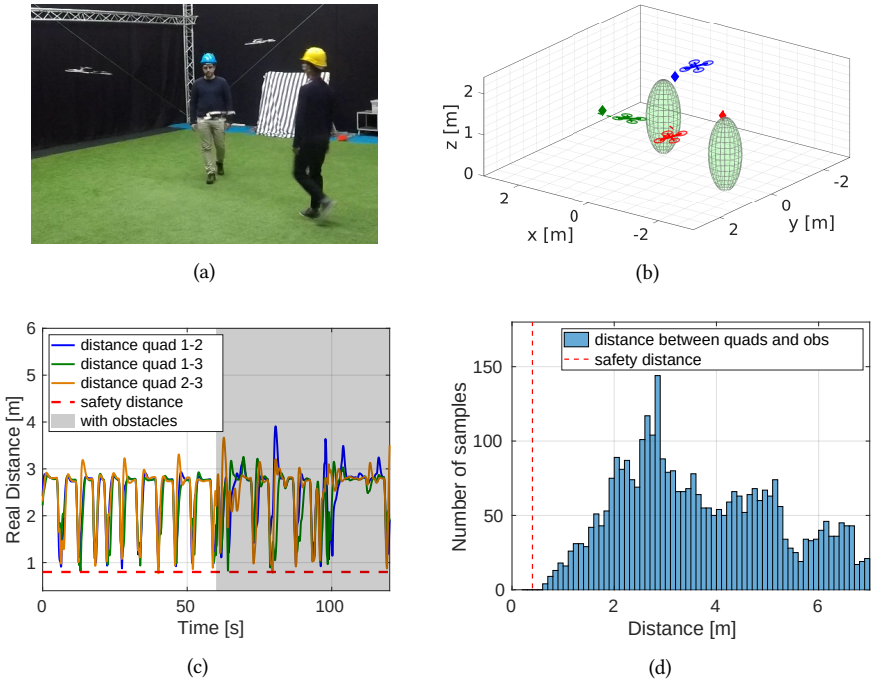
Setup

We validate our proposed approach with a team of Parrot Bebop 2 quadrotors flying in a shared space with walking human obstacles. The pose of each quadrotor and obstacle (human) is obtained using an external motion capture system (OptiTrack) and their velocities obtained via a standard Kalman filter running at a high rate. Control commands are sent to the quadrotor via ROS. During the experiment, the humans walked at a speed with mean 0.8 m/s and the maximum 1.2 m/s. They could change their speeds and make small turns in the workspace.

Results

Experiments in two representative scenarios are conducted: with and without obstacles. In the first scenario, three quadrotors, initially distributed in a virtual horizontal circle, are required to swap their positions multiple times. Then in the second scenario, two moving obstacles (walking humans) join the space while the three quadrotors keeps changing their positions while avoiding the humans at the same time. Fig. 6.5a presents a snapshot from the experiment. Fig. 6.5c shows distance between each pair of the three quadrotors over time during the experiment. It can be seen that they maintained a safe distance of 0.8 m over the entire run even after the two walking humans joined the space which makes it more confined. In Fig. 6.5d we cumulate the distance between each quadrotor and human obstacle that is computed as the closest distance from the quadrotor center to the obstacle ellipsoid's surface. The results show that a minimum safe separation of 0.4 m to the obstacles is achieved. In sum, the demonstration shows that our proposed approach works well for multi-robot motion planning in dynamic environments which is

decentralized and communication-free.



(a) (b) (c) (d)

Figure 6.5: Experimental results with three quadrotors flying in a shared space with two walking humans. (a) A snapshot of the experiment. (b) Schematic of quadrotors, humans, and planned trajectories. (c) Distance between the quadrotors over time. The shaded grey area indicates the two walking humans join the space. (d) Histogram of the quadrotor-obstacle distance during the experiments.

6.6 Conclusion

In this chapter, we presented a decentralized multi-robot MPC-based motion planning approach that accounts for the robot's interactions with obstacles and other robots through the use of a RNN-based trajectory prediction model. We showed that our proposed interaction-aware RNN model generalizes well with different numbers of robots and obstacles, and is able to provide more accurate trajectory predictions than the constant velocity model in a variety of scenarios. In simulations with six quadrotors, we showed that our decentralized planner outperforms the planner using a constant velocity model for trajectory prediction and can achieve a comparable level of performance to the centralized sequential planner while being communication-free. We also validated our approach in real-world experiments with three quadrotors flying in a shared space with walking humans.

7

Conclusions and future work

7.1 Conclusions

This thesis investigated the problem of multi-robot motion planning under uncertainty. In real-world settings, various uncertainties exist in multi-robot systems, such as robot localization and sensing uncertainties, motion disturbances, and uncertain behaviors of other decision-making agents. We showed that these uncertainties can be accounted for in multi-robot motion planning by formulating collision chance constraints, i.e., the collision probability between robots and obstacles to be below a specified risk threshold. In particular, several local motion planning methods that can achieve probabilistic collision avoidance have been developed, which are summarized in the following.

In Chapter 3, a Chance-Constrained Nonlinear Model Predictive Control (CCNMPC) method was presented. The method explicitly formulated chance constraints on the robot collision probability taking into account robot localization, sensing, and motion uncertainties. We showed that the chance constraints can be reformulated into deterministic constraints using a local linearization technique, thus making the MPC tractable and solvable in real time. We adopted the method in an autonomous MAV with onboard vision-based localization and obstacle detection, which was shown to be able to avoid walking humans at a maximum speed of 2.4 m/s. In addition, we incorporated the CCNMPC into three multi-robot planning strategies and compared their performance. It was shown that the sequential and distributed planning strategies in which the robots communicate their future planned trajectories outperform the decentralized strategy in terms of safety where the robots predict their neighbors' trajectories based on the constant velocity model, especially in crowded environments.

Chapter 4 presented a decentralized communication-free algorithm for multi-robot collision avoidance under uncertainty, which relies on the concept of Buffered Uncertainty-Aware Voronoi Cell (B-UAVC). The B-UAVC was defined as a local safe region (position space) for each robot among other robots and obstacles. We showed that the collision probability between robots and obstacles is below a specified threshold if each robot is constrained to navigate within its corresponding B-UAVC. The algorithm was applied to systems with a large number of robots and heterogeneous robot teams, showing robust collision avoidance under uncertainty in robot localization and sensing. Comparing to the CCNMPC method with robots communicating trajectories, the B-UAVC method is communication-free, but it results in more conservative motions of robots and deadlocks of the system.

Chapter 5 presented a method to compute a probabilistic safe control space for each robot in a multi-robot system by formulating Chance-Constrained Safety Barrier Certificates (CC-SBC). The CC-SBC is defined via a set of chance constraints on the robot's control input. We showed that by enforcing the control input of each robot to be within its CC-SBC, the collision probability between robots is guaranteed to be smaller than a specified threshold. In addition, we showed that the CC-SBC constraints can be reformulated into deterministic quadratic constraints, based on which a quadratically constrained quadratic program (QCQP) was formulated. By solving the QCQP, the robot can obtain a safe control action that minimally modifies a given nominal controller. Hence, the CC-SBC method can

be used as a probabilistic safety filter for multi-robot systems.

In Chapter 6, a novel trajectory prediction model based on Recurrent Neural Networks (RNN) that can learn multi-robot motion behaviors from demonstrated trajectories was developed. The model was shown able to run efficiently online and provide more accurate trajectory predictions than the constant velocity model in a variety of scenarios. The model was then incorporated with the MPC framework for decentralized multi-robot motion planning, which was shown to achieve a comparable level of performance to the centralized sequential planner developed in Chapter 3 while being communication-free.

7.2 Future work

Although this thesis has provided a step forward towards probabilistic motion planning for multi-robot systems under uncertainty, many challenges and avenues for future research remain. In the following, we recommend several research avenues in motion planning under uncertainty, in multi-robot coordination, and in extending this work to perceptive motion planning.

Motion planning under uncertainty

Planning under non-Gaussian uncertainty. The work in this thesis assumes that uncertainties are Gaussian-distributed, which have unbounded support. However, this assumption is not always realistic and reasonable since many uncertainties in real-world scenarios are non-Gaussian distributions. For example, autonomous navigation among humans typically requires the prediction of human motions, which may be multi-modal trajectories represented by Gaussian mixture models (GMM) [156]. Hence, motion planning for the robot will need to develop algorithms that can handle non-Gaussian uncertainties.

Risk assessment and risk-aware planning. This thesis addresses the problem of motion planning under uncertainty in a probabilistic manner, in which the key idea is to formulate and approximate collision chance constraints, such that the robot collision probability is below a specified threshold. In fact, chance constraint is closely related to the concept of Value at Risk (VaR), which is one of the commonly used risk metrics. Using the VaR metric or chance constraint may lead to unsafe robot behaviors since it does not capture risk in the tail of cost distributions [165]. Some recent works propose to use the Conditional Value at Risk (CVaR) as a risk metric and have developed CVaR-constrained motion planning algorithms [166]. However, the algorithms are computationally intractable for real-time motion planning. Developing reasonable and efficient risk assessment methods and risk-aware algorithms for online motion planning are still open.

