Risk-Aware Planning and Control in Extreme Environments

Thesis by

Anushri Dixit

In Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2023
Defended 2nd February, 2023

Anushri Dixit
ORCID: 0000-0002-9698-2189

ACKNOWLEDGEMENTS

I would like to thank my advisor, Prof. Joel Burdick, for his persistent support throughout this Ph.D. My journey through graduate school has not been easy, but I've always known I could trust Joel to have my back. His deep knowledge and perspective on robotics problems never ceases to amaze me. I can only hope to be as incredible a mentor as Joel one day.

I would also like to thank my thesis committee - Profs. Aaron Ames, Richard Murray, Soon-Jo Chung, and Eric Mazumdar. I have always come out of our meetings about my research progress with new ideas and fresh directions.

Thank you to my group members and office mates across the years - Amanda Bouman, Skylar Wei, Thomas Touma, Daniel Pastor Moreno, Carl Folkestad, Ellen Novoseller, Jeff Edlund, and Richard Cheng for fun research discussions and maintaining a constant flow of coffee in the office.

Thanks to Sonya Lincoln: your support has made planning all the travel for conferences and field trips so much easier.

I would like to thank my collaborators, especially everyone from JPL Team CoSTAR. The field experience that I have shared with all members has been a crucial part of my growth as a researcher and the work with everyone on SubT has given me a new outlook on robotics research. I want to especially thank Drs. Ali Agha, Kyon Otsu, David Fan, Matt Anderson, Ben Morrell, Mike Wolf, and Brett Lopez - for their patience to teach me how to develop resilient robots and their mentoring during various stages of my graduate school career.

I want to thank Dr. Reza Ahmadi: it was a pleasure to work with you and have you as my mentor. You have always been willing to talk about research ideas on any topic: be it control of tracked vehicles, SOS tools, differential inclusions, or risk-aware control. Thank you Profs. Lars Lindemann and Margaret Chapman for being great collaborators. Lars, it's been such fun to talk with you about academia, risk-aware control, SoCal and everything in between.

I would not be writing this thesis without my friends' support. Nachiket Naik, Ng Cai Tong, and Irene Dutta - you guys are the best roommates and friends I could've asked for to share this journey through graduate school with. I will always fondly remember our many chats over coffee at Red Door. I also want to thank my friends

from graduate school - Widi Moestopo, Lena Maxey, Prithvi Akella, Jagganadh Kumar, Maegan Tucker, Gunho Kim, Nikhilesh Alatur, and Carmen Amo Alonso and my friends from undergrad - Tanvi Kulkarni, Malavika Bindhi, and Aneri Muni.

Words cannot express how thankful I am to have my family's love and support through this endeavor, but I will try. First and foremost, thanks mom and dad - I would not be here without your unwavering faith in me through thick and thin. I aspire to be like both of you - a fantastic educator like mom, who was my first (and best) math teacher, and a great engineer like dad, whose practical skills and creativity have been amazing to witness growing up. Aditi and Ameya: you have been there for me when I've missed home and needed to be reminded to have some fun. Sid, my partner-in-crime, I am so grateful for your unending love and encouragement to be the best, most authentic version of myself. Thank you all for believing in me, especially on the days that I didn't. Finally, this thesis is dedicated to my grandfather. Ajoba, I miss you every day and wish you were here to read this manuscript.

ABSTRACT

Safety-critical control and planning for autonomous systems operating in unstructured environments is a challenging problem must be addressed as autonomous vehicles, surgical robots, and autonomous industrial robots become more pervasive. This thesis addresses some of the issues in safety critical autonomy by introducing new techniques for computationally tractable and efficient safety-critical control. The approach developed in this thesis arises from taking a deeper look at two questions: 1) How can we obtain better uncertainty quantification of the disturbances that affect autonomous systems either as a result of unmodeled changes in the environment or due to sensor imperfections? 2) Given richer uncertainty quantification techniques, how do incorporate the diverse uncertainty descriptions into the control and planning framework without sacrificing the tractability and efficiency of existing approaches?

I address the above two questions by developing risk-aware control and planning techniques for traversal of a mobile robot over static but extreme terrain and in the presence of dynamic obstacles. We first look at algorithms for risk-aware terrain assessment, and extensively test them on wheeled and legged robots that were deployed in subterranean tunnel, urban, and cave environments for search and rescue operations in the DARPA Subterranean Challenge. I then present a theory for risk-aware model predictive control in static environments and in the presence of dynamic obstacles. Coherent risk measures are applied to this planning and control framework in order to account for diverse uncertainty descriptions. Computationally tractable reformulations of the optimal control problem are realized through constraint tightening techniques.

I then investigate algorithms for uncertainty assessment and prediction of apriori unknown, dynamic obstacles using data-driven techniques. We use a technique from signal processing literature called Singular Spectrum Analysis for making linear predictions of dynamic obstacles. The obstacle motion predictions are equipped with error predictions to account for the uncertainty in the sensing heuristically using bootstrapping techniques. We use a statistical tool, Adaptive Conformal Inference, to further calibrate the heuristic error prediction online to obtain true uncertainty prediction while using nonstationary data to analyze the performance of the data-driven predictor. These techniques provide reactive, real-time, risk-aware obstacle avoidance in dynamic environments.

PUBLISHED CONTENT AND CONTRIBUTIONS

[1] Ali Agha, Kyohei Otsu, Benjamin Morrell, David D. Fan, Rohan Thakker, Angel Santamaria-Navarro, Sung-Kyun Kim, Amanda Bouman, Xianmei Lei, Jeffrey Edlund, Muhammad Fadhil Ginting, Kamak Ebadi, Matthew Anderson, Torkom Pailevanian, Edward Terry, Michael Wolf, Andrea Tagliabue, Tiago Stegun Vaquero, Matteo Palieri, Scott Tepsuporn, Yun Chang, Arash Kalantari, Fernando Chavez, Brett Lopez, Nobuhiro Funabiki, Gregory Miles, Thomas Touma, Alessandro Buscicchio, Jesus Tordesillas, Nikhilesh Alatur, Jeremy Nash, William Walsh, Sunggoo Jung, Hanseob Lee, Christoforos Kanellakis, John Mayo, Scott Harper, Marcel Kaufmann, Anushri Dixit, Gustavo Correa, Carlyn Lee, Jay Gao, Gene Merewether, Jairo Maldonado-Contreras, Gautam Salhotra, Maira Saboia Da Silva, Benjamin Ramtoula, Yuki Kubo, Seyed Fakoorian, Alexander Hatteland, Taeyeon Kim, Tara Bartlett, Alex Stephens, Leon Kim, Chuck Bergh, Eric Heiden, Thomas Lew, Abhishek Cauligi, Tristan Heywood, Andrew Kramer, Henry A. Leopold, Chris Choi, Shreyansh Daftry, Olivier Toupet, Inhwan Wee, Abhishek Thakur, Micah Feras, Giovanni Beltrame, George Nikolakopoulos, David Shim, Luca Carlone, and Joel Burdick. NeBula: Quest for robotic autonomy in challenging environments; TEAM CoSTAR at the DARPA subterranean challenge. Field Robotics, 2021. URL `https://arxiv.org/abs/2103.11470`.
Contribution: Traversability anaysis and low-level motion planning.

[2] Mohamadreza Ahmadi, Anushri Dixit, Joel W. Burdick, and Aaron D. Ames. Risk-averse stochastic shortest path planning. In Conference on Decision and Control, 2021. URL `https://arxiv.org/abs/2103.14727`.
Contribution: Helped with proofs and paper writing.

[3] Prithvi Akella, Anushri Dixit, Mohamadreza Ahmadi, Joel W. Burdick, and Aaron D. Ames. Sample-Based Bounds for Coherent Risk Measures: Applications to Policy Synthesis and Verification. arXiv, 2022. doi: 10.48550/ARXIV.2204.09833. URL `https://arxiv.org/abs/2204.09833`.
Contribution: Helped with theory development.

[4] Sharmita Dey, David Fan, Robin Schmid, Anushri Dixit, Kyohei Otsu, Thomas Touma, Arndt F. Schilling, and Ali akbar Aghamohammadi. PrePARE: predictive proprioception for agile failure event detection in robotic exploration of extreme terrains. In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022.
Contribution: Performed experiments.

[5] Anushri Dixit, M. Ahmadi, and J. W. Burdick. Risk-Sensitive Motion Planning using Entropic Value-at-Risk. In European Control Conference, 2021. URL

https://arxiv.org/abs/2011.11211.
Contribution: Developed theory, algorithm, and code.

[6] Anushri Dixit, Mohamadreza Ahmadi, and Joel W. Burdick. Distributionally robust model predictive control with total variation distance. IEEE Control Systems Letters, 6:3325–3330, 2022. doi: 10.1109/LCSYS.2022.3184921.
Contribution: Developed theory, algorithm, and code.

[7] Anushri Dixit, Mohamadreza Ahmadi, and Joel W. Burdick. Risk-Averse Receding Horizon Motion Planning. 2022.
Contribution: Developed theory, algorithm, and code.

[8] Anushri Dixit*, David D. Fan*, Kyohei Otsu, Sharmita Dey, Ali-Akbar Agha-Mohammadi, and Joel W. Burdick. STEP: Stochastic Traversability Evaluation and Planning for Risk-Aware Off-road Navigation; Results from the DARPA Subterranean Challenge, 2022.
Contribution: Developed algorithm and code; field testing and data analysis.

[9] Anushri Dixit*, Lars Lindemann*, Skylar Wei, Matthew Cleaveland, George J. Pappas, and Joel W. Burdick. Adaptive conformal prediction for motion planning among dynamic agents. In Submitted to Learning for Dynamics and Control (L4DC) Conference, 2022. URL https://arxiv.org/pdf/2212.00278.pdf.
Contribution: Developed theory, algorithm, and code.

[10] David D. Fan*, Kyohei Otsu*, Yuki Kubo, Anushri Dixit, Joel Burdick, and Ali-Akbar Agha-Mohammadi. STEP: Stochastic traversability evaluation and planning for safe off-road navigation. In Robotics: Science and Systems, 2021. URL https://arxiv.org/abs/2103.02828.
Contribution: Modeled risk-aware traversability tools; code writing; helped with geometric planner development.

[11] Skylar X. Wei*, Anushri Dixit*, Shashank Tomar, and Joel W. Burdick. Moving obstacle avoidance: A data-driven risk-aware approach. IEEE Control Systems Letters, 7:289–294, 2022. doi: 10.1109/LCSYS.2022.3181191.
Contribution: Problem design; developed theory for risk-aware planning; code debugging.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

Chapter 1

INTRODUCTION

Providing safety and performance guarantees for motion planning and control algorithms is an important prerequisite for developing trustworthy robots. While this is a well-studied problem with tractable algorithms for real-time implementation when the robot and environment models are exactly known, the guarantees fail in real-world, unstructured settings. For example - autonomous vehicles are highly successful in clear driving conditions. But unlike human drivers, autonomous vehicles are not as adaptable to sudden environmental changes (like weather or other erratic drivers). How does one translate this human-like intelligence into principles for autonomous robots so that they can operate successfully, not only in structured environments, but also in the real world?

Simplified models of robot dynamics and of the environment are very useful because they make the control and planning algorithms tractable and allow real-time implementation. However, when robots operate in a real-world setting where the environment is dynamic and unstructured, common distributional assumptions used to develop the planning algorithms are no longer valid and consequently, the safety guarantees no longer hold. The work described in this thesis takes steps to close the gap between the simplifying assumptions made by control and planning algorithms and the real world to account for a wider class of uncertainty descriptions while retaining the tractability of the current state-of-the-art approaches. The process of bridging this gap is tackled first by modeling the uncertainty in the robot's perception in static and dynamic environments, and then accounting for this uncertainty in the interpretation of the environment by using a risk-aware approach in the motion planning framework.

1.1 Uncertainty modeling

In a first step to get a richer risk assessment for autonomous robots, we must understand where the uncertainty in planning is arising from. Consider a robot operating in a post-earthquake industrial warehouse. The environments which are of interest to robotic operations are highly risky, containing difficult geometries (e.g. rubble, slopes) and non-forgiving hazards (e.g. large drops, sharp rocks). Determining where the robot may safely travel is a non-trivial problem, compounded

by several issues: 1) Sensor noise, sparsity, and occlusion induces biases and uncertainty in analysis of safety. 2) Environments often pose a mix of various sources of static risk, including slopes, rough terrain, low traction, narrow passages, etc. 3) The motion of dynamic obstacles (e.g. falling debris) is often unknown a priori and requires reactive behaviors.

Let's first consider motion planning over completely unknown static, extreme terrain. Motion planning over uneven terrain given only sensor measurements requires classification of the terrain as *traversable* or not. Most traversability analyses are dependent on sensor type and measured through geometry-based, appearance-based, or proprioceptive methods [106]. Geometry-based methods often rely on building a 2.5D terrain map which is used to extract features such as maximum, minimum, and variance of the height and slope of the terrain [56]. Planning algorithms for such methods take into account the stability of the robot on the terrain [61]. In [52, 104], the authors estimate the probability distributions of states based on the kinematic model of the vehicle and the terrain height uncertainty. Some methods incorporate sensor and state uncertainty to obtain a probabilistic terrain estimate in the form of a grid-based elevation map like in [43].

Now let's consider environments with dynamic, moving obstacles. One needs to be able to predict the motion of the obstacle and incorporate it into the planning framework. One way to account for obstacle motion is to assume a limit on their modeled motions. For example, in [132], the authors assumed a priori knowledge of the obstacle dynamics or motion patterns. The authors of ARMTD [67] and CHOMP [109] provided an offline planner while assuming that the obstacles are quasi-static. Or, one can plan the agent's path off-line using a Probabilistic Roadmap (PRM) in a field of static obstacles and then replan when dynamical behaviors are observed [137]. However, without prior knowledge of an obstacle's behavior, a worst-case analysis of unsafe obstacle locations can lead to conservative behavior. Potential fields (PFs) are actively used for dynamic obstacle avoidance: e.g., recent works [80] apply artificial PFs with stochastic reachable sets in Human-Centered environments. Slow moving and simple (linear or double integrator-like) dynamics are assumed. Switching-based planning methods detect and classify dynamic obstacle behavior against a set of trajectories, such as constant speed, linear, and projectile-like motion [69, 85]. Classification-based methods require distinguishable obstacle behaviors and prior knowledge about the dynamic environment to generate a set of trajectories.

While many prior works assume perfect knowledge of the environment, an important challenge is to also account for uncertainty in perception. Existing works address the problem by making simplifying assumptions, such as linear system dynamics and bounded or Gaussian uncertainty distributions [10, 110, 130]. However, addressing the problem in its full generality for nonlinear dynamics and arbitrary distributions is an open problem. In this thesis, we consider uncertainty quantification and risk assessment in static but extreme environments and in the presence of unknown moving obstacles.

1.2 Risk-aware Control

When uncertainty in the interpretation of the environment is modeled using approximate sensor models and data-driven methods, we obtain environment (or uncertainty) models that are constantly changing. If we assume that the distribution of the uncertainty is known a priori - the motion planning techniques yield paths that are safe only when the world behaves exactly as modeled. Instead, this work incorporates the changing uncertainty distributions within the planning and control pipeline to be able to provide safety guarantees even when the world does not behave as modeled.

Model Predictive Control (MPC) is widely used for robotic motion planning because it incorporates robot dynamics and state and control constraints in a receding horizon fashion [21, 97]. There are many ways to incorporate uncertainty in MPC. Robust MPC accounts for worst-case disturbances in a set of bounded uncertainties [16]. This approach is often too conservative, since it does not account for the distribution of the uncertainties. Stochastic MPC [93] minimizes the expected value of a cost function, while respecting a bound on the probability of violating the state and control constraints. The chance constraints in stochastic MPC do not usefully account for events in the tail of the uncertainty distribution, and the policy that results from an expected cost function minimizes the cost on average. In this thesis, we optimize for policies that have risk-averse behavior: the policies are not as conservative as in the robust case but account for "risky" outcomes in the tail of the uncertainty distribution and therefore perform better in practice.

There are many ways to incorporate risk into a control strategy [70], such as chance constraints [64, 103], exponential utility functions [78], and distributional robustness [30, 144]. However, applications in autonomy and robotics require more "nuanced assessments of risk" [92]. Artzner *et. al.* [12] characterized a set of *co-*

herent risk measures that have natural and desirable properties. This thesis focuses on these risk measures, which are widely used in finance and operations research, among other fields. Some examples of coherent risk measures are Conditional Value-at-Risk (CVaR) and Entropic Value-at-Risk (EVaR).

Motion planning based on coherent risk measures has previously been considered. In [123], the authors provided an MPC scheme for a discrete-time dynamical system with process noise whose objective was a CVaR measure. They further provided Lyapunov conditions for risk-sensitive exponential stability. In [62], the authors devised an MPC scheme to avoid randomly moving obstacles using a CVaR risk metric. Similar results were obtained in [37] on EVaR metric for obstacle avoidance with additional guarantees of recursive feasibility and finite-time task completion while following a set of waypoints. Risk-sensitive obstacle avoidance has also been tackled through CVaR-based control barrier functions in [5] with application to bipedal robot locomotion. In [125], the authors considered multistage risk-averse and risk-constrained optimal control for general coherent risk measures with conic representations. A scenario tree-based branch MPC framework with feedback policies that account for a tradeoff between robustness and performance through CVaR metrics was proposed in [31].

Recently, there has been a push to develop motion planning techniques that are robust to a set of probability distributions within which the true distribution of uncertainty lies. This is called distributionally robust motion planning. It is important to note that coherent risk measures have heretofore provided distributional robustness in the cost, but not in the chance constraints that one may need to satisfy in SMPC. This thesis also extends the use of coherent risk measures to provide distributional robustness in the chance constraints.

Distributionally robust chance constraints (DRCCs) have been well studied in stochastic optimization. A popular metric for enforcing distributional robustness is the Wasserstein distance. In [143] the author proposed a tight inner and outer approximation of the DRCC with a Wasserstein ambiguity set using a CVaR reformulation. In [33], the authors enforced DRCCs with Wasserstein distance ambiguity set in an MPC setting. Optimal control using distributionally robust CVaR constraints with second-order moment ambiguity sets can be posed as a semidefinite program in [138].

Clearly, there is a need to quantify the risk in our modeling assumptions based on the robot's perception of the environment. Towards this end, we first consider risk-

aware traversability analysis in static, extreme environments. Next, we develop a theoretical framework for risk-aware model predictive control that is distributionally-robust to disturbances. Finally, we look at prediction of dynamic obstacles with uncertainty quantification and risk-aware obstacle avoidance.

1.3 Thesis Structure

Chapter 2: Preliminaries

We provide the necessary background on coherent risk measures and model predictive control - topics that are used repeatedly in the following sections.

Chapter 3: Risk-Aware Planning in a Static Environment

We introduce a risk-aware traversability and planning framework for a static environment with extensive field results. The use of the mean value of the cost often accounts for the average measurement, but not the cost with low probability of occurrence that can be very risky for the robot. We model the traversability estimate as a Gaussian random variable (for each grid cell) with the variance given by the sensor uncertainty and the expected value calculated using different sources of risk like - obstacle height, slope, step height, negative obstacles, water/mud, etc. The random variable is then used to compute a Conditional Value-at-Risk (CVaR) cost map. Unlike other traversability cost analysis techniques, our method implicitly accounts for high risk, low probability events using CVaR. The CVaR cost is used in the geometric and local planner to plan paths over extreme terrain that account for multiple sources of geometric, confidence-based, and semantic risk. This framework was deployed in the DARPA Subterranean Challenge.

Chapter 4: Distributionally-robust MPC with Total Variation Distance

We consider the thoery behind risk-aware model predictive control in static environments with distributionally-robust chance constraints and compare the complexity of the control problem to state-of-the-art model predictive control techniques. We show that obtaining distributional robustness in the constraints for Total Variation Distance ambiguity sets is equivalent to a simple tightening of the chance constraint. The constraint tightenings provide intuitive approximations of the MPC optimization that reduce the number of optimization variables so as to make the complexity of the distributionally-robust MPC comparable to stochastic MPC.

Chapter 5: Risk-Aware MPC with dynamic obstacles

We study the risk-aware control problem in the presence of dynamic obstacles, whose uncertainty description is known apriori, using MPC. We develop an obstacle-avoidance MPC framework that formulates the obstacle avoidance constraint using coherent risk measures. To handle disturbances, or process noise, in the state dynamics, the state constraints are tightened in a risk-aware manner to obtain a disturbance feedback policy (a policy that is reactive to the disturbances that the robot will experience in the future) that provides risk-sensitive recursive feasibility while (probabilistically) guaranteeing finite-time task completion.

Chapter 6: Data-driven Prediction and Risk-Aware MPC with unknown obstacles

We look at the data-driven prediction of apriori unknown obstacles and provides a risk-aware MPC framework to obtain reactive control strategies for dynamic obstacle avoidance purely from online data. Compared to state-of-the-art techniques, our method replaces the need for obstacle trajectory/model classification while allowing online predictions that allow for highly reactive behavior when the predictions are incorporated in a multi-step planning framework. Extracting a dynamics model from data is challenging, especially when the available data is limited, noisy, and partial. We use Singular Spectrum Analysis to separate noise from the underlying signal and to extract a predictive distribution of obstacle behavior. We generate a bootstrap of predictive models and forecast a set of obstacle trajectories. The bootstrapped forecasts are treated as a distribution of possible obstacle models at each time step and used in a model predictive control (MPC) framework in a distributionally-robust manner. The MPC optimization incorporates distributionally-robust chance constraints such that the optimization is robust to all distributions with the mean and variance of the bootstraps.

Chapter 7: Data-driven MPC using Adaptive Conformal Prediction

We provide an uncertainty quantification technique for the above data-driven predictor to be able to provide distribution-free uncertainty region predictions using conformal inference, a data-driven technique to get calibrated prediction sets from the output of a black-box predictor. Hence, we supplement the data-driven, dynamic obstacle predictions with true coverage guarantees.

Chapter 8: Conclusion

We list the contributions of this thesis and provides some future directions that could improve the work presented in this thesis.

Chapter 2

PRELIMINARIES

We formally consider a probability space $(\Omega, \mathcal{F}, P)$, where Ω, $\mathcal{F}$, and P are the sample space, σ-algebra over Ω, and probability measure over $\mathcal{F}$ respectively. A random variable $X : \Omega \to \mathbb{R}$ denotes the cost of each outcome. The set of all cost random variables defined on Ω is given by $\mathcal{X}$. We use $\boldsymbol{X} \in \mathbb{R}^n$ to denote a random vector of length n.

2.1 Distributionally Robust Optimization

Let $\boldsymbol{u} \in \mathcal{U} \subset \mathbb{R}^{n_u}$ be the decision vector of a stochastic program with the random vector $\boldsymbol{X}$. A classical stochastic program is given by,

$$\begin{aligned} \inf_{\boldsymbol{u}\in\mathcal{U}} \quad & \mathbb{E}[g(\boldsymbol{X}, \boldsymbol{u})] \\ \text{s.t.} \quad & P\Big(h(\boldsymbol{X}, \boldsymbol{u}) > 0\Big) \leq \epsilon, \end{aligned}$$

where the functions $g(\cdot), h(\cdot) \in \mathbb{R}$ are the cost random variable and the optimization constraints respectively.

When the uncertainty is bounded, we can write a classical robust optimization program as,

$$\begin{aligned} \inf_{\boldsymbol{u}\in\mathcal{U}} \quad & \sup_{\boldsymbol{X}\in\mathcal{X}} \quad g(\boldsymbol{X}, \boldsymbol{u}) \\ \text{s.t.} \quad & h(\boldsymbol{X}, \boldsymbol{u}) > 0, \quad \forall \boldsymbol{X} \in \mathcal{X}. \end{aligned}$$

We consider a stochastic optimization problem for which the distribution of random variables is ambiguous and lies in a set of probability measures called an *ambiguity set*. The distributionally robust optimization program is hence given by,

$$\begin{aligned} \inf_{\boldsymbol{u}\in\mathcal{U}} \quad & \sup_{Q\in\mathcal{Q}} \quad \mathbb{E}_Q[g(\boldsymbol{X}, \boldsymbol{u})] \\ \text{s.t.} \quad & Q\Big(h(\boldsymbol{X}, \boldsymbol{u}) > 0\Big) \leq \epsilon, \quad \forall Q \in \mathcal{Q}, \end{aligned}$$

where $\mathcal{Q}$ is the set of probability measures that described the ambiguity set.

2.2 Risk Measures

A risk measure is a function that maps a cost random variable to a real number, $\rho : \mathcal{X} \to \mathbb{R}$. For constrained stochastic optimization programs, chance constraints can be reformulated by a commonly used risk measure called the *Value-at-Risk* (VaR). For a given confidence level $\alpha \in (0, 1)$, $\text{VaR}_{1-\alpha}$ denotes the $(1 - \alpha)$-quantile value of the cost variable X and is defined as,

$$\text{VaR}_{1-\alpha}(X) := \inf\{z \mid \mathbb{P}(X \leq z) \geq \alpha\}.$$

It follows that $\text{VaR}_{1-\alpha}(X) \leq 0 \implies \mathbb{P}(X \leq 0) \geq \alpha$. However, VaR is generally nonconvex and hard to compute. We now introduce convex and monotonic risk measures. In particular, we are interested in coherent risk measures [12] that satisfy the following properties.

Definition 1 (Coherent Risk Measures) *Consider two random variables, X, $X' \in \mathcal{X}$. A coherent risk measure, $\rho : \mathcal{X} \to \mathbb{R}$, is a risk measure that satisfies the following properties:*

1. ***Monotonicity*** $X \leq X' \implies \rho(X) \leq \rho(X')$,
2. ***Translational invariance*** $\rho(X + c) = \rho(X) + c$, $\forall c \in \mathbb{R}$,
3. ***Positive homogeneity*** $\rho(\alpha X) = \alpha\rho(X)$, $\forall \alpha \geq 0$,
4. ***Subadditivity*** $\rho(X + X') \leq \rho(X) + \rho(X')$.

Another nice property of coherent risk measures is that they can be written as the worst-case expectation over a convex, bounded, and closed set of probability mass (or density) functions (pdf/pmf). This is the dual representation of a risk measure, and this set is referred to as the *risk envelope*.

Definition 2 (Representation Theorem [12]) *Every coherent risk measure can be represented in its dual form as,*

$$\rho(X) := \sup_{Q \in \mathcal{Q}} \mathbb{E}_Q(X),$$

where the risk envelope $\mathcal{Q} \subset \{Q \ll P\}$ is convex and closed.

The above theorem gives a distributionally robust interpretation of coherent risk measures. We will discuss distributionally optimization in the next section.

While coherent risk measures act on a one-dimensional cost random variable, in this thesis, we write $\rho(\boldsymbol{X})$, where $\boldsymbol{X}$ is a vector of cost random variables of length n, to mean $\rho(\boldsymbol{X}) = \Big[\rho(X_1), \ldots, \rho(X_n)\Big]^T$.

Some examples of coherent risk measures and their dual representation are reviewed next. These example measures will be used in subsequent chapters.

Conditional Value-at-Risk

For a given confidence level $\alpha \in (0, 1)$, value-at-risk $\text{VaR}_{1-\alpha}$ denotes the $(1-\alpha)$-quantile value of the cost variable $X \in \mathcal{L}_p(\Omega, \mathcal{F}, \mathbb{P})$. The conditional value-at-risk $\text{CVaR}_{1-\alpha}$ measures the expected loss in the $(1-\alpha)$-tail given that the threshold $\text{VaR}_{1-\alpha}$ has been crossed. $\text{CVaR}_{1-\alpha}$ is found as

$$\text{CVaR}_{1-\alpha}(X) := \inf_{z \in \mathbb{R}} \mathbb{E}\left[z + \frac{(X-z)^+}{1-\alpha}\right], \tag{2.1}$$

where $(\cdot)^+ = \max\{\cdot, 0\}$. A value of $\alpha \simeq 0$ corresponds to a risk-neutral case; whereas, a value of $\alpha \to 1$ is rather a risk-averse case. $\mathcal{Q}$ is the risk envelope defined by,

$$\mathcal{Q} := \left\{Q \ll P \,|\, 0 \leq \frac{dQ}{dP} \leq \frac{1}{\alpha}\right\}, \tag{2.2}$$

where $\frac{dQ}{dP}$ is called the Radon–Nikodym derivative and it gives the rate of change of density of one density function, Q, w.r.t the other, P. For a discrete probability distribution, the risk envelope translates to

$$\mathcal{Q} := \left\{q \in \Delta_J \,|\, 0 \leq q(j) \leq \frac{p(j)}{\alpha} \,\forall j \in \{1, \ldots, J\}\right\} \tag{2.3}$$

where Δ_J is the probability simplex, $\Delta_J := \{q \in \mathbb{R}^J \,|\, q \geq 0,\ \sum_{j=1}^J q(j) = 1\}$.

CVaR provides a convex upper bound of VaR, or equivalently, a convex inner approximation of a chance constraint, i.e.,

$$\text{VaR}_{1-\alpha}(X) \leq \text{CVaR}_{1-\alpha}(X) \leq 0 \implies \mathbb{P}(X \leq 0) \geq \alpha. \tag{2.4}$$

Entropic Value-at-Risk

EVaR, derived using the Chernoff inequality for VaR, is the tightest upper bound for VaR and CVaR. The $\text{EVaR}_{1-\alpha}$ of random variable X is given by

$$\text{EVaR}_{1-\alpha}(X) := \inf_{z>0}\left[z^{-1}\ln\frac{\mathbb{E}[e^{Xz}]}{1-\alpha}\right]. \tag{2.5}$$

Similar to $\text{CVaR}_{1-\alpha}$, for $\text{EVaR}_{1-\alpha}$, the limit $\alpha \to 0$ corresponds to a risk-neutral case; whereas, $\alpha \to 1$ corresponds to a risk-averse case. In fact, it was demonstrated in [7, Proposition 3.2] that $\lim_{\alpha\to 1}\text{EVaR}_{1-\alpha}(X) = \text{ess sup}(X)$, where $\text{ess sup}(X)$ is the worst case value of X.

For EVaR, the risk envelope $\mathcal{Q}$ for a continuous random variable with the pdf P is defined as the epigraph of the KL divergence,

$$\mathcal{Q} := \left\{Q \ll P \,|\, D_{KL}(Q||P) := \int \frac{dQ}{dP}\big(\ln\frac{dQ}{dP}\big)dP \le -\ln(1-\alpha)\right\}, \tag{2.6}$$

where $D_{KL}(Q||P)$ denotes the KL divergence between the distributions Q and P. For some $x, y \in \mathbb{R}$, $D_{KL}(x||y)$ can be written in the form of the exponential cone, K_{exp}:

$$t \ge x\ln(x/y) \iff (y, x, -t) \in K_{exp}.$$

Similarly, for a discrete random variable $X \in \{x_1, x_2, \ldots, x_J\}$ with pmf given by $p = [p(1), p(2), \ldots, p(J)]^T$, where $p(j) = \mathbb{P}(X = x_j)$, $j \in \mathbb{Z}_1^J$, the KL divergence is given as

$$D_{KL}(q||p) := \sum_{j=1}^{J} q(j)\ln\left(\frac{q(j)}{p(j)}\right),\ q, p \in \Delta_J = \{q \in \mathbb{R}^J \,|\, q \ge 0,\ \sum_{j=1}^{J} q(j) = 1\}.$$

g-entropic risk measures

Let g be a convex function with $g(1) = 0$, and β be a nonnegative number. The g-entropic risk measure [7], $\text{ER}_{g,\beta}$, with divergence level β for a random variable $X \in \mathcal{L}_p(\Omega, \mathcal{F}, \mathbb{P})$ is defined as,

$$\text{ER}_{g,\beta}(X) := \sup_{Q\in\mathcal{Q}} E_Q(X), \tag{2.7}$$

where, $\mathcal{Q} = \{Q \ll P : \int g\big(\frac{dQ}{dP}\big)\, dP \le \beta\}$.