Multi-robot coordination

When, to whom, and what to communicate. The multi-robot coordination strategies employed in this thesis either require the robots to communicate with each other or do not need the robots to communicate at all. The former fully-communication strategy typically has better performance than the communication-free strategy. However, communication

is not always available nor reliable in bandwidth-limited environments. Hence, a potential research avenue is to develop efficient communication strategies for multi-robot coordination, which may include when, to whom, and what to communicate in a multi-robot system. Some recent works have been exploring the topic [139, 167–169] using deep learning or reinforcement learning techniques. But the methods are limited in homogeneous robot teams and in deterministic scenarios. Future research focus can be given to developing communication strategies for coordination among heterogeneous multi-robot systems and taking into account uncertainty.

Integration of global knowledge. The motion planning methods presented in this thesis are local and distributed/decentralized. Hence, deadlocks may happen, particularly in crowded situations. Future research shall explore avoiding deadlocks as well as congestions in multi-robot systems. A promising solution is to integrate global knowledge of the environment and system to guide local motion planning. For example, [99] presents a method to synthesize a local collision avoidance controller with a policy that is learned from a global motion planner, thus improving its performance of avoiding deadlocks. But the method is limited to single and double integrator robots. Developing methods to integrate global knowledge into local motion planning for multi-robot systems is still challenging and open.

Perceptive motion planning

This thesis focuses on multi-robot local motion planning and collision avoidance under uncertainty, in which the objective for the robots is to reach given goal locations while being probabilistically collision-free. Future research shall explore motion planning for higher-level tasks, particularly perception-driven motion planning and information gathering motion planning. Potential research questions include *informative multi-robot motion planning under uncertainty* and *task-oriented multi-robot coordination*. The methods presented in this work can be used as local safety controllers and contributing to the study of the mentioned research topics.

A

Quadrotor dynamics model

We use the Parrot Bebop 2 quadrotor in our experiments. The state of the quadrotor is

$$\mathbf{x} = [\mathbf{p}^T, \mathbf{v}^T, \phi, \theta, \psi]^T \in \mathbb{R}^9,$$

where $\mathbf{p} = [p_x, p_y, p_z]^T \in \mathbb{R}^3$ is the position, $\mathbf{v} = [v_x, v_y, v_z]^T \in \mathbb{R}^3$ the velocity, and ϕ, θ, ψ the roll, pitch and yaw angles of the quadrotor. The control inputs to the quadrotor are

$$\mathbf{u} = [\phi_c, \theta_c, v_{z_c}, \dot{\psi}_c]^T \in \mathbb{R}^4,$$

where ϕ_c and θ_c are commanded roll and pitch angles, v_{z_c} the commanded velocity in vertical z direction, and $\dot{\psi}_c$ the commanded yaw rate.

The dynamics of the quadrotor position and velocity are

$$\begin{aligned}
\dot{\mathbf{p}} &= \mathbf{v}, \\
\begin{bmatrix} \dot{v}_x \\ \dot{v}_y \end{bmatrix} &= R_Z(\psi) \begin{bmatrix} \tan\theta \\ -\tan\phi \end{bmatrix} g - \begin{bmatrix} k_{D_x} v_x \\ k_{D_y} v_y \end{bmatrix}, \\
\dot{v}_z &= \frac{1}{\tau_{v_z}} (k_{v_z} v_{z_c} - v_z),
\end{aligned}$$

where $g = 9.81$ m/s^2 is the Earth's gravity, $R_Z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix}$ is the rotation matrix along the z-body axis, k_{D_x} and k_{D_y} the drag coefficient, k_{v_z} and τ_{v_z} the gain and time constant of vertical velocity control.

The attitude dynamics of the quadrotor are

$$\begin{aligned}
\dot{\phi} &= \frac{1}{\tau_\phi} (k_\phi \phi_c - \phi), \\
\dot{\theta} &= \frac{1}{\tau_\theta} (k_\theta \theta_c - \theta), \\
\dot{\psi} &= \dot{\psi}_c,
\end{aligned}$$

A

where k_ϕ, k_θ and τ_ϕ, τ_θ are the gains and time constants of roll and pitch angles control respectively.

B

Procedure to compute the best linear separator

The procedure to compute the best linear separator between two gaussian distributions [135] is summarized in the following.

The objective is to solve the following minimax problem:

$$(\mathbf{a}_{ij}, b_{ij}) = \arg \min_{\mathbf{a}_{ij}\in\mathbb{R}^d, b_{ij}\in\mathbb{R}} \max (\Pr_i, \Pr_j),$$

where

$$\Pr_i(\mathbf{a}_{ij}^T\mathbf{p} > b_{ij}) = 1 - \Phi((b_{ij} - \mathbf{a}_{ij}^T\hat{\mathbf{p}}_i)/\sqrt{\mathbf{a}_{ij}^T\Sigma_i\mathbf{a}_{ij}}),$$
$$\Pr_j(\mathbf{a}_{ij}^T\mathbf{p} \leq b_{ij}) = 1 - \Phi((\mathbf{a}_{ij}^T\hat{\mathbf{p}}_j - b_{ij})/\sqrt{\mathbf{a}_{ij}^T\Sigma_j\mathbf{a}_{ij}}).$$

Let $u_1 = \frac{b_{ij}-\mathbf{a}_{ij}^T\hat{\mathbf{p}}_i}{\sqrt{\mathbf{a}_{ij}^T\Sigma_i\mathbf{a}_{ij}}}$, $u_2 = \frac{\mathbf{a}_{ij}^T\hat{\mathbf{p}}_j-b_{ij}}{\sqrt{\mathbf{a}_{ij}^T\Sigma_j\mathbf{a}_{ij}}}$. As the function $\Phi(\cdot)$ is monotonic, the original minimax problem is equivalent to

$$(\mathbf{a}_{ij}, b_{ij}) = \arg \max_{\mathbf{a}_{ij}\in\mathbb{R}^d, b_{ij}\in\mathbb{R}} \min (u_1, u_2).$$

We can write u_1 in the following form for a given u_2,

$$u_1 = \frac{\mathbf{a}_{ij}^T\hat{\mathbf{p}}_{ij} - u_2\sqrt{\mathbf{a}_{ij}^T\Sigma_j\mathbf{a}_{ij}}}{\sqrt{\mathbf{a}_{ij}^T\Sigma_i\mathbf{a}_{ij}}},$$

where $\hat{\mathbf{p}}_{ij} = \hat{\mathbf{p}}_j - \hat{\mathbf{p}}_i$. For each given u_2, u_1 needs to be maximized. Hence, we can differentiate the above equation with respect to $\mathbf{a}_{ij}$ and set the derivative to equal to zero, which leads to

$$\mathbf{a}_{ij} = [t\Sigma_i + (1-t)\Sigma_j]^{-1}\hat{\mathbf{p}}_{ij}, \tag{B.1}$$

where $t \in (0,1)$ is a scaler. Thus according to definition of u_1 and u_2, we have

$$b_{ij} = \mathbf{a}_{ij}^T \hat{\mathbf{p}}_i + t\mathbf{a}_{ij}^T \Sigma_i \mathbf{a}_{ij} = \mathbf{a}_{ij}^T \hat{\mathbf{p}}_j - (1-t)\mathbf{a}_{ij}^T \Sigma_j \mathbf{a}_{ij}. \tag{B.2}$$

It is proved that $u_1 = u_2$ must be hold for the solution of the minimax problem [135], which leads to

$$\mathbf{a}_{ij}^T [t^2 \Sigma_i - (1-t)^2 \Sigma_j] \mathbf{a}_{ij} = 0. \tag{B.3}$$

Thus, one can first solve for t by combining Eqs. (B.1) and (B.3) via numerical iteration efficiently. Then $\mathbf{a}_{ij}$ and b_{ij} can be computed using Eqs. (B.1) and (B.2).

C

Towards online active information gathering motion planning

Facilitating Unmanned Aerial Vehicles (UAVs) path planning for autonomous structural surface inspection [170–174] has drawn significant attention from the research community [175, 176]. While most existing works focus on planning a global path that can provide full coverage of the surface, two main disadvantages are observed: a) the methods inherently assume that the surface information is uniformly distributed, hence ignoring potential spatial correlations of the information field (e.g. temperature distribution); b) the path is planned offline, so it cannot actively integrate measurements obtained during execution into the planning to improve the performance. To overcome the two issues, Zhu et al. [177] propose an online informative path planning (IPP) approach for active information gathering of a 3D surface, which can enhance data acquisition efficiency by taking account of spatial correlations of the information field and planning the path online in a receding horizon manner.