The definition (2.7) describes the g-entropic risk measures in terms of their dual representation. To obtain the primal form, we can use the generalized Donsker-Vardhan variational formula [7],

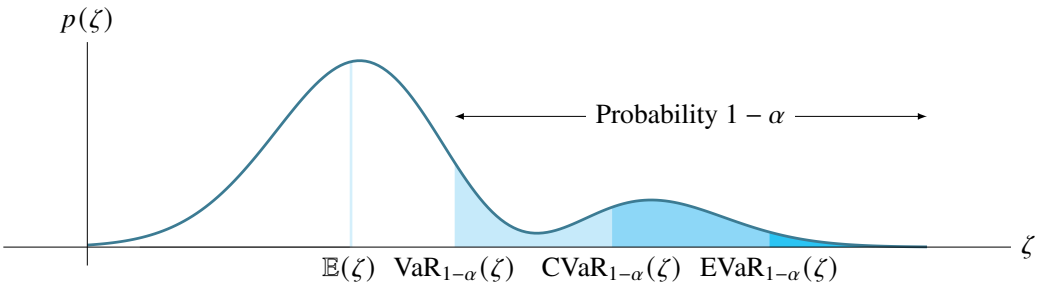


Figure 2.1: Comparison of the mean, VaR, and CVaR for a given confidence $\alpha \in (0, 1)$. The axes denote the values of the stochastic variable ζ with pdf $p(\zeta)$. The shaded area denotes the $\%(1-\alpha)$ of the area under $p(\zeta)$. If the goal is to minimize ζ, using $\mathbb{E}(\zeta)$ as a performance measure is misleading because tail events with low probability of occurrence are ignored. VaR gives the value of ζ at the $(1-\alpha)$-tail of the distribution. But, it ignores the values of ζ with probability below $1-\alpha$. CVaR is the average of the values of VaR with probability less than $1-\alpha$ (average of the worst-case values of ζ in the $1-\alpha$ tail of the distribution). Note that $\mathbb{E}(\zeta) \leq \text{VaR}_{1-\alpha}(\zeta) \leq \text{CVaR}_{1-\alpha}(\zeta) \leq \text{EVaR}_{1-\alpha}(\zeta)$. Hence, $\text{EVaR}_{1-\alpha}(\zeta)$ is a more risk-sensitive measure.

$$\inf_{\mu\in\mathbb{R}}\{\mu + E_P(g^*(X-\mu))\} = \sup_{Q\ll P}\{E_Q(X) - g\big(\frac{dQ}{dP}\big)\, dP\},$$

where g^* is the conjugate (the Legendre-Fenchel transform) of g. Both CVaR and EVaR have been proven to be g-entropic risk measures. Another g-entropic risk measure that we'll consider in this work is using the total variation distance [120]:

$$\text{TVD}_\alpha(X) = \sup_{Q\in\mathcal{Q}} E_Q(X) = \alpha \sup_{x\in\Omega} x + (1-\alpha)\text{CVaR}_{1-\alpha}(X)$$

where the risk envelope is given by,

$$\mathcal{Q} := \Big\{q \in \Delta_J : \frac{1}{2}\sum_{j=1}^{J} |q(j) - p(j)| \leq \alpha\Big\}.$$

2.3 Model Predictive Control

Consider a linear, discrete-time system given by

$$\boldsymbol{x}(t+1) = A\boldsymbol{x}(t) + B\boldsymbol{u}(t) + D\boldsymbol{\delta}(t) \tag{2.8}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^{n_x}$ and $\boldsymbol{u}(t) \in \mathbb{R}^{n_u}$ are the system state and controls at time t, respectively. The system is affected by a stochastic, additive, process noise $\boldsymbol{\delta}_t \in \mathbb{R}^{n_d}$.

Consider there are r_x state constraints that take the form

$$\mathcal{X} := \{\boldsymbol{x} \in \mathbb{R}^{n_x} | F_x\boldsymbol{x} \leq g_x\}, F_x \in \mathbb{R}^{r_x \times n_x},\ g_x \in \mathbb{R}^{r_x}.$$

We also have r_u control constraints that take the form

$$\mathcal{U} := \{\boldsymbol{u} \in \mathbb{R}^{n_u} | F_u \boldsymbol{u} \leq g_u\}, F_u \in \mathbb{R}^{r_u \times n_x},\ g_u \in \mathbb{R}^{r_u}.$$

Consider that the goal is to steer the system to a set

$$\mathcal{X}_F := \{\boldsymbol{x} \in \mathbb{R}^{n_x} | F_f \boldsymbol{x} \leq g_f\}, F_f \in \mathbb{R}^{r_f \times n_x},\ g_x \in \mathbb{R}^{r_f},$$

while minimizing the control effort and deviation from the desired trajectory, i.e., we want to minimize the following cost,

$$J(\boldsymbol{x}, \boldsymbol{u}) := \boldsymbol{x}^T Q \boldsymbol{x} + \boldsymbol{u}^T R \boldsymbol{u},$$

where $Q \in \mathbb{R}^{n_x \times n_x}$ and $R \in \mathbb{R}^{n_u \times n_u}$ are the weights on each of the state and control costs. Model Predictive Control (MPC) provides an optimization-based framework to compute the best control input for the next N-steps while satisfying the state and control constraints. If we have a system starting at an initial condition at time t, that is given by $\boldsymbol{x}(t) = \boldsymbol{x}_0$, the MPC optimization is given by,

$$J_t^*(\boldsymbol{x}(t)) = \min_{U_t} \quad \mathbb{E}\Big[\boldsymbol{x}_{t+N|t}^T P \boldsymbol{x}_{t+N|t} + \sum_{k=t}^{t+N-1} \left(\boldsymbol{x}_{k|t}^T Q \boldsymbol{x}_{k|t} + \boldsymbol{u}_{k|t}^T R \boldsymbol{u}_{k|t}\right)\Big] \tag{2.9}$$

$$\text{s.t.} \quad \boldsymbol{x}_{k+1|t} = A\boldsymbol{x}_{k|t} + B\boldsymbol{u}_{k|t} + D\boldsymbol{\delta}_{k|t}, \tag{2.10}$$

$$\text{Prob}(\boldsymbol{x}_{k|t} \notin \mathcal{X}) \leq \epsilon,\ \boldsymbol{u}_{k|t} \in \mathcal{U}, \tag{2.11}$$

$$\text{Prob}(\boldsymbol{x}_{T+N|t} \notin \mathcal{X}_F) \leq \epsilon \tag{2.12}$$

$$\boldsymbol{x}_{t|t} = \boldsymbol{x}(t) \tag{2.13}$$

$\forall k \in \{t, t+1, \ldots t+N-1\}$

Chapter 3

RISK-AWARE PLANNING IN STATIC ENVIRONMENTS

For a robot to autonomously navigate over a priori unknown, extreme terrain, it must first evaluate the risk of traversal over different regions and then incorporate this risk in the planning framework. In this work, we evaluate the *traversability cost map* for a static environment by computing the CVaR cost of traversal using the first two moments of the cost random variable. The risk cost map is incorporated into geometric and kinodynamic planners to enable the robot to account for high-cost, low-probability scenarios in its motion plan.

This chapter was adapted from:

Anushri Dixit*, David D. Fan*, Kyohei Otsu, Sharmita Dey, Ali-Akbar Agha-Mohammadi, and Joel W. Burdick. STEP: Stochastic Traversability Evaluation and Planning for Risk-Aware Off-road Navigation; Results from the DARPA Subterranean Challenge, 2022.

3.1 Introduction

Autonomous traversal over extreme, hazardous terrain is an open problem with many applications to extra-terrestrial [103], disaster-struck [1], and subterranean environments [74]. The robots operate in terrain is uneven and highly risky with noisy sensor measurements and localization uncertainty. Autonomous motion planning in such conditions requires a framework that can account for the traversability risk arising from sources like rubble, sudden drops, muddy/slippery areas, while also considering the uncertainty in the robots estimates. The framework needs to be able to make tractable reformulations of the nonconvex constraints arising from these risks so that it can plan reactive motions in real time.

In this work, we will evaluate the *traversability* cost of a region as the cost accounting for the hazards from the uneven terrain (geometric, confidence-aware analysis) and type of terrain (semantic analysis), see Figure 3.1. We will treat this traversability cost as a random variable and account for the uncertainty in the traversability cost using Conditional Value-at-Risk (CVaR). Using the CVaR assessment of the terrain, we obtain a geometric path and a low-level model predictive control (MPC) plan that accounts for the risk in the terrain.

Figure 3.1: Top left: Boston Dynamics Spot quadruped robot exploring Valentine Cave at Lava Beds National Monument, CA. Top middle, bottom middle (second image from the left): Clearpath Husky robot exploring Arch Mine in Beckley, WV. Bottom left: Spot exploring abandoned Satsop power plant in Elma, WA.

In particular, we take an MPC approach known as Sequential Quadratic Programming (SQP), which iteratively solves locally quadratic sub-problems to converge to a globally (more) optimal solution [20]. Particularly in the robotics domain, this approach is well-suited due to its reduced computational costs and flexibility for handling a wide variety of costs and constraints [13, 47, 82, 95, 116]. A common criticism of SQP-based MPC (and nonlinear MPC methods in general) is that they can suffer from being susceptible to local minima. We address this problem by incorporating a trajectory library (which can be predefined and/or randomly generated, e.g. as in [73]) to use in a preliminary trajectory selection process. We use this as a means to find more globally optimal initial guesses for the SQP problem to refine locally. Another common difficulty with risk-constrained nonlinear MPC problems is ensuring recursive feasibility [87]. We bystep this problem by dynamically relaxing the severity of the risk constraints while penalizing CVaR in the cost function.

In this work, we propose STEP (Stochastic Traversability Evaluation and Planning), that pushes the boundaries of the state-of-the-practice to enable safe, risk-aware, and high-speed ground traversal of unknown environments. Specifically, our contributions include:

1. Uncertainty-aware 2.5D traversability evaluation which accounts for localization error, sensor noise, and occlusion, and combines multiple sources of

traversability risk.

2. An approach for combining these traversability risks into a unified risk-aware CVaR planning framework.

3. A highly efficient MPC architecture for robustly solving non-convex risk-constrained optimal control problems.

4. A suite of recovery behaviors to account for fast response to failure scenarios.

5. Risk-based gait adaptation for quadrupedal robots (in our case, the Boston Dynamics Spot platform).

6. Real-world demonstration of real-time CVaR planning on wheeled and legged robotic platforms in unknown and risky environments.

3.2 Risk-Aware Traversability and Planning

Problem Statement

We first define the problem of risk-aware traversability and motion planning. Let x_k, u_k, z_k denote the robot's state, action (or control input), and observation (or sensory measurement) at the k-th time step. A path $x_{0:N} = \{x_0, x_1, \cdots, x_N\}$ is composed of a sequence of poses. A policy is a mapping from state to control $u = \pi(x)$. A map is represented as $m = (m^{(1)}, m^{(2)}, \cdots)$ where m^i is the i-th element of the map (e.g., a cell in a grid map). The robot's dynamics model captures the physical properties of the vehicle's motion, such as inertia, mass, dimension, shape, and kinematic and control constraints:

$$x_{k+1} = f(x_k, u_k) \tag{3.1}$$

$$g(u_k) \succ 0 \tag{3.2}$$

$$h(m, x_k) \succ 0 \tag{3.3}$$

where $g(u_k)$, $h(m, x_k)$ are vector-valued functions that encode the control and state constraints/limits respectively.

Following [106], we define *traversability* as the capability for a ground vehicle to reside over a terrain region under an admissible state. We represent traversability as a cost, i.e. a continuous value computed using a terrain model, the robotic vehicle model, and kinematic constraints, which represents the degree to which we wish the robot to avoid a given state:

$$r = \mathcal{R}(m, x, u) \tag{3.4}$$

where $r \in \mathbb{R}$, and $\mathcal{R}(\cdot)$ is a terrain assessment model. This model captures various unfavorable events such as collision, getting stuck, tipping over, high slippage, to name a few. Each mobility platform has its own assessment model to reflect its mobility capability.

Associated with the true traversability value is a distribution over possible values based on the current understanding about the environment and robot actions. In most real-world applications where perception capabilities are limited, the true value can be highly uncertain. To handle this uncertainty, consider a map belief, i.e., a probability distribution $p(m|x_{0:k}, z_{0:k})$, over a possible set $\mathcal{M}$. Then, the traversability estimate is also represented as a random variable $R : (\mathcal{M}\times\mathcal{X}\times\mathcal{U}) \longrightarrow \mathbb{R}$. We call this probabilistic mapping from map belief, state, and controls to possible traversability cost values a *terrain assessment model*.

A risk metric $\rho(R) : R \rightarrow \mathbb{R}$ is a mapping from a random variable to a real number which quantifies some notion of risk. In order to assess the risk of traversing along a path $x_{0:N}$ with a policy π, we wish to define the cumulative risk metric associated with the path, $J(x_0, \pi)$. To do this, we need to evaluate a sequence of random variables $R_{0:N}$. To quantify the stochastic outcome as a real number, we use the dynamic, time-consistent risk metric given by compounding the one-step risk metrics [115]:

$$J(x_0, \pi; m) = R_0 + \rho_0\big(R_1 + \rho_1\big(R_2 + \ldots + \rho_{N-1}\big(R_N\big)\big)\big) \tag{3.5}$$

where $\rho_k(\cdot)$ is a one-step coherent risk metric at time k. This one-step risk gives us the cost incurred at time-step $k + 1$ from the perspective of time-step k. Any distortion risk metric compounded as given in (3.5) is time-consistent (see [92] for more information on distortion risk metrics and time-consistency). We use the Conditional Value-at-Risk (CVaR) as the one-step risk metric:

$$\rho(R) = \text{CVaR}_\alpha(R) = \inf_{z\in\mathbb{R}} \mathbb{E}\left[z + \frac{(R - z)_+}{1 - \alpha}\right] \tag{3.6}$$

where $(\cdot)_+ = \max(\cdot, 0)$, and $\alpha \in (0, 1]$ denotes the *risk probability level*. We note that the results in this chapter are also easily extended to other tail risk measures like Entropic Value-at-Risk and total variation distance-based risk.

We formulate the objective of the problem as follows: Given the initial robot configuration x_S and the goal configuration x_G, find an optimal control policy π^* that moves the robot from x_S to x_G while 1) minimizing time to traverse, 2)

minimizing the cumulative risk metric along the path, and 3) satisfying all kinematic and dynamic constraints. For a quadrupedal robot like the Boston Dynamics Spot robot, the framework must additionally also select the best gait type based on the risk accrued while moving from x_S to x_G as a part of the optimal policy π^*.

Hierarchical Risk-Aware Planning

We propose a hierarchical approach to address the aforementioned risk-aware motion planning problem by splitting the motion planning problem into geometric and kinodynamic domains. We consider the geometric domain over long horizons, while we solve the kinodynamic problem over a shorter horizon. This is convenient for several reasons: 1) Solving the full constrained CVaR minimization problem over long timescales/horizons becomes intractable in real-time. 2) Geometric constraints play a much larger role over long horizons, while kinodynamic constraints play a much larger role over short horizons (to ensure dynamic feasibility at each timestep). 3) A good estimate (upper bound) of risk can be obtained by considering position information only. This is done by constructing a position-based traversability model $\mathcal{R}_{\text{pos}}$ by marginalizing out non-position related variables from the terrain assessment model, i.e. if the state $x = [p_x, p_y, x_{\text{other}}]^\intercal$ consists of position and non-position variables (e.g. orientation, velocity), then

$$\mathcal{R}_{\text{pos}}(m, p_x, p_y) \geq \mathcal{R}(m, x, u) \quad \forall x_{\text{other}}, u \tag{3.7}$$

Geometric Planning: The objective of geometric planning is to search for an *optimistic* risk-minimizing path, i.e. a path that minimizes an upper bound approximation of the true CVaR value. For efficiency, we limit the search space only to the geometric domain. We are searching for a sequence of poses $x_{0:N}$ which ends at x_G and minimizes the position-only risk metric in (3.5), which we define as $J_{\text{pos}}(x_{0:N})$. The path optimization problem can be written as:

$$x^*_{0:N} = \arg\min_{x_{0:N}} \left[J_{\text{pos}}(x_{0:N}) + \lambda \sum_{k=0}^{N-1} \|x_k - x_{k+1}\|^2 \right] \tag{3.8}$$

$$s.t. \quad \phi(m, x_k) \succ 0 \tag{3.9}$$

where the constraints $\phi(\cdot)$ encode position-dependent traversability constraints (e.g. constraining the vehicle to avoid obstacles and prohibit lethal levels of risk) and $\lambda \in \mathbb{R}$ weighs the tradeoff between risk and path length.

Kinodynamic Planning: We then solve a kinodynamic planning problem to track the optimal geometric path, minimize the risk metric, and respect kinematic and

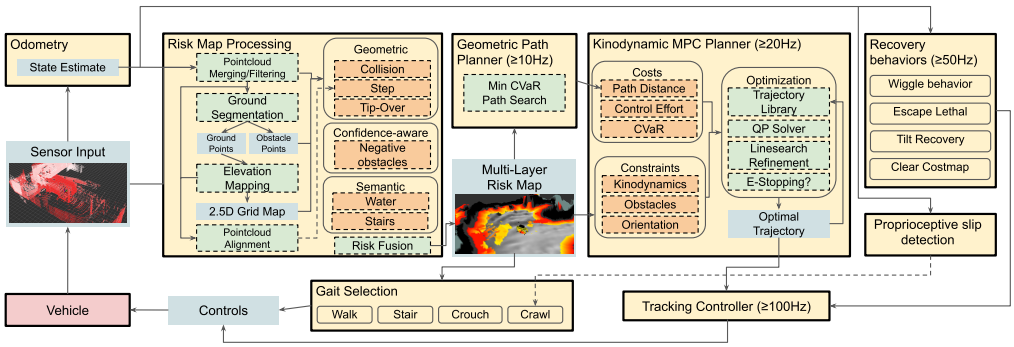


Figure 3.2: Overview of system architecture for STEP. From left to right: Odometry aggregates sensor inputs and relative poses. Next, Risk Map Processing merges these pointclouds and creates a multi-layer risk map. The map is used by the Geometric Path Planner and the Kinodynamic MPC Planner. An optimal trajectory is found and sent to the Tracking Controller, which produces control inputs to the robot.

dynamics constraints. The goal is to find a control policy π^* within a local planning horizon $T \leq N$ which tracks the path $X^*_{0:N}$. The optimal policy can be obtained by solving the following optimization problem:

$$\pi^* = \arg\min_{\pi \in \Pi} \left[J(x_0, \pi) + \lambda \sum_{k=0}^{T} \|x_k - x_k^*\|^2 \right] \tag{3.10}$$

$$s.t.\ \forall k \in [0, \cdots, T]: \quad x_{k+1} = f(x_k, u_k) \tag{3.11}$$

$$g(u_k) \succ 0 \tag{3.12}$$

$$h(m, x_k) \succ 0 \tag{3.13}$$

where the constraints $g(u)$ and $h(m, x_k)$ are vector-valued functions which encode controller limits and state constraints, respectively.

Recovery from Unfavorable States

Recovery planning is a particular type of planning problem where the initial state is not safe. For example, the robot might need to start planning from the state where it touches walls with its bumpers, or the state where the body is tilted on top of rubble. The recovery from those unfavorable states involves finding a safe control without violating the safety constraints under smaller margin conditions.

3.3 STEP for Unstructured Terrain

This section discusses how we compute traversability risk and efficiently solve the risk-aware trajectory optimization problem. At a high level, our approach takes the

following steps (see Figure 3.2): 1) Assuming some source of localization with uncertainty, aggregate sensor measurements to create an uncertainty-aware map. 2) Perform ground segmentation to isolate the parts of the map the robot can potentially traverse. 3) Compute risk and risk uncertainty using geometric properties of the pointcloud (optionally, include other sources of risk, e.g. semantic or other sensors). 4) Aggregate these risks to compute a 2.5D CVaR risk map. 5) Solve for an optimistic CVaR minimizing path over long ranges with a geometric path planner. 7) Solve for a kinodynamically feasible trajectory which minimizes CVaR while staying close to the geometric path and satisfying all constraints.

Pointcloud Processing and Mapping

Multi-sensor Merging Our pointcloud pipeline starts from merging pointclouds from different sensors. One robot can have multiple units of the same senor to increase coverage, or have heterogeneous sensors that produce pointclouds using different mechanics (e.g., active LiDAR, passive stereo cameras). After applying sensor-specific filters that remove noise or body occlusion, these pointclouds are merged using extrinsic calibration information. If the sensors are not time-synchronized, we use odometry to compensate motion offset.

Temporal Pointcloud Merging The merged pointclouds are aggregated over a fixed time window to construct a local pointcloud map. We maintain a pose graph on the time window to incorporate history updates in the odometry estimate (e.g., loop closures). Based on the latest pose graph, we reconstruct a full aggregated pointcloud. We annotate each point by the time offset to the latest pointcloud. This allows us to propagate odometry uncertainty to each point in the fusion phase.

Ground Segmentation The aggregated pointcloud is segmented into ground and obstacle points using 3D pointcloud segmentation techniques. We leverage the work in [66] that allows efficient ground segmentation based on line fitting in the cylindrical coordinates. We extended the work to also handle challenges prevalent in subterranean environments, such as low ceiling or negative obstacles. The ground segmentation is critical for elevation mapping in an occluded environment where the ground is not observed by the sensors and the measurements of walls/ceiling make false ground planes.

Elevation Mapping We construct 2.5D height map using the segmented ground points. After splitting points to each grid map cell and sorting by time stamps, we apply the Kalman filter to estimate the height of the ground. We set up the

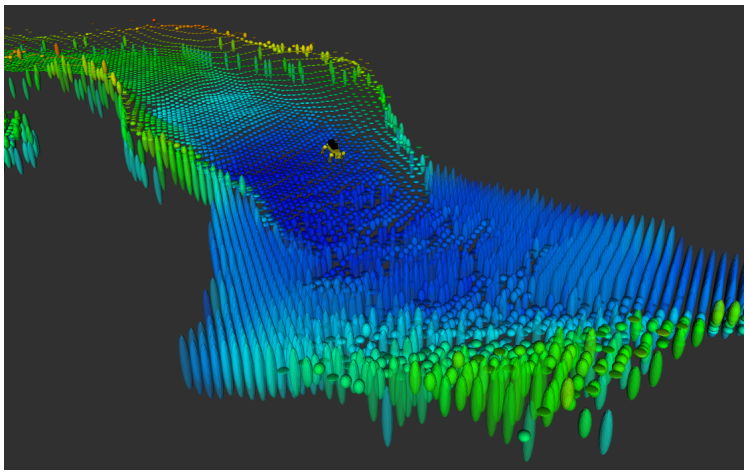


Figure 3.3: Estimation of ground height with uncertainty

filter to place more weight on the recent measurements which are less affected by the odometry error. Figure 3.3 shows the visualization of ground height estimation with uncertainty. Note that there is higher level of uncertainty for the areas that do not have enough measurements. These mapping uncertainties are used to adjust confidence in a later traversability estimation process.

Modeling Assumptions

Among many representation options for rough terrain, we use a 2.5D grid map in this work for its efficiency in processing and data storage [42]. The map is represented as a collection of terrain properties (e.g., height, risk) over a uniform grid.

For different vehicles we use different robot dynamics models. For example, for a system which produces longitudinal/lateral velocity and steering (e.g. legged platforms), the state and controls can be specified as:

$$x = [p_x, p_y, p_\theta, v_x, v_y, v_\theta]^\top \tag{3.14}$$

$$u = [a_x, a_y, a_\theta]^\top \tag{3.15}$$

While the dynamics $x_{k+1} = f(x_k, u_k)$ can be written as,

$$x_{k+1} = x_k + \Delta t \begin{bmatrix} v_x \cos(p_\theta) - v_y \sin(p_\theta) \\ v_x \sin(p_\theta) + v_y \cos(p_\theta) \\ \kappa v_x + (1-\kappa) v_\theta \\ a_x \\ a_y \\ a_\theta \end{bmatrix}.$$

We let $\kappa \in [0, 1]$ be a constant which adjusts the amount of turning-in-place the vehicle is permitted. In differential drive or ackermann steered vehicles we can remove the lateral velocity component of these dynamics if desired. However, our general approach is applicable to any vehicle dynamics model.

Terrain assessment models

The traversability cost is assessed as the combination of multiple risk factors. These factors are designed to capture potential hazards for the target robot in the specific environment (Figure Figure 3.4). Such factors include:

- *Collision*: quantified by the distance to the closest obstacle point.
- *Step size*: the height gap between adjacent cells in the grid map.
- *Tip-over*: a function of slope angles and the robot's orientation.
- *Sensor Uncertainty*: sensor and localization error increase the variance of traversability estimates.
- *Negative Obstacles*: detected by checking the lack of measurement points in a cell.
- *Slippage*: quantified by geometry and the surface material of the ground.

Geometry-based risk sources

Geometry-based risk sources include collision with obstacles, too-large step sizes in terrain, and impassable slopes. These geometry-based risks are constructed using geometric analysis of elevation map estimates and LiDAR pointcloud points. We construct these risks per grid cell, with the following methods. First, using the ground estimates from the ground segmentation pipeline, we obtain a lower bound on the height of the ground, i.e. the height at which the robot would place its foot or wheel if occupying that cell with its foot/wheel. Above this ground estimate, we can determine the relative height of other occupying LiDAR points in the temporally merged pointcloud with respect to the ground. Points which occur above or below a certain height threshold which outline the body of the robot are treated as obstacles, and the corresponding grid cells at these points are marked as untraversable. Similar analysis is performed for step size risk, which checks the height gap between adjacent cells in the elevation map. For adjacent cells which

exhibit too high a height gap, these cells are marked as untraversable. Finally, checking the normals of the elevation map gives an overall estimation of slope (which the size of the normal calculation averaged over the approximate size of the robot), and areas with too high slope values are marked as high risk or untraversable.

Note that in all these analyses, the uncertainty of the elevation map plays a large role. Sensor and localization uncertainty corrupts elevation measurements to a varying degree, proportional with distance from the robot. Therefore we adapt various detection or risk thresholds with distance from the robot, to obtain a more robust result. For example, for the Husky robot, ground clearance beneath the robot belly is 10cm. Detecting a 10cm step size in elevation at 100m away requires an angular localization accuracy of < 0.06 degrees. This may be infeasbile, and therefore adapting the step size threshold with distance helps to reduce spurious detections at longer ranges.

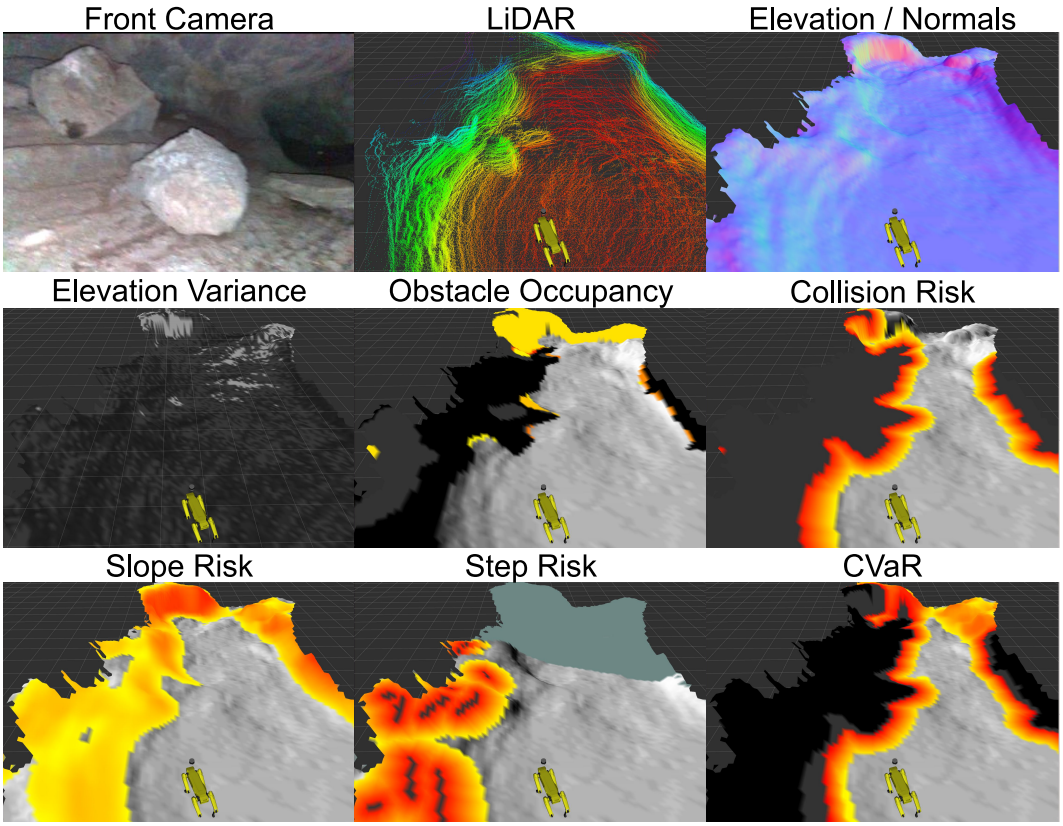


Figure 3.4: Multi-layer geometric risk analysis, which first aggregates recent point-clouds (top). Then, each type of analysis (slope, step, collision, etc.) generates a risk map along with uncertainties (middle rows). These risks are aggregated to compute the final CVaR map (bottom).

Confidence-based risk sources

One example of confidence-based risk sources is the detection of negative obstacles. To detect negative obstacles, we estimate areas in the pointcloud that have no returns and use 2-D ray-tracing to find "gap" areas that are not occluded by obstacles, small steps, or upward slopes. However, if this is the only criteria for negative obstacle detection, many false positives for negative obstacles are observed. One such instance is shown in Figure 3.5 when the robot turns the corner into a new room from a narrow passageway and in the time-taken for the LiDAR returns from all areas of the new room to reach the sensor, false-positives of negative obstacles are detected. To address this, we account for the confidence in the gaps in the pointcloud by estimating whether these areas have been sufficiently covered so far, i.e., the robot has sensed the area from different positions and for long enough to ensure that the gaps in the pointcloud are not caused due to the sensor and environment configuration, see Algorithm 1 for a detailed description of the method to check whether a given region passes the coverage check.

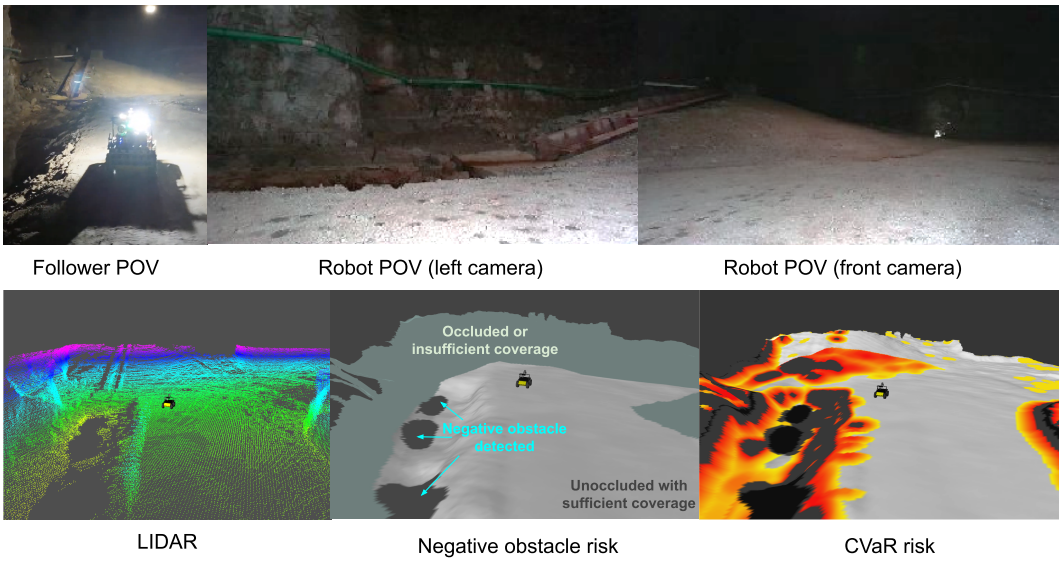


Figure 3.5: Confidence-based risk analysis: the scene is illustrated through the point-of-view (POV) of a third-person (top left) and robot (top middle and top right). The aggregated pointcloud (bottom left) has regions of no returns from the area on the left side of the robot. These holes in the pointcloud are marked as negative obstacles in the risk layer (bottom middle) only when there are no returns from these regions despite them being unoccluded and sufficiently covered by laser strike pattern. This risk layer is aggregated with the geometric risk layers to compute the final CVaR map (bottom right).