Related work

In path planning for inspection of a given surface, most existing works formulate it as an offline coverage planning problem which tries to find a path that can provide full coverage of the surface [178]. Early works including [179, 180] divide non-planar surfaces into regions using exact cellular decompositions that are then covered in a spiral sweeping pattern. Alternatively, [181] presents a two-step process for inspection path planning which first constructs a set of viewpoints and then finds a short path connecting them by solving a traveling salesman problem (TSP). By employing an iterative strategy with re-meshing techniques, [182] proposes a path planning framework that provides uniform coverage of 3D structures. Besides those optimization-based approaches, several sampling-based

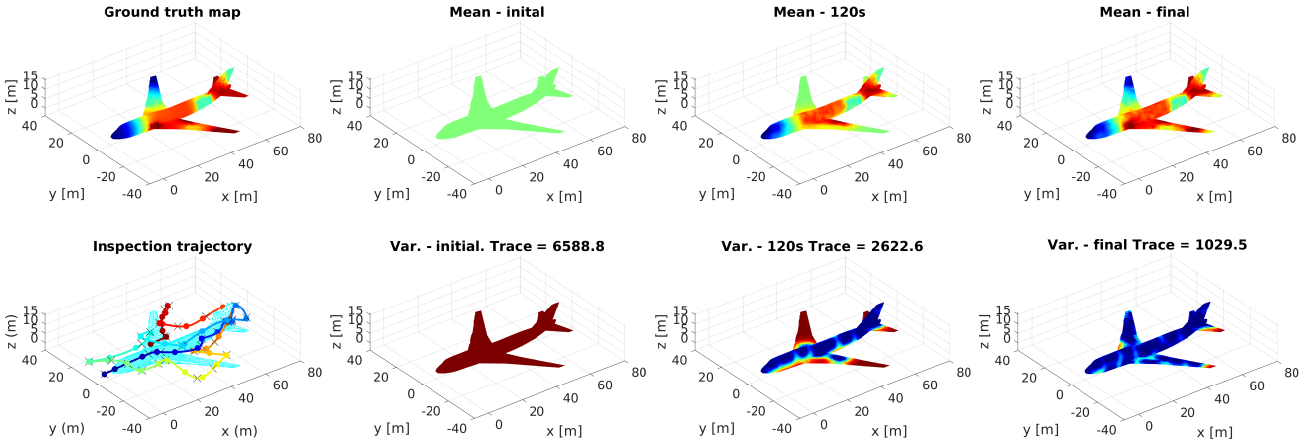
coverage planning methods [176, 183, 184] are presented by exploring the configuration space with a random tree. In contrast to these offline path planning works, there are also several online planning approaches for surface inspection. [185] adapts the next-best-view (NBV) planner [186] to automated surface acquisition by enforcing the surface portions scanning and overlap constraints. Recently [187] presents a receding horizon NBV planner for inspection of a given surface.

While the aforementioned path planning approaches can generate offline coverage paths or online NBV paths, they all inherently assume that the information on the surface is uniformly distributed, hence ignoring potential spatial correlations of the information field. To model such information spatial correlations, Gaussian processes (GPs) [188] have been used as a popular mapping method and successfully applied in terrain mapping [189], target search [190] and environmental monitoring [191]. However, the normal GP formulation is limited to 2D terrains or 3D Euclidean space. To facilitate mapping 3D surfaces, Gaussian process implicit surfaces (GPISs) [192] have been used for surface reconstruction [193–195], object shape estimation [196], and pipeline thickness mapping [197]. The key idea is to represent the surface using a function that specifies whether a point in space is on the surface, outside the surface, or inside the surface. However, GPISs are limited to modeling surface geometry and cannot be directly applied to general surface information fields (e.g. temperature distribution over a surface). Recently, manifold Gaussian processes (mGPs) [198] have been developed to map information fields to complex domains and surfaces with heat kernels [199], generalized Matérn kernels [200], and geodesic Gaussian kernels [201, 202]. We use manifold Gaussian processes with geodesic kernel functions to map the surface information fields and plan informative paths based on the map.

Contribution

An online informative path planning (IPP) approach for active information gathering on 3D surfaces is proposed. The approach builds upon previous works [189, 191] in which an informative path planning framework is presented for 2D terrain mapping and environmental monitoring. We adopt a similar framework and adapt it for 3D surface active information gathering. In particular, we use manifold Gaussian processes (mGPs) with geodesic kernel functions to map the surface information fields which encode their spatial correlations. For data measurement and map update, a probabilistic sensor (camera) model with limited field-of-view (FoV) is considered. Based on the map, a continuous trajectory is optimized by maximizing an information acquisition metric in a receding horizon fashion. The planned trajectory is collision-free and respects the UAV dynamics.

We showed in simulations that the proposed IPP approach could achieve faster map uncertainty and error reduction than the coverage path planner and a random inspection strategy. It was also shown that taking spatial correlations into planning using mGP for mapping could significantly improve the information gathering efficiency. We also validated the approach in a simulated airplane surface temperature inspection mission, as shown in Fig. C.1.



(a) Ground truth map (top) and inspection trajectory (bottom).

(b) Map initial mean (top) and variance (bottom) at 0 s.

(c) Map mean (top) and variance (bottom) at 120 s.

(d) Map final mean (top) and variance (bottom) at 240 s.

Figure C.1: Simulation results of the airplane inspection task using the proposed IPP method.

Bibliography

[1] Peter R. Wurman, Raffaello D'Andrea, and Mick Mountz. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI Magazine*, 29(1):9–19, 2008.

[2] Joseph L Baxter, EK Burke, Jonathan M Garibaldi, and Mark Norman. Multi-robot search and rescue: A potential field based approach. In *Autonomous robots and agents*, pages 9–16. Springer, 2007.

[3] Shushman Choudhury, Kiril Solovey, Mykel J Kochenderfer, and Marco Pavone. Efficient large-scale multi-drone delivery using transit networks. *Journal of Artificial Intelligence Research*, 70:757–788, 2021.

[4] Tobias Nägeli, Lukas Meier, Alexander Domahidi, Javier Alonso-Mora, and Otmar Hilliges. Real-time planning for automated multi-view drone cinematography. *ACM Transactions on Graphics*, 36(4):1–10, 2017.

[5] Wolfgang Hönig. *Motion coordination for large multi-robot teams in obstacle-rich environments*. PhD thesis, University of Southern California, 2019.

[6] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, Jose Neira, Ian Reid, and John J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.

[7] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to autonomous mobile robots*. MIT press, 2011.

[8] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on intelligent vehicles*, 1(1):33–55, 2016.

[9] Zachary Kingston, Mark Moll, and Lydia E. Kavraki. Sampling-based methods for motion planning with constraints. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1):159–185, 2018.

[10] Eduardo F Camacho and Carlos Bordons. *Model predictive control: Classical, robust and stochastic*. Springer International Publishing, 2016.

[11] Shen Peng. *Chance constrained problem and its applications*. PhD thesis, Universite Paris Sud, 2019.

[12] Noel E. Du Toit. *Robot motion planning in dynamic, cluttered, and uncertain environments: The partially closed-loop receding horizon control approach*. PhD thesis, California Institute of Technology, 2010.

[13] Sumeet Singh, Anirudha Majumdar, Jean-Jacques Slotine, and Marco Pavone. Robust online motion planning via contraction theory and convex optimization. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5883–5890. IEEE, 2017.

[14] Carlos Quintero-Pena, Anastasios Kyrillidis, and Lydia E Kavraki. Robust optimization-based motion planning for high-dof robots under sensing uncertainty. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[15] Dimitris Bertsimas, David B Brown, and Constantine Caramanis. Theory and applications of robust optimization. *SIAM review*, 53(3):464–501, 2011.

[16] Vivek Bithar. *Robust mpc based motion planning and control of autonomous ground vehicles*. PhD thesis, The Ohio State University, 2020.

[17] Changchun Liu, Andrew Gray, Chankyu Lee, J. Karl Hedrick, and Jiluan Pan. Nonlinear stochastic predictive control with unscented transformation for semi-autonomous vehicles. In *2014 American Control Conference (ACC)*, pages 5574–5579. IEEE, 2014.

[18] Lucas Janson, Edward Schmerling, and Marco Pavone. Monte carlo motion planning for robot trajectory optimization under uncertainty. In *Robotics Research*, pages 343–361. Springer, 2018.

[19] Chonhyon Park, Jae S Park, and Dinesh Manocha. Fast and bounded probabilistic collision detection for high-dof trajectory planning in dynamic environments. *IEEE Transactions on Automation Science and Engineering*, 15(3):980–991, 2018.

[20] Siyu Dai, Shawn Schaffert, Ashkan Jasour, Andreas Hofmann, and Brian Williams. Chance constrained motion planning for high-dimensional robots. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8805–8811. IEEE, 2019.

[21] Andrew Blake, Alejandro Bordallo, Kamen Brestnichki, Majd Hawasly, Svetlin Valentinov Penkov, Subramanian Ramamoorthy, and Alexandre Silva. Fpr—fast path risk algorithm to evaluate collision probability. *IEEE Robotics and Automation Letters*, 5(1):1–7, 2019.

[22] Brandon Luders, Mangal Kothari, and Jonathan How. Chance constrained rrt for probabilistic robustness to environmental uncertainty. In *AIAA Guidance, Navigation, and Control Conference*, 2010.

[23] Jur Van Den Berg, Stephen J Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. In *Robotics Research*, pages 3–19. Springer, 2011.

[24] Pete Penny Florence, John Carter, and Russ Tedrake. Integrated perception and control at high speed: Evaluating collision avoidance maneuvers without maps. In *WAFR: Workshop on Algorithmic Foundations of Robotics*, 2016.

[25] Sachin Patil, Jur van den Berg, and Ron Alterovitz. Estimating probability of collision for safe motion planning under Gaussian motion and sensing uncertainty. In *2012 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3238–3244. IEEE, 2012.

[26] Alain Lambert, Dominique Gruyer, and Guillaume Saint Pierre. A fast monte carlo algorithm for collision probability estimation. In *2008 10th International Conference on Control, Automation, Robotics and Vision*, pages 406–411. IEEE, 2008.

[27] Lars Blackmore, Masahiro Ono, Askar Bektassov, and Brian C. Williams. A probabilistic particle-control approximation of chance-constrained stochastic predictive control. *IEEE Transactions on Robotics*, 26(3):502–517, 2010.

[28] Edward Schmerling and Marco Pavone. Evaluating trajectory collision probability through adaptive importance sampling for safe motion planning. In *Robotics: Science and Systems*, 2017.

[29] Noel E. Du Toit and Joel W. Burdick. Probabilistic collision checking with chance constraints. *IEEE Transactions on Robotics*, 27(4):809–815, 2011.

[30] Lars Blackmore, Masahiro Ono, and Brian C Williams. Chance-constrained optimal path planning with obstacles. *IEEE Transactions on Robotics*, 27(6):1080–1094, 2011.

[31] Jason Hardy and Mark Campbell. Contingency planning over probabilistic obstacle predictions for autonomous road vehicles. *IEEE Transactions on Robotics*, 29(4):913–929, 2013.

[32] Russell A Paielli and Heinz Erzberger. Conflict probability estimation for free flight. *Journal of Guidance, Control, and Dynamics*, 20(3):588–596, 1997.

[33] Jae Sung Park, Chonhyon Park, and Dinesh Manocha. Efficient probabilistic collision detection for non-convex shapes. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1944–1951. IEEE, 2017.

[34] Jae Sung Park and Dinesh Manocha. Efficient probabilistic collision detection for non-gaussian noise distributions. *IEEE Robotics and Automation Letters*, 5(2):1024–1031, 2020.

[35] A.M. Mathai and Serge B. Provost. *Quadratic forms in random variables: Theory and applications*. Marcel Dek ker, 1992.

[36] Ali Akbar Mohsenipour. *On the Distribution of Quadratic Expressions in Various Types of Random Vectors*. PhD thesis, The University of Western Ontario, 2012.

[37] Imhof P.J. Computing the distribution of quadratic forms in normal variables. *Biometrika*, 48(3):419–426, 1961.

[38] Samuel Kotz, Norman L Johnson, and DW Boyd. Series representations of distributions of quadratic forms in normal variables. I. Central case. *The Annals of Mathematical Statistics*, 38(3):823–837, 1967.

[39] Samuel Kotz, N. L. Johnson, and D. W. Boyd. Series representations of distributions of quadratic forms in normal variables II. Noncentral case. *The Annals of Mathematical Statistics*, 38(3):838–848, 1967.

[40] Huan Liu, Yongqiang Tang, and Hao Helen Zhang. A new chi-square approximation to the distribution of non-negative definite quadratic forms in non-central normal variables. *Computational Statistics & Data Analysis*, 53(4):853–856, 2009.

[41] Tareq Y. Al-Naffouri, Muhammed Moinuddin, Nizar Ajeeb, Babak Hassibi, and Aris L. Moustakas. On the distribution of indefinite quadratic forms in gaussian random variables. *IEEE Transactions on Communications*, 64(1):153–165, 2016.

[42] F. Kenneth Chan. *Spacecraft Collision Probability*. American Institute of Aeronautics and Astronautics, Inc., Washington, DC, 2008.

[43] Lukas Hewing, Alexander Liniger, and Melanie N. Zeilinger. Cautious nmpc with gaussian process dynamics for autonomous miniature race cars. In *2018 European Control Conference (ECC)*, pages 1341–1348. IEEE, 2018.

[44] Allen Wang, Xin Huang, Ashkan Jasour, and Brian C Williams. Fast risk assessment for autonomous vehicles using learned models of agent futures. In *Robotics: Science and Systems*, 2020.

[45] Pablo A Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical programming*, 96(2):293–320, 2003.

[46] Ashkan M Jasour. *Convex approximation of chance constrained problems: application is systems and control*. PhD thesis, The Pennsylvania State University, 2017.

[47] Ashkan M Jasour, Andreas Hofmann, and Brian C Williams. Moment-sum-of-squares approach for fast risk estimation in uncertain environments. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 2445–2451. IEEE, 2018.

[48] Ashkan M. Jasour and Brian Charles Williams. Risk contours map for risk bounded motion planning under perception uncertainties. In *Robotics: Science and Systems*, 2019.

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

[50] Charles Dawson, Andreas Hofmann, and Brian Williams. Fast certification of collision probability bounds with uncertain convex obstacles. *arXiv preprint arXiv:2003.07792*, 2020.

[51] Xinbo Geng and Le Xie. Data-driven decision making in power systems with probabilistic guarantees: Theory and applications of chance-constrained optimization. *Annual Reviews in Control*, 47:341–363, 2019.

[52] Masahiro Ono and Brian C Williams. Iterative risk allocation: A new approach to robust model predictive control with a joint chance constraint. In *2008 IEEE Conference on Decision and Control (CDC)*, pages 3427–3432. IEEE, 2008.

[53] Lars Blackmore, Masahiro Ono, and Brian C Williams. Chance-constrained optimal path planning with obstacles. *IEEE Transactions on Robotics*, 27(6):1080–1094, 2011.

[54] Shinji Kataoka. A stochastic programming model. *Econometrica*, 31(1/2):181–196, 1963.

[55] Marcio da Silva Arantes, Claudio Fabiano Motta Toledo, Brian Charles Williams, and Masahiro Ono. Collision-free encoding for chance-constrained nonconvex path planning. *IEEE Transactions on Robotics*, 35(2):433–448, 2019.

[56] Monimoy Bujarbaruah, Xiaojing Zhang, Marko Tanaskovic, and Francesco Borrelli. Adaptive stochastic mpc under time-varying uncertainty. *IEEE Transactions on Automatic Control*, 66(6):2840–2845, 2020.

[57] Manuel Castillo-Lopez, Philippe Ludivig, Seyed Amin Sajadi-Alamdari, Jose Luis Sanchez-Lopez, Miguel A Olivares-Mendez, and Holger Voos. A real-time approach for chance-constrained motion planning with dynamic obstacles. *IEEE Robotics and Automation Letters*, 5(2):3620–3625, 2020.

[58] Fan Chung and Linyuan Lu. *Complex graphs and networks*. American Mathematical Soc., 2006.

[59] Allen Wang, Ashkan Jasour, and Brian C. Williams. Non-gaussian chance-constrained trajectory planning for autonomous vehicles under agent uncertainty. *IEEE Robotics and Automation Letters*, 5(4):6041–6048, 2020.

[60] M C Campi and S Garatti. The exact feasibility of randomized solutions of uncertain convex programs. *SIAM Journal on Optimization*, 19(3):1211–1230, 2008.

[61] Marco C. Campi, Simone Garatti, and Maria Prandini. The scenario approach for systems and control design. *Annual Reviews in Control*, 33(2):149–157, 2009.

[62] G.C. Calafiore and M.C. Campi. The scenario approach to robust control design. *IEEE Transactions on Automatic Control*, 51(5):742–753, 2006.

[63] Sergio Grammatico, Xiaojing Zhang, Kostas Margellos, Paul Goulart, and John Lygeros. A scenario approach for non-convex control design. *IEEE Transactions on Automatic Control*, 61(2):1–1, 2016.

[64] Marco Claudio Campi, Simone Garatti, and Federico Alessandro Ramponi. A general scenario theory for nonconvex optimization and decision making. *IEEE Transactions on Automatic Control*, 63(12):4067–4078, 2018.

[65] Oscar de Groot, Bruno Brito, Laura Ferranti, Dariu Gavrila, and Javier Alonso-Mora. Scenario-based trajectory optimization in uncertain dynamic environments. *IEEE Robotics and Automation Letters*, 6(3):5389–5396, 2021.

[66] M. C. Campi and S. Garatti. A sampling-and-discarding approach to chance-constrained optimization: Feasibility and optimality. *Journal of Optimization Theory and Applications*, 148(2):257–280, 2011.

[67] Raffaele Soloperto, Johannes Kohler, Frank Allguwer, and Matthias A. Muller. Collision avoidance for uncertain nonlinear systems with moving obstacles using robust model predictive control. In *2019 European Control Conference (ECC)*, pages 811–817. IEEE, 2019.

[68] Brian Axelrod, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Provably safe robot navigation with obstacle uncertainty. *The International Journal of Robotics Research*, 37(13-14):1760–1774, 2018.

[69] Charles Dawson, Ashkan Jasour, Andreas Hofmann, and Brian Williams. Provably safe trajectory optimization in the presence of uncertain convex obstacles. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6237–6244. IEEE, 2020.

[70] Sri Sai Naga Jyotish Poonganam, Bharath Gopalakrishnan, Venkata Seetharama Sai Bhargav Kumar Avula, Arun Kumar Singh, K. Madhava Krishna, and Dinesh Manocha. Reactive navigation under non-parametric uncertainty through hilbert space embedding of probabilistic velocity obstacles. *IEEE Robotics and Automation Letters*, 5(2):2690–2697, 2020.

[71] Ashkan Jasour, Weiqiao Han, and Brian Williams. Convex risk bounded continuous-time trajectory planning in uncertain nonconvex environments. In *Robotics: Science and Systems*, 2021.

[72] Herbert G Tanner and Amit Kumar. Formation stabilization of multiple agents using decentralized navigation functions. In *Robotics: Science and Systems*, 2005.

[73] Farbod Fahimi, C. Nataraj, and Hashem Ashrafiuon. Real-time obstacle avoidance for multiple mobile robots. *Robotica*, 27(2):189–198, 2009.