Algorithm 1 Algorithm for computing whether a given area is unoccluded and sufficiently covered, i.e., $m^{\text{cover}}_{\text{MSE}} >= 0$

Input: m^{unocc}: Grid map of unoccluded grid points from the current robot state x_t (using ray-tracing),
$x_t = [p_x, p_y, p_\theta, v_x, v_y, v_\theta]^T$: Current robot state,
d^{cover}: Threshold for sufficient coverage.

Output: m^{unocc}_N: Number of times each grid cell in the grid map has been observed unoccluded by the robot so far,
$m^{\text{cover}} = \{m^{\text{cover}}_{\mu_x}, m^{\text{cover}}_{\mu_y}, m^{\text{cover}}_{\text{MSE}}\}$: Multi-layer map indicating whether each grid cell being evaluated for presence of negative obstacles has been observed unoccluded from different points to ensure sufficient coverage.

1: **for** $i = 0$ to $|m^{\text{unocc}}|$ **do**

2: # *Update* m^{unocc}_N

$$m^{\text{unocc},(i)}_N = \begin{cases} 1 + m^{\text{unocc},(i)}_N & \text{if } m^{\text{unocc},(i)} = 1 \text{ (i.e., } i^{\text{th}} \text{ grid cell is unoccluded)} \\ m^{\text{unocc},(i)}_N & \text{if } m^{\text{unocc},(i)} = 0 \text{ (i.e., } i^{\text{th}} \text{ grid cell is occluded)} \end{cases}$$

3: # *Update mean x and y robot positions from where each grid cell has been observed unoccluded*

$$\{m^{\text{cover},(i)}_{\mu_x}, m^{\text{cover},(i)}_{\mu_y}\} = \begin{cases} \{m^{\text{cover},(i)}_{\mu_x} + \frac{p_x - m^{\text{cover},(i)}_{\mu_x}}{m^{\text{unocc},(i)}_N}, m^{\text{cover},(i)}_{\mu_y} + \frac{p_y - m^{\text{cover},(i)}_{\mu_y}}{m^{\text{unocc},(i)}_N}\} & \text{if } m^{\text{unocc},(i)} = 1 \\ \{m^{\text{cover},(i)}_{\mu_x}, m^{\text{cover},(i)}_{\mu_y}\} & \text{if } m^{\text{unocc},(i)} = 0 \end{cases}$$

4: # *Update mean squared deviation of the average robot positions from which a given grid cell is observed*
to check if it crosses the threshold for sufficient coverage, d^{cover}

$$m^{\text{cover},(i)}_{\text{MSE}} = \begin{cases} (p_x - m^{\text{cover},(i)}_{\mu_x})^2 + (p_y - m^{\text{cover},(i)}_{\mu_y})^2 - (d^{\text{cover}})^2 & \text{if } m^{\text{unocc},(i)} = 1 \\ -(d^{\text{cover}})^2 & \text{if } m^{\text{unocc},(i)} = 0 \end{cases}$$

5: **end for**

Semantics-based risk sources

We account for changes in the terrain features based on changes in the intensity of LiDAR returns. Areas that comprise a lot of mud and water have low intensity of returns. This enables detection of muddy regions where the robot may get stuck or fall down. Areas with deep water levels are similarly detected as negative obstacles because the LiDAR returns are very sparse. This combined with low intensity of returns allows us to detect lethal regions with deep water levels.

Traversability Cost

To efficiently compute the Conditional Value-at-Risk (CVaR) traversability cost for $l > 1$ risk factors, we assume each risk factor R_l is an independent random variable which is normally distributed, with mean μ_l and variance σ_l. We take a weighted average of the risk factors to obtain the terrain assessment model, $R = \sum_l w_l R_l$, which will also be normally distributed as $R \sim \mathcal{N}(\mu, \sigma^2)$. Let φ and $\mathbf{\Phi}$ denote

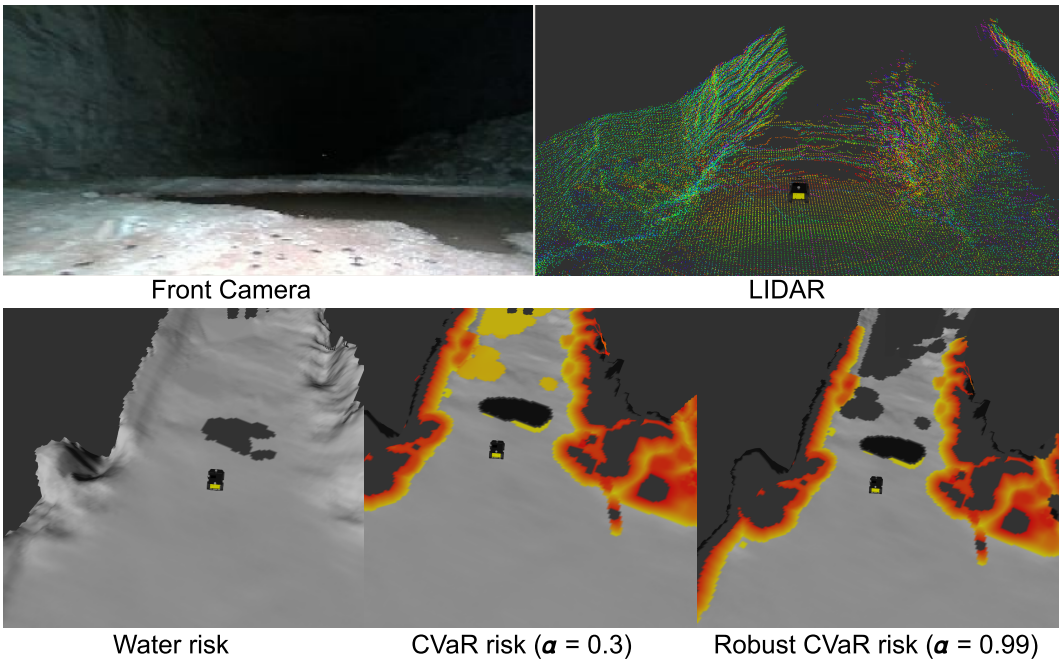


Figure 3.6: Semantics-based risk analysis: the robot camera (top left) shows a puddle of water in front of the robot. The aggregated pointcloud (top right) has regions of no returns from the area in front of the robot. These holes in the pointcloud are marked as water in the risk layer (bottom left) when the holes in the pointcloud are present with low intensity returns in the area near the hole. This risk layer is aggregated with the geometric risk layers to compute the final CVaR map (bottom middle and bottom right) with different levels of robustness α. We clearly see the effects on the risk map, where higher values of α result assigning high cost to unknown regions.

the probability density function and cumulative distribution function of a standard normal distribution respectively. The corresponding CVaR is computed as:

$$\rho(R) = \mu + \sigma \frac{\varphi(\mathbf{\Phi}^{-1}(\alpha))}{1-\alpha} \tag{3.16}$$

The CVaR cost accounts for the expected cost in the $(1-\alpha)$ tail of the distribution of the terrain assessment model, R, thus also accounting for high-risk, low probability events. We construct the R such that the expectation of R is positive, to keep the CVaR value positive.

Construction of the mean and variance of each risk factor depends on the type of risk. For example, collision risk is determined by checking for points above the elevation map, and the variance is derived from the elevation map variance, which is mainly a function of localization error. In contrast, negative obstacle risk is determined by looking for gaps in sensor measurements. These gaps tend to be a function of sensor sparsity, so the risk variance increases with distance from the sensor frame.

Risk-aware Geometric Planning

In order to optimize (3.8) and (3.9), the geometric planner computes an optimal path that minimizes the position-dependent dynamic risk metric in (3.5) along the path. Substituting (3.16) into (3.5), we obtain:

$$\begin{aligned} J_{\text{pos}}(x_{0:N}) &= R_0 + \rho_0\big(R_1 + \rho_1\big(R_2 + \ldots + \rho_{T-1}(R_T)\big)\big) \\ &= R_0 + \rho\Bigg(R_1 + \rho\Big(R_2 + \ldots + \rho(R_{T-1} + \mu_T + \sigma_T \frac{\varphi(\mathbf{\Phi}^{-1}(\alpha))}{1-\alpha})\Big)\Bigg) \\ &\vdots \\ &= \mu_0 + \sum_{k=1}^{N}\left[\mu_k + \sigma_k \frac{\varphi(\mathbf{\Phi}^{-1}(\alpha))}{1-\alpha}\right] \end{aligned}$$

We use the A* algorithm to solve (3.8) over a 2D grid. A* requires a path cost $g(n)$ and a heuristic cost $h(n)$, given by:

$$g(n) = J_{\text{pos}}(x_{0:n}) + \lambda \sum_{k=0}^{n-1} \|x_k - x_{k+1}\|^2, \tag{3.17}$$

$$h(n) = \lambda \, \|x_n - x_G\|^2. \tag{3.18}$$

For the heuristic cost we use the shortest Euclidean distance to the goal. The parameter lambda is a relative weighting between the distance penalty and risk penalty and can be thought of as having units of (traversability cost / m). We use a relatively small value, which means we are mainly concerned with minimizing traversability costs.

Risk-aware Kinodynamic Planning

The geometric planner produces a path, i.e. a sequence of poses. We wish to find a kinodynamically feasible trajectory which stays near this path, while satisfying all constraints and minimizing the CVaR cost. To solve (3.10)-(3.13), we use a risk-aware kinodynamic MPC planner, whose steps we outline (Figure 3.7, Algorithm 2, Figure 3.8).

Trajectory library: Our kinodynamic planner begins with selecting the best candidate trajectory from a trajectory library, which stores multiple initial control and state sequences. The selected trajectory is used as initial solution for solving a full optimization problem. The trajectory library can include: 1) the trajectory accepted in the previous planning iteration, 2) a stopping (braking) trajectory, 3) a

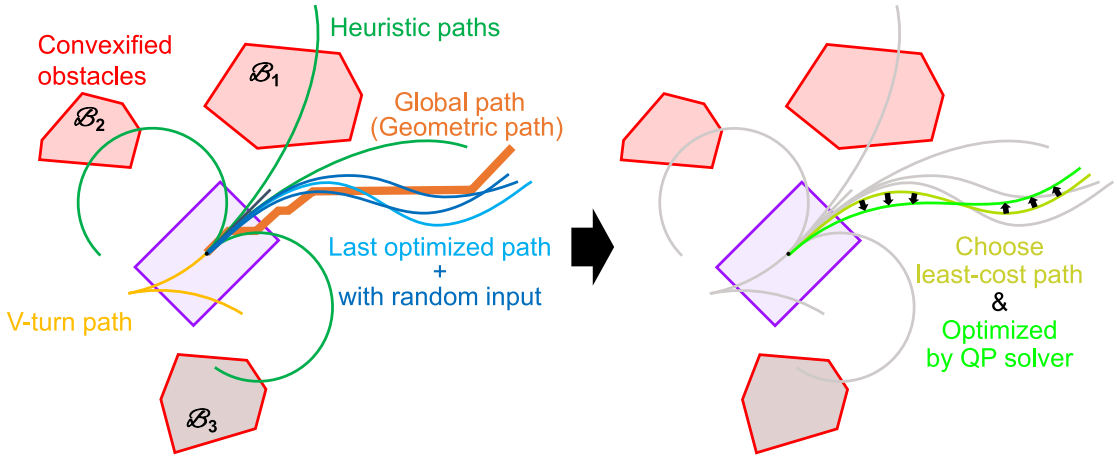


Figure 3.7: Diagram of kinodynamic MPC planner, which begins with evaluating various paths within a trajectory library. The lowest cost path is chosen as a candidate and optimized by the QP solver.

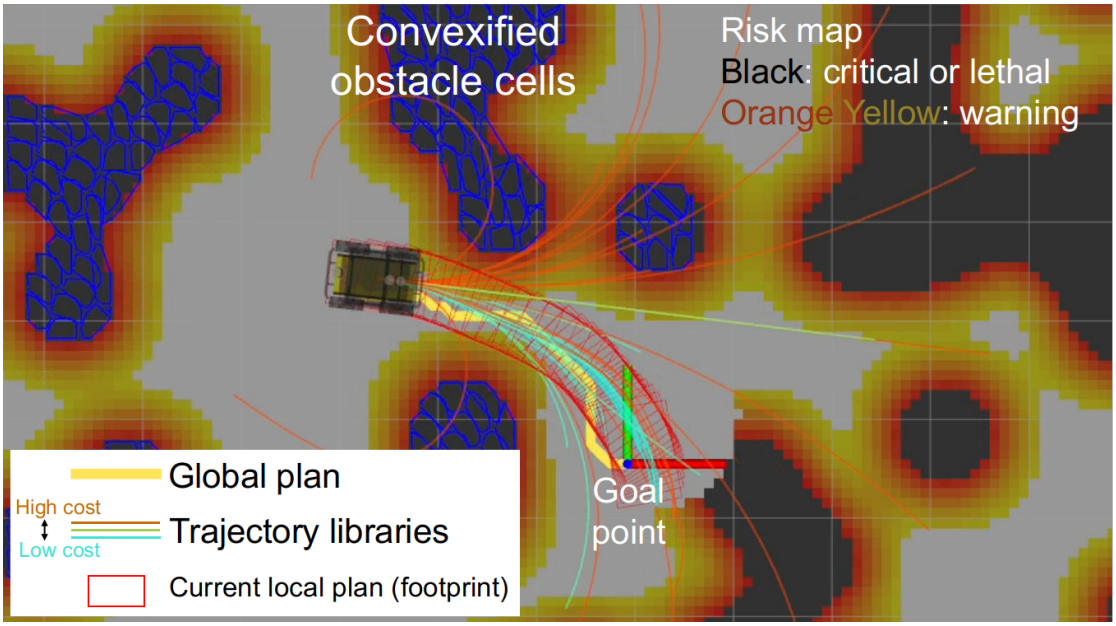


Figure 3.8: Diagram of kinodynamic MPC planner, which begins with evaluating various paths within a trajectory library. The lowest cost path is chosen as a candidate and optimized by the QP solver.

geometric plan following trajectory, 4) heuristically defined trajectories (including v-turns, u-turns, and varying curvatures), and 5) randomly perturbed control input sequences.

QP Optimization: Next, we construct a non-linear optimization problem with costs

Algorithm 2 Kinodynamic MPC Planner (sequences $\{\text{var}_k\}_{k=0:T}$ are expressed as $\{\text{var}\}$ for brevity)

Input: current state x_0, current control sequence (previous solution) $\{u^*\}^{(j)}$
Output: re-planned trajectory $\{x^*\}^{(j+1)}$, re-planned control sequence $\{u^*\}^{(j+1)}$

 # Initialization
$\{x^{\text{r}}\}$ = updateReferenceTrajectory()
$\{u^*\}^{(j)}$ = stepControlSequenceForward($\{u^*\}^{(j)}$)
 # Loop process
for $i = 0$ to qp_iterations **do**
  l = generateTrajectoryLibrary(x_0)
  $[\{x^{\text{c}}\}, \{u^{\text{c}}\}]$ = chooseCandidateFromLibrary(l)
  $[\{\delta x^*\}, \{\delta u^*\}]$ = solveQP($\{x^{\text{c}}\}, \{u^{\text{c}}\}, \{x^{\text{r}}\}$)
  $[\gamma, solved]$ = lineSearch($\{x^{\text{c}}\}, \{\delta x^*\}, \{u^{\text{c}}\}, \{\delta u^*\}$)
  $u^c_k = u^{\text{c}}_k + \gamma \delta u^*_k, \ \forall k = 0 : T$
  $\{x^c\}$ = rollOutTrajectory($x_0, \{u^c\}$)
end for
if solved **then**
  $\{x^*\}^{(j+1)}, \{u^*\}^{(j+1)} = \{x^c\}, \{u^c\}$
else
  $\{x^*\}^{(j+1)}, \{u^*\}^{(j+1)}$ = getStoppingTrajectory()
end if
return $\{x^*\}^{(j+1)}, \{u^*\}^{(j+1)}$

and constraints (3.10–3.13). We linearize the problem about the initial solution and solve iteratively in a sequential quadratic programming (SQP) fashion [101]. Let $\{\hat{x}_k, \hat{u}_k\}_{k=0:T}$ denote an initial solution. Let $\{\delta x_k, \delta u_k\}_{k=0:T}$ denote deviation from the initial solution. We introduce the solution vector variable X:

$$X = \begin{bmatrix} \delta x_0^{\text{T}} & \cdots & \delta x_T^{\text{T}} & \delta u_0^{\text{T}} & \cdots & \delta u_T^{\text{T}} \end{bmatrix}^{\text{T}} \tag{3.19}$$

We can then write the MPC in the form:

$$\text{minimize} \quad \frac{1}{2} X^{\text{T}} P X + q^{\text{T}} X \tag{3.20}$$

$$\text{subject to} \quad l \leq AX \leq u \tag{3.21}$$

where P is a positive semi-definite weight matrix, q is a vector to define the first order term in the objective function, A defines inequality constraints and l and u provide their lower and upper limit. In the next subsection we describe these costs and constraints in detail. This is a quadratic program, which can be solved using commonly available QP solvers such as our implementation uses the OSQP solver, which is a robust and highly efficient general-purpose solver for convex QPs [126].

Linesearch: The solution to the SQP problem returns an optimized variation of the control sequence $\{\delta u^*_k\}_{k=0:T}$. We then use a linesearch procedure to determine

the amount of deviation $\gamma > 0$ to add to the current candidate control policy π: $u_k = u_k + \gamma \delta u_k^*$.

Stopping Sequence: If no good solution is found from the linesearch, we pick the lowest cost trajectory from the trajectory library with no collisions. If all trajectories are in collision, we generate an emergency stopping sequence to slow the robot as much as possible (a collision may occur, but hopefully with minimal energy).

Tracking Controller: Having found a feasible and CVaR-minimizing trajectory, we send it to a tracking controller to generate closed-loop tracking behavior at a high rate (>100Hz), which is specific to the robot type (e.g. a simple cascaded PID, or legged locomotive controller).

Optimization Costs and Constraints

Costs: Note that the cost (3.10) includes the CVaR risk. To linearize the cost for representation in a QP, we compute the Jacobian and Hessian of ρ with respect to the state x. We efficiently approximate these terms via numerical differentiation.

Kinodynamic constraints: Similar to the cost, we linearize the system dynamics (3.11) with respect to x and u. Depending on the dynamics model, this may be done analytically.

Control limits: The constraint function $g(u)$ in (3.12) limits the range of the control inputs. For example in the 6-state dynamics case, we limit maximum accelerations: $|a_x| < a_x^{\max}$, $|a_y| < a_y^{\max}$, and $|a_\theta| < a_\theta^{\max}$.

State limits: Within the function $h(m, x)$ in (3.13), we encode velocity constraints: $|v_x| < v_x^{\max}$, $|v_y| < v_y^{\max}$, and $|v_\theta| < v_\theta^{\max}$. We also constrain the velocity of the vehicle to be less than some scalar multiple of the risk in that region, along with maximum allowable velocities:

$$|v_\theta| < \gamma_\theta \, \rho(R_k) \tag{3.22}$$

$$\sqrt{v_x^2 + v_y^2} < \gamma_v \, \rho(R_k) \tag{3.23}$$

This constraint reduces the energy of interactions the robot has with its environment in riskier situations, preventing more serious damage.

Position risk constraints: The function $h(m, x_k)$ also encodes constraints on position and orientation to prevent the robot from hitting obstacles. The general form of this constraint is:

$$\rho(R_k) < \rho^{\max} \tag{3.24}$$

To formulate the above constraint, we locate areas on the map where the risk ρ is greater than the maximum allowable risk. These areas are marked as obstacles, and are often highly non-convex. To obtain a convex and tractable approximation of this highly non-convex constraint, we decompose obstacles into non-overlapping 2D convex polygons, and create a signed distance function which determines the minimum distance between the robot's footprint (also assumed to be a convex polygon) and each obstacle [116]. Let $\mathcal{A}, \mathcal{B} \subset \mathbb{R}^2$ be two convex sets, and define the distance between them as:

$$\text{dist}(\mathcal{A}, \mathcal{B}) = \inf\{\|T\| \mid (T + \mathcal{A}) \cap \mathcal{B} \neq \emptyset\} \tag{3.25}$$

where T is a translation. When the two sets are overlapping, define the penetration distance as:

$$\text{penetration}(\mathcal{A}, \mathcal{B}) = \inf\{\|T\| \mid (T + \mathcal{A}) \cap \mathcal{B} = \emptyset\} \tag{3.26}$$

Then we can define the signed distance between the two sets as:

$$\text{sd}(\mathcal{A}, \mathcal{B}) = \text{dist}(\mathcal{A}, \mathcal{B}) - \text{penetration}(\mathcal{A}, \mathcal{B}) \tag{3.27}$$

We then include within the function $h(m, x_k)$ a constraint to enforce the following inequality:

$$\text{sd}(\mathcal{A}_{\text{robot}}, \mathcal{B}_i) > 0 \quad \forall i \in \{0, \cdots, N_{\text{obstacles}}\}. \tag{3.28}$$

Note that the robot footprint $\mathcal{A}_{\text{robot}}$ depends on the current robot position and orientation: $\mathcal{A}_{\text{robot}}(p_x, p_y, p_\theta)$, while each obstacle $\mathcal{B}_i(m)$ is dependent on the information in the map (See Figure 3.9).

Orientation constraints: We wish to constrain the robot's orientation on sloped terrain so as to prevent the robot from rolling over or performing dangerous maneuvers. To do this, we add constraints to the function $h(m, x_k)$ which limit the roll and pitch of the robot as it settles on the surface of the ground. Denote the position as $p = [p_x, p_y]^\intercal$ and the position/yaw as $s = [p_x, p_y, p_\theta]^\intercal$. Let the robot's pitch be ψ and roll be ϕ in its body frame. Let $\omega = [\psi, \phi]^\intercal$. The constraint will have the form $|\omega| \prec \omega^{\max}$. At p, we compute the surface normal vector, call it $n^w = [n_x^w, n_y^w, n_z^w]^\intercal$, in the world frame. Let $n^r = [n_x^r, n_y^r, n_z^r]^\intercal$, be the surface normal in the body frame, where we rotate by the robot's yaw: $n^r = R_\theta n^w$ (see Figure 3.9), where R_θ is a basic rotation matrix by the angle θ about the world z axis. Then, we define the robot

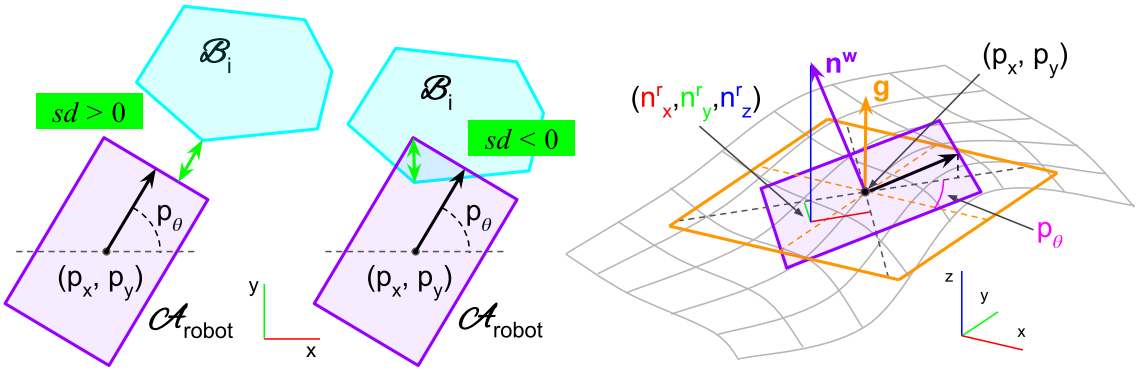


Figure 3.9: Left: Computing convex to convex signed distance function between the robot footprint and an obstacle. Signed distance is positive with no intersection and negative with intersection. Right: Robot pitch and roll are computed from the surface normal rotated by the yaw of the robot. Purple rectangle is the robot footprint with surface normal n^w. $\boldsymbol{g}$ denotes gravity vector, $n^r_{x,y,z}$ are the robot-centric surface normal components used for computing pitch and roll.

pitch and roll as $\omega = g(n^r)$ where:

$$\omega = g(n^r) = \begin{bmatrix} \text{atan2}(n^r_x, n^r_z) \\ -\text{atan2}(n^r_y, n^r_z) \end{bmatrix} \tag{3.29}$$

Note that ω is a function of s. Creating a linearly-constrained problem requires a linear approximation of the constraint:

$$|\nabla_s \omega(s)\delta s + \omega(s)| < \omega^{\max} \tag{3.30}$$

The linearization is realized by finding the gradients of the orientation constraints with respect to position and yaw separately.

Box Constraint: Note that if δx and δu are too large, linearization errors will dominate. To mitigate this effect we also include box constraints within (3.12) and (3.13) to maintain a bounded deviation from the initial solution: $|\delta x| < \epsilon_x$ and $|\delta u| < \epsilon_u$.

Adding Slack Variables: To further improve the feasibility of the optimization problem we introduce auxilliary slack variables for constraints on state limits, position risk, and orientation. For a given constraint $h(x) > 0$ we introduce the slack variable ϵ, and modify the constraint to be $h(x) > \epsilon$ and $\epsilon < 0$. We then penalize large slack variables with a quadratic cost: $\lambda_\epsilon \epsilon^2$. These slack variables are incorporated into the QP problem (3.20) and (3.21).

Dynamic Risk Adjustment

The CVaR metric allows us to dynamically adjust the level and severity of risk we are willing to accept. Selecting low α reverts towards using the mean cost as a metric, leading to optimistic decision making while ignoring low-probability but high cost events. Conversely, selecting a high α leans towards conservatism, reducing the likelihood of fatal events while reducing the set of possible paths. We adjust α according to two criteria: 1) Mission-level states, where depending on the robot's role, or the balance of environment and robot capabilities, the risk posture for individual robots may differ. 2) Recovery Behaviors, where if the robot is trapped in an unfavorable condition, by gradually decreasing α, an escape plan can be found with minimal risk. These heuristics are especially useful in the case of risk-aware planning, because the feasibility of online nonlinear MPC is difficult to guarantee. When no feasible solution is found for a given risk level α, a riskier but feasible solution can be quickly found and executed.

Gait Selection

We considered two platforms for testing our traversability framework - a Clearpath Husky (wheeled robot) and Boston Dynamics Spot (quadrupedal legged robot). In the case of the Spot, multiple locomotion gaits are available, namely, *walking gait* (standard operation, most stable), *stair climbing gait* (reduces the robot speed and pitches the robot to better see the stairs while descending), *crawling gait* (three feet touch the ground at all times, the most stable gait)[1]. Furthermore, the height of the robot can also be dynamically adjusted; the height reduction to fit in small spaces is henceforth called the *crouch gait*. These locomotion gaits provide greater mobility for the Spot in areas that would not be traversable with the standard walking gait.

- **Stair gait selection** Stair detection was performed using geometric analysis of LiDAR and camera disparity pointcloud data. To detect stairs, we use a plane fitting method similar to [117]. We compute normals and perform clustering to identify contiguous surfaces in the pointcloud of similar slope. We then isolate flat and vertical planes to detect potential stair surfaces. After finding candidate stair surfaces, the larger stair direction and overall plane is determined. Finally, the intersection of the ground plane that the robot currently sits on and the plane of the stairs is used to determine the start of

[1]Further descriptions of the gaits can be found here: `https://support.bostondynamics.com/s/article/Operating-Spot`

the stairs. Once the start of the stairs (either up or down) is determined, the robot can align itself to this location and begin a stair walking gait.

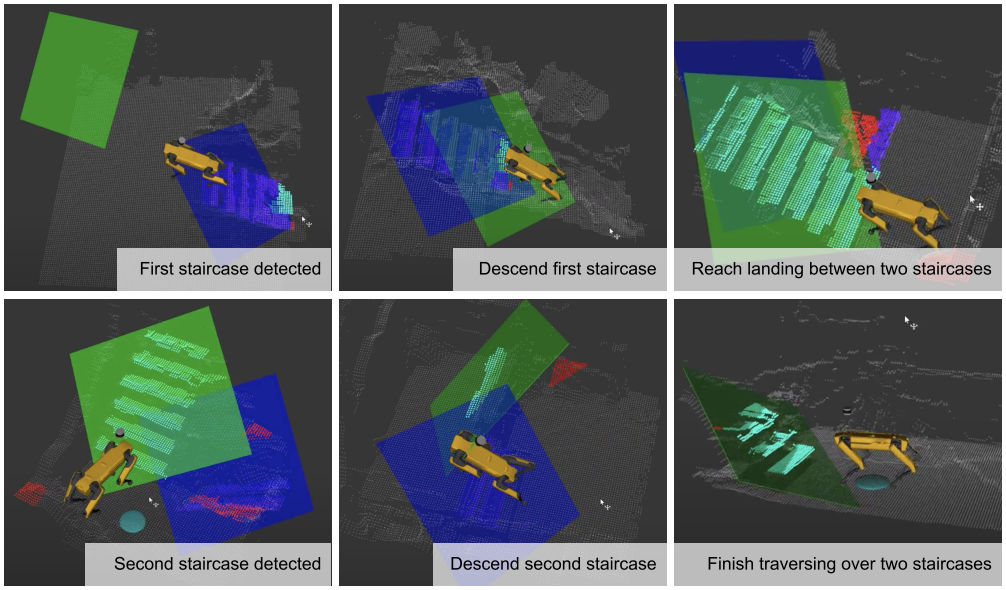


Figure 3.10: Stair detection with plane-fitting method

- **Crouch gait through slip detection** A learning-based slip prediction module was implemented leveraging the joint state information of the quadruped at each time instant. We train an ensemble model to predict the probability of slip, $p(slip)$, at each time instant, based on the robot limb kinematics and kinetics. Slip-annotations from multiple field trials across different terrains (e.g., rocky, sandy, muddy terrains) formed the ground truth for the model training. During slip prediction using the trained model, if the $p(slip) > 0.5$, the *crawling gait* mode is enabled for the robot to traverse the potentially slippery terrain. For more details on the implementation of the slip-predictive model see [35].

- **Crouch behavior in low ceiling areas** If the robot detects a region with low ceiling that is within 16 cm of the robot height (the limit to which the robot can bend), the robot enables *crouch gait* that reduces its height of the robot by 16 cm. Once the robot passes through the low ceiling area, it switches back to the *walking gait*.

Recovery Behaviors

Recovery behaviors are fast, reactive actions ensure that if the robot is doing something that is hazardous that will end its mission, the robot can recover from that

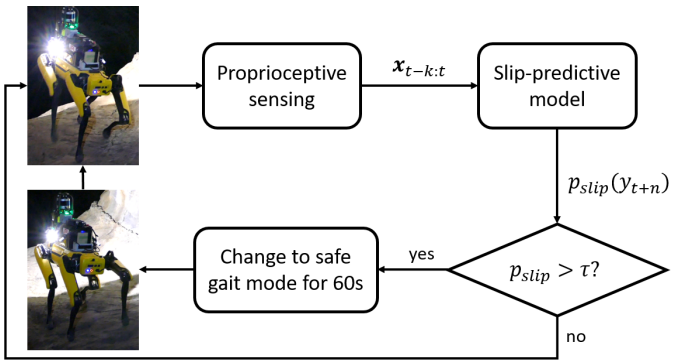


Figure 3.11: Gait change through slip detection

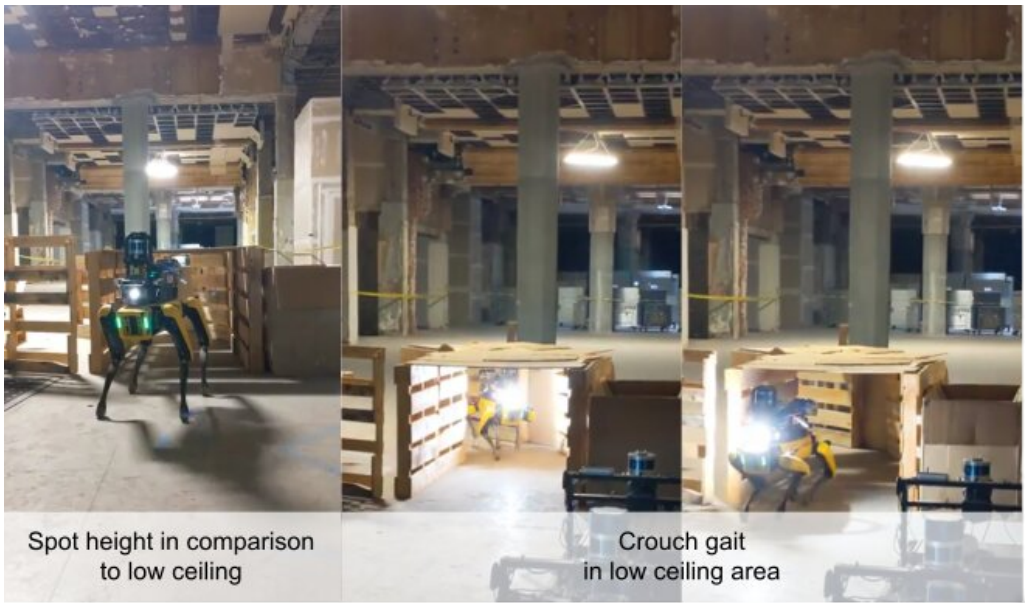


Figure 3.12: Crouch behavior in low ceiling areas

situation quickly and safely. In the case that a recovery behavior is activated, the robot temporarily pauses following the commands from the MPC pipeline to get to a safe state from where it can resume the kinodynamic planning pipeline.