[74] Yan Yongjie and Zhang Yan. Collision avoidance planning in multi-robot based on improved artificial potential field and rules. In *2009 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 1026–1031. IEEE, 2009.

[75] Oussama Khatib and Mobile Robots. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–98, 1986.

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

[77] Jur Van den Berg, Ming Lin, and Dinesh Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *2008 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1928–1935. IEEE, 2008.

[78] Javier Alonso-Mora, Andreas Breitenmoser, Martin Rufli, Paul Beardsley, and Roland Siegwart. Optimal reciprocal collision avoidance for multiple non-holonomic robots. In *Distributed autonomous robotic systems*, pages 203–216. Springer, 2013.

[79] Daman Bareiss and Jur van den Berg. Generalized reciprocal collision avoidance. *The International Journal of Robotics Research*, 34(12):1501–1514, 2015.

[80] Javier Alonso-Mora, Paul Beardsley, and Roland§ Siegwart. Cooperative collision avoidance for nonholonomic robots. *IEEE Transactions on Robotics*, 34(2):404–420, 2018.

[81] Dingjiang Zhou, Zijian Wang, Saptarshi Bandyopadhyay, and Mac Schwager. Fast, on-line collision avoidance for dynamic vehicles using buffered voronoi cells. *IEEE Robotics and Automation Letters*, 2(2):1047–1054, 2017.

[82] Li Wang, Aaron D. Ames, and Magnus Egerstedt. Safety barrier certificates for collisions-free multirobot systems. *IEEE Transactions on Robotics*, 33(3):661–674, 2017.

[83] Dh Shim, H.J. Kim, and Shankar Sastry. Decentralized nonlinear model predictive control of multiple flying robots. In *2003 IEEE Conference on Decision and Control (CDC)*, pages 3621–3626. IEEE, 2003.

[84] Mina Kamel, Javier Alonso-Mora, Roland Siegwart, and Juan Nieto. Robust collision avoidance for multiple micro aerial vehicles using nonlinear model predictive control. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 236–243. IEEE, 2017.

[85] Senthil Hariharan Arul and Dinesh Manocha. Dcad: decentralized collision avoidance with dynamics constraints for agile quadrotor swarms. *IEEE Robotics and Automation Letters*, 5(2):1191–1198, 2020.

[86] Daniel Morgan, Giri P. Subramanian, Soon Jo Chung, and Fred Y. Hadaegh. Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming. *International Journal of Robotics Research*, 35(10):1261–1285, 2016.

[87] Carlos E Luis, Marijan Vukosavljev, and Angela P Schoellig. Online trajectory generation with distributed model predictive control for multi-robot motion planning. *IEEE Robotics and Automation Letters*, 5(2):604–611, 2020.

[88] Stephen Boyd, Neal Parikh, and Eric Chu. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

[89] Huarong Zheng, Rudy R Negenborn, and Gabriël Lodewijks. Fast admm for distributed model predictive control of cooperative waterborne agvs. *IEEE Transactions on Control Systems Technology*, 25(4):1406–1413, 2016.

[90] Ruben Van Parys and Goele Pipeleers. Distributed model predictive formation control with inter-vehicle collision avoidance. In *2017 11th Asian Control Conference (ASCC)*, pages 2399–2404. IEEE, 2017.

[91] Laura Ferranti, Rudy R. Negenborn, Tamás Keviczky, and Javier Alonso-Mora. Coordination of multiple vessels via distributed nonlinear model predictive control. In *2018 European Control Conference (ECC)*, pages 2523–2528. IEEE, 2018.

[92] Felix Rey, Zhoudan Pan, Adrian Hauswirth, and John Lygeros. Fully decentralized admm for coordination and collision avoidance. In *2018 European Control Conference (ECC)*, pages 825–830. IEEE, 2018.

[93] Thulasi Mylvaganam, Mario Sassano, and Alessandro Astolfi. A differential game approach to multi-agent collision avoidance. *IEEE Transactions on Automatic Control*, 62(8):4229–4235, 2017.

[94] Wilko Schwarting, Alyssa Pierson, Javier Alonso-Mora, Sertac Karaman, and Daniela Rus. Social behavior for autonomous vehicles. *Proceedings of the National Academy of Sciences*, 116(50):24972–24978, 2019.

[95] Brian Reily, Terran Mott, and Hao Zhang. Decentralized and communication-free multi-robot navigation through distributed games. *arXiv preprint arXiv:2012.09335*, 2021.

[96] Zijian Wang, Riccardo Spica, and Mac Schwager. Game theoretic motion planning for multi-robot racing. In *Distributed Autonomous Robotic Systems*, pages 225–238. Springer, 2019.

[97] Pinxin Long, Wenxi Liu, and Jia Pan. Deep-learned collision avoidance policy for distributed multiagent navigation. *IEEE Robotics and Automation Letters*, 2(2):656–663, 2017.

[98] Qingbiao Li, Fernando Gama, Alejandro Ribeiro, and Amanda Prorok. Graph neural networks for decentralized path planning. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 1901–1903, 2020.

[99] Benjamin Riviere, Wolfgang Hönig, Yisong Yue, and Soon Jo Chung. Glas: global-to-local safe autonomy synthesis for multi-robot motion planning with end-to-end learning. *IEEE Robotics and Automation Letters*, 5(3):4249–4256, 2020.

[100] Yu Fan Chen, Miao Liu, Michael Everett, and Jonathan P. How. Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 285–292. IEEE, 2017.

[101] Michael Everett, Yu Fan Chen, and Jonathan P How. Motion planning among dynamic, decision-making agents with deep reinforcement learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3052–3059. IEEE, 2018.

[102] Samaneh Hosseini Semnani, Hugh Liu, Michael Everett, Anton de Ruiter, and Jonathan P How. Multi-agent motion planning for dense and dynamic environments via deep reinforcement learning. *IEEE Robotics and Automation Letters*, 5(2):3221–3226, 2020.

[103] Tingxiang Fan, Pinxin Long, Wenxi Liu, and Jia Pan. Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios. *The International Journal of Robotics Research*, 39(7):856–892, 2020.

[104] Guni Sharon, Roni Stern, Ariel Felner, and Nathan R Sturtevant. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, 219:40–66, 2015.

[105] Wolfgang Hönig, James A. Preiss, T. K.Satish Kumar, Gaurav S. Sukhatme, and Nora Ayanian. Trajectory planning for quadrotor swarms. *IEEE Transactions on Robotics*, 34(4):856–869, 2018.

[106] Lukas Hewing, Kim P. Wabersich, Marcel Menner, and Melanie N. Zeilinger. Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3(1):269–296, 2020.

[107] Jaime F Fisac, Neil F Lugovoy, Vicenç Rubies-Royo, Shromona Ghosh, and Claire J Tomlin. Bridging hamilton-jacobi safety analysis and reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8550–8556. IEEE, 2019.

[108] Bruno Brito, Michael Everett, Jonathan P How, and Javier Alonso-Mora. Where to go next: Learning a subgoal recommendation policy for navigation in dynamic environments. *IEEE Robotics and Automation Letters*, 6(3):4616–4623, 2021.

[109] Daniel Claes, Daniel Hennes, Karl Tuyls, and Wim Meeussen. Collision avoidance under bounded localization uncertainty. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1192–1198. IEEE, 2012.

[110] Daniel Claes and Karl Tuyls. Multi robot collision avoidance in a shared workspace. *Autonomous Robots*, 42(8):1749–1770, 2018.

[111] Daniel Lyons, J. Calliess, and U. D. Hanebeck. Chance constrained model predictive control for multi-agent systems with coupling constraints. In *2012 American Control Conference (ACC)*, pages 1223–1230. IEEE, 2012.

[112] Bharath Gopalakrishnan, Arun Kumar Singh, Meha Kaushik, K. Madhava Krishna, and Dinesh Manocha. Prvo: Probabilistic reciprocal velocity obstacle for multi robot navigation under uncertainty. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1089–1096. IEEE, 2017.

[113] Senthil Hariharan Arul and Dinesh Manocha. Swarmcco: Probabilistic reactive collision avoidance for quadrotor swarms under uncertainty. *IEEE Robotics and Automation Letters*, 6(2):2437–2444, 2021.

[114] Mingyu Wang and Mac Schwager. Distributed collision avoidance of multiple robots with probabilistic buffered voronoi cells. In *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pages 169–175. IEEE, 2019.

[115] Wenhao Luo, Wen Sun, and Ashish Kapoor. Multi-robot collision avoidance under uncertainty with probabilistic safety barrier certificates. In *2020 Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[116] Tobias Nageli, Javier Alonso-Mora, Alexander Domahidi, Daniela Rus, and Otmar Hilliges. Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization. *IEEE Robotics and Automation Letters*, 2(3):1696–1703, 2017.

[117] Alexei Yu Uteshev and Marina V Goncharova. Point-to-ellipse and point-to-ellipsoid distance equation analysis. *Journal of Computational and Applied Mathematics*, 328:232–251, 2018.

[118] Ali Mesbah, Stefan Streif, Rolf Findeisen, and Richard D Braatz. Stochastic nonlinear model predictive control with probabilistic constraints. In *2014 American Control Conference (ACC)*, pages 2413–2419. IEEE, 2014.