Tilt recovery The tilt recovery behavior tracks the odometry received from the state estimation pipeline for the wheeled robot. The odometry is a combination of wheel odometry and inertial measurements. It consists of the pose and twist measurements. If the pitch of the robot is above a certain threshold, the robot immediately backtracks along the same path that got the robot in the lethal pitch state at a low speed.

Wiggle behavior We monitor the velocity of the robot obtained from the wheel, inertial odometry to ensure that the robot is not stuck during mission. If the robot

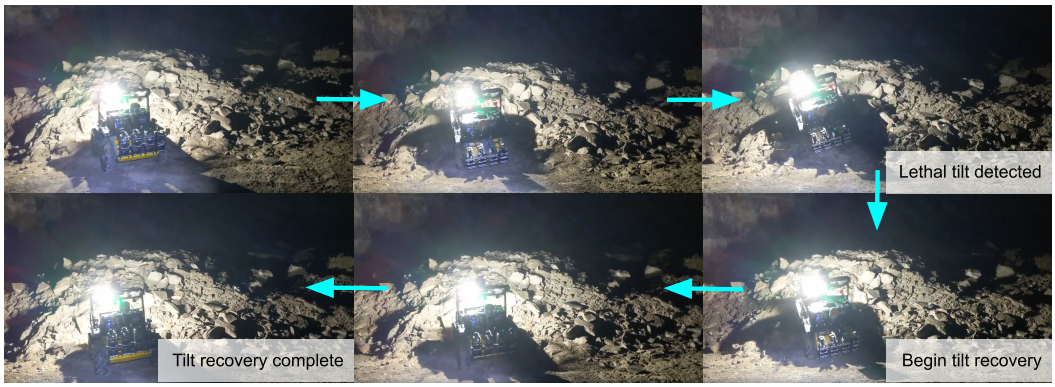


Figure 3.13: Tilt recovery behavior when the robot pitch exceeds the safe limit.

does not move for a certain time period even though the MPC pipeline is commanding the robot to move, the robot is considered to be stuck. In the case that the robot is stuck, we deploy a wiggle behavior that moves the wheels of the robot in such a way that, unintruded, the robot would travel in a figure-8 pattern. The allows the robot to wiggle back and forth till it is not stuck anymore.

Escape lethal If the robot finds itself trapped in a lethal zone for some period of time, it is useful to have a behavior which attempts to escape the high risk regions with minimum risk. This is accomplished in the following way. First, a goal location is chosen which is not in a lethal area, close to the robot, and which minimizes the amount of lethal area the robot would need to traverse to reach the goal in a straight line. Next, the risk threshold of the traversability cost maps are changed, to allow traversal through high risk map cells which would normally be treated as constraints or obstacles. Then, the goal is sent to the global and local planners, and executed on the robot. If the goal is reached, the recovery behavior is ended, but if the goal is still not reachable, the risk threshold can be further adjusted to allow more and more risky behavior in an attempt to escape. In this way, the CVaR risk threshold is useful for adapting the risk profile of the robot in real-time.

3.4 Experiments

In this section, we report the experimental and field performance of STEP. We first present a comparative study between different adjustable risk thresholds in simulation on a wheeled differential drive platform. Then, we demonstrate real-world performance using a wheeled robot deployed in an abandoned subway filled with clutter, and a legged platform deployed in a lava tube environment.

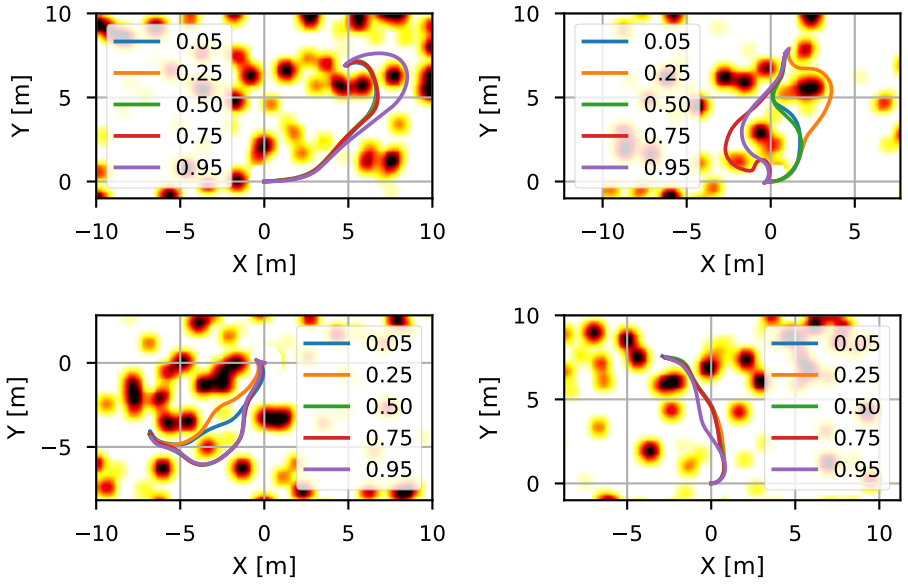


Figure 3.14: Path distributions from four simulated runs. The risk level α spans from 0.1 (close to mean-value) to 0.95 (conservative). Smaller α typically results in a shorter path, while larger α chooses statistically safe paths.

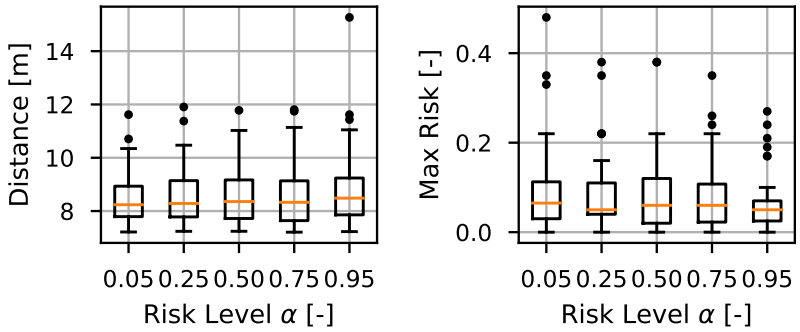


Figure 3.15: Distance vs risk trade-off from 50 Monte-Carlo simulations. Left: Distributions of path distance. Right: Distributions of max risk along the traversed paths. Box plot uses standard quartile format and dots are outliers.

Simulation Study

To assess statistical performance, we perform 50 Monte-Carlo simulations with randomly generated maps and goals. These results were also included in our previous work [41]. Random traversability costs are assigned to each grid cell. The following assumptions are made: 1) no localization error, 2) no tracking error, and 3) a simplified perception model with artificial noise. We give a random goal 8 m away and evaluate the path cost and distance. We use a differential-drive dynamics model (no lateral velocity).

We compare STEP using different α levels. Figure 3.14 shows the distribution of

paths for different planning configurations. The optimistic (close to mean-value) planner $\alpha = 0.05$ typically generates shorter paths, while the conservative setting $\alpha = 0.95$ makes long detours to select statistically safer paths. The other α settings show distributions between these two extremes, with larger α generating similar paths to the conservative planner and smaller α generating more time-optimal paths. Statistics are shown in Figure 3.15.

Hardware Results

We deployed STEP on two different robots (wheeled and legged) in different challenging environments (an abandoned subway, a lava tube, a limestone mine, and a combination of different subterranean environments in the DARPA Subterranean Challenge). The robot was equipped with custom sensing and computing units, and driven by JPL's NeBula autonomy software [1]. 3 Velodyne VLP-16s were used for collecting LiDAR data. Localization was provided onboard by a LiDAR-based SLAM solution [39, 105]. The entire autonomy stack runs on an Intel Core i7 CPU. The typical CPU usage for the traversability stack is about a single core.

Previous Results

Earlier versions of STEP (without semantic and confidence-aware risk sources, gait adaptations, and the entire suite of recovery behaviors) were tested in multiple field locations. For the sake of completeness, we have included these results, published in [41], from:

1. Abandoned subway filled with industrial clutter in Downtown Los Angeles, CA in Figure 3.16,
2. Valentine Cave in Lava Beds National Monument, Tulelake, CA in Figure 3.17

Results from the Kentucky Underground and LA Subway

We tested the various new parts of our traversability pipeline during fully autonomous runs in our tests at Kentucky Underground Storage (KU), Wilmore, KY. The results of the cost map for confidence-based and semantics-based risk are seen in Figure 3.5 and Figure 3.6. We also tested the tilt recovery behavior on a pile of rubble in KU by forcing the robot into a hazardous tilt position, the robot was able to recover from this risky configuration as seen in Figure 3.13.

Figure 3.16: Traversability analysis results for Husky in an abandoned subway experiment. Top left to right: Risk maps at three varying risk levels: $\alpha = 0.1, 0.5, 0.9$, respectively. Colors correspond to CVaR value (white: safe ($r <= 0.05$), yellow to red: moderate ($0.05 < r <= 0.5$), black: risky ($r > 0.5$)). Also shown are the most recent LiDAR measurements (green points). Bottom left and middle: Front and right on-board cameras observing the same location. Bottom right: Completed top-down map of the environment after autonomous exploration. Bright dots are pillars, which are visible in the camera images. The effect of these risk analyses results in intuitive outcomes - for example, a low pile of metal, while probably traversable, should be avoided if possible. When the region has inadequate sensor coverage, the risk will be high. When the robot is closer and the sensor coverage is good, then the CVaR cost will decrease, yielding a more accurate risk assessment. This results in more efficient and safer planning when compared to deterministic methods.

We also tested new results in an abandoned subway with multiple levels filled with industrial clutter in Downtown Los Angeles, CA. For example, the LA Subway envirnoment served as a test bed for the stair gait and crouch gait behaviors that are shown in Figure 3.10 and Figure 3.12 respectively.

Results from the DARPA Subterranean Challenge

The traversability framework was deployed in the DARPA Subterranean Challenge held in Louisville Mega Cavern, Louisville, KY. The course consisted of 3 different environments - tunnel, urban, and cave. Figure 3.18 shows the course map and different sections that the robot explored. The sections showcased in Figure 3.18 have a high difficulty rating as listed in the course layout guide posted by DARPA[2]. The terrain challenges of each of the sections are listed below:

[2]The course guide can be accessed here: `https://bitbucket.org/subtchallenge/finals_ground_truth/src/master/course_design/Finals_Course_Callouts.pdf`

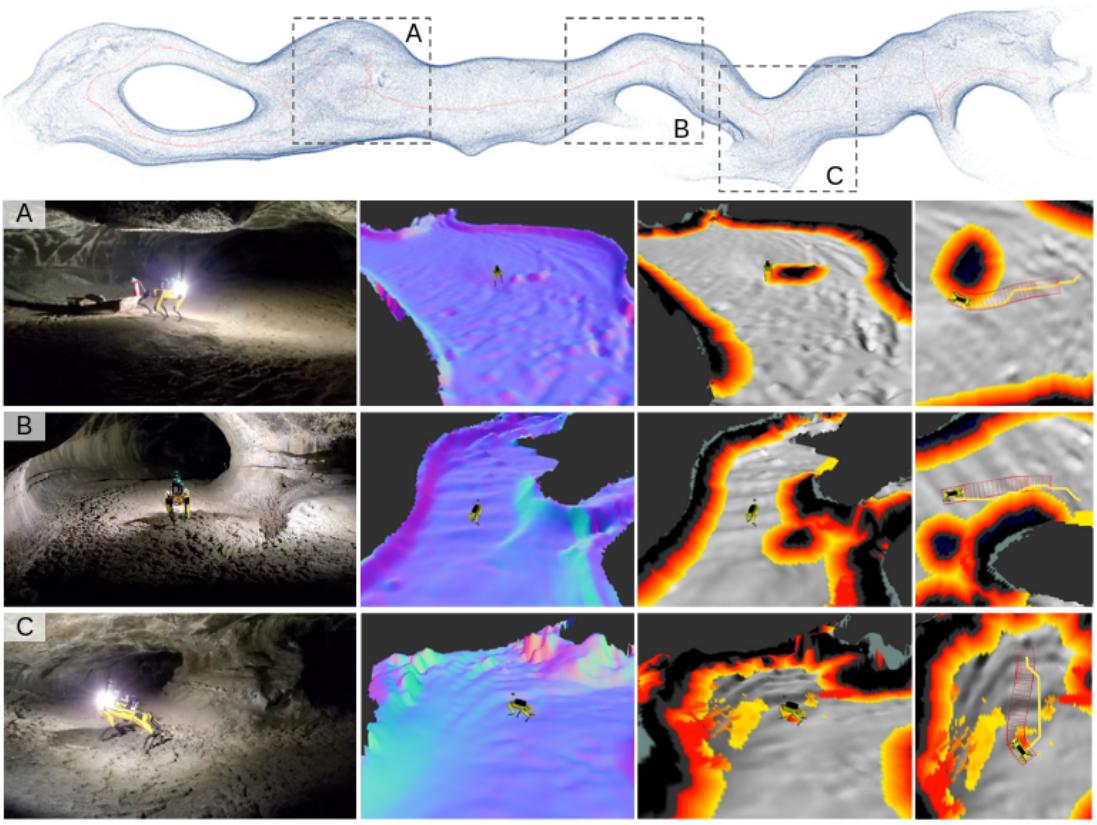


Figure 3.17: Traversability analysis results for the Valentine Cave experiment. From left to right: Third-person view, elevation map (colored by normal direction), risk map (colored by risk level. white: safe ($r <= 0.05$), yellow to red: moderate ($0.05 < r <= 0.5$), black: risky ($r > 0.5$)), and planned geometric/kinodynamic paths (yellow lines/red boxes).

Region A An office-like area that consists of narrow corridors ($5' \times 8'$) and small rooms ($8' \times 8'$) for the robots to explore.

Region B A warehouse-like area with a lot of shelving and clutter imitating an industrial warehouse after an earthquake.

Region C A connection between the urban and tunnel part of the course. The stairs act as a negative obstacle for the wheeled robots. The presence of this negative obstacle in a narrow corridor makes the drop harder to detect.

Region D A constrained passage with ground, wall, and ceiling obstacles - with vertical pipes and debris.

Region E A narrow cave opening mimicking a region that humans have to crawl through. The ground slopes upwards and the ceiling slopes downwards creating issues for ground segmentation and low-ceiling detection.

Region F A small limestone cave with rubble and loose rock piles.

Figure 3.19 shows the results of the traversability analysis and the geometric and kinodynamic plans of the robot in the different scenarios listed above. We see that the robot is able to plan safe paths in all of these scenarios and successfully navigate the different terrain hazards.

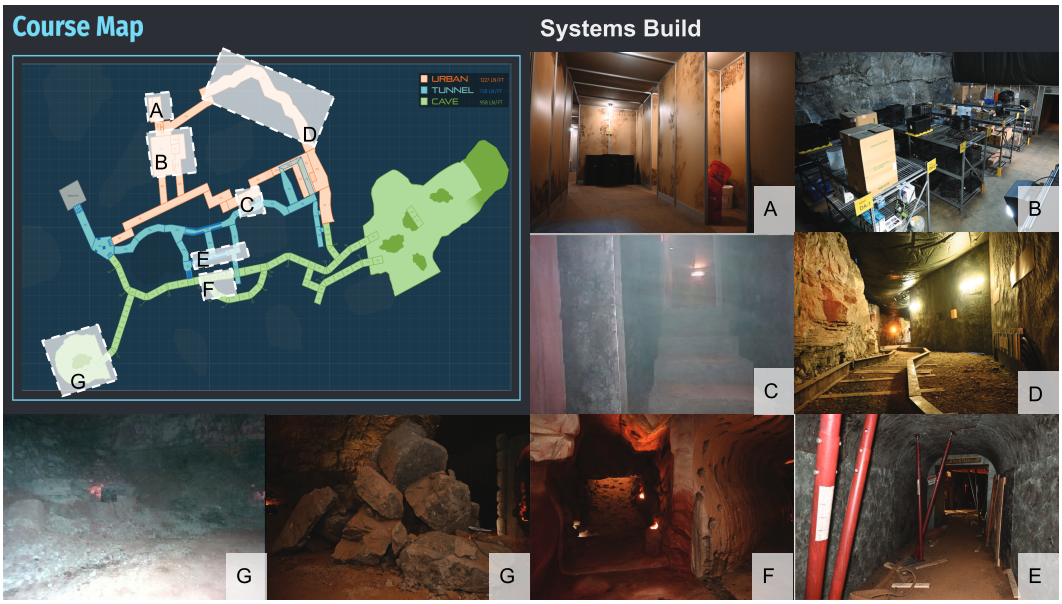


Figure 3.18: The layout of the competition course with snapshots of the systems build of the areas traversed by the robots that have interesting terrain and high difficulty.

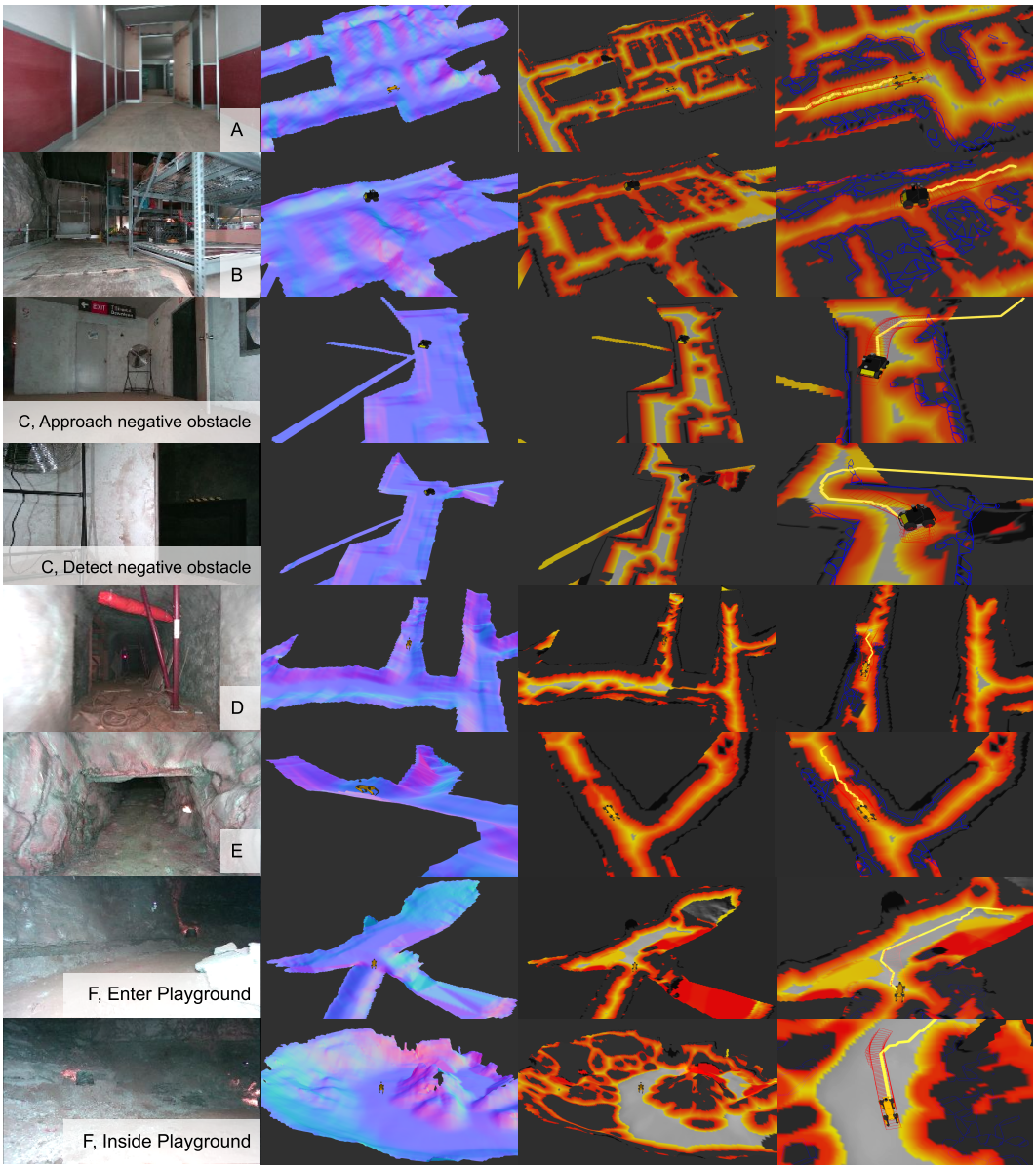


Figure 3.19: Traversability analysis results for the DARPA Subterranean Challenge. From left to right: robot front camera view, elevation map (colored by normal direction), risk map (colored by risk level- white: safe ($r <= 0.05$), yellow to red: moderate ($0.05 < r <= 0.5$), black: risky ($r > 0.5$)), and planned geometric/kinodynamic paths (yellow lines/red boxes).

Chapter 4

DISTRIBUTIONALLY-ROBUST MPC WITH TOTAL VARIATION DISTANCE

The previous chapter considered risk-aware traversability evaluation in a priori unknown environments. The risk-based cost map was used in the geometric and kinodynamic planner directly in the cost and constraints. However, in the previous chapter, we did not consider the theory behind the risk evaluation of the cost and constraints used in the planning framework. This chapter studies a method to compute control in the cases when the uncertainty distribution (like the distribution of the process noise) is not known exactly, but is known to lie in a *set of distributions*. Hence, this chapter studies the problem of distributionally robust model predictive control (MPC) using total variation distance ambiguity sets. For a discrete-time linear system with additive disturbances, we provide a conditional value-at-risk reformulation of the MPC optimization problem that is distributionally robust in the expected cost and chance constraints. The distributionally robust chance constraint (DRCC) is over-approximated as a simpler, tightened chance constraint that reduces the computational burden. Numerical experiments support our results on probabilistic guarantees and computational efficiency.

This chapter was adapted from:

Anushri Dixit, Mohamadreza Ahmadi, and Joel W. Burdick. Distributionally robust model predictive control with total variation distance. IEEE Control Systems Letters, 6:3325–3330, 2022. doi: 10.1109/LCSYS.2022.3184921.

4.1 Introduction and Background

There are many ways to incorporate uncertainty in MPC. Robust MPC accounts for worst-case disturbances in a set of bounded uncertainties [16]. This approach is often too conservative, since it does not account for the distribution of the uncertainties. Stochastic MPC (SMPC) [93] minimizes the expected value of a cost function, while respecting a bound on the probability of violating state and control constraints (also called chance constraints). Risk-aware MPC methods use coherent risk measures [37, 125] to account for variations in the underlying distribution of uncertainty. This is convenient since one often only has an estimate of the true uncertainty distribution. As discussed in the preliminaries, this notion of allowing

for variation in the underlying distribution is called distributional robustness. It is important to note that coherent risk measures have heretofore provided distributional robustness in the cost, but not in the chance constraints that one may need to satisfy in SMPC. This chapter extends the use of coherent risk measures to provide distributional robustness in the chance constraints.

The total variation distance (TVD) is a commonly used bounding metric on probability spaces. Intuitively, it provides an upper bound on the difference of probabilities that an event occurs under two measures [53], see Figure 4.1 for an illustration of TVD. In [136], the authors provide a TVD-based distributionally robust solution of the linear, quadratic regulator and use this formulation for a drop–shipping retail fulfillment application. In [148], the authors consider a data-driven strategy to solve iterative tasks using a MPC scheme. This framework is amenable to general ambiguity sets, including TVD.

Definition 3 (Total Variation Distance Risk [120]) *The total variation distance-based risk measure for a cost random variable, X is given by,*

$$TVD_{1-\alpha}(X) := \sup_{Q \in \mathcal{Q}_{TVD}} E_Q(X) = \alpha \sup_{x \in \Omega} x + (1-\alpha) CVaR_{1-\alpha}(X),$$

where the ambiguity set of $TVD_{1-\alpha}(X)$ is given by,

$$\mathcal{Q}_{TVD} := \left\{ q \in \Delta_J : d_{TV}(p,q) = \frac{1}{2} \sum_{j=1}^{J} |q(j) - p(j)| \leq \alpha \right\},$$

where Δ_J is the probability simplex, $\Delta_J := \{q \in \mathbb{R}^J \mid q \geq 0, \sum_{j=1}^{J} q(j) = 1\}$ and p refers to the probability mass function associated with the random variable X. Gibbs and Su [53] survey the relationships between total variation and various other probability metrics/distances.

We provide a deterministic approximation of a stochastic MPC optimization problem with a distributionally robust objective and DRCC with TVD ambiguity sets for a discrete distribution. This is achieved by over-approximating the DRCC in the form of a simple, but more conservative, chance constraint that is further simplified using CVaR. The objective is also reformulated as a CVaR objective. The resulting MPC optimization is an efficient quadratic program.

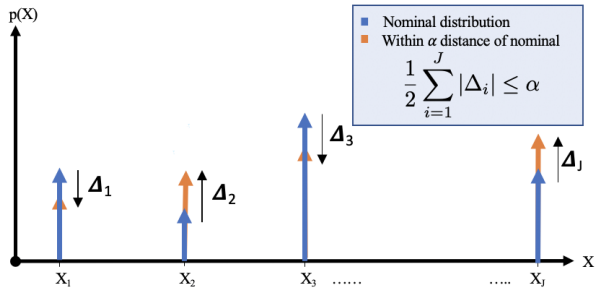


Figure 4.1: Depiction of the Total Variation Distance (TVD) between discrete probability distributions. Our method is robust to any distribution within TVD $\leq \alpha$ of a nominal distribution, where $\alpha \in (0, 1)$.

4.2 Problem formulation

We consider a class of discrete-time systems given by

$$\boldsymbol{x}(t+1) = A\boldsymbol{x}(t) + B\boldsymbol{u}(t) + D\boldsymbol{\delta}(t) \tag{4.1}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^{n_x}$ and $\boldsymbol{u}(t) \in \mathbb{R}^{n_u}$ are the system state and controls at time t, respectively. The system is affected by a stochastic, additive, process noise $\boldsymbol{\delta}_t \in \mathbb{R}^{n_d}$.

Assumption 1 (Discrete process noise) *The process noise $\boldsymbol{\delta}$ consists of i.i.d. samples of a discrete distribution given by the probability mass function (pmf), $\boldsymbol{p} = [p(1), p(2), \ldots, p(J)]^T$. For this distribution, we also define the index set $\mathcal{D} = \{1, \ldots, J\}$.* [1]

Consider there are r state constraints that take the form

$$\mathcal{X} := \{\boldsymbol{x} \in \mathbb{R}^{n_x} | F_x \boldsymbol{x} \leq g_x\}, F_x \in \mathbb{R}^{r \times n_x}, g_x \in \mathbb{R}^r.$$

In this work, we are interested in satisfying these state constraints in a distributionally robust manner,

$$\begin{aligned} 1 - \epsilon &\leq \min_{Q \in \boldsymbol{Q}} \mathbb{P}_Q(\boldsymbol{x} \in \mathcal{X}) \\ &= 1 - \max_{Q \in \boldsymbol{Q}} \mathbb{P}_Q(\boldsymbol{x} \notin \mathcal{X}) \\ &= 1 - \max_{Q \in \boldsymbol{Q}} \mathbb{E}_Q(\mathbf{1}_{\boldsymbol{x} \notin \mathcal{X}}) \\ &= 1 - \rho(\mathbf{1}_{\boldsymbol{x} \notin \mathcal{X}}), \end{aligned}$$

[1] The results in this work can be extended to continuous distributions using sample average approximation [77] or other sample-based techniques [18].

where we used the property that the probability of an event can be expressed as the expected value of its indicator to express the DRCC as the risk of an indicator. Hence,

$$\min_{Q \in \mathcal{Q}_{\text{TVD}}} \mathbb{P}_Q(\boldsymbol{x} \in \mathcal{X}) \geq 1 - \epsilon \iff \text{TVD}_{1-\alpha}(\mathbf{1}_{\boldsymbol{x} \notin \mathcal{X}}) \leq \epsilon. \tag{4.2}$$

Problem 1 *Consider the discrete-time system* (5.1)*. Given a deterministic initial condition* $\boldsymbol{x}_0 \in \mathbb{R}^{n_x}$*, state constraints* $\mathcal{X} \subset \mathbb{R}^{n_x}$*, convex polytopic control constraints* $\mathcal{U} \subset \mathbb{R}^{n_u}$*, a convex stage cost* $c : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}_{\geq 0}$*, a horizon* $N \in \mathbb{N}$*, and risk tolerance* $\epsilon \in (0, 1)$ *for state constraints, compute the receding horizon controller* $\boldsymbol{u} = \{\boldsymbol{u}_k\}_{k=0}^{N-1}$ *such that the total cost* $\mathcal{J}(x(t), \boldsymbol{u})$ *is minimized while satisfying the distributionally robust constraints* (5.9)*, i.e., the solution to the following optimization problem,*

$$\min_{\boldsymbol{u}} \quad \mathcal{J}(x(t), \boldsymbol{u}) := TVD_{1-\alpha}\left(\sum_{k=0}^{N-1} c(\boldsymbol{x}_k, \boldsymbol{u}_k)\right) \tag{4.3a}$$

$$s.t. \quad \boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + B\boldsymbol{u}_k + D\boldsymbol{\delta}_k, \tag{4.3b}$$

$$TVD_{1-\alpha}(\mathbf{1}_{\boldsymbol{x}_k \notin \mathcal{X}}) \leq \epsilon, \tag{4.3c}$$

$$\boldsymbol{u}_k \in \mathcal{U}, \quad \boldsymbol{x}_0 = \boldsymbol{x}(t), \, \forall k \in \{0, \dots N-1\}, \tag{4.3d}$$

where, $\boldsymbol{x}_k = \boldsymbol{x}(t+k|t)$ *and* $\boldsymbol{u}_k = \boldsymbol{u}(t+k|t)$*.*

4.3 MPC reformulation

The batch form of Equation (5.11b) can be re-written as

$$\begin{aligned}\boldsymbol{x}_{k+1} &= A^{k+1}\boldsymbol{x}_0 + \sum_{i=0}^{k}\left(A^i B\boldsymbol{u}_{k+1-i} + A^i D\boldsymbol{\delta}_{k+1-i}\right), \\ &= A^{k+1}\boldsymbol{x}_0 + \boldsymbol{B}_{k+1}\bar{\boldsymbol{u}}_{k+1} + \boldsymbol{D}_{k+1}\bar{\boldsymbol{\delta}}_{k+1},\end{aligned}$$

where, $\boldsymbol{B}_k$, $\bar{\boldsymbol{u}}_k$, $\boldsymbol{D}_k$, and $\bar{\boldsymbol{\delta}}_k$ are given as,

$$\begin{aligned}\boldsymbol{B}_k &= \begin{bmatrix} A^{k-1}B & A^{k-2}B & \dots & B \end{bmatrix} \in \mathbb{R}^{n_x \times k n_u} \\ \bar{\boldsymbol{u}}_k &= \begin{bmatrix} \boldsymbol{u}_1^T & \boldsymbol{u}_2^T & \dots & \boldsymbol{u}_k^T \end{bmatrix}^T \in \mathbb{R}^{k n_u \times 1} \\ \boldsymbol{D}_k &= \begin{bmatrix} A^{k-1}D & A^{k-2}D & \dots & D \end{bmatrix} \in \mathbb{R}^{n_x \times k n_d} \\ \bar{\boldsymbol{\delta}}_k &= \begin{bmatrix} \boldsymbol{\delta}_1^T & \boldsymbol{\delta}_2^T & \dots & \boldsymbol{\delta}_k^T \end{bmatrix}^T \in \mathbb{R}^{k n_d \times 1}.\end{aligned}$$

Clearly, as k increases, the disturbance effects compound. At each step k, the distribution of $\boldsymbol{x}_k$ is given by the joint distribution of $(\boldsymbol{\delta_1}, \ldots, \boldsymbol{\delta_k})$. Let $\boldsymbol{p}_k$ denote the probability mass function of this joint distribution. The size of this joint pmf is J^k (see Assumption 1).

The following key result provides an over approximation of the DRCC for TVD ambiguity sets.

Lemma 1 (Risk Reformulation) *If Assumption 2 holds, then* (5.11e) *is satisfied if the following constraint is satisfied,*

$$\mathbb{P}(\boldsymbol{x}_k \notin \mathcal{X}) + \alpha \leq \epsilon. \tag{4.4}$$

Proof: We express the risk constraint through its dual representation. Note that we are finding the worst-case expectation within the TVD-based risk envelope. The distribution that gives us this worst-case expectation has the pmf $\boldsymbol{q} \circ \boldsymbol{p}_k$ (where $\circ$ denotes element-wise multiplication) in the following optimization,

$$\begin{aligned}
\mathrm{TVD}_{1-\alpha}(\mathbf{1}_{\boldsymbol{x}_k \notin \mathcal{X}}) = \max_{\boldsymbol{q}} \; & \sum_{j=1}^{J^k} q(j) p_k(j) \mathbf{1}_{\boldsymbol{x}_k^j \notin \mathcal{X}} \\
\text{s.t.} \; & \sum_{j=1}^{J^k} q(j) p_k(j) = 1, \; q(j) \geq 0, && \text{(4.5a)} \\
& \sum_{j=1}^{J^k} \frac{1}{2} |q(j) p_k(j) - p_k(j)| \leq \alpha. \\
= \min_{\boldsymbol{\lambda_1} \in \mathbb{R}^{J^k}, \lambda_2, \nu \in \mathbb{R}} \; & \max_{\boldsymbol{q}} \quad \mathcal{L}(\boldsymbol{\lambda}_1, \lambda_2, \nu) && \text{(4.5b)} \\
\text{s.t.} \; & \quad \boldsymbol{\lambda}_1 \geq 0, \; \lambda_2 \geq 0.
\end{aligned}$$

where, $\mathcal{L}(\boldsymbol{\lambda}_1, \lambda_2, \nu)$ is the Lagrangian of the constrained optimization given in (4.5a) given by,

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\lambda}_1, \lambda_2, \nu) = & \sum_{j=1}^{J^k} q(j) \left[p_k(j) \mathbf{1}_{\boldsymbol{x}_k^j \notin \mathcal{X}} + \nu p_k(j) + \lambda_1(j) \right] \\
& - \lambda_2 \Big(\sum_{j=1}^{J^k} p_k(j) \underbrace{|q(j) - 1|}_{f(q(j))} - 2\alpha \Big) - \nu.
\end{aligned}$$

The inner maximization of (4.5b) can be solved by using convex conjugate of the function $f(x) = |x - 1|$ given by,

$$f^*(y) = \begin{cases} y & |y| \le 1 \\ +\infty & |y| > 1 \end{cases}.$$

Hence, we obtain

$$\max_{\boldsymbol{q}} \mathcal{L}(\boldsymbol{\lambda}_1, \lambda_2, \nu) = \lambda_2 \sum_{j=1}^{J^k} p_k(j) f^*\big((\lambda_2 p_k(j))^{-1}\big(p_k(j)\mathbf{1}_{\boldsymbol{x}_k^j \notin \mathcal{X}} + \nu p_k(j) + \lambda_1(j)\big)\big) + 2\lambda_2\alpha - \nu.$$

Now we substitute the above convex conjugate f^* into (4.5b)[2],

$$\begin{aligned}
&\mathrm{TVD}_{1-\alpha}(\mathbf{1}_{\boldsymbol{x}_k \notin \mathcal{X}}) \\
&= \min_{\boldsymbol{\lambda_1}, \lambda_2, \nu} \sum_{j=1}^{J^k} \big(p_k(j)\mathbf{1}_{\boldsymbol{x}_k^j \notin \mathcal{X}} + \nu p_k(j) + \lambda_1(j)\big) + 2\lambda_2\alpha - \nu \qquad (4.6a) \\
&\quad \text{s.t.} \quad \boldsymbol{\lambda}_1 \ge 0,\ \lambda_2 \ge 0, \\
&\quad -1 \le (\lambda_2 p_k(j))^{-1}\big(p_k(j)\mathbf{1}_{\boldsymbol{x}_k^j \notin \mathcal{X}} + \nu p_k(j) + \lambda_1(j)\big) \le 1 \\
&= \min_{\boldsymbol{\lambda_1}, \lambda_2, \nu} \mathbb{P}(\boldsymbol{x}_k \notin \mathcal{X}) + \nu + \sum_{j=1}^{J^k} \lambda_1(j) + 2\lambda_2\alpha - \nu \\
&\quad \text{s.t.} \quad \boldsymbol{\lambda}_1 \ge 0,\ \lambda_2 \ge 0, \qquad (4.6b) \\
&\qquad -\lambda_2 p_k(j) - \nu p_k(j) - \lambda_1(j) \le p_k(j)\mathbf{1}_{\boldsymbol{x}_k^j \notin \mathcal{X}}, \\
&\qquad p_k(j)\mathbf{1}_{\boldsymbol{x}_k^j \notin \mathcal{X}} \le \lambda_2 p_k(j) - \nu p_k(j) - \lambda_1(j), \\
&\le \min_{\boldsymbol{\lambda_1}, \lambda_2, \nu} \mathbb{P}(\boldsymbol{x}_k \notin \mathcal{X}) + \sum_{j=1}^{J^k} \lambda_1(j) + 2\lambda_2\alpha \\
&\quad \text{s.t.} \quad \boldsymbol{\lambda}_1 \ge 0,\ \lambda_2 \ge 0, \qquad (4.6c) \\
&\qquad -\lambda_2 p_k(j) - \nu p_k(j) - \lambda_1(j) \le 0, \\
&\qquad p_k(j) \le \lambda_2 p_k(j) - \nu p_k(j) - \lambda_1(j), \\
&= \mathbb{P}(\boldsymbol{x}_k \notin \mathcal{X}) + \alpha. \qquad (4.6d)
\end{aligned}$$

In the above equations, we first substituted f^* and constrained the argument of the conjugate to lie in $[-1, 1]$ in (4.6a) considering that the conjugate is unbounded

[2]We can take the inverse of λ_2 in the conjugate because the solution of (4.5a) always lies on the boundary of the TVD constraint $\sum_{j=1}^{J} \frac{1}{2}|q(j)p_k(j) - p_k(j)| \le \alpha$, i.e., the optimal $\lambda_2 > 0$.

outside this range. Afterwards, we noted that

$$\mathbb{P}(\boldsymbol{x}_k \notin \mathcal{X}) = \sum_{j=1}^{J^k} p_k(j)\mathbf{1}_{\boldsymbol{x}_k^j \notin \mathcal{X}},$$

and re-arranged the inequality constraints on $(\lambda_2 p_k(j))^{-1}\big(p_k(j)\mathbf{1}_{\boldsymbol{x}_k^j \notin \mathcal{X}} + \nu p_k(j) + \lambda_1(j)\big)$ to obtain (4.6b).

Next, we make the following constraints stricter in (4.6c)

$$\begin{aligned} -\lambda_2 p_k(j) - \nu p_k(j) - \lambda_1(j) \leq 0 \implies \\ -\lambda_2 p_k(j) - \nu p_k(j) - \lambda_1(j) \leq p_k(j)\mathbf{1}_{\boldsymbol{x}_k^j \notin \mathcal{X}}, \\ p_k(j) \leq \lambda_2 p_k(j) - \nu p_k(j) - \lambda_1(j) \implies \\ p_k(j)\mathbf{1}_{\boldsymbol{x}_k^j \notin \mathcal{X}} \leq \lambda_2 p_k(j) - \nu p_k(j) - \lambda_1(j), \end{aligned}$$

to get a upper bound on (4.6b) that is independent of the state x_k^j. Finally, we note that $(\boldsymbol{\lambda}_1, \lambda_2, \nu) = (\mathbf{0}, 0.5, -0.5)$ satisfies the KKT condition [24] and hence is the optimal solution to (4.6c) to complete the proof. ■

Remark 1 *A natural interpretation of the tightening provided in Lemma 1 can be seen in Figure 4.1: one could have predicted that the probability of constraint violation can vary at most by* 2α*. The above proof provides a state-independent way to realize a similar tightening. It may be more conservative than the TVD constraint value, but it simply approximates the distributionally robust chance constraint by a more conservative chance constraint (with lower violation probability). If* $\alpha > \epsilon$*, then the chance constraint becomes* $\mathbb{P}(\boldsymbol{x}_k \notin \mathcal{X}) < 0$*, which is impossible to satisfy and indicates that the conservativeness of the chance constraint should be reduced by decreasing* α *or increasing* ϵ*.*

Lemma 2 *If Assumption 2 holds, the TVD constraint* (5.11e) *is satisfied if the following constraint is satisfied,*

$$F_x\tilde{\boldsymbol{x}}_k + CVaR_{\epsilon-\alpha}\big(F_x\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k\big) \leq g_x, \tag{4.7}$$

where $\tilde{\boldsymbol{x}}_k$ *is the undisturbed nominal state:* $\boldsymbol{x}_k = \tilde{\boldsymbol{x}}_k + D\bar{\boldsymbol{\delta}}_k$*.*

Proof: In [100], Nemirovski and Shapiro showed that CVaR provides a convex conservative approximation of the chance constraint. We have shown in Lemma 1

that (5.11e) is satisfied if (4.4) holds. Hence, a conservative approximation of $\mathbb{P}(\boldsymbol{x}_k \notin \mathcal{X}) = \mathbb{P}(F_x\boldsymbol{x}_k - g_x > 0) \leq \epsilon - \alpha$ is given by,

$$\begin{aligned} 0 &\geq \mathrm{CVaR}_{\epsilon-\alpha}(F_x\boldsymbol{x}_k - g_x) \\ &= \mathrm{CVaR}_{\epsilon-\alpha}(F_x\tilde{\boldsymbol{x}}_k + F_x\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k - g_x)) \\ &= F_x\tilde{\boldsymbol{x}}_k - g_x + \mathrm{CVaR}_{\epsilon-\alpha}(F_x\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k), \end{aligned}$$

where we obtain the first step by plugging in $\boldsymbol{x}_k = \tilde{\boldsymbol{x}}_k + D\bar{\boldsymbol{\delta}}_k$ and the next step follows from the translational invariance property of coherent risk measures. ■

Lemmas 1, 2 provide a simple tightening of the state constraints (5.11e).

Remark 2 *The chance constraint can be further tightened using the positive homogeneity property of coherent risk measures and i.i.d assumption on all disturbances,*

$$F_x\tilde{\boldsymbol{x}}_k + \|F_xD\|_1 CVaR_{\epsilon-\alpha}(|\delta|) \leq g_x. \tag{4.8}$$

This tightening further reduces the size of the optimization problem, as $\|F_xD\|_1 CVaR_{\epsilon-\alpha}(|\delta|)$ can be expressed with approximately J^k fewer optimization variables and $2J^k$ fewer constraints for each $k \in \{0, \ldots N-1\}$when computed online.

Lemma 3 (Cost function) *If the cost function given in* (4.3a) *is expressed as a quadratic cost with*

$$c(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{x}_k^T Q\boldsymbol{x}_k + \boldsymbol{u}_k^T R\boldsymbol{u}_k,$$

then the MPC cost $\mathcal{J}(\boldsymbol{x}_t, \boldsymbol{u})$ is equivalently expressed as:

$$\begin{aligned} &\mathcal{J}(\boldsymbol{x}_t, \boldsymbol{u}) \\ &= \min_{m,z,\boldsymbol{s}} \sum_{k=0}^{N-1} c(\tilde{\boldsymbol{x}}_k, \boldsymbol{u}_k) + \alpha m + (1-\alpha)\sum_{j=1}^{J^N} p_N(j)\big(z + \frac{s(j)}{1-\alpha}\big) && \text{(4.9a)} \\ &\quad s.t. \quad m \geq \sum_{k=0}^{N-1} \big(\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k^j + 2A^k\boldsymbol{x}_0 + 2\boldsymbol{B}_k\bar{\boldsymbol{u}}_k\big)^T Q\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k^j, && \text{(4.9b)} \\ &\qquad\quad s(j) + z \geq \sum_{k=0}^{N-1} \big(\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k^j + 2A^k\boldsymbol{x}_0 + 2\boldsymbol{B}_k\bar{\boldsymbol{u}}_k\big)^T Q\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k^j, && \text{(4.9c)} \\ &\qquad\quad s(j) \geq 0, \quad \forall j \in \{1, \ldots, J^N\} && \text{(4.9d)} \end{aligned}$$

Proof: Consider (4.3a) with a quadratic stage cost $c(\boldsymbol{x}_k, \boldsymbol{u}_k)$.

$$\begin{aligned}\mathcal{J}(\boldsymbol{x}_0, \boldsymbol{u}) &= \text{TVD}_{1-\alpha}\left(\sum_{k=0}^{N-1} c(\boldsymbol{x}_k, \boldsymbol{u}_k)\right) \\ &= \alpha \sup_{(\boldsymbol{\delta_1}, \ldots, \boldsymbol{\delta_k})} \sum_{k=0}^{N-1} c(\boldsymbol{x}_k, \boldsymbol{u}_k) + (1-\alpha)\text{CVaR}_{1-\alpha}\left(\sum_{k=0}^{N-1} c(\boldsymbol{x}_k, \boldsymbol{u}_k)\right)\end{aligned}$$

Our goal is to find the worst-case value and the CVaR of the total stage cost. Consider the nominal state update equation,

$$\tilde{\boldsymbol{x}}_{k+1} = A^{k+1}\boldsymbol{x}_0 + \boldsymbol{B}_{k+1}\bar{\boldsymbol{u}}_{k+1} \tag{4.10}$$

We can write the quadratic stage cost as a function of the nominal state (without any disturbance effects) as,

$$\begin{aligned}&c(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{x}_k^T Q \boldsymbol{x}_k + \boldsymbol{u}_k^T R \boldsymbol{u}_k \\ &= (A^k\boldsymbol{x}_0 + \boldsymbol{B}_k\bar{\boldsymbol{u}}_k + \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k)^T Q (A^k\boldsymbol{x}_0 + \boldsymbol{B}_k\bar{\boldsymbol{u}}_k + \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k) + \boldsymbol{u}_k^T R \boldsymbol{u}_k \\ &= (A^k\boldsymbol{x}_0 + \boldsymbol{B}_k\bar{\boldsymbol{u}}_k)^T Q (A^k\boldsymbol{x}_0 + \boldsymbol{B}_k\bar{\boldsymbol{u}}_k) + 2(A^k\boldsymbol{x}_0)^T Q \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k \\ &\qquad + 2(\boldsymbol{B}_k\bar{\boldsymbol{u}}_k)^T Q \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k + (\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k)^T Q \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k + \boldsymbol{u}_k^T R \boldsymbol{u}_k \\ &= \tilde{\boldsymbol{x}}_k^T Q \tilde{\boldsymbol{x}}_k + \boldsymbol{u}_k^T R \boldsymbol{u}_k + (\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k)^T Q \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k + 2(A^k\boldsymbol{x}_0)^T Q \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k + 2(\boldsymbol{B}_k\bar{\boldsymbol{u}}_k)^T Q \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k \\ &= c(\tilde{\boldsymbol{x}}_k, \boldsymbol{u}_k) + \left(\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k + 2A^k\boldsymbol{x}_0 + 2\boldsymbol{B}_k\bar{\boldsymbol{u}}_k\right)^T Q \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k.\end{aligned}$$

We know from the translational invariance property of coherent risk measures that,

$$\text{TVD}_{1-\alpha}\left(\sum_{k=0}^{N-1} c(\boldsymbol{x}_k, \boldsymbol{u}_k)\right) = \sum_{k=0}^{N-1} c(\tilde{\boldsymbol{x}}_k, \boldsymbol{u}_k) + \text{TVD}_{1-\alpha}\left(\sum_{k=0}^{N-1}\left(\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k + 2A^k\boldsymbol{x}_0 + 2\boldsymbol{B}_k\bar{\boldsymbol{u}}_k\right)^T Q \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k\right).$$

Now, the above TVD is expressed as a combination of the worst-case value and the CVaR. The worst-case value is,

$$\begin{aligned}&\sup_{(\boldsymbol{\delta_1}, \ldots, \boldsymbol{\delta_k})} \sum_{k=0}^{N-1} \left(\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k + 2A^k\boldsymbol{x}_0 + 2\boldsymbol{B}_k\bar{\boldsymbol{u}}_k\right)^T Q \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k \\ &= \min_{m \in \mathbb{R}} \quad m \\ &\qquad \text{s.t.} \quad m \geq \sum_{k=0}^{N-1} \left(\boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k^j + 2A^k\boldsymbol{x}_0 + 2\boldsymbol{B}_k\bar{\boldsymbol{u}}_k\right)^T Q \boldsymbol{D}_k\bar{\boldsymbol{\delta}}_k^j \qquad \forall j \in \{1, \ldots, J^N\}.\end{aligned}$$

Hence the cost of the MPC can be rewritten as the minimization given by (4.9). ■

Using the reformulations afforded by Lemmas 1, 2, 3, we can reformulate the MPC optimization given in (5.11).

Theorem 4 *If there exists a solution to the following quadratic program,*

$$\min_{\boldsymbol{u},m,z,\boldsymbol{s}} \sum_{k=0}^{N-1} c(\tilde{\boldsymbol{x}}_k, \boldsymbol{u}_k) + \alpha m + (1-\alpha) \sum_{j=1}^{J^N} p_N(j)\left(z + \frac{s(j)}{1-\alpha}\right) \tag{4.11a}$$

$$s.t. \quad \tilde{\boldsymbol{x}}_{k+1} = A\tilde{\boldsymbol{x}}_k + B\boldsymbol{u}_k, \tag{4.11b}$$

$$F_x\tilde{\boldsymbol{x}}_k + CVaR_{\epsilon-\alpha}\big(F_x \boldsymbol{D}_k \bar{\boldsymbol{\delta}}_k\big) \leq g_x, \tag{4.11c}$$

$$\boldsymbol{u}_k \in \mathcal{U}, \quad \tilde{\boldsymbol{x}}_0 = \boldsymbol{x}(t), \tag{4.11d}$$

$$\text{(4.9b), (4.9c), (4.9d), } \forall k \in \{0, \dots N-1\}. \tag{4.11e}$$

then the solution is a feasible solution of (5.11).

Proof: We showed in Lemmas 1, 2 that satisfying (4.11c) also satisfies (5.11e). We further showed in Lemma 3 that the cost function can be reformulated as a minimization. Plugging this cost function into the original MPC gives us a min-min optimization problem that can be combined into a one-layer optimization given by (5.24). This is true because the feasible solution to the one-layer optimization (5.24) must be a feasible solution for the min-min problem and vice versa. Hence the optimal value of both optimizations must be equal. ■

4.4 Numerical Experiments

We compare our method, **DRMPC** given in (5.24) and **tight DRMPC (TDRMPC)** that uses constraint (4.8) in place of (4.11c), against chance constrained stochastic MPC (SMPC) methods that evaluate the chance constraint using mixed integer variables as seen in [18, 89] and a CVaR MPC (CMPC) approach inspired by [63, 123, 125]. In [63], the authors consider a CVaR-constrained MPC whereas in [123] the authors considered a CVaR cost. Similar to [125], although we don't consider dynamic risk, we consider CVaR cost and constraints in the MPC problem for the most consistent comparison to our method. The MPC optimizations considered for both these approaches are given below.

$$\begin{aligned} \min_{\boldsymbol{u}} \quad & \mathbb{E}\left(\sum_{k=0}^{N-1} c(\boldsymbol{x}_k, u_k) \right) && \textbf{(SMPC)} \\ \text{s.t.} \quad & \boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + B\boldsymbol{u}_k + D\boldsymbol{\delta}_k, \\ & \mathbb{P}(F_x\boldsymbol{x}_k - g_x > 0) \leq \epsilon, \\ & \boldsymbol{x}_0 = \boldsymbol{x}(t). \\ \min_{\boldsymbol{u}} \quad & \text{CVaR}_{1-\alpha}\left(\sum_{k=0}^{N-1} c(\boldsymbol{x}_k, \boldsymbol{u}_k) \right) && \textbf{(CMPC)} \\ \text{s.t.} \quad & \boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + B\boldsymbol{u}_k + D\boldsymbol{\delta}_k, \\ & \text{CVaR}_{\epsilon}(F_x\boldsymbol{x}_k - g_x) \leq 0, \\ & \boldsymbol{x}_0 = \boldsymbol{x}(t). \end{aligned}$$

To illustrate the effectiveness and the advantages of the proposed method, we compare it to chance constrained stochastic MPC and CVaR MPC. We perturb the probability mass function of the disturbance to demonstrate the proposed method's distributional robustness. We look at a simple two-dimensional discrete system $\boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + Bu_k + D\delta_k$, with

$$A = \begin{bmatrix} 1.0475 & -0.0463 \\ 0.0463 & 0.9690 \end{bmatrix}, \; B = D = \begin{bmatrix} 0.028 \\ -0.0195 \end{bmatrix}.$$

The control constraints are $-20 \leq u_k \leq 20$, the state constraints are $-\begin{bmatrix} 4 & 4 \end{bmatrix}^T \leq \boldsymbol{x}_k \leq \begin{bmatrix} 4 & 4 \end{bmatrix}^T$, and the disturbance lies in the set $\delta_k \in \{-1,\ 0,\ 1\}$ with probabilities $\boldsymbol{p} = \begin{bmatrix} 0.1 & 0.8 & 0.1 \end{bmatrix}$ respectively. We run 100 random simulations for each value of $\epsilon \in \{0.09,\ 0.2,\ 0.5,\ 0.9\}$ such that each simulation has 35 runs of the MPC optimization. The initial system state, $\boldsymbol{x}_0$, lies somewhere between $(3.1, 3.0)^T$ and $(4.1, 4.0)^T$. For each Monte-Carlo simulation, we randomly choose an initial condition in this range.

Discussion: The results are summarized in Table 4.1. We have two comparisons for each value of ϵ: the nominal case wherein we do not allow any perturbations to the original distribution $\boldsymbol{p}$ of the disturbances ($\alpha = 0$), and another allowing random variations in the distribution $\boldsymbol{p}$ with a total variation distance α. As seen in Table 4.1, for the 100 simulations, the percentage of violation of the constraints is consistently lower for the distributionally robust formulation. When $\alpha = 0$, the chance constraint

	ϵ	**0.09**		**0.2**		**0.5**		**0.9**	
	α	*0*	*0.05*	*0*	*0.15*	*0*	*0.4*	*0*	*0.8*
Violations	SMPC	3.08	3.17	11.9	12.2	14.6	15.9	23.4	24.1
	CMPC	0	0	2.71	3.03	2.97	4.69	2.97	9.28
	DRMPC	0	0	2.71	0	2.97	0	2.97	0
	TDRMPC	0	0	0.69	0	2.7	0	2.89	0
Cost	SMPC	1.12	1.13	1.19	1.19	1.22	1.21	1.31	1.31
	CMPC	1.1	1.11	1.13	1.12	1.19	1.18	1.21	1.23
	DRMPC	1.1	1.1	1.13	1.01	1.19	1.1	1.21	1.1
	TDRMPC	1.01	1.02	1.01	1.02	1.12	1.02	1.18	1.01

Table 4.1: Summary of results from Monte-Carlo simulations. The percentage of constraint violations and the average cost of each simulation ($\times 10^4$) are compared.

and cost for CMPC and DRMPC are equivalent and the results are the same for both. However, as soon as we allow for α perturbations in the distribution of the process noise, we see that DRMPC allows much fewer constraint violations and has a consistently lower cost. The TDRMPC is even more risk-averse than DRMPC due to further constraint tightening and we see the smallest percentage of constraint violations with behaviors that have consistently lower cost than SMPC and CMPC.

Figure 4.2 depicts one such Monte-Carlo simulation. The constraint $y_k \leq 4$, where y is the second component of our state $\boldsymbol{x}_k$, is violated by both the SMPC and CVaR MPC controllers while the DRMPC and tight DRMPC controllers do not violate constraints. However, this risk-averse behavior comes at the cost of slower convergence to the origin.

The average times (in seconds) for each MPC iteration are

$$\text{SMPC: } 0.32, \text{CMPC: } 0.47, \text{DRMPC: } 0.54, \text{TDRMPC: } 0.11$$

run using YALMIP [86] and a Gurobi solver [60] in MATLAB (on a 2.7 GHz Quad-Core Intel Core i7 processor). Thus, TDRMPC provides extra safety and reduced computational effort compared to SMPC and CMPC (with essentially no cost penalty in this example). These results motivate and justify our risk-based chance constraint formulation with the novel constraint tightening approximations.

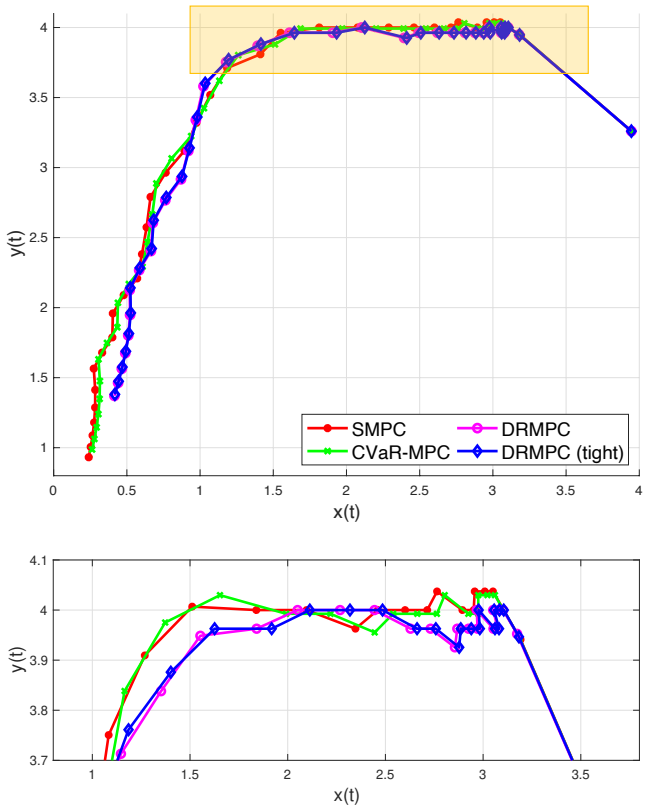


Figure 4.2: **Top:** Comparison of the four controllers through visualization of one of the 100 simulations ($\epsilon = 0.5, \alpha = 0.4$). **Bottom:** Yellow region of the top figure zoomed in for clarity on the behavior near the boundary of the state constraint set, $y_k \leq 4$.

Chapter 5

RISK-AWARE MPC WITH DYNAMIC OBSTACLES

The previous chapters investigated the planning problem in static but extreme environments in the presence of sensor and localization uncertainty. This chapter establishes a theoretical framework for risk-aware planning in the presence of moving obstacles. We study the problem of risk-averse receding horizon motion planning for agents with uncertain dynamics, in the presence of stochastic, dynamic obstacles. We propose a model predictive control (MPC) scheme that formulates the obstacle avoidance constraint using coherent risk measures. To handle disturbances, or process noise, in the state dynamics, the state constraints are tightened in a risk-aware manner to provide a disturbance feedback policy. We also propose a waypoint following algorithm that uses the proposed MPC scheme for discrete distributions and prove its risk-sensitive recursive feasibility while guaranteeing finite-time task completion. We further investigate some commonly used coherent risk metrics, namely, conditional value-at-risk (CVaR), entropic value-at-risk (EVaR), and g-entropic risk measures, and propose a tractable incorporation within MPC. We illustrate our framework via simulation studies.

This chapter was adapted from:

Anushri Dixit, M. Ahmadi, and J. W. Burdick. Risk-Sensitive Motion Planning using Entropic Value-at-Risk. In European Control Conference, 2021. URL `https://arxiv.org/abs/2011.11211`.

Anushri Dixit, Mohamadreza Ahmadi, and Joel W. Burdick. Risk-Averse Receding Horizon Motion Planning. 2022.

5.1 Introduction

Autonomous robots must increasingly plan motions in unstructured and uncertain environments with safety guarantees. Some applications where safe planning is required include autonomous traversal over extreme terrain in GPS-denied subterranean environments [23, 41], inspection of planetary environments [34], search and rescue missions caused by natural disasters [96, 118], and autonomous driving [113]. These applications present challenges at all the levels of planning and control [114]. The lowest control level requires a good physical model for accurate

motion prediction. To ensure robustness and safety, these models are often equipped with low-level controllers that leverage tools from robust control and invariant set theory [8]. At the intermediate level, algorithms must plan paths that are dynamically feasible, obstacle-free, and account for uncertainty in the motion dynamics, sensor measurements, and the environment. Several existing algorithms (model predictive control and A*-based graph planners to name a few) tackle some or all of these issues [19, 71]. At the highest level, robots must reason about their (uncertain) environment and decide on what tasks to do. Partially Observable Markov Decision Processes (POMDPs) are popular models for such sequential planning tasks [76]. Our work looks at the problem of obstacle avoidance using model predictive control (MPC) techniques.

This chapter provides a framework for risk-averse model predictive control with obstacle avoidance constraints. This work allows for a linear discrete-time system to be affected by both process noise as well as measurement noise in the sensing of obstacle position and orientation. The control input is parameterized as a disturbance feedback policy. Additionally, the MPC scheme in this work allows for a general class of coherent risk measures. Coherent risk measures can be expressed as a distributionally-robust expectation, i.e, the risk is equivalently expressed as the worst-case expectation over a convex, closed set of distributions, as discussed in the Preliminaries. We use this property of distributional robustness extensively throughout this chapter. We reformulate the risk-aware MPC with obstacle avoidance constraints as a convex, mixed-integer program.

Notation: We denote by $\mathbb{R}^n$ the n-dimensional Euclidean space, $\mathbb{R}_{\geq 0}$ the non-negative reals, and $\mathbb{Z}_{\geq 0}$ the set of non-negative integers. The index set, $\{k, k+1, \ldots, k+N\}$ is denoted by $\mathbb{Z}_k^{k+N}$. Throughout the chapter, a bold font denotes a vector and $(\cdot)^\top$ is its transpose, *e.g.,* $\boldsymbol{a} = (a_1, \ldots, a_n)^\top$, with $n \in \{1, 2, \ldots\}$. For vector $\boldsymbol{a}$, we use $\boldsymbol{a} \geq (\leq) \mathbf{0}$ to denote element-wise non-negativity (non-positivity), $\boldsymbol{a} \equiv \mathbf{0}$ to show all elements of $\boldsymbol{a}$ are zero, and $|\boldsymbol{a}|$ to denote the element-wise absolute value of $\boldsymbol{a}$. For vectors $a, b \in \mathbb{R}^n$, we denote their inner product by $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$, *i.e.,* $\langle \boldsymbol{a}, \boldsymbol{b} \rangle = \boldsymbol{a}^\top \boldsymbol{b}$. For a finite set $\mathcal{A}$, denote its power set $2^{\mathcal{A}}$. For a probability space $(\Omega, \mathcal{F}, \mathbb{P})$ and a constant $p \in [1, \infty)$, $\mathcal{L}_p(\Omega, \mathcal{F}, \mathbb{P})$ denotes the vector space of real valued random variables X for which $\mathbb{E}|X|^p < \infty$. For probability density functions $P(X)$ and $Q(X)$, $P \ll Q$ implies that P is absolutely continuous with respect to Q, *i.e.,* if $Q(X) = 0$, then $P(X) = 0$.