[119] Roberto Armellin, Pierluigi Di Lizia, Franco Bernelli-Zazzera, and Martin Berz. Asteroid close encounters characterization using differential algebra: the case of apophis. *Celestial Mechanics and Dynamical Astronomy*, 107(4):451–470, 2010.

[120] Monica Valli, Roberto Armellin, Pierluigi Di Lizia, and MR Lavagna. Nonlinear mapping of uncertainties in celestial mechanics. *Journal of Guidance, Control, and Dynamics*, 36(1):48–63, 2013.

[121] Ya Zhong Luo and Zhen Yang. A review of uncertainty propagation in orbital mechanics. *Progress in Aerospace Sciences*, 89:23–39, 2017.

[122] Wilko Schwarting, Javier Alonso-Mora, Liam Pauli, Sertac Karaman, and Daniela Rus. Parallel autonomy in automated vehicles: Safe motion generation with minimal intervention. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1928–1935. IEEE, 2017.

[123] Mina Kamel, Javier Alonso-Mora, Roland Siegwart, and Juan Nieto. Robust collision avoidance for multiple micro aerial vehicles using nonlinear model predictive control. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 236–243, 2017.

[124] Ke Sun, Kartik Mohta, Bernd Pfrommer, Michael Watterson, Sikang Liu, Yash Mulgaonkar, Camillo J. Taylor, and Vijay Kumar. Robust stereo visual inertial odometry for fast autonomous flight. *IEEE Robotics and Automation Letters*, 3(2):965–972, 2017.

[125] Hai Zhu, Jelle Juhl, Laura Ferranti, and Javier Alonso-Mora. Distributed multi-robot formation splitting and merging in dynamic environments. In *2019 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9080–9086. IEEE, 2019.

[126] Boris Houska, Hans Joachim Ferreau, and Moritz Diehl. ACADO toolkit-An open-source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods*, 32(3):298–312, 2011.

[127] Andrea Zanelli, Alexander Domahidi, J Jerez, and Manfred Morari. Forces nlp: an efficient implementation of interior-point methods for multistage nonlinear nonconvex programs. *International Journal of Control*, 93(1):13–29, 2020.

[128] Robin Deits and Russ Tedrake. Computing large convex regions of obstacle-free space through semidefinite programming. In *Algorithmic foundations of robotics XI*, pages 109–124. Springer, 2015.

[129] Robin Deits and Russ Tedrake. Efficient mixed-integer planning for uavs in cluttered environments. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 42–49. IEEE, 2015.

[130] Sikang Liu, Michael Watterson, Kartik Mohta, Ke Sun, Subhrajit Bhattacharya, Camillo J. Taylor, and Vijay Kumar. Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments. *IEEE Robotics and Automation Letters*, 2(3):1688–1695, 2017.

[131] Jesus Tordesillas, Brett T Lopez, and Jonathan P How. Faster: Fast and safe trajectory planner for flights in unknown environments. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1934–1940. IEEE, 2019.

[132] Omur Arslan and Daniel E. Koditschek. Sensor-based reactive navigation in unknown convex sphere worlds. *International Journal of Robotics Research*, 38(2-3):196–223, 2019.

[133] Alyssa Pierson, Wilko Schwarting, Sertac Karaman, and Daniela Rus. Weighted buffered voronoi cells for distributed semi-cooperative behavior. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5611–5617. IEEE, 2020.

[134] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. *Spatial tessellations: Concepts and applications of Voronoi diagrams*. John Wiley & Sons, 2009.

[135] T. W. Anderson and R. R. Bahadur. Classification into two multivariate normal distributions with different covariance matrices. *The Annals of Mathematical Statistics*, 33(2):420–431, 1962.

[136] Larry C. Andrews. *Special functions of mathematics for engineers*, volume 49. SPIE press, 1997.

[137] Mikhail K Kozlov, Sergei P Tarasov, and Leonid G Khachiyan. The polynomial solvability of convex quadratic programming. *USSR Computational Mathematics and Mathematical Physics*, 20(5):223–228, 1980.

[138] A. Astolfi. Exponential stabilization of a wheeled mobile robot via discontinuous control. *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, 121(1):121–126, 1999.

[139] Álvaro Serra-Gómez, Bruno Brito, Hai Zhu, Jen Jen Chung, and Javier Alonso-Mora. With whom to communicate: Learning efficient communication for multi-robot collision avoidance. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11770–11776. IEEE, 2020.

[140] Hai Zhu and Javier Alonso-Mora. B-uavc: buffered uncertainty-aware voronoi cells for probabilistic multi-robot collision avoidance. In *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pages 162–168. IEEE, 2019.

[141] Andreas Breitenmoser and Alcherio Martinoli. On combining multi-robot coverage and reciprocal collision avoidance. In *Springer Tracts in Advanced Robotics*, volume 112, pages 49–64. Springer Japan, Tokyo, 2016.

[142] Lifeng Zhou, Vasileios Tzoumas, George J Pappas, and Pratap Tokekar. Resilient active target tracking with multiple robots. *IEEE Robotics and Automation Letters*, 4(1):129–136, 2018.

[143] Soon-Jo Chung, Aditya Avinash Paranjape, Philip Dames, Shaojie Shen, and Vijay Kumar. A survey on aerial swarm robotics. *IEEE Transactions on Robotics*, 34(4):837–855, 2018.

[144] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 European Control Conference (ECC)*, pages 3420–3431. IEEE, 2019.

[145] Aaron D. Ames, Xiangru Xu, Jessy W. Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8):3861–3876, 2017.

[146] Hai Zhu, Francisco Martinez Claramunt, Bruno Brito, and Javier Alonso-Mora. Learning interaction-aware trajectory predictions for decentralized multi-robot motion planning in dynamic environments. *IEEE Robotics and Automation Letters*, 6(2):2256–2263, 2021.

[147] Edward Schmerling, Karen Leung, Wolf Vollprecht, and Marco Pavone. Multimodal probabilistic model-based planning for human-robot interaction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3399–3406. IEEE, 2018.

[148] David Fridovich-Keil, Andrea Bajcsy, Jaime F. Fisac, Sylvia L. Herbert, Steven Wang, Anca D. Dragan, and Claire J. Tomlin. Confidence-aware motion prediction for real-time collision avoidance. *The International Journal of Robotics Research*, 39(2-3):250–265, 2020.

[149] Andrey Rudenko, Luigi Palmieri, Michael Herman, Kris M. Kitani, Dariu M. Gavrila, and Kai O. Arras. Human motion trajectory prediction: a survey. *The International Journal of Robotics Research*, 39(8):895–935, 2020.

[150] Dirk Helbing and Péter Molnár. Social force model for pedestrian dynamics. *Physical Review E*, 51(5):4282–4286, 1995.

[151] Dirk Helbing and Benno Tilch. Generalized force model of traffic dynamics. *Physical Review E*, 58(1):133, 1998.

[152] Annemarie Turnwald, Daniel Althoff, Dirk Wollherr, and Martin Buss. Understanding human avoidance behavior: interaction-aware decision making based on game theory. *International Journal of Social Robotics*, 8(2):331–351, 2016.

[153] Dave W. Oyler, Yildiray Yildiz, Anouck R. Girard, Nan I. Li, and Ilya V. Kolmanovsky. A game theoretical model of traffic with multiple interacting drivers for use in autonomous vehicle development. In *2016 American Control Conference (ACC)*, pages 1705–1710. IEEE, 2016.

[154] Markus Kuderer, Shilpa Gulati, and Wolfram Burgard. Learning driving styles for autonomous vehicles from demonstration. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2641–2646. IEEE, 2015.

[155] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–971. IEEE, 2016.

[156] Bruno Brito, Hai Zhu, Wei Pan, and Javier Alonso-mora. Social-vrnn: One-shot multi-modal trajectory prediction for interacting pedestrians. In *2020 Conference on Robot Learning (CoRL)*, 2020.

[157] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B Choy, Philip H.S. Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2165–2174. IEEE, 2017.

[158] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2255–2264. IEEE, 2018.

[159] Jim Mainprice and Dmitry Berenson. Human-robot collaborative manipulation planning using early prediction of human motion. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 299–306. IEEE, 2013.

[160] Jae Sung Park, Chonhyon Park, and Dinesh Manocha. I-planner: Intention-aware motion planning using learning-based human motion prediction. *The International Journal of Robotics Research*, 38(1):23–39, 2019.

[161] Mark Pfeiffer, Giuseppe Paolo, Hannes Sommer, Juan Nieto, Rol Siegwart, and Cesar Cadena. A data-driven model for interaction-aware pedestrian motion prediction in object cluttered environments. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5921–5928. IEEE, 2018.

[162] Christoph Scholler, Vincent Aravantinos, Florian Lay, and Alois Knoll. What the constant velocity model can teach us about pedestrian motion prediction. *IEEE Robotics and Automation Letters*, 5(2):1696–1703, 2020.

[163] Sepp Hochreiter and J Urgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[164] Paul J Werbos. Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[165] Anirudha Majumdar and Marco Pavone. How should a robot assess risk? Towards an axiomatic theory of risk in robotics. In *Robotics Research*, pages 75–84. Springer, 2020.

[166] Astghik Hakobyan, Gyeong Chan Kim, and Insoon Yang. Risk-aware motion planning and control using cvar-constrained optimization. *IEEE Robotics and Automation Letters*, 4(4):3924–3931, 2019.