We follow the following convention for indices that appear most commonly as

sub/superscripts in this chapter:

- k indicates the state at time $t+k$, $\boldsymbol{x}_k = \boldsymbol{x}(t+k|t)$,
- j denotes the values associated with the j^{th} possible occurrence of the random variable (from the sample space),
- l, i represent the l^{th} obstacle and the i^{th} edge of the obstacle respectively.

5.2 Problem Statement

We consider a class of discrete-time dynamical systems given by

$$\begin{aligned} \boldsymbol{x}(t+1) &= A\boldsymbol{x}(t) + B\boldsymbol{u}(t) + D\boldsymbol{\delta}(t), \\ \boldsymbol{y}(t) &= C\boldsymbol{x}(t), \end{aligned} \tag{5.1}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^{n_x}$, $\boldsymbol{y}(t) \in \mathbb{R}^{n_y}$, and $\boldsymbol{u}(t) \in \mathbb{R}^{n_u}$ are the system state, output, and controls at time t, respectively. The system is affected by a stochastic, additive, process noise $\boldsymbol{\delta}(t) \in \mathbb{R}^{n_x}$. In fact, the noise term $\boldsymbol{\delta}$ can represent exogenous disturbances or unmodeled dynamics (see the case study in [5] for such modeling method applied to bipedal robots). We posit the following assumption about the process noise.

Assumption 2 (Discrete process noise) *The process noise $\boldsymbol{\delta}$ is assumed to consist of i.i.d. samples of a discrete distribution given by the probability mass function (pmf), $p_\delta = [p_\delta(1), p_\delta(2), \ldots, p_\delta(J_\delta)]^T$. For this distribution, we also define the index set $\mathcal{D} = \mathbb{Z}_1^{J_\delta}$.*

We also consider L *moving* obstacles with index $l \in \mathbb{Z}_1^L$ that can be approximated by a convex polytope defined by m_l half-spaces in $\mathbb{R}^{n_x}$

$$\bar{O}_l(t) = \{\boldsymbol{o} \in \mathbb{R}^{n_x} \mid \boldsymbol{c}_{i,l}^T(\boldsymbol{o} - \boldsymbol{a}_l(t)) \leq \boldsymbol{d}_{i,l},\ \forall i \in \mathbb{Z}_1^{m_l}\}. \tag{5.2}$$

We allow each polytopic obstacle $\bar{O}_l$, $l \in \mathbb{Z}_1^L$, centered at $\boldsymbol{a}_l$ at time t to move randomly w.r.t. the nominal trajectory. The random set defining obstacle $\bar{O}_l$, $l \in \mathbb{Z}_1^L$, at time t can be written as a random rotation R_l and random translation w_l of $\bar{O}_l$. Hence, we can rewrite the obstacle at time t as a random set, O_l, as

$$\begin{aligned} O_l(t) &= R_l(t)\bar{O}_l(t) + \boldsymbol{w}_l(t) \\ &= \left\{\boldsymbol{o}' = R_l(t)(\boldsymbol{o} - \boldsymbol{a}_l(t)) + \boldsymbol{a}_l(t) + \boldsymbol{w}_l(t) \mid \boldsymbol{c}_{i,l}^T\boldsymbol{o} \leq \boldsymbol{d}_{i,l},\ \forall i \in \mathbb{Z}_1^{m_l}\right\} \\ &= \left\{\boldsymbol{o}' \mid \boldsymbol{c}_{i,l}^T\Big(R_l(t)^{-1}\big(\boldsymbol{o}' - \boldsymbol{a}_l(t) - \boldsymbol{w}_l(t)\big) + \boldsymbol{a}_l(t)\Big) \leq \boldsymbol{d}_{i,l},\ \forall i \in \mathbb{Z}_1^{m_l}\right\}. \end{aligned} \tag{5.3}$$

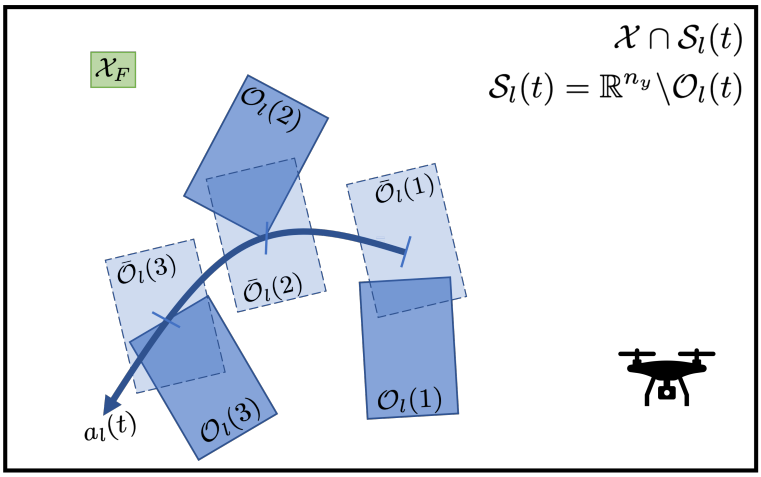


Figure 5.1: A representation of the environment for the motion planning problem. The light blue polytopes, $\bar{O}(t)$, represent the nominal obstacle set centered at the nominal trajectory $\boldsymbol{a}_l(t)$. We allow for random rotations and translations about this trajectory. This random obstacle set is given by the darker blue polytope $O(t)$. The safe set, $\mathcal{S}_l(t)$, is the region outside the obstacle set. The goal of the drone in the figure is to plan a path to the terminal set $\mathcal{X}_F$.

In other words, we allow the l^{th} obstacle moving with the nominal trajectory $\boldsymbol{a}_l(t)$ to rotate and translate randomly. The random movement of the obstacle about the nominal trajectory is described by the set $O_l(t)$ in the above equations.

Assumption 3 (Discrete measurement noise) *The random rotations and translations are sampled from a joint probability distribution whose sample space has cardinality J_o, i.e., $\Omega_l = \{(R_l^1, \boldsymbol{w}_l^1), \ldots, (R_l^J, \boldsymbol{w}_l^{J_o})\}$. A random rotation and translation is picked from this set with pmf given by $p_l = [p_l(1), p_l(2), \ldots, p_l(J_o)]^T$. For this distribution, we also define the index set $\mathcal{J} = \{1, \ldots, J_o\}$.*

The nominal safe set is defined as the region outside of the polytopic obstacles

$$\begin{aligned} \bar{\mathcal{S}}_l(t) &= \mathbb{R}^{n_y} \backslash \bar{\mathcal{O}}_l(t) \\ &= \left\{ \boldsymbol{o} \,|\, \exists i \in \mathbb{Z}_1^{m_l}, \;\; c_{i,l}^T(\boldsymbol{o} - \boldsymbol{a}_l(t)) \geq d_{i,l} \right\}. \end{aligned} \tag{5.4}$$

Similarly, the random safe set is given by $\mathcal{S}_l(t) = \mathbb{R}^{n_y} \backslash O_l(t)$. For obstacle avoidance, we are interested in the minimum distance to the safe set,

$$\zeta(\boldsymbol{y}(t), \mathcal{S}_l(t)) = \text{dist}(\boldsymbol{y}(t), \mathcal{S}_l(t)) := \min_{\boldsymbol{z} \in \mathcal{S}_l(t)} ||\boldsymbol{y}(t) - \boldsymbol{z}||. \tag{5.5}$$

Our goal is to bound the risk of collision with the randomly moving obstacles by evaluating the distance from the probabilistic safe set (which is the complement of

the obstacle set) and constraining it to be below a threshold, ϵ_l,

$$\rho_{1-\alpha}\left[\zeta(\boldsymbol{y}(t), \mathcal{S}_l(t))\right] \leq \epsilon_l, \quad \forall l \in \mathbb{Z}_1^L. \tag{5.6}$$

The obstacle avoidance constraint (5.6) is a risk safety constraint with confidence level α and risk tolerance (also referred to as risk threshold) ϵ_l for each obstacle $l \in \mathbb{Z}_1^L$. Note that this implies that we allow the coherent risk of the distance from the safe set to be at most ϵ_l in $1-\alpha$ worst realizations. Henceforth, we represent $\rho_{1-\alpha}$ as ρ for simplicity.

Let the state constraints take the form $\mathcal{X} := \{x \in \mathbb{R}^{n_x} | F_x x \leq g_x\}, F_x \in \mathbb{R}^{r \times n_x}, g_x \in \mathbb{R}^r$, which can represent physical constraints on a robot. Given that the system is subject to noise δ, we want to satisfy the following state constraints in the risk-averse sense:

$$\rho(f_{x,n}^T x(t+k|t) - g_{x,n}) \leq \epsilon_x, \quad \forall k \in \mathbb{Z}_1^N, n \in \mathbb{Z}_1^r, \tag{5.7}$$

where $F_x = \begin{bmatrix} f_{x,1}^T & f_{x,2}^T & \dots & f_{x,r}^T \end{bmatrix}^T, g_x = \begin{bmatrix} g_{x,1}^T & g_{x,2}^T & \dots & g_{x,r}^T \end{bmatrix}^T$. We write this constraint in shorthand as, $\rho(\boldsymbol{x}_k \notin \mathcal{X}) \leq \epsilon_x$.

Similarly, we consider control constraints of the form $\mathcal{U} := \{\boldsymbol{u} \in \mathbb{R}^{n_u} | F_u \boldsymbol{u} \leq g_u\}, F_u \in \mathbb{R}^{s \times n_u}, g_u \in \mathbb{R}^s$, for example, representing actuator limitations, and we want to satisfy the following risk constraint

$$\rho(f_{u,n}^T u(t+k|t) - g_{u,n}) \leq \epsilon_u, \quad \forall k \in \mathbb{Z}_0^{N-1}, n \in \mathbb{Z}_1^s, \tag{5.8}$$

where $F_u = \begin{bmatrix} f_{u,1}^T & f_{u,2}^T & \dots & f_{u,s}^T \end{bmatrix}^T, g_u = \begin{bmatrix} g_{u,1}^T & g_{u,2}^T & \dots & g_{u,s}^T \end{bmatrix}^T$. We write this constraint in shorthand as, $\rho(\boldsymbol{u}_k \notin \mathcal{U}) \leq \epsilon_u$.

Note that the uncertainty in the control input $\boldsymbol{u}$ arises from the description of the control policy as a function of the disturbances. This disturbance feedback policy will be elaborated upon shortly. If we choose to have hard constraints on the control input, the risk level can be set to a conservative value, $\alpha \to 1$, for the control constraints. For ease of presentation, we keep the risk level constant across all the constraints.

Lastly, we also consider terminal constraints of the form $\mathcal{X}_F := \{\boldsymbol{x} \in \mathbb{R}^{n_x} | F_f \boldsymbol{x} \leq g_f\}, F_f \in \mathbb{R}^{v \times n_x}, g_f \in \mathbb{R}^v$ and we want to satisfy,

$$\rho(f_{f,n}^T x(t+N|t) - g_{f,n}) \leq \epsilon_f, \quad \forall n \in \mathbb{Z}_1^v \tag{5.9}$$

where, $F_f = \begin{bmatrix} f_{f,1}^T & f_{f,2}^T & \dots & f_{f,v}^T \end{bmatrix}^T, g_f = \begin{bmatrix} g_{f,1}^T & g_{f,2}^T & \dots & g_{f,v}^T \end{bmatrix}^T$. We write this constraint in shorthand as, $\rho(\boldsymbol{x}_N \notin \mathcal{X}_F) \leq \epsilon_f$.

Remark 3 *The total number of risk constraints are $L + r + s + v$ for L obstacles, r state constraints, s control constraints, and v terminal constraints. With some abuse of notation, we write the risk $\rho_{1-\alpha}$ to mean the adjusted risk level $1 - \alpha'$ such that,*

$$1 - \alpha' = \frac{(1 - \alpha)}{L + 1}$$

where α is the risk confidence level of the entire system and α' is the adjusted risk level for the risk constraints and cost to attain the true confidence α.

Assumption 4 *We assume that the measures of risk (used for safety, state, and control constraints and the cost function) are coherent risk measures that can be represented in their dual form as:*

$$\rho(X) := \sup_{Q \in \boldsymbol{Q}} E_Q(X),$$

where, $\boldsymbol{Q}$ is a convex, closed set that we represent as $\boldsymbol{Q} = \left\{\boldsymbol{g}(q, \alpha) \leq 0, \sum_{j=1}^{J} p(j)q(j) = 1,\ q(j) \geq 0,\ \forall j \in \mathcal{J}\right\}$, and $\boldsymbol{g}(q, \alpha)$ is a convex function in q. We assume that if $\boldsymbol{g}(q, \alpha)$ is of dimension > 1, all its elements constitute a single function applied to all the components of q separately. We also assume that $\alpha \to 0$ corresponds to the risk-neutral case with $\rho(X) \to \mathbb{E}(X)$ and $\alpha \to 1$ corresponds to worst case (robust) with $\rho(X) \to \max \delta$.

In this work, we parameterize the control policy as an affine function of the process noise, i.e., we solve for a disturbance feedback policy. In [147], the authors reduced the number of decision variables during the computation of the affine disturbance feedback policy and showed that for a linear, time-invariant system, this policy is equivalent to using state feedback policies. We apply this simplified affine disturbance feedback (SADF) as,

$$u_k = \sum_{m=0}^{k-1} K_{k-m}\delta_m + \eta_k, \tag{5.10a}$$

$$\implies \boldsymbol{u}_N = \boldsymbol{K}_N\boldsymbol{\delta}_N + \boldsymbol{\eta}_N, \tag{5.10b}$$

where, u_i is an affine function of the disturbances, δ, from time t to $t+i$, K_{k-m}, η_k are

the decision variables in the MPC optimization, and for a N step (horizon) problem,

$$\begin{aligned}
\boldsymbol{K}_N &= \begin{bmatrix} 0 & \dots & \dots & 0 & 0 \\ K_1 & 0 & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & 0 & 0 \\ K_{N-1} & K_{N-2} & \dots & K_1 & 0 \end{bmatrix} \\
\boldsymbol{u}_N &= \begin{bmatrix} u_0 & u_1 & \dots & u_{N-1} \end{bmatrix}^T \\
\boldsymbol{\delta}_N &= \begin{bmatrix} \delta_0 & \delta_1 & \dots & \delta_{N-1} \end{bmatrix}^T \\
\boldsymbol{\eta}_N &= \begin{bmatrix} \eta_0 & \eta_1 & \dots & \eta_{N-1} \end{bmatrix}^T.
\end{aligned}$$

We are now ready to present the problem we are interested in solving in this chapter.

Problem 2 *Consider the discrete-time dynamical system (5.1) and the randomly moving obstacles O_l, $l \in \mathbb{Z}_1^L$, as defined in* (5.2) *and* (5.3)*. Given an initial condition $x_0 \in \mathbb{R}^{n_x}$, a goal set $\mathcal{X}_f \subset \mathbb{R}^{n_x}$, state constraints $\mathcal{X} \subset \mathbb{R}^{n_x}$, control constraints $\mathcal{U} \subset \mathbb{R}^{n_u}$, an immediate convex cost function $r : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}_{\geq 0}$, a horizon $N \in \mathbb{N}_{\geq 0}$, and risk tolerances ϵ_l, $\epsilon_x, \epsilon_u, \epsilon_F$, for for obstacle, state, control and terminal constraints respectively, compute the receding horizon controller $\{u_k\}_{k=0}^{N-1}$ such that $x(t+N) \in \mathcal{X}_f$ while satisfying the risk-sensitive safety constraints* (5.6)*, i.e., the solution to the following problem*

$$\min_{\boldsymbol{K}_N, \boldsymbol{\eta}_N} \quad J(x(t), \boldsymbol{u}) := \rho_{1-\alpha}\Big(\sum_{k=0}^{N-1} (r(\boldsymbol{x}_k, \boldsymbol{u}_k) \Big) \tag{5.11a}$$

$$\text{s.t.} \quad \boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + B\boldsymbol{u}_k + D\delta_k, \tag{5.11b}$$

$$\boldsymbol{y}_k = C\boldsymbol{x}_k, \tag{5.11c}$$

$$u_k = \sum_{m=0}^{k-1} K_{k-m}\delta_m + \eta_i, \tag{5.11d}$$

$$\rho_{1-\alpha}(\boldsymbol{x}_{k+1} \notin \mathcal{X}) \leq \epsilon_x, \tag{5.11e}$$

$$\rho_{1-\alpha}(\boldsymbol{u}_k \notin \mathcal{U}) \leq \epsilon_u, \tag{5.11f}$$

$$\rho_{1-\alpha}\big(\zeta(\boldsymbol{y}_k, \mathcal{S}_l(t+k))\big) \leq \epsilon_l, \forall l \in \mathbb{Z}_1^L, \tag{5.11g}$$

$$\rho_{1-\alpha}(x_N \notin \mathcal{X}_F) \leq \epsilon_f, \tag{5.11h}$$

$$\boldsymbol{x}_0 = \boldsymbol{x}(t). \tag{5.11i}$$

Note that although the obstacles O_l are assumed to be convex polytopes (5.2), the safe set $\mathcal{S}_l(t+k)$ in (5.4) is nonconvex in $\boldsymbol{y}_k$. Hence, the minimum distance to

$\mathcal{S}_l(t+k)$, $\zeta(\boldsymbol{y}_k, \mathcal{S}_l(t+k))$, is also nonconvex in $\boldsymbol{y}_k$. Therefore, the risk-sensitive safety constraint (5.11g) is a nonconvex constraint in the decision variable u, which renders optimization problem (5.11) nonconvex as well.

The next section will *reformulate* the state, control, and safety constraints (5.11e), (5.11f), (5.11g) in order to obtain a convex mixed-integer relaxation of (5.11), which yields locally optimal solutions. Nonetheless, every such locally optimal solutions satisfies the constraints of optimization (5.11) including the risk-sensitive safety constraint (5.11g).

5.3 Risk-Constrained Receding Horizon Planning

This section breaks down the receding horizon control problem into several parts. First, we modify the state and control constraints by finding efficient approximations that rigorously satisfy the risk constraints. Next, we specifically look at a tractable reformulation of the risk-obstacle avoidance constraint. Note that the risk-averse state and control constraint tightening can be computed offline because it depends only on the risk from the process noise. On the other hand, the risk-averse obstacle avoidance constraint depends on the distance of the system from the obstacle, which is constantly varying and hence needs to be computed online. We then rewrite the non-convex safe set as a set of mixed-integer constraints. We reformulate the terminal constraint by adding discrete states such that we can reach the goal in finite-time. Finally, we provide an efficient, tractable reformulation of the risk cost function. Note that the proofs of the lemmas and propositions are provided in the Appendix.

State and Control Constraint Tightening

Lemma 5 (Tightened state constraint) *Assuming the control policy* (5.11d)*, a tightened set of state constraints,*

$$f_{x,n}^T\big(A^k x_0 + \boldsymbol{B}_k \boldsymbol{\eta}_k\big) + \|f_{x,n}^T\big(\boldsymbol{B}_k \boldsymbol{K}_k + \boldsymbol{D}_k\big)\|_1 \rho(|\delta|) \leq \epsilon_x + g_{x,n},\ \forall k \in \mathbb{Z}_1^N, n \in \mathbb{Z}_1^r \tag{5.12}$$

where, $\boldsymbol{B}_k = \begin{bmatrix} A^{k-1}B & A^{k-2}B & \dots & B \end{bmatrix}$*, and* $\boldsymbol{D}_k = \begin{bmatrix} A^{k-1}D & A^{k-2}D & \dots & D \end{bmatrix}$*, guarantees that* (5.11e) *holds.*

Proof: First, rewrite the risk state constraint (5.9) as,

$$\rho(f_{x,n}^T(A^k x_0 + \boldsymbol{B}_k \boldsymbol{u}_k + \boldsymbol{D}_k \boldsymbol{\delta}_k) - g_{x,n}) \leq \epsilon_x, \quad \forall k \in \mathbb{Z}_1^N, n \in \mathbb{Z}_1^r$$

where, $\boldsymbol{B}_k = \begin{bmatrix} A^{k-1}B & A^{k-2}B & \dots & B \end{bmatrix}$, and $\boldsymbol{D}_k = \begin{bmatrix} A^{k-1}D & A^{k-2}D & \dots & D \end{bmatrix}$. We replace $\boldsymbol{u}_k$ with the SADF control policy (5.11d),

$$\begin{aligned}
\epsilon_x &\geq \rho\big(f_{x,n}^T\big(A^k x_0 + \boldsymbol{B}_k(\boldsymbol{\eta}_k + \boldsymbol{K}_k\boldsymbol{\delta}_k) + \boldsymbol{D}_k\boldsymbol{\delta}_k\big) - g_{x,n}\big) \\
&= f_{x,n}^T\big(A^k x_0 + \boldsymbol{B}_k\boldsymbol{\eta}_k\big) + \rho\big(f_{x,n}^T(\boldsymbol{B}_k\boldsymbol{K}_k + \boldsymbol{D}_k)\boldsymbol{\delta}_k\big) - g_{x,n} \\
&= f_{x,n}^T\big(A^k x_0 + \boldsymbol{B}_k\boldsymbol{\eta}_k\big) + \rho\big(f_{x,n}^T \sum_{m=0}^{k-1} \big(A^{k-m-1}(BK_{k-m} + D)\big)\delta_m\big) - g_{x,n}
\end{aligned} \tag{5.13}$$

The second term on the right-hand side of the above inequality can be simplified by using the subadditivity, monotonicity, and positive homogeneity properties of coherent risk measures and then i.i.d. nature of the disturbances respectively,

$$\begin{aligned}
&\rho\left(f_{x,n}^T \sum_{m=0}^{k-1} \big(A^{k-m-1}(BK_{k-m} + D)\big)\delta_m\right) \\
&\leq \sum_{m=0}^{k-1} \rho\left(f_{x,n}^T\big(A^{k-m-1}(BK_{k-m} + D)\big)\delta_m\right) && \text{(Subadditivity)} \\
&\leq \sum_{m=0}^{k-1} \rho\left(|f_{x,n}^T\big(A^{k-m-1}(BK_{k-m} + D)\big)|\,|\delta_m|\right) && \text{(Monotonicity)} \\
&\leq \sum_{m=0}^{k-1} |f_{x,n}^T\big(A^{k-m-1}(BK_{k-m} + D)\big)|\,\rho(|\delta_m|) && \text{(Positive Homogeneity)} \\
&\leq \|f_{x,n}^T\big(\boldsymbol{B}_k\boldsymbol{K}_k + \boldsymbol{D}_k\big)\|_1\rho(|\delta|). && \text{(i.i.d disturbances)}
\end{aligned}$$

Hence, satisfying the tightened constraint (5.12) implies satisfaction of the state constraint (5.9). ■

The above tightening is useful because $\rho(|\delta|)$ is independent of the disturbance feedback matrices. Hence, $\rho(|\delta|)$ can be computed offline.

Similarly, input constraints of the form (5.11f) are enforced as,

$$f_{u,n}^T\boldsymbol{\eta_i} + \|f_{u,n}^T\boldsymbol{K}_k\|_1\rho(|\delta|) \leq \epsilon_u + g_{u,n}, \quad k \in \mathbb{Z}_0^{N-1}, n \in \mathbb{Z}_1^s, \tag{5.14}$$

and terminal constraints of the form (5.11h) are enforced as,

$$f_{f,n}^T\big(A^N x_0 + \boldsymbol{B}_N\boldsymbol{\eta}_N\big) + \|f_{f,n}^T\big(\boldsymbol{B}_N\boldsymbol{K}_N + \boldsymbol{D}_N\big)\|_1\rho(|\delta|) \leq \epsilon_f + g_{f,n},\ \forall n \in \mathbb{Z}_1^v. \tag{5.15}$$

Now we reformulate the risk that arises from the moving obstacle (5.11g). This safety constraint is given by,

$$\rho\big[\zeta(\boldsymbol{y}_k, \mathcal{S}_l(t+k))\big] = \rho\left(\min_{z \in \mathcal{S}_l(t+k)} ||\boldsymbol{y}(t+k|t) - \boldsymbol{z}||\right) \leq \epsilon_l, \quad \forall l \in \mathbb{Z}_1^L \tag{5.16}$$

In (5.16), the safe set at time $t+k$, $\mathcal{S}_l(t+k)$, is a random variable that is a function of the discrete measurement noise and the output, $\boldsymbol{y}(t+k|k)$, is a random variable that is a function of the process noise $(\delta_0, \delta_1, \ldots, \delta_k)$. Hence the distance of the output $\boldsymbol{y}_k$ from the safe set is given by a random variable, $\zeta(\boldsymbol{y}_k, \mathcal{S}_l(t+k))$, that has a joint distribution of the measurement and process noise. This joint distribution has a sample space of cardinality $J = |\mathcal{D}|^k|\mathcal{J}| = (J_\delta)^k J_o$ and a pmf given by $\boldsymbol{p} = [p(1), p(2), \ldots, p(J)]^T$.

Lemma 6 (Safety Constraint Reformulation) *If Assumptions 3 and 4 hold, then the L.H.S. of constraint* (5.11g) *is equivalent to*

$$\begin{aligned} \min_{\boldsymbol{\lambda}_1, \lambda_2, \nu, h_{l,k}} \quad & \lambda_2 g^*\left(\lambda_2^{-1}\big(\boldsymbol{p}(h_{l,k}+\nu)+\boldsymbol{\lambda}_1\big)\right) - \nu \\ s.t. \quad & \boldsymbol{\lambda}_1 \geq 0,\ \lambda_2 \geq 0, \\ & \boldsymbol{y}_k + \frac{\boldsymbol{c}_{i,l}}{||\boldsymbol{c}_{i,l}||} h_{l,k}^j \in \mathcal{S}_l^j(t+k), \qquad \forall j \in \mathbb{Z}_1^J, \exists i \in \mathbb{Z}_1^{m_l}, \end{aligned} \tag{5.17}$$

where, $\boldsymbol{\lambda}_1 \in \mathbb{R}^J, \lambda_2, \nu, h_{l,k} \in \mathbb{R}$ *and* g^* *is the convex conjugate [17] of the convex function* g *that describes the risk envelope of a coherent risk measure.*

Proof: To find the distance of $\boldsymbol{y}_k$ from the safe set, $\zeta(\boldsymbol{y}_k, \mathcal{S}_l(t+k))$, we define a set of variables $h_{l,k}^j \geq 0$, $l \in \mathbb{Z}_1^L$ and $k = 0, \ldots, N-1$ satisfying

$$\boldsymbol{y}_k + \frac{\boldsymbol{c}_{i,l}}{||\boldsymbol{c}_{i,l}||} h_{l,k}^j = \boldsymbol{z} \tag{5.18}$$

$\forall j \in \mathbb{Z}_1^J, \forall k \in \mathbb{Z}_0^{N-1}$ and for some $i \in \mathbb{Z}_1^{m_l}$, which is the distance from each $\boldsymbol{y}_k$ to a point $\boldsymbol{z} \in \mathcal{X}$. If $\boldsymbol{z} \in \mathcal{S}_l^j(t+k)$, then minimizing $h_{l,k}$ defines the line segment connecting $\boldsymbol{y}_k$ and $\boldsymbol{z}$, which is the minimum distance to the set $\mathcal{S}_l^j(t+k)$. Therefore,

$$\begin{aligned} \zeta(\boldsymbol{y}_k, \mathcal{S}_l(t+k)) &= \min_{\boldsymbol{z} \in \mathcal{S}_l(t+k)} ||\boldsymbol{y}(t+k) - \boldsymbol{z}|| \\ &= \begin{cases} \min_{h_{l,k}} & h_{l,k} \\ \text{s.t.} & \boldsymbol{y}_k + \frac{\boldsymbol{c}_{i,l}}{||\boldsymbol{c}_{i,l}||} h_{l,k}^j \in \mathcal{S}_l^j(t+k), \end{cases} \end{aligned} \tag{5.19}$$

and we denote $h_{l,k}^*$ as the solution to (5.19).

Next, substitute the dual form of coherent risk measures (given by the representation theorem) from (2.5) into the L.H.S. of (5.11g). Then, replace the risk envelope $\mathcal{Q}$

with the convex representation given in Assumption 4. That is, $\rho(\zeta(\boldsymbol{y}_k, \mathcal{S}_l(t+k))) = \max_{Q\in\mathcal{Q}} \mathbb{E}_Q\left[\zeta(\boldsymbol{y}_k, \mathcal{S}_l(t+k))\right] = \max_{Q\in\mathcal{Q}} \mathbb{E}_Q\left[h^*_{l,k}\right]$, where (5.19) is used.

$$\begin{aligned} \max_{q(1),\ldots,q(J)} \quad & \mathbb{E}_Q\left[h^*_{l,k}\right] \\ \text{s.t.} \quad & g(q) \leq 0,\ -q(j) \leq 0, \quad \forall j \in \mathbb{Z}_1^J, \\ & \sum_{j\in\mathbb{Z}_1^J} p(j)q(j) = 1. \end{aligned} \tag{5.20}$$

The dual of this problem is given by,

$$\begin{aligned} \min_{\boldsymbol{\lambda}_1,\lambda_2,\nu} \max_{\boldsymbol{q}} \Big\{ & \sum_{j\in\mathbb{Z}_1^J} \left[q(j)p(j)h^*_{l,k} + \lambda_1^j q(j) + \nu p(j)q(j)\right] - \lambda_2 g(q) - \nu \Big\} \\ \text{s.t.} \quad & \boldsymbol{\lambda}_1 \geq 0,\ \lambda_2 \geq 0, \end{aligned} \tag{5.21}$$

where, $\boldsymbol{\lambda}_1 = \left[\lambda_1^1, \ldots, \lambda_1^J\right] \in \mathbb{R}^J$, λ_2 and $\nu \in \mathbb{R}$ are the dual variables. We conclude that (5.21) and (5.20) are equivalent because strong duality holds by Slater's condition [25]. Slater's condition is satisfied by showing *strict feasibility* for (5.20), i.e., there exists a feasible solution to (5.20) such that the inequality constraints hold with strict inequalities. One such solution occurs when $q(j) = 1$, $\forall j \in \mathbb{Z}_1^J$. We can find the maximum value of the Lagrangian in the objective of (5.21) when we know the exact form of the function g.

We can equivalently write the inner maximization of (5.21) in the form of the convex conjugate of g given by $g*$

$$\begin{aligned} \min_{\boldsymbol{\lambda}_1,\lambda_2,\nu} \quad & \lambda_2 g^*\left(\lambda_2^{-1}\left(\boldsymbol{p}(h^*_{l,k} + \nu) + \boldsymbol{\lambda}_1\right)\right) - \nu \\ \text{s.t.} \quad & \boldsymbol{\lambda}_1 \geq 0,\ \lambda_2 \geq 0. \end{aligned} \tag{5.22}$$

The above minimization is convex in the dual variables because the perspective operation preserves convexity [25]. We know from the conjugacy theorem ([17], Proposition 1.6.1) that properness of g implies properness of g^*. Finally, substituting minimization problem (5.19) for $h^*_{l,k}$ in optimization (5.22) gives (5.17). ■

Remark 4 *The term $\boldsymbol{p}(h^*_{l,k}+\nu)$ in* (5.22) *is just the expected value, i.e., $\boldsymbol{p}(h^*_{l,k}+\nu) = \mathbb{E}_P(h^*_{l,k} + \nu)$.*

Remark 5 *Note that for simplicity, we are assuming that $g(q) \in \mathbb{R}$. The above proof, however, can easily be extended to a vector-valued function $\boldsymbol{g}$ of dimension*

> 1 *under Assumption 4. In this case,* $\boldsymbol{g}$ *is a function* $\bar{g}$ *applied to each component of* q*, i.e.,*

$$\boldsymbol{g}(q) = \begin{bmatrix} \bar{g}(q(1)) & \bar{g}(q(2)) & \dots & \bar{g}(q(J)) \end{bmatrix}^T.$$

The equivalent safety constraint reformulation is then given by,

$$\begin{aligned} \min_{\boldsymbol{\lambda}_1, \boldsymbol{\lambda_2}, \nu, h_{l,k}} \quad & \sum_{j \in \mathbb{Z}_1^J} \left\{ \lambda_2^j \, \bar{g}^* \left((\lambda_2^j)^{-1} \left(p(j)(h_{l,k} + \nu) + \boldsymbol{\lambda}_1^j \right) \right) \right\} - \nu \\ s.t. \quad & \boldsymbol{\lambda}_1 \geq 0, \; \boldsymbol{\lambda}_2 \geq 0, \\ & \boldsymbol{y}_k + \frac{\boldsymbol{c}_{i,l}}{||\boldsymbol{c}_{i,l}||} h_{l,k}^j \in \mathcal{S}_l^j(t+k), \;\; \forall j \in \mathbb{Z}_1^J \end{aligned} \tag{5.23}$$

Similar results could be obtained for g-entropic risk measures using the Donsker-vardhan variational formula, as shown in [7].

The above reformulation applies to all coherent risk measures that satisfy Assumption 4. Next, we present this formulation for a few specific risk measures studied in our examples.

CVaR

For a CVaR constraint, the convex function, $g(q) = \begin{bmatrix} q(1) - \frac{1}{1-\alpha} & \dots & q(J) - \frac{1}{1-\alpha} \end{bmatrix}^T$ is separated into $\bar{g}(q(j)) = q(j) - \frac{1}{1-\alpha} \; \forall j \in \mathbb{Z}_1^J$. The convex conjugate $\bar{g}^*(q^*) = \frac{1}{1-\alpha}$ if $q^* = 1$ and $\bar{g}^*(q^*) = +\infty$ otherwise can be applied to (5.23) and simplified to get a linear program,

$$\begin{aligned} & \text{CVaR}_{1-\alpha}(\zeta(\boldsymbol{y}_k, \mathcal{S}_l(t+k))) = \text{CVaR}_{1-\alpha}(h_{l,k}^*) \\ = & \min_{\boldsymbol{\lambda}_1, \boldsymbol{\lambda_2}, \nu} \sum_{j=1}^{J} \left\{ \lambda_2^j \, \bar{g}^* \left((\lambda_2^j)^{-1} \left(p(j)(h_{l,k}^{j,*} + \nu) + \boldsymbol{\lambda}_1^j \right) \right) \right\} - \nu \\ & \text{s.t.} \quad \boldsymbol{\lambda}_1 \geq 0, \; \boldsymbol{\lambda}_2 \geq 0 \\ = & \min_{\boldsymbol{\lambda}_1, \boldsymbol{\lambda_2}, \nu} \sum_{j=1}^{J} \frac{\lambda_2^j}{1-\alpha} - \nu \\ & \text{s.t.} \quad \boldsymbol{\lambda}_1 \geq 0, \; \boldsymbol{\lambda}_2 \geq 0, \\ & \qquad p(j)(h_{l,k}^{j,*} + \nu) + \lambda_1^j = \lambda_2^j, \quad \forall j \in \mathbb{Z}_1^J. \end{aligned}$$

EVaR

For EVaR, the risk envelope constitutes $g(q) = \sum_{j\in\mathbb{Z}_1^J} p(j)q(j)\ln(q(j)) + \ln(1-\alpha)$ and,

$$g^*(q^*) = \sum_{j=1}^{J} p(j)\exp\left(\frac{q^*(j)-1}{p(j)}\right) - \ln(1-\alpha)$$

We substitute the convex conjugate into (5.22) to obtain the following exponential cone optimization,

$$\begin{aligned}
&\mathrm{EVaR}_{1-\alpha}(\zeta(\boldsymbol{y}_k, \mathcal{S}_l(t+k))) = \mathrm{EVaR}_{1-\alpha}(h_{l,k}^*) \\
&= \min_{\boldsymbol{\lambda}_1,\lambda_2,\nu} \lambda_2 g^*\left(\lambda_2^{-1}\left(\boldsymbol{p}(h_{l,k}^{j,*}+\nu) + \boldsymbol{\lambda}_1\right)\right) - \nu \\
&\quad \text{s.t.} \quad \boldsymbol{\lambda}_1 \geq 0,\ \lambda_2 \geq 0 \\
&= \min_{\boldsymbol{\lambda}_1,\lambda_2,\nu,\boldsymbol{s}} \lambda_2 \sum_{j=1}^{J} p(j)\exp\left(\frac{\lambda_2^{-1}(p(j)(h_{l,k}^{j,*}+\nu)+\lambda_1^j)}{p(j)}\right) - \lambda_2\ln(1-\alpha) - \nu \\
&\quad \text{s.t.} \quad \boldsymbol{\lambda}_1 \geq 0,\ \lambda_2 \geq 0 \\
&= \min_{\boldsymbol{\lambda}_1,\lambda_2,\nu} \lambda_2 \sum_{j=1}^{J} p(j)s(j) - \lambda_2\ln(1-\alpha) - \nu \\
&\quad \text{s.t.} \quad \boldsymbol{\lambda}_1 \geq 0,\ \lambda_2 \geq 0, \\
&\qquad \left(s(j),\ \lambda_2 p(j),\ p(j)(h_{l,k}^{j,*}+\nu)+\lambda_1^j\right) \in K_{exp},\ \forall j \in \mathbb{Z}_1^J.
\end{aligned}$$

Total variational distance ambiguity sets

The risk envelope for TVD is given by, $g(q) = \sum_{j\in\mathbb{Z}_1^J} |q(j)p(j) - p(j)| - 2\alpha$. When the conjugate $g^*(q^*) = \sum_{j\in\mathbb{Z}_1^J} q^*(j) + 2\alpha$ when $|q^*(j)/p(j)| \leq 1$ is substituted into (5.22), one obtains a linear program,

$$
\begin{aligned}
&\mathrm{TVD}_\alpha(\zeta(\mathcal{S}_l(t+k))) = \mathrm{TVD}_\alpha(h^*_{l,k}) \\
&= \min_{\boldsymbol{\lambda}_1,\lambda_2,\nu} \lambda_2 g^*\left(\lambda_2^{-1}\big(\boldsymbol{p}(h^{j,*}_{l,k}+\nu)+\boldsymbol{\lambda}_1\big)\right) - \nu \\
&\quad \text{s.t.} \quad \boldsymbol{\lambda}_1 \geq 0,\ \lambda_2 \geq 0 \\
&= \min_{\boldsymbol{\lambda}_1,\lambda_2,\nu} \sum_{j=1}^{J} \big(p(j)(h^{j,*}_{l,k}+\nu)+\lambda_1^j\big) + 2\lambda_2\alpha - \nu \\
&\quad \text{s.t.} \quad \boldsymbol{\lambda}_1 \geq 0,\ \lambda_2 \geq 0 \\
&\qquad |\lambda_2^{-1}\big(p(j)(h^*_{l,k}+\nu)+\lambda_1^j\big)| \leq p(j), \quad \forall j \in \mathbb{Z}_1^J \\
&= \min_{\boldsymbol{\lambda}_1,\lambda_2,\nu} \sum_{j=1}^{J} \big(p(j)(h^{j,*}_{l,k}+\nu)+\lambda_1^j\big) + 2\lambda_2\alpha - \nu \\
&\quad \text{s.t.} \quad \boldsymbol{\lambda}_1 \geq 0,\ \lambda_2 \geq 0, \\
&\qquad -\lambda_2 p(j) \leq p(j)(h^{j,*}_{l,k}+\nu)+\lambda_1^j \leq \lambda_2 p(j), \quad \forall j \in \mathbb{Z}_1^J.
\end{aligned}
$$

Now that we have seen how our general reformulation of the risk obtained in Lemma 6 can be applied to various examples of coherent risk measures, we are ready to present the MPC optimization that incorporates all the risk constraint reformulations we have obtained so far.

Theorem 7 *Consider the MPC optimization given by (5.11) with confidence level α and risk tolerances ϵ_l, $l \in \mathbb{Z}_1^L$. If Assumption 1 holds, then* (5.11) *is equivalent to a minimization over $\mathcal{V} = \{\boldsymbol{K}_N, \boldsymbol{\eta}_N, \boldsymbol{\lambda}_1, \lambda_2, \nu, h_{l,k}\}$ given by*

$$
\begin{aligned}
\min_{\mathcal{V}} \quad & J(\boldsymbol{x}(t), \boldsymbol{u}) := \rho\left(\sum_{k=0}^{N-1} r(\boldsymbol{x}_k, \boldsymbol{u}_k)\right) && \text{(5.24a)} \\
s.t. \quad & \lambda_2 g^*\left(\lambda_2^{-1}\big(\boldsymbol{p}(h_{l,k}+\nu)+\boldsymbol{\lambda}_1\big)\right) - \nu \leq \epsilon_l, && \text{(5.24b)} \\
& \boldsymbol{\lambda}_1 \leq 0, && \text{(5.24c)} \\
& \lambda_2 \geq 0 && \text{(5.24d)} \\
& \boldsymbol{y}_k + \frac{\boldsymbol{c}_{i,l}}{||\boldsymbol{c}_{i,l}||} h^j_{l,k} \in \mathcal{S}_l^j(t+k), \quad \forall j \in \mathbb{Z}_1^J && \text{(5.24e)} \\
& (5.11b), (5.11c), (5.11d), (5.11e), (5.11f), (5.11h), (5.11i)(5.11h). && \text{(5.24f)}
\end{aligned}
$$

Proof: We can substitute the result from Lemma 6 in (5.11) to get

$$\min_{\boldsymbol{K}_N, \boldsymbol{\eta}_N} \quad J(\boldsymbol{x}(t), \boldsymbol{u}) := \sum_{k=0}^{N-1} r(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{5.25a}$$

$$\text{s.t.} \quad (5.11b), (5.11c), (5.11d), (5.11e), (5.11f), (5.11h), (5.11i)(5.11h), \tag{5.25b}$$

$$(5.17) \leq \epsilon_l, \quad l \in \mathbb{Z}_1^L. \tag{5.25c}$$

Suppose we have an optimal solution to (5.25) given by $(\boldsymbol{K}_N^*, \boldsymbol{\eta}_N^*)$. As (5.25) is feasible, its constraints must be satisfied; this implies the inner minimization (5.17) must also be feasible, with solution $(\boldsymbol{\lambda}_1^*, {\lambda_2}^*, \nu^*, h_{l,k}^*)$. Hence, $(\boldsymbol{K}_N^*, \boldsymbol{\eta}_N^*, \boldsymbol{\lambda}_1^*, {\lambda_2}^*, \nu^*, h_{l,k}^*)$ must also be a feasible solution to (5.24) and yield the same objective value. Conversely, denote the optimal solution to (5.24) as $(\boldsymbol{K}_N^*, \boldsymbol{\eta}_N^*, \boldsymbol{\lambda}_1^*, {\lambda_2}^*, \nu^*, h_{l,k}^*)$. The pair $(\boldsymbol{K}_N^*, \boldsymbol{\eta}_N^*)$ must be feasible for (5.25) and gives the same objective value. Hence, the above optimization (5.25) is equivalent to the one-layer optimization (5.24). ■

We have now incorporated the results from Lemma 6 into the MPC formulation to get an equivalent formulation (5.24) of the original MPC problem given by (5.11). We can incorporate the results from Lemma 5 by replacing constraints (5.11e), (5.11f), (5.11h) with the tightened constraints (5.12), (5.14), (5.15) respectively. However, it remains to express the cost (5.24a) and obstacle avoidance constraint (5.24e) just in terms of the optimization variables $\boldsymbol{K}_N, \boldsymbol{\eta}_N$ instead of the dependence on $\boldsymbol{x}_k, \boldsymbol{u}_k$. We will reformulate the obstacle avoidance constraint in terms of the optimization variables and provide a mixed-integer reformulation of the nonconvex safe set in the next subsection.

Mixed-Integer Reformulation of the MPC optimization

This subsection frames the nonconvex safe set as a set of disjunctive inequalities. These inequalities are incorporated in our optimization by introducing a set of binary variables and using the Big-M reformulation [139]. The safe set (5.4) is defined as the region outside the obstacle l. Given that an obstacle can rotate and translate by $R_l(t+k)$ and $\boldsymbol{w}_l(t+k)$ from its nominal trajectory, we can write the safe set at $t+k$ as the region outside $O_l(t+k)$ described in (5.3). It can equivalently be expressed as a result of the rotation and translation of the nominal safe set itself

$$\begin{aligned} \mathcal{S}_l(t+k) &= \mathbb{R}^{n_y} \backslash O_l(t+k) \\ &= R_l(t+k)\bar{\mathcal{S}}_l(t+k) + \boldsymbol{w}_l(t+k). \end{aligned} \tag{5.26}$$

In (5.24e), $\mathcal{S}_l(t+k)$ is a nonconvex set. For some obstacle $l \in \mathbb{Z}_1^L$, (5.24e) can be rewritten as

$$R_l(t+k)^{-1}\left(\boldsymbol{y}_k + \frac{\boldsymbol{c}_{i,l}}{||\boldsymbol{c}_{i,l}||}h_{l,k} - \boldsymbol{w}_l(t+k)\right) \in \bar{\mathcal{S}}_l(t+k).$$

We know that $\boldsymbol{y}_k = C\boldsymbol{x}_k$ such that $\boldsymbol{x}_k = A^k x_0 + \boldsymbol{B}_k(\boldsymbol{\eta}_k + \boldsymbol{K}_k\boldsymbol{\delta}_k + \boldsymbol{D}_k\boldsymbol{\delta}_k)$, where $\boldsymbol{\delta}_k = \begin{bmatrix}\delta_1 & \delta_2 & \dots \delta_k\end{bmatrix}^T$ is the process noise, i.e.,

$$R_l(t+k)^{-1}\big(A^k x_0 + \boldsymbol{B}_k(\boldsymbol{\eta}_k + \boldsymbol{K}_k\boldsymbol{\delta}_k) + \boldsymbol{D}_k\delta_k + \frac{\boldsymbol{c}_{i,l}}{||\boldsymbol{c}_{i,l}||}h_{l,k} - \boldsymbol{w}_l(t+k)\big) \in \bar{\mathcal{S}}_l(t+k).$$

In the above equation, as described at the before Lemma 6, the safe set at time $t+k$, $\mathcal{S}_l(t+k)$ is a random variable that is a function of the discrete measurement noise and the output, $\boldsymbol{y}(t+k|k)$, that in turn is a random variable that is a function of the process noise $(\delta_0, \delta_1, \dots, \delta_k)$. Hence, distance of the output from the safe set, $h_{l,k}$, is a random variable that has a joint distribution of the measurement and process noise. This joint distribution has a sample space of cardinality $J = |\mathcal{D}|^k|\mathcal{J}| = (J_\delta)^k J_o$ and a pmf given by $\boldsymbol{p} = [p(1), p(2), \dots, p(J)]^T$.

$$R_l^j(t+k)^{-1}\big(A^k x_0 + \boldsymbol{B}_k\boldsymbol{\eta}_k + (\boldsymbol{B}_k\boldsymbol{K}_k + \boldsymbol{D}_k)\boldsymbol{\delta}_k^j + \frac{\boldsymbol{c}_{i,l}}{||\boldsymbol{c}_{i,l}||}h_{l,k}^j - \boldsymbol{w}_l^j(t+k)\big) \in \bar{\mathcal{S}}_l{}^j(t+k),$$

where $\{R_l^j(t+k),\ w_l^j(t+k),\ \boldsymbol{\delta}_k^j\}$ such that $j \in \mathbb{Z}_1^J$ is a realization of the measurement and process noise from its joint distribution. Given that the obstacles are convex polygons of the form (5.2), we write the safe region as the union of regions outside of the halfspaces that define an obstacle as follows

$$\bigvee_{i=1}^{m_l} \boldsymbol{c}_{i,l}^T\Bigg[R_l^j(t+k)^{-1}\big(A^k x_0 + \boldsymbol{B}_k\boldsymbol{\eta}_k + \big(\boldsymbol{B}_k\boldsymbol{K}_k + \boldsymbol{D}_k\big)\boldsymbol{\delta}_k^j + \frac{\boldsymbol{c}_{i,l}}{||\boldsymbol{c}_{i,l}||}h_{l,k}^j - \boldsymbol{w}_l^j(t+k) - \boldsymbol{a}_l(t+k)\big) + \boldsymbol{a}_l(t+k)\Bigg] \geq d_{i,l}. \quad (5.27)$$

Because the above disjunctive inequalities, however, are hard to enforce, we reformulate the constraint using a Big-M reformulation. The reformulation converts the disjunctive inequalities into a set of constraints described using binary variables, $\gamma_i^j \in \{0,1\}$ and a large positive constant M. The value of M depends on the bounds on $h_{l,k}^j$ (determined from the size of obstacle l) and $\boldsymbol{y}_k$ (dependent on the state and control inputs). It can be computed using linear programming. The Big-M reformulation of (5.27) is as follows

$$\boldsymbol{c}_{i,l}^T\Bigg[R_l^j(t+k)^{-1}\Bigg(A^k x_0 + \boldsymbol{B}_k\boldsymbol{\eta}_k + \big(\boldsymbol{B}_k\boldsymbol{K}_k + \boldsymbol{D}_k\big)\boldsymbol{\delta}_k^j + \frac{\boldsymbol{c}_{i,l}}{||\boldsymbol{c}_{i,l}||}h_{l,k}^j - \boldsymbol{w}_l^j(t+k) - \boldsymbol{a}_l(t+k)\Bigg) + \boldsymbol{a}_l(t+k)\Bigg] \geq d_{i,l} - M\gamma_i^j, \tag{5.28a}$$

$$\sum_{i=1}^{m_l}\gamma_i^j \leq m_l - 1, \qquad \forall i \in \mathbb{Z}_1^{m_l},\ j \in \mathbb{Z}_1^J. \tag{5.28b}$$

Inequalities (5.28) provide output constraints that satisfy the risk-sensitive obstacle avoidance constraint by taking into account measurement noise and process noise. However, the cardinality of the joint distribution that describes the distance from the obstacle, $h_{l,k}$, increases exponentially with the horizon, k. With exponentially increasing mixed-integer variables, the optimization soon becomes intractable.

To account for this, we introduce a new random variable, $\delta_{max,k}$, whose cumulative distribution function is defined as follows

$$\begin{aligned}\mathbb{P}(\delta_{max,k} \leq x) :=& \mathbb{P}(|\delta_0| \leq x\,,\ |\delta_2| \leq x\,,\ \ldots\,,\ |\delta_{k-1}| \leq x)\\ =& \mathbb{P}(|\delta_0| \leq x)\mathbb{P}(|\delta_2| \leq x)\ldots\mathbb{P}(|\delta_{k-1}| \leq x)\\ =& \mathbb{P}(|\delta_1| \leq x)^k.\end{aligned}$$

We can find a conservative (inner) approximation of (5.28a) using, $\delta_{max,k}$, as follows,

$$\begin{aligned}\boldsymbol{c}_{i,l}^T R_l(t+k)^{-1}\big(\boldsymbol{B}_k\boldsymbol{K}_k + \boldsymbol{D}_k\big)\boldsymbol{\delta}_k \geq& -\|\boldsymbol{c}_{i,l}^T R_l(t+k)^{-1}\big(\boldsymbol{B}_k\boldsymbol{K}_k + \boldsymbol{D}_k\big)\|_1\|\boldsymbol{\delta}_k\|_\infty\\ =& -\|\boldsymbol{c}_{i,l}^T R_l(t+k)^{-1}\big(\boldsymbol{B}_k\boldsymbol{K}_k + \boldsymbol{D}_k\big)\|_1\delta_{max,k}.\end{aligned}$$

Hence, we can rewrite (5.28a) as,

$$-\big\|\boldsymbol{c}_{i,l}^T R_l^j(t+k)^{-1}(\boldsymbol{B}_k\boldsymbol{K}_k + \boldsymbol{D}_k)\big\|_1\delta_{max,k}^j + \boldsymbol{c}_{i,l}^T\Bigg[R_l^j(t+k)^{-1}\Bigg(A^k x_0 + \boldsymbol{B}_k\boldsymbol{\eta}_k + \frac{\boldsymbol{c}_{i,l}}{||\boldsymbol{c}_{i,l}||}h_{l,k}^j - \boldsymbol{w}_l^j(t+k) - \boldsymbol{a}_l\Bigg) + \boldsymbol{a}_l\Bigg] \geq d_{i,l} - M\gamma_i^j \tag{5.29}$$

Notice that in the above inequality, we have, with some abuse of notation, reduced the cardinality of the joint distribution that describes, $h_{l,k}$, from $J = |\mathcal{D}|^k|\mathcal{J}| = (J_\delta)^k J_o$ to $J = |\mathcal{D}||\mathcal{J}| = J_\delta J_o$. This means that the number of constraints no longer increase exponentially with horizon length. Notice that the approximation is not conservative at the beginning of the horizon, i.e., when $k = 1$, the distribution described by $\delta_{max,k}$ is the same as δ_1.

Terminal constraints

In order to steer the system to the target region in finite time, we follow the suggestion of [112] and define a new discrete state $\psi \in \{0, 1\}$, such that $\psi = 0$ implies that the task has been completed at an earlier step or at the current step and $\psi = 1$ means that the task has not yet been completed at the current time. The update equation of ψ is then given by

$$\psi_{k+1} = \psi_k - \mu_k, \tag{5.30}$$

where $\mu_k \in \{0, 1\}$ is a discrete input.

Our goal to drive the system to a terminal set $\mathcal{X}_F$ is given by the tightened state constraint (5.15). Additionally, we incorporate the following constraints

$$f_{f,n}^T\big(A^k x_0 + \boldsymbol{B}_k \boldsymbol{\eta}_k\big) + \|f_{f,n}^T\big(\boldsymbol{B}_k \boldsymbol{K}_k + \boldsymbol{D}_k\big)\|_1 \rho(|\delta|) \leq \epsilon_f + g_{f,n} + \mathbb{1} M(1 - \mu_k), \tag{5.31}$$

$\forall k \in \mathbb{Z}_1^N, n \in \mathbb{Z}_1^v$, where $\mathbb{1} \in \mathbb{R}^{n_x}$ is a vector of 1's. Here, $\mu_k = 1$ if the task of reaching the goal, $\mathcal{X}_F = \{\boldsymbol{x} \in \mathbb{R}^{n_x} | F_f \boldsymbol{x} \leq g_f\}$ is completed at time step $t + k + 1$. Equation (5.30) implies that ψ jumps from $1 \rightarrow 0$, signaling completion of the task. After the task completion, all other MPC problem constraints can be relaxed by adding the term $M(1 - \psi_k)$ to them, i.e., any constraints of the form $C_1 \boldsymbol{s}_k + C_2 \gamma_i + C_3 \geq 0$ are modified to $C_1 \boldsymbol{s}_k + C_2 \gamma_i + C_3 + \mathbb{1} M(1 - \psi_k) \geq 0$, $\forall i, k$ where $\boldsymbol{s}_k = [\boldsymbol{K_k}, \boldsymbol{\eta_k}, \boldsymbol{\lambda_1}, \boldsymbol{\lambda_2}, \nu, h_{l,k}]$. We also add the following terminal constraint at time $t + N$ to ensure that the task is completed

$$\psi_N = 0. \tag{5.32}$$

Note that the discrete state ψ need not be a binary variable as long as we enforce the constraint,

$$0 \leq \psi_k \leq 1, \quad k = 1, 2, \ldots, N. \tag{5.33}$$

The MPC objective function is then modified as

$$\begin{aligned} \min_{\mathcal{V}} \quad J(\mathcal{V}) &:= \rho\Bigg(\sum_{k=0}^{N-1} \big(r(\boldsymbol{u}_k) + \psi_k\big) \Bigg) \\ &= \rho\Bigg(\sum_{k=0}^{N-1} r\Big(\sum_{m=0}^{k-1} K_{k-m} \delta_m + \eta_k \Big) \Bigg) + \sum_{k=0}^{N-1} \psi_k \end{aligned} \tag{5.34}$$

where $\mathcal{V} = \{\boldsymbol{K_N}, \boldsymbol{\eta_N}, \boldsymbol{\lambda_1}, \boldsymbol{\lambda_2}, \nu, h_{l,k}\}$ and $r(\boldsymbol{u}_k)$ is a convex function of the control input with $r(0) = 0$.

MPC Objective

For the MPC cost (5.34), consider the case, $r(\boldsymbol{u}) = \|R\boldsymbol{u}\|_1$, where $R \in \mathbb{R}^{n_u}$,

$$\begin{aligned} J(t) &= \rho\Big(\sum_{k=0}^{N-1} \|R\boldsymbol{u}_k\|_1 \Big) + \sum_{k=0}^{N-1} \psi_k, \\ &= \rho\Big(\sum_{k=0}^{N-1} \Big\| \sum_{m=0}^{k-1} RK_{k-m}\delta_m + R\eta_k \Big\|_1 \Big) + \sum_{k=0}^{N-1} \psi_k. \end{aligned} \tag{5.35}$$

This subsection introduces two methods to compute the control effort risk given by $\rho\big(\sum_{k=0}^{N-1} r(\boldsymbol{u_k}) \big)$. The first method provides an exact value of the risk and the second method provides an approximation. The first method loosely follows the steps taken to calculate the moving obstacle risk (see Lemma 6) and will be more computationally expensive because the control effort $r(\boldsymbol{u}_k)$ is a joint distribution of $(\delta_0, \ldots, \delta_{N-1})$ that grows with the horizon length N. The second method will utilize the constraint tightening tools used in Lemma 5 to approximate the value of the control effort risk. This approximation will be more computationally efficient. The examples in Section 5.4 contrast the two methods.

Exact computation of control effort risk

We define the control effort as a random variable, $Z := \sum_{k=0}^{N-1} \Big\| \sum_{m=0}^{k-1} K_{k-m}\delta_m + \eta_k \Big\|_1$. The sample space of Z consists of the joint probability distribution of $(\delta_0, \ldots, \delta_{N-1})$, which has cardinality $|\mathcal{D}|^N$. All the realizations of Z can be vectorized as $\boldsymbol{z} = \Big[z(1), z(2), \ldots, z(|\mathcal{D}|^N) \Big]$. Note that $z(j) = \sum_{k=0}^{N-1} \Big\| \sum_{m=0}^{k-1} K_{k-m}\delta_m^j + \eta_k \Big\|_1$, $\forall j \in \mathbb{Z}_1^{|\mathcal{D}|^N}$, where δ_m^j is a realization of δ_m from the joint pmf. If the pmf is denoted by $\boldsymbol{p}_\Delta \in \mathbb{R}^{|\mathcal{D}|^N}$, then,

$$\begin{aligned} &\rho\Big(\sum_{k=0}^{N-1} \Big\| \sum_{m=0}^{k-1} RK_{k-m}\delta_m + R\eta_k \Big\|_1 \Big) \\ &= \begin{cases} \max_{q(1),\ldots,q(|\mathcal{D}|^N)} & \mathbb{E}_Q\big[Z\big] \\ \quad \text{s.t.} & g(q) \le 0 \\ & -q(j) \le 0 \qquad \forall j \in \mathbb{Z}_1^{|\mathcal{D}|^N} \\ & \sum_{j=1}^{J} p_\Delta(j) q(j) = 1 \end{cases} \\ &= \begin{cases} \min_{\boldsymbol{\xi}_1, \xi_2, \vartheta} & \xi_2 g^*\Big(\xi_2^{-1}\big(\boldsymbol{p}_\Delta(\boldsymbol{z} + \vartheta) - \boldsymbol{\xi}_1 \big) \Big) - \vartheta \\ \quad \text{s.t.} & \boldsymbol{\xi}_1 \le 0 \\ & \xi_2 \ge 0 \end{cases}, \end{aligned} \tag{5.36}$$

where, $\boldsymbol{\xi}_1 \in \mathbb{R}^{|\mathcal{D}|^N}$, $\boldsymbol{\xi}_2, \vartheta \in \mathbb{R}$ are the dual variables, see Lemma 6 for details on how we find the dual function and obtain the conjugate in the above minimization. Note that the number of constraints grow exponentially with the horizon length.

Approximation of control effort risk

We can alternatively approximate the cost as,

$$\begin{aligned}
&\rho\Big(\sum_{k=0}^{N-1} \Big\| \sum_{m=0}^{k-1} RK_{k-m}\delta_m + R\eta_k \Big\|_1\Big) \\
&\leq \sum_{k=0}^{N-1} \Big(\rho\big(\|\sum_{m=0}^{k-1}(RK_{k-m}\delta_m)\|_1\big) + \|R\eta_k\|_1\Big) \qquad \text{(Subadditivity)} \qquad (5.37)\\
&\leq \sum_{k=0}^{N-1} \Big(\Big\| \sum_{m=0}^{k-1} RK_{k-m}\Big\|_1 \rho\big(|\delta|\big) + \|R\eta_k\|_1\Big) \qquad \text{(i.i.d disturbances)},
\end{aligned}$$

where we obtained the first inequality by using the subadditivity of norms and then the translational invariance property of coherent risk measures. The second inequality results from observing that all disturbances are i.i.d and can be replaced by δ. We use the homogeneity of norms and coherent risk measures to obtain the final result (similar to Lemma 5).

The above cost approximation eliminates the additional $|\mathcal{D}|^N$ constraints that result from (5.36). This approximation deprioritizes task completion; i.e., when we substitute this upper bound into (5.35), the term $\sum_{k=0}^{N-1} \psi_k$ has less weight compared to when we use the exact cost (5.36). Another approximation of the true cost would be using the random variable $\delta_{max,N}$ as seen in (5.29). In using an approximation of $(\delta_0, \ldots, \delta_{N-1})$ via $\delta_{max,N}$, we reduce the number of constraints from $|\mathcal{D}|^N$ to $|\mathcal{D}|$.

Properties of MPC

We now combine all the parts of the MPC into one optimization.

$$
\begin{aligned}
\min_{\mathcal{V}} \quad & \sum_{k=0}^{N-1} \left(\Big\| \sum_{m=0}^{k-1} RK_{k-m} \Big\|_1 \rho(|\delta|) + \|R\eta_k\|_1 \right) \\
& \sum_{j=1}^{J} \left\{ \lambda_2^j \, \bar{g}^* \left((\lambda_2^j)^{-1} \big(p(j)(h_{l,k}^j + \nu) + \boldsymbol{\lambda}_1^j \big) \right) \right\} - \nu \leq \epsilon_l + M_k \\
& \text{L.H.S (5.29)} \geq d_{i,l} - M\gamma_i^j - M_k \\
& F_x\big(A^k x_0 + \boldsymbol{B}_k \boldsymbol{\eta}_k\big) + \|F_x\big(\boldsymbol{B}_k \boldsymbol{K}_k + \boldsymbol{D}_k\big)\|_1 \rho(|\delta|) \leq \epsilon_x + g_x + M_k \\
& F_u \boldsymbol{\eta_k} + \|F_u \boldsymbol{K}_k\|_1 \rho(|\boldsymbol{\delta}|)\big) \leq \epsilon_u + g_u + M_k \\
& (5.30),\ (5.31),\ (5.32),\ (5.33).
\end{aligned}
\tag{5.38}
$$

where $M_k = M(1 - \psi_k)$ and $\mathcal{V} = \{\boldsymbol{K}_N, \boldsymbol{\eta}_N, \boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \nu, h_{l,k}\}$. The constraints must hold $\forall k \in \mathbb{Z}_1^{N-1}$, $l \in \mathbb{Z}_1^L$, $j \in \mathbb{Z}_1^J$, and $i \in \mathbb{Z}_1^{m_l}$. The solution to the deterministic MPC problem (5.38) is also a solution to (5.11). The convex, mixed-integer relaxation of a nonconvex optimization problem in (5.38) results in locally optimal solutions.

Proposition 1 (Risk-sensitive recursive feasibility) *If the optimization* (5.38) *is feasible at time t, then it is feasible at time* $t + 1$ *with confidence* α.

Proof: Assume that the feasible solution to (5.38) at time t is given by the control policy $\{(0, \eta_0^*), (K_1^*, \eta_1^*), \ldots, (K_{N-1}^*, \eta_{N-1}^*)\}$. At time t, we apply the control input $u_0 = \eta_0^*$. Since (5.12), (5.14) hold for all $\delta_0 \leq \rho(|\delta|)$, if $\delta_0 > \rho(|\delta|)$, the state and control constraints may not hold and the MPC optimization may no longer be feasible. Similarly, if the distance to the obstacle is greater than the risk of the distance, the MPC may no longer be feasible, i.e., if $h_{l,0} > \rho(h_{l,0})$.