[167] Yen Cheng Liu, Junjiao Tian, Nathaniel Glaser, and Zsolt Kira. When2com: Multi-agent perception via communication graph grouping. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 4105–4114, 2020.

[168] Yen Cheng Liu, Junjiao Tian, Chih Yao Ma, Nathan Glaser, Chia Wen Kuo, and Zsolt Kira. Who2com: Collaborative perception via learnable handshake communication. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 6876–6883, 2020.

[169] Yuanzhao Zhai, Bo Ding, Xuan Liu, Hongda Jia, Yong Zhao, and Jie Luo. Decentralized multi-robot collision avoidance in complex scenarios with selective communication. *IEEE Robotics and Automation Letters*, 6(4):8379–8386, 2021.

[170] Weibin Gu, Dewen Hu, Liang Cheng, Yabing Cao, Alessandro Rizzo, and Kimon P. Valavanis. Autonomous wind turbine inspection using a quadrotor. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 709–715, 2020.

[171] Siyuan Chen, Debra F. Laefer, Eleni Mangina, S. M. Iman Zolanvari, and Jonathan Byrne. UAV bridge inspection through evaluated 3d reconstructions. *Journal of Bridge Engineering*, 24(4):05019001, 2019.

[172] Jong-Woo Kim, Sung-Bae Kim, Jeong-Cheon Park, and Jin-Won Nam. Development of crack detection system with unmanned aerial vehicles and digital image processing. *Advances in structural engineering and mechanics (ASEM15)*, 33(3):25–29, 2015.

[173] Julien Miranda, Stanislas Larnier, Ariane Herbulot, and Michel Devy. UAV-based inspection of airplane exterior screws with computer vision. In *14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 421–427, 2019.

[174] Pipeline inspection with thermal diagnostics. https://www.drone-thermal-camera.com/drone-uav-thermography-inspection-pipeline/, 2019. Accessed: 2020-10-11.

[175] Andreas Bircher, Mina Kamel, Kostas Alexis, Michael Burri, Philipp Oettershagen, Sammy Omari, Thomas Mantel, and Roland Siegwart. Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots. *Autonomous Robots*, 40(6):1059–1078, 2016.

[176] Andreas Bircher, Kostas Alexis, Ulrich Schwesinger, Sammy Omari, Michael Burri, and Roland Siegwart. An incremental sampling-based approach to inspection planning: the rapidly exploring random tree of trees. *Robotica*, 35(6):1327–1340, 2017.

[177] Hai Zhu, Jen Jen Chung, Nicholas Lawrance, Roland Siegwart, and Javier Alonso-Mora. Online informative path planning for active information gathering of a 3d surface. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021.

[178] Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous systems*, 61(12):1258–1276, 2013.

[179] P.N. Atkar, H. Choset, A.A. Rizzi, and E.U. Acar. Exact cellular decomposition of closed orientable surfaces embedded in R3. In *2001 IEEE International Conference on Robotics and Automation (ICRA)*, pages 699–704. IEEE, 2001.

[180] Ercan U. Acar, Howie Choset, Alfred A. Rizzi, Prasad N. Atkar, and Douglas Hull. Morse decompositions for coverage tasks. *The International Journal of Robotics Research*, 21(4):331–344, 2002.

[181] Franz S. Hover, Ryan M. Eustice, Ayoung Kim, Brendan Englot, Hordur Johannsson, Michael Kaess, and John J. Leonard. Advanced perception, navigation and planning for autonomous in-water ship hull inspection. *International Journal of Robotics Research*, 31(12):1445–1464, 2012.

[182] Kostas Alexis, Christos Papachristos, Roland Siegwart, and Anthony Tzes. Uniform coverage structural inspection path–planning for micro aerial vehicles. In *2015 IEEE International Symposium on Intelligent Control (ISIC)*, pages 59–64. IEEE, 2015.

[183] Brendan Englot and Franz S. Hover. Sampling-based coverage path planning for inspection of complex structures. In *Proceedings of the 22nd International Conference on Automated Planning and Scheduling (ICAPS)*, pages 29–37, 2012.

[184] Georgios Papadopoulos, Hanna Kurniawati, and Nicholas M. Patrikalakis. Asymptotically optimal inspection planning using systems with differential constraints. In *2013 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4126–4133. IEEE, 2013.

[185] Richard Pito. A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):1016–1030, 1999.

[186] C. Connolly. The determination of next best views. In *1985 IEEE International Conference on Robotics and Automation (ICRA)*, pages 432–435. IEEE, 1985.

[187] Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. Receding horizon path planning for 3D exploration and surface inspection. *Autonomous Robots*, 42(2):291–306, 2018.

[188] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

[189] Marija Popović, Teresa Vidal-Calleja, Gregory Hitz, Jen Jen Chung, Inkyu Sa, Roland Siegwart, and Juan Nieto. An informative path planning framework for UAV-based terrain monitoring. *Autonomous Robots*, 44(6):889–911, 2020.

[190] Ajith Anil Meera, Marija Popovic, Alexander Millane, and Roland Siegwart. Obstacle-aware adaptive informative path planning for uav-based target search. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 718–724. IEEE, 2019.

[191] Gregory Hitz, Enric Galceran, Marie-Ève Garneau, François Pomerleau, and Roland Siegwart. Adaptive continuous-space informative path planning for online environmental monitoring. *Journal of Field Robotics*, 34(8):1427–1449, 2017.

[192] Oliver Williams and Andrew Fitzgibbon. Gaussian process implicit surfaces. Technical report, Microsoft Research, 2006.

[193] Soohwan Kim and Jonghyuk Kim. Gpmap: A unified framework for robotic mapping based on sparse gaussian processes. In *Field and service robotics*, pages 319–332. Springer, 2015.

[194] Bhoram Lee, Clark Zhang, Zonghao Huang, and Daniel D. Lee. Online continuous mapping using gaussian process implicit surfaces. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6884–6890. IEEE, 2019.

[195] Lan Wu, Raphael Falque, Victor Perez-Puchalt, Liyang Liu, Nico Pietroni, and Teresa Vidal-Calleja. Skeleton-based conditionally independent gaussian process implicit surfaces for fusion in sparse to dense 3d reconstruction. *IEEE Robotics and Automation Letters*, 5(2):1532–1539, 2020.

[196] Stanimir Dragiev, Marc Toussaint, and Michael Gienger. Gaussian process implicit surfaces for shape estimation and grasping. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2845–2850. IEEE, 2011.

[197] Teresa Vidal-Calleja, Daobilige Su, Freek De Bruijn, and Jaime Valls Miro. Learning spatial correlations for Bayesian fusion in pipe thickness mapping. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 683–690. IEEE, 2014.

[198] Sadeep Jayasumana, Richard Hartley, Mathieu Salzmann, Hongdong Li, and Mehrtash Harandi. Kernel methods on riemannian manifolds with gaussian rbf kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(12):2464–2477, 2015.

[199] Mu Niu, Pokman Cheung, Lizhen Lin, Zhenwen Dai, Neil Lawrence, and David Dunson. Intrinsic gaussian processes on complex constrained domains. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 81(3):603–627, 2019.

[200] Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Peter Deisenroth. Matérn gaussian processes on riemannian manifolds. In *2020 Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[201] Masashi Sugiyama, Hirotaka Hachiya, Christopher Towell, and Sethu Vijayakumar. Geodesic Gaussian kernels for value function approximation. *Autonomous Robots*, 25(3):287–304, 2008.

[202] Enrique del Castillo, Bianca M. Colosimo, and Sam Davanloo Tajbakhsh. Geodesic Gaussian processes for the parametric reconstruction of a free-form surface. *Technometrics*, 57(1):87–99, 2015.

Glossary

Notation

Throughout this thesis, scalers are denoted by plain lowercase letters, e.g. x, vectors by bold lowercase, e.g. $\mathbf{x}$, matrices by plain uppercase, e.g. M, and sets by calligraphic uppercase, e.g. $\mathcal{X}$.

Specific sets

$\mathbb{R}$	real numbers.
$\mathbb{R}^n$	real n-vectors.
$\mathbb{R}^{m\times n}$	real $m\times n$ matrices.
$\mathbb{R}_+, \mathbb{R}_{++}$	nonnegative, positive real numbers.
$\mathbb{Z}$	integers.
$\mathbb{Z}_+$	nonnegative integers.
$\mathbb{N}$	natural numbers.
$\mathbb{S}^n$	symmetric $n\times n$ matrices.
$\mathbb{S}^n_+, \mathbb{S}^n_{++}$	symmetric positive semi-definite, positive definite, $n\times n$ matrices.

Vectors and matrices

x^k	the k-th element of vector. $\mathbf{x}$
M^{ij}	the i-th row and j-th column element of M.
$\mathbf{x}^T, M^T$	transpose of a vector $\mathbf{x}$, or matrix M.
$\mathrm{diag}(\mathbf{x})$	diagonal matrix with diagonal entries $\mathbf{x}$.
$\mathrm{tr}(M)$	trace of matrix M.
$\mathbf{0}$	vector with all components zeros.
$\mathbf{1}$	vector with all components one.

Norms and distances

$\|\mathbf{x}\|$	Euclidean norm of vector $\mathbf{x}$.
$\|\mathbf{x}\|^2$	squared Euclidean norm of vector $\mathbf{x}$.
$\|\mathbf{x}\|^2_Q$	weighted squared norm of vector $\mathbf{x}$ with weights matrix Q.
$\|\mathbf{x}\|_\infty$	infinity norm of vector $\mathbf{x}$.
$\mathrm{dist}(\mathcal{A}, \mathcal{B})$	distance between sets (or points) $\mathcal{A}$ and $\mathcal{B}$.