Let us assume for simplicity that $\delta_0 \leq \rho(|\delta|)$, $h_{l,0} \leq \rho(h_{l,0})$ hold at time t. The optimization is feasible at time $t + 1$ if there exists a feasible input at time $t + N$ that does not violate constraints. Since $\psi_N^* = 0$ by virtue of the terminal constraint, all the constraints in the optimization are relaxed thereafter. Note that the state $\psi_N = 0$ is invariant due to (5.30) and (5.33) and that $\mu_k = 0$ at all time after the task has been completed. Therefore, once the optimization constraints are relaxed they will remain this way. A control input $(K_N, \eta_N) = (0, 0)$ ensures that the optimization is feasible. At time $t + 1$, a feasible solution to (5.38) is given by the control sequence $\{(0, K_1^* \delta_0 + \eta_1^*), \ldots, (K_{N-1}^*, \eta_{N-1}^*), (0, 0)\}$.

Finally, we aim to quantify the probability of the constraints at time $t+1$ no longer being satisfied by the control input $u_0 = \eta_0^*$.

$$
\begin{aligned}
&\mathbb{P}\{\text{MPC infeasible}\} \\
&\quad \leq \mathbb{P}\{\delta_0 > \rho(|\delta|) \cup h_{l,0} > \rho(h_{l,0})\} \\
&\quad \leq \mathbb{P}\{\delta_0 \geq \rho(|\delta|)\} + \mathbb{P}\{h_{l,0} \geq \rho(h_{l,0})\} \\
&\quad = \mathbb{P}\{\delta_0 - \mathbb{E}(|\delta|) \geq \rho(|\delta|) - \mathbb{E}(|\delta|)\} + \mathbb{P}\{h_{l,0} - \mathbb{E}(h_{l,0}) \geq \rho(h_{l,0}) - \mathbb{E}(h_{l,0})\} \\
&\quad \text{(Subtracting } \mathbb{E}(|\delta|),\ \mathbb{E}(h_{l,0}) \text{ from both sides)} \\
&\quad \leq \frac{\sigma_\delta^2}{\sigma_\delta^2 + \big(\rho(|\delta|) - \mathbb{E}(|\delta|)\big)^2} + \frac{\sigma_h^2}{\sigma_h^2 + \big(\rho(h_{l,0}) - \mathbb{E}(h_{l,0})\big)^2} \text{(Cantelli's inequality)}
\end{aligned}
$$

where, σ_δ^2, σ_h^2 are the variances of δ and $h_{l,0}$ respectively. Note that as α increases, the risk gets larger as a greater value of α corresponds to a more risk-averse setting. Hence, the upper bound on $\mathbb{P}\{\text{MPC infeasible}\}$ gets smaller.

We know that when $\alpha \to 0$ (risk-neutral), $\rho(|\delta|) \to \mathbb{E}(|\delta|)$, $\rho(h_{l,0}) \to \mathbb{E}(h_{l,0})$, and $\mathbb{P}\{\text{MPC infeasible}\} \leq 1$. Similarly, $\alpha \to 1$ (risk-averse), $\rho(|\delta|) \to \max|\delta|$, $\rho(h_{l,0}) \to \max h_{l,0}$, and $\mathbb{P}\{\text{MPC infeasible}\} \to 0$ (because $\mathbb{P}\{\delta_0 > \rho(|\delta|) \cup h_{l,0} > \rho(h_{l,0})\} \to 0$). Hence, (5.38) is feasible at time $t+1$ if it is feasible at time t with increasing probability as the confidence level α increases. ■

Remark 6 *Proposition 1 provides a loose bound on the infeasibility of the MPC. In the case of CVaR, TVD, and EVaR, we know that the risk measures are upper bounds for VaR (see Figure 1) and can hence provide tighter bounds on the likelihood of infeasibility. Recall from Remark 3 that the confidence level is adjusted to account for multiple risk constraints. Hence, we can quantify the bounds on probability of MPC infeasibility in terms of the confidence α,*

$$
\begin{aligned}
\mathbb{P}\{\textit{MPC infeasible at } t+1|t\} &\leq \mathbb{P}\{\delta_0 > \rho(|\delta|) \cup h_{l,0} > \rho(h_{l,0})\} \\
&\leq \mathbb{P}\{\delta_0 \geq \rho(|\delta|)\} + \sum_{l=1}^{L} \mathbb{P}\{h_{l,0} \geq \rho(h_{l,0})\} \\
&\leq \frac{1-\alpha}{L+1} + L\frac{1-\alpha}{L+1} \qquad \textit{(VaR probability bound)} \\
&\leq 1-\alpha.
\end{aligned}
$$

MPC is often used to plan local trajectories given a reference trajectory or a set of waypoints from a higher-level global planner like A* or RRT [62, 88]. Let

Algorithm 3 Follow waypoints

```
Number of waypoints visited, W = 0
while W < K do
  Initialize (x_0, ψ_0) = (w_W, 1)
  Set desired goal x_des = w_{W+1}
  while ψ_0 ≠ 0 do
    Solve (5.38) to obtain policy {(0, η*_0), ..., (K*_{N-1}, η*_{N-1})}
    Update x_0 = A x_0 + B η*_0 + D δ_0
    Update ψ_0 = ψ_0 − μ_0
    if x_0 = x_des then
      W = W + 1
    end if
  end while
end while
```

$\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_K\}$ be a given a sequence of waypoints. We call a waypoint $\boldsymbol{w}_{j+1}$ *N-step reachable* from $\boldsymbol{w}_j$, if there exists a feasible solution to (5.38) with $\boldsymbol{x}_0 = \boldsymbol{w}_j$ and $\boldsymbol{x}_K = \boldsymbol{w}_{j+1}$.[1]

Proposition 2 (Finite-time task completion) *Assuming that the waypoint $\boldsymbol{w}_{j+1}$ is N-step reachable from $\boldsymbol{w}_j$, $\forall j \in \mathbb{Z}_1^{K-1}$, Algorithm 3 gives a sequence of control inputs to move from $\boldsymbol{w}_0$ to $\boldsymbol{w}_K$ in finite time with confidence α.*

Proof: Consider the simple case of starting from $\boldsymbol{w}_0$ and reaching $\boldsymbol{w}_1$, i.e., when we have exactly two waypoints. We implement Algorithm 3 till $\psi_0 = 0$. Let J_t^* be the cost of the MPC optimization (5.38) at time t. The feasible solution to (5.38) at t is given by the input sequence $\{(0, \eta_0^*), (K_1^*, \eta_1^*), \ldots, (K_{N-1}^*, \eta_{N-1}^*)\}$ and the state sequence $\{\psi_0^*, \psi_1^*, \ldots, \psi_K^*\}$. At time $t+1$, with confidence α, the cost of the MPC optimization is $J_{t+1}^* \leq J_t^* - \|R\eta_0\|_1 - \psi_0^*$. This is true because we know from Proposition 1 that, with confidence α, at time $t+1$, $\{(0, K_1^*\delta_0 + \eta_1^*), \ldots, (K_{N-1}^*, \eta_{N-1}^*), (0,0)\}$ is a feasible control input with $\psi(t+K|t+1) = 0$, i.e., J_t^* will incur no additional cost from $(K_N, \eta_N|t+1) = (0,0)$ and $\psi(t+K+1|t+1) = 0$. Since $J_{t+1}^* - J_t^* \leq -\|R\eta_0\|_1 - \psi_0^*$, the cost decreases by at least 1 at each time step till the task is completed. Considering that J_t^* is always positive and finite, it will take a finite number of steps to reach $J_k^* = 0$, $k \geq t$. Hence, the policy to take the

[1] We assume that we obtain these waypoints from a higher-level planner like A* or RRT. Analyzing the N-step reachability of the waypoints is out of the scope of this thesis and we consider it an avenue of future work.

system from $\boldsymbol{w}_0$ to $\boldsymbol{w}_1$ will be implemented in finite time, with confidence α. By induction, the system will reach $\boldsymbol{w}_K$ from $\boldsymbol{w}_0$ in finite time with confidence α. ■

5.4 Numerical Results

To illustrate the effectiveness of this method, we present numerical experiments that were run on MATLAB using the YALMIP toolbox [86] with a Gurobi solver [60] (for CVaR and TVD) and a Mosek solver [11] (for EVaR) .

Simple 2D system

We first look at the two-dimensional discrete system $x_{k+1} = Ax_k + Bu_k + D\delta_k$ that is similar to the example we considered in [37], but with process noise.

$$A = \begin{bmatrix} 1.0475 & -0.0463 \\ 0.0463 & 0.9690 \end{bmatrix},\ B = \begin{bmatrix} 0.028 \\ -0.0195 \end{bmatrix},\ D = \begin{bmatrix} 0.028 \\ -0.0195 \end{bmatrix}.$$

The process noise can take values, $\delta_k \in \begin{bmatrix} -0.2 & -0.1 & 0 & 0.1 & 0.2 \end{bmatrix}$.
The control constraints are

$$-100 \leq u_k \leq 100.$$

One randomly moving obstacle interferes with the MPC solution path that would be found in the absence of any obstacles. The obstacle rotates either 0 or $\pi/4$ degrees. The obstacle can translate along the x-axis by $0, -0.25,$ or 0.25 m. The horizon length is $N = 8$.

We compare the disturbance feedback policies obtained by using three different risk measures - CVaR, EVaR, and TVD by comparing the total cost of reaching the goal, the percentage of infeasible optimizations, and the average computation time for each MPC iteration.

Cost Comparison: A fair comparison of using the exact cost versus the over approximation of the cost can be made only if the constraints of the MPC (5.38) remain the same. For 50 Monte Carlo simulations, we compare the MPC trajectories obtained when using CVaR risk. We compare the trajectory costs resulting from using a) the *exact cost* as computed in (5.36) and b) the *upper bound* of the exact cost (5.37), see Table 5.1. The average time taken for each MPC iteration is also provided. We see that the time taken for each MPC iteration is significantly higher when the exact CVaR cost is used. The control effort is also higher when using the exact cost. This is because the over-approximated cost always penalizes higher control effort more than task completion (control is parameterized as an affine

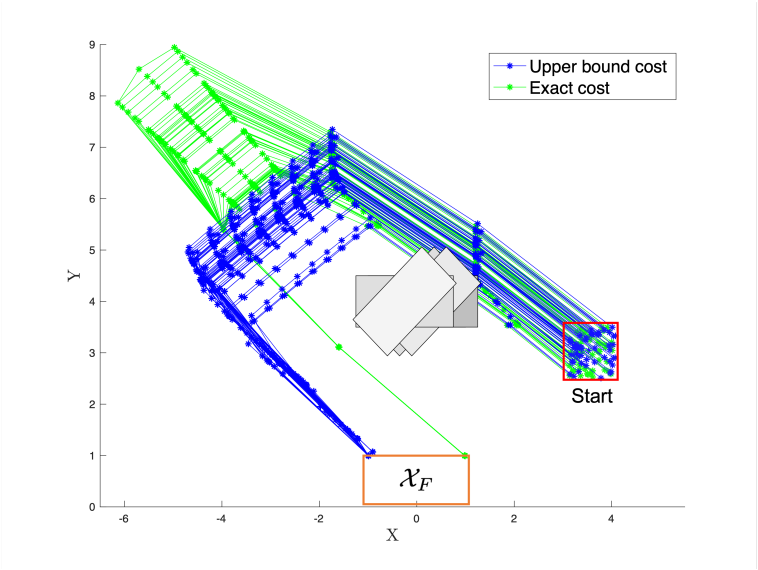


Figure 5.2: Comparison of the trajectories obtained using the exact cost (5.36) and the upper bound cost (5.37). The gray rectangles show possible obstacle configurations. The darker rectangle has a higher probability of occurrence and the lighter rectangle has a lower probability of occurrence.

function of the disturbance). Figure 5.2 shows us the qualitative difference between the 50 Monte Carlo simulations when $\alpha = 0.9$. We emphasize that task completion is prioritized when the exact CVaR cost is used.

α	Average cost ($\times 10^3$)	
	Exact	Upper bound
0.1	4.44	3.94
0.4	4.51	3.94
0.8	5.36	3.93
0.9	6.20	3.89
Time(s)	83.68	6.32

Table 5.1: Average trajectory cost for CVaR MPC using different costs

Feasibility comparison: For different risk levels, α, we also compare the number of times the MPC optimization is infeasible when we use CVaR, EVaR, and TVD risk measures, with the cost (5.37). The results for 50 Monte Carlo simulations are summarized in Table 5.2. It has been shown that VaR $\leq$ CVaR $\leq$ EVaR and CVaR $\leq$ TVD, see Section **??**. Proposition 1 provides us a loose bound for the probability of infeasibility of the MPC. Table 5.2 shows us that as α increases (increasing conservativeness), the percentage of infeasible optimizations decreases. Furthermore, the actual likelihood of infeasibility is much lower than the bounds

obtained in Proposition 1. Thus, the bounds of Proposition 1 are verified in this case, though the degree of tightness is unknown.

α	MPC infeasible (%)		
	CVaR	EVaR	TVD
0.1	5.3	1.64	0
0.4	5.9	1.09	0
0.8	6.4	0.18	0
0.9	2.7	0	0
Time (s)	6.32	42.33	3.61

Table 5.2: Results for infeasibility of risk-aware MPC

Quadcopter

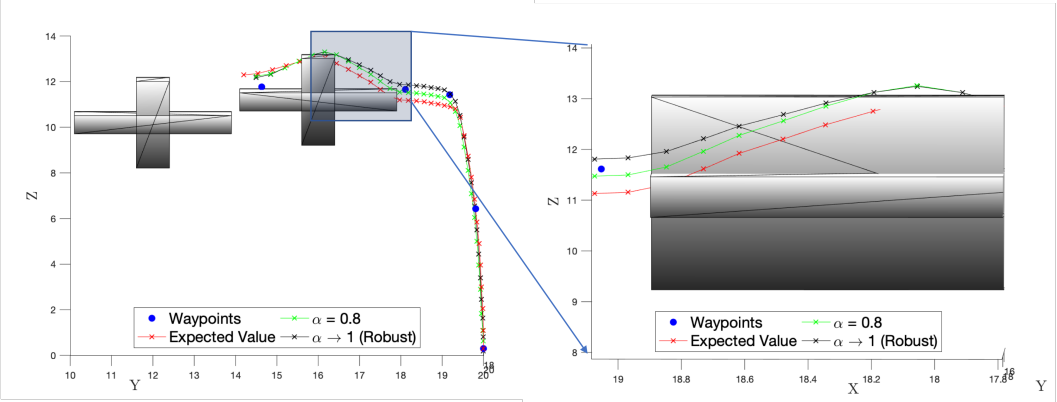


Figure 5.3: A comparison of the TVD MPC trajectories with the expectation-based MPC ($\alpha \to 0$) trajectory. On the right, the shaded blue region is zoomed in from a different perspective to illustrate the behavior near one of the random realizations of the obstacle.

We consider a quadcopter that must follow given waypoints while avoiding randomly moving obstacles and adhering to state and control constraints, Algorithm 3. The quadcopter is described by the states $(x, y, z, \phi, \theta, \varphi, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\varphi})^T$. The position of the quadcopter in 3D space is given by the coordinates x, y, z and the roll, pitch, and yaw are given by ϕ, θ, φ respectively. The model of the quadcopter is given by

$$\ddot{x} = -g\theta,\ \ddot{y} = g\theta,\ \ddot{z} = -\frac{u_1}{m} - g,$$

$$\ddot{\phi} = \frac{u_2}{I_{xx}},\ \ddot{\theta} = \frac{u_3}{I_{yy}},\ \ddot{\varphi} = \frac{u_4}{I_{zz}},$$

where m is the quadcopter's mass, g is the acceleration due to gravity, and I_{xx}, I_{yy}, I_{zz} are the quadcopter moments of inertia about the x, y, z-axes of the system. The

control inputs are given by u_1, u_2, u_3, u_4. We used the following parameters: $m = 0.65\text{kg}$, $l = 0.23\text{m}$, $I_{xx} = 0.0075\text{kg.m}^2$, $I_{yy} = 0.0075\text{kg.m}^2$, $I_{zz} = 0.0013\text{kg.m}^2$, $g = 9.81\text{m.s}^{-2}$ [37, 62].

The TVD risk constraint has two parameters: the confidence level, α, and the risk-threshold, ϵ. We chose $\alpha \in \{0.8, 0.99\}$, $\epsilon = 0.04$. The waypoints are given by regions of size $[-0.5, 0.5]^3$ around the waypoint center (denoted by o in Figure 5.3). We chose a horizon length of $N = 15$ for the MPC optimization. We considered the case of having one randomly translating and rotating obstacle. The obstacle is a rectangular box of size 2x1x4 m^3; it can rotate by $\frac{\pi}{2}$ and translate by 4m along the y-axis and 1m along the z-axis. Figure 5.3 shows all the different configurations of this obstacle at different times. We further allow for process noise $\delta \in \{-0.2, -0.1, 0, 0.1, 0.2\}$. The continuous-time system is discretized with a time-interval of 0.02 sec. As the risk-averseness of the system grows, the trajectory followed by the quadcopter maintains a greater distance from all possible configurations of the moving obstacle.

Chapter 6

DATA-DRIVEN PREDICTION AND RISK-AWARE MPC WITH UNKNOWN OBSTACLES

So far, this thesis has investigated risk-aware planning in a priori unknown static, extreme environments. The previous chapter also considered risk-aware MPC in the presence of moving obstacles but with the completely known uncertainty descriptions to compute risk-aware feedback policies with recursive feasibility and task completion guarantees. This chapter answers two questions: 1) how does one predict the motion of a priori unknown moving obstacles and also quantify the uncertainty in the prediction? 2) Given this prediction model with uncertainty, how can one obtain a risk-aware obstacle avoidance maneuver? Given noisy measurements of the a priori unknown obstacle trajectory, a bootstrapping technique predicts a set of obstacle trajectories. The bootstrapped predictions are incorporated in the MPC optimization using a risk-aware methodology so as to provide probabilistic guarantees on obstacle avoidance. Note, however, that the bootstrapping technique is a heuristic for the uncertainty quantification of the obstacle trajectory predictions and the risk-level we account for in the planning may or may not represent the true risk-level for obstacle avoidance. The next chapter will expand on these issues and take a deeper look at adaptive uncertainty quantification that provides true probabilistic coverage. The methods introduced in this chapter are validated using simulations of a multi-rotor drone that avoids various moving obstacles.

This chapter was adapted from:

Skylar X. Wei*, Anushri Dixit*, Shashank Tomar, and Joel W. Burdick. Moving obstacle avoidance: A data-driven risk-aware approach. IEEE Control Systems Letters, 7:289–294, 2022. doi: 10.1109/LCSYS.2022.3181191.

6.1 Introduction

Emerging applications of robots in urban, cluttered, and potentially hostile environments have increased the importance of online path planning with obstacle behavior classification and avoidance [41, 134]. Traditionally, robot-obstacle interaction is formulated as the problem of planning a collision-free path from a starting position to a goal [81]. In environments with an arbitrary number of moving obstacles and agents with bounded velocity, this problem is known to be NP-hard [28].

This chapter presents a new framework for discovering the dynamics of a priori unknown moving obstacles, forecasting their trajectories, and providing risk-aware optimal avoidance strategies. It replaces the need for obstacle trajectory/model classification while allowing online computation. Extracting a dynamics model from data is challenging [26], especially when the available data is limited, noisy, and partial. To tackle partial measurements, we leverage Takens' embedding theorem [127], which uses partial observations to produce an attractor that is diffeomorphic to the full-state attractor. We then use Singular Spectrum Analysis (SSA) [57, 58] to separate noise from the underlying signal and to extract a predictive model of obstacle behavior. Our use of time delay embedding is also the basis of Eigensystem Realization Algorithm (ERA) in linear system identification [72]. Inspired by [45], we use a classical bootstrap to forecast a set of obstacle trajectories with statistical quantification. An MPC planner then incorporates the set of obstacle forecasts as an affine conservative approximation of a distributionally-robust chance constraint (DRCC). This constraint is then efficiently recast in a risk-aware manner, allowing an MPC optimization based on sequential convex programming [13, 95].

We demonstrate our approach on three scenarios that exhibit increasingly complicated dynamical behavior. Monte-Carlo simulations verify the planner's ability to uphold the user chosen chance constraint. The risk-aware reformulation not only gives provable probabilistic collision avoidance guarantees, but also allows an on-line execution of the planner.

Notation: The set of positive integers, natural numbers, real numbers, and positive reals are denoted as $\mathbb{Z}_+$, $\mathbb{N}$, $\mathbb{R}$, and $\mathbb{R}_+$, respectively. We denote the sequence of consecutive integers $\{i, \cdots, i+k\}$ as $\mathbb{Z}^{i:i+k}$. The finite sequence $\{a_1, \cdots, a_k\}$ of scalars or vectors a is denoted as $\{a\}_1^k$. The expression $I_{n\times n}$ denotes n by n identity matrices and $\mathbf{1} = [1, 1, 1]^T$.

6.2 SSA Preliminaries

Consider a discrete-time multivariate stochastic process $\{\mathbf{o}^m\}_1^N$ where m denotes the m^{th} observable measurement of the process, and N is the total number of available observations. Suppose that the true stochastic process model of the observables is $\hat{\mathbf{o}}^m = \mathbf{o}^m + \gamma^m$, where γ^m denotes a random discrete-time zero-mean measurement noise, and $\mathbf{o}^m$ is the noiseless observable that captures the governing laws, which can be composed of *trends*, *seasons*, and *stationary* time series. Singular Spectrum Analysis [57] separates the true signal $\mathbf{lo}^m$ and the noise γ^m and extracts a

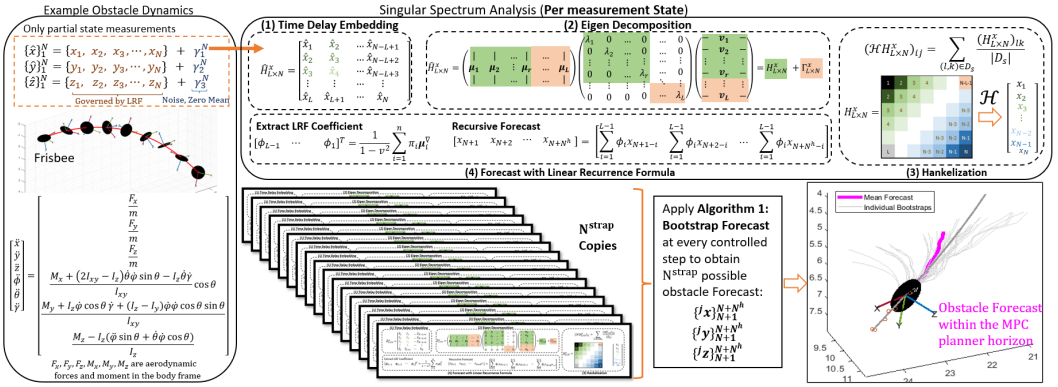


Figure 6.1: A description of bootstrap-SSA-forecast architecture in forecasting the trajectory of a Frisbee where the stochastic observables (corrupted by zero-mean, noise) consist of $\{\hat{\mathbf{o}}\}_1^N = [\{\hat{x}\}_1^N, \{\hat{y}\}_1^N, \{\hat{z}\}_1^N]$, the Frisbee's center positions with respect to an inertial frame. The SSA analysis and bootstrap forecast is applied to every observable state. Despite its 12-state governing dynamics [68] and with only center position measurements of the Frisbee, we show an example N^{strap} forecasts of the Frisbee trajectory for future time steps $\{1, 2, \cdots, N^h\}$ using our proposed framework.

recursive governing dynamic model of $\mathbf{o}^m$ that can generate a short term accurate forecast. Figure 6.1 describes this method.

Time Delay Embedding

Takens' method of delays [127] can reconstruct qualitative features of the full-state phase-space from delayed partial observations. The m^{th}-state observables $\hat{\mathbf{o}}^m$ are delay embedded into a trajectory (Hankel) matrix $H^m_{[L\times N]}$, Figure 6.1 gives an example of the Hankel matrix for state x. Parameter L is the time delay length, and N is the time series length. Repeating patterns in the Hankel matrix represent underlying trends and oscillations, which can be extracted from its covariance matrix: $X^m = H^m_{[L,N]}(H^m_{[L,N]})^T$.

Eigen Decomposition

To recover the true signal $\mathbf{o}^m$, we seek the best, low-rank matrix approximation of this signal by thresholding the eigenvalues of X^m [2]. The symmetric covariance matrix X^m has a spectral decomposition $U\Sigma U^T$, where Σ is a diagonal matrix with real eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_L$. The matrix of left eigenvectors $U = \left[\mu_1, \cdots, \mu_L\right]$ is orthogonal. The truncated right eigenvectors $\mathrm{l}V = [\nu_1, \cdots, \nu_L]^T \in \mathbb{R}^{L\times N}$ of X^m can be found as $V = U\Sigma$. Suppose λ is the optimal threshold and $\lambda_n \geq \lambda \geq \lambda_{n+1}$, which

partitions the Hankel matrix $H^m_{[L,N]}$ as:

$$H^m_{[L,N]} = \underbrace{\sum_{p=1}^{n} \sqrt{\lambda_p}\mu_p \nu_p^T}_{\triangleq H^o_{[L,N]}} + \underbrace{\sum_{p=n+1}^{L} \sqrt{\lambda_p}\mu_p \nu_p^T}_{\triangleq H^\gamma_{[L,N]}} . \tag{6.1}$$

Hankelization

Matrix $H^o_{[L,N]}$ in (6.1) should maintain a Hankel structure: minor variations in its k^{th} secondary diagonals result from insufficient noise removal.[1] A Hankelization step performs secondary diagonal averaging in order to find the matrix $\mathcal{H}O$ that is closest to $H^o_{[L,K]}$ with respect to the Frobenius norm among all $L \times N$ Hankel matrices [57]. The operator $\mathcal{H}$ acting on $L \times N$ matrix $H^y_{[L,N]}$ entry wise is defined as: for the $(i,j)^{th}$ element of matrix $H^o_{[L,N]}$ and $i + j = s$, define a set $D_s \triangleq \{(l,n) : l + n = s, l \in \mathbb{Z}^{1:L}, n \in \mathbb{Z}^{1:N}\}$, is mapped to $(i,j)^{\text{th}}$ element of the Hankelized $\mathcal{H}H^o_{[L,N]}$ via the expression in Figure 6.1 (for the case of $\mathbf{o}^m = \mathbf{x}$), where $|D_s|$ denotes the number of elements in set D_s .

Forecast with Linear Recurrence Formula

Definition 4 *A time series $Y_N = \{y\}_1^N$ admits an L-decomposition of order not larger than d, denoted by $ord_L(Y_N) \leq d$, if there exist two systems of functions $\varrho_k : \mathbb{Z}^{0:L-1} \to \mathbb{R}, \vartheta_k : \mathbb{Z}^{0:L-1} \to \mathbb{R}$, such that $y_{i+j} = \sum_{k=1}^{d} \vartheta_k(i)\varrho_k(j)$, $\{i,j\} \in \mathbb{Z}^{0:L-1} \times \mathbb{Z}^{0:L-1}$ $\forall k \in \mathbb{Z}^{1:d}$.*

If $\text{ord}_L(Y_N) = d$, then the series Y_N admits a *L-decomposition of the order d* and both systems of functions $(\varrho_1, \cdots, \varrho_d)$ and $(\vartheta_1, \cdots, \vartheta_d)$ are linearly independent [59].

Definition 5 *A time series $\{y\}_1^N$ is governed by a linear recurrent relations/formula (LRF), if there exist coefficients $\{\phi\}_1^d$ and $\phi_d \neq 0$ such that*

$$y_{i+d} = \sum_{k=1}^{d} \phi_k y_{i+d-k}, \quad \forall i \in \mathbb{Z}^{0:N-d}, d < N . \tag{6.2}$$

[1] The k^{th} secondary diagonals of a matrix M are also the k^{th} diagonals of M flipped horizontally with respect to its middle column.

Real-valued time series governed by LRFs consists of sums of products of polynomials, exponentials and sinusoids [57].

Theorem 8 *[57] Let $\mu_i^{1:L-1}$ be the vector of the first $L-1$ components of a left eigenvector μ_i of $H_{[L,N]}^m$, and let π_i be the L^{th} component of eigenvector μ_i. Let $v^2 \triangleq \sum_{i=1}^d \pi_i^2$. Under Assumptions 6 and 7 (see below), the LRF coefficients ϕ_i where $i \in [1, L-1]$ can be computed as:*

$$\begin{bmatrix} \phi_{L-1} & \phi_{L-2} & \cdots & \phi_1 \end{bmatrix}^T = \frac{1}{1-v^2} \sum_{i=1}^{d} \pi_i \mu_i^{1:L-1}, \tag{6.3}$$

and $\mathbf{y}$ *evolves as the LRF:* $\mathbf{y}_{N+1} = \sum_{j=1}^{L-1} \phi_j \mathbf{y}_{N-j}$.

6.3 Problem Statement

Consider the linear, discrete-time dynamical agent model:

$$\mathbf{x}_{i+1} = A\mathbf{x}_i + B\mathbf{u}_i, \qquad \mathbf{y}_{i+1} = G\mathbf{x}_{i+1} \tag{6.4}$$

where $\mathbf{x}_i \in \mathbb{R}^{n_\mathbf{x}}$, $\mathbf{u}_i \in \mathbb{R}^{n_\mathbf{u}}$, and $\mathbf{y}_i \in \mathbb{R}^{n_\mathbf{y}}$, for all $i \in \mathbb{N}$, correspond to the system states, controls, and outputs at time index i respectively. The state transition, actuation, and measurement matrices are $A \in \mathbb{R}^{n_\mathbf{x} \times n_\mathbf{x}}$, $B \in \mathbb{R}^{n_\mathbf{x} \times n_\mathbf{u}}$, and $G \in \mathbb{R}^{n_\mathbf{y} \times n_\mathbf{x}}$ respectively. Constant matrix $C \in \mathbb{R}^{3 \times n_\mathbf{x}}$ maps the system's states (6.4) to the system's x, y, z positions with respect to inertial frame E. We model the k^{th} obstacle, $k \in \mathbb{Z}^{1:N^{\text{obs}}}$, as a sphere. The obstacle occupies the point set $O_k(\mathbf{c}_k, r_k) = \{\mathbf{x} \in \mathbb{R}^3 : \|\mathbf{c}_k - \mathbf{x}\|_2 \leq r_k\}$, where $\mathbf{c}_k \in \mathbb{R}^3$ and $r_k \in \mathbb{R}_+$ are the k^{th} obstacle's center and radius.

We consider the case where the agent (6.4) is tasked with following a reference output trajectory $\mathbf{y}^{\text{ref}}$ which need not consider obstacle information. While following this path, the agent may encounter N^{obs} spherical, stationary or moving obstacles. The obstacle-free region is the open set:

$$\mathcal{S} \triangleq \left\{ \mathbb{R}^3 \setminus \cup_{k=1}^{N^{\text{obs}}} O_k \right\}. \tag{6.5}$$

Assumption 5 *Obstacles can be detected and localized at the same rate (f^+ Hz) as the planner update. Only measurements of an obstacle's geometric center with respect to frame E are assumed, and they are corrupted by a zero-mean noise. We can estimate the radius, r_k, of the k^{th} obstacle as $\hat{r}_k$, and the estimate satisfies $\hat{r}_k \geq r_k$.* [2]

[2] Note Assumption 5 **does not** imply full state measurement.

Assumption 6 *All obstacle measurements, admit an L-decomposition of order d, are governed by LRFs* (6.2) *whose LRF coefficients can be uniquely defined.*

Assumption 7 *We assume that the obstacles' velocities are bounded by v_{max}, and the initial distances between all obstacles and the agent are significantly greater than $\frac{dv_{max}}{f^+}$.*

Problem 3 ***[Prediction]*** *Consider a multivariate stochastic process where observables $\{x\}_1^N$, $\{y\}_1^N$, and $\{z\}_1^N$ denote the spherical obstacle's true center location in reference frame, E. The measurements are corrupted by independent, zero-mean noises $\{\gamma_1\}_1^N$, $\{\gamma_2\}_1^N$, and $\{\gamma_3\}_1^N$ (see Figure 6.1). Under Assumptions 5-7, we seek to predict the obstacle position at times $N+1$ to $N+N^h$ using these measurements.*

Due to limited and noisy partial data and the lack of explicit dynamics models, we estimate a Bootstrap distribution of the obstacle predictions, denoted by the random set $\mathcal{O}^{\text{pred}}$, from time index $N+1$ to $N+N^h$ and calculate its first and second moments. We account for errors in the forecastS due to poor signal and noise separation and bandwidth limits (due to limited training data and incorrect choices of embedding length L) by solving a DRCC MPC problem.

Problem 4 ***[Planning