Functions and derivatives

$f : \mathcal{A} \to \mathcal{B}$	f is a function mapping the set $\mathcal{A}$ into the set $\mathcal{B}$.
∇f	gradient of function f.
$\nabla^2 f$	Hessian of function f.
$\dot{\mathbf{x}}$, $\frac{d\mathbf{x}}{dt}$, or $\frac{\partial \mathbf{x}}{\partial t}$	derivative of $\mathbf{x}$ w.r.t time t.
$L_f h(\mathbf{x})$	Lie derivative, $L_f h(\mathbf{x}) = f(\mathbf{x})\frac{\partial h(\mathbf{x})}{\partial t}$.

Probability theory

$p(\cdot)$	probability density function.
$\Pr(\cdot)$	probability of an event.
$\mathbb{E}(\cdot)$	expectation of a random variable.
$\mathbb{V}(\cdot)$	Variance of a random variable.
$\hat{\mathbf{x}}$	mean of random variable $\mathbf{x}$.
δ	probability threshold.
$\mathcal{N}(\boldsymbol{\mu}, \Sigma)$	Gaussian distribution with mean $\boldsymbol{\mu}$ and variance Σ.

Model predictive control

N	number of time steps in receding horizon planning.
Δt	time step.
τ	planning horizon length, $\tau = N\Delta t$.
$.^k$	the super index k indicates value of the variable at stage k.
$J^k(\cdot)$	the k-th stage cost.
$J^N(\cdot)$	the terminal stage cost.

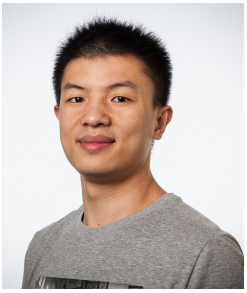
Multi-robot system

d	dimension of the workspace, $d \in \{2, 3\}$.
$\mathcal{W}$	workspace of the multi-robot system, $\mathcal{W} \subseteq \mathbb{R}^d$.
n	number of robots in the multi-robot system, $n \in \mathbb{N}$.
$\mathcal{I}$	the set of robots, $\mathcal{I} = \{1, \dots, n\} \subset \mathbb{N}$.
i, j	the i-th, j-th robot, $i, j \in \mathcal{I}$.
$\mathbf{x}_i, \mathcal{X}_i$	robot i's state and state space, $\mathbf{x}_i \in \mathcal{X}_i \subset \mathbb{R}^{n_x}$.
$\mathbf{u}_i, \mathcal{U}_i$	robot i's control and control space, $\mathbf{u}_i \in \mathcal{U}_i \subset \mathbb{R}^{n_u}$.
$\mathbf{f}_i$	the function to model robot i's dynamics.
$\mathbf{p}_i, \mathbf{v}_i, \mathbf{a}_i$	robot i's position, velocity and acceleration, $\mathbf{p}_i, \mathbf{v}_i, \mathbf{u}_i \in \mathbb{R}^d$.
n_o	number of obstacles in the workspace, $n_o \in \mathbb{N}$.
$\mathcal{I}_o$	the set of obstacles, $\mathcal{I}_o = \{1, \dots, n_o\} \subset \mathbb{N}$.
o	the o-th obstacle, $o \in \mathcal{I}_o$.
$\mathbf{p}_o, \mathbf{v}_o$	obstacle o's position and velocity, $\mathbf{p}_o, \mathbf{v}_o \in \mathbb{R}^d$.
$\mathcal{T}_i^0$	robot i's current planned trajectory, $\mathcal{T}_i^0 = \{\mathbf{x}_i^{1:N}\}$.

List of abbreviations

MAV	micro aerial vehicle
MPC	model predictive control
NMPC	nonlinear model predictive control
DMPC	distributed model predictive control
CCNMPC	chance-constrained nonlinear model control
CCO	chance-constrained optimization
CVM	constant velocity model
KF	Kalman filter
EKF	extended Kalman filter
UKF	unscented Kalman filter
MVG	multivariate Gaussian distribution
GMM	Gaussian mixture model
PDF	probability density function
CDF	cumulative density function
VC	Voronoi cell
BVC	buffered Voronoi cell
UAVC	uncertainty-aware Voronoi cell
B-UAVC	buffered uncertainty-aware Voronoi cell
VO	velocity obstacle
RVO	reciprocal velocity obstacle
ORCA	optimal reciprocal collision avoidance
CBF	control barrier function
SBC	safety barrier certificates
CC-SBC	chance-constrained safety barrier certificates
MDP	Markov decision process
POMDP	Partially observable Markov decision process
RL	reinforcement learning
NN	neural network
DNN	deep neural network
CNN	convolutional neural network
RNN	recurrent neural network
ROS	robot operating system

Curriculum vitæ



Hai ZHU was born in March 1993 in Hubei, China. He obtained the B.Sc. and M.Sc. degrees in Aerospace Engineering from the National University of Defense Technology, Changsha, China in 2015 and 2017, respectively. From July to October in 2014, he was a Mitacs Globalink research intern in the Spacecraft Dynamics Control and Navigation Laboratory (SDCNLab), York University, Toronto, Canada, supervised by Prof. Jinjun Shan.

In October 2017, he was sponsored by the Chinese Scholarship Council (CSC) to become a PhD candidate at the Department of Cognitive Robotics, Delft University of Technology, Delft, The Netherlands. In his PhD project, he worked on probabilistic motion planning for multi-robot systems under uncertainty, under the supervision of Dr. Javier Alonso-Mora and Prof. Robert Babuska. During summer 2020, he spent three months in the Autonomous System Lab (ASL), ETH Zurich, Zurich, Switzerland, as a visiting scholar to conduct research on online informative path planning for active information gathering, supervised by Dr. Jen Jen Chung and Dr. Nicholas Lawrance. He received the IEEE ICRA Best Paper Award on Multi-Robot Systems in 2019.

His research interests include robot motion planning, planning under uncertainty, multi-robot coordination, model predictive control, and learning-based control.

List of publications

Referred journals

- Á. Serra-Gómez, **H. Zhu**, B. Brito, W. Böhmer, and J. Alonso-Mora, "Learning scalable and efficient communication policies for multi-robot collision avoidance," submitted to *The International Journal of Robotics Research*, under review.
- C. Salmi, L. Knoedler, **H. Zhu**, B. Brito, and J. Alonso-Mora, "Improving pedestrian prediction models with self-supervised continual learning," submitted to *IEEE Robotics and Automation Letters*, under review.
- **H. Zhu**, B. Brito, and J. Alonso-Mora, "Decentralized probabilistic multi-robot collision avoidance using buffered uncertainty-aware voronoi cells," *Autonomous Robots*, online available.
- **H. Zhu***, F.M. Claramunt*, B. Brito, and J. Alonso-Mora, "Learning interaction-aware trajectory predictions for decentralized multi-robot motion planning in dynamic environments," *IEEE Robotics and Automation Letters*, 6(2):2256-2263, Apr. 2021.
- Q. Chen*, **H. Zhu***, L. Yang, X. Chen, S. Pollin, and E. Vinogradov, "Edge computing assisted autonomous flight for uav: Synergies between vision and communications," *IEEE Communications Magazine*, 59(1):28-33, Jan. 2021.
- **H. Zhu**, and J. Alonso-Mora, "Chance-constrained collision avoidance for mavs in dynamic environments," *IEEE Robotics and Automation Letters*, 4(2):776-783, Apr. 2019.

Referred conference proceedings

- **H. Zhu**, and J. Alonso-Mora, "Chance-constrained safety barrier certificates for multi-robot collision avoidance under uncertainty," submitted to *2022 IEEE International Conference on Robotics and Automation (ICRA)*, under review.
- A. Ray, A. Pierson, **H. Zhu**, J. Alonso-Mora, and D. Rus, "Multi-robot task assignment for aerial tracking with viewpoint constraints," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2021.
- **H. Zhu**, J.J. Chung, N.R.J. Lawrance, R. Siegwart, and J. Alonso-Mora, "Online informative path planning for active information gathering of a 3d surface," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2021.
- B. Brito, **H. Zhu**, W. Pan, and J. Alonso-Mora, "Social-vrnn: One-shot multi-modal trajectory prediction for interacting pedestrians," in *Conference on Robot Learning (CoRL)*, Nov. 2020.
- Á. Serra-Gómez, B. Brito, **H. Zhu**, J.J Chung, and J. Alonso-Mora, "With whom to communicate: Learning efficient communication for multi-robot collision avoidance," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2020.

* indicates equal contributions.

- J. Lin*, **H. Zhu***, and J. Alonso-Mora, "Robust vision-based obstacle avoidance for micro aerial vehicles in dynamic environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2020.
- **H. Zhu**, and J. Alonso-Mora, "B-uavc: Buffered uncertainty-aware voronoi cells for probabilistic multi-robot collision avoidance," in *Proceedings of the IEEE International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, Aug. 2019.
- **H. Zhu**, J. Juhl, L. Ferranti, and J. Alonso-Mora, "Distributed multi-robot formation splitting and merging in dynamic environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2019. **Best Paper Award on Multi-Robot Systems.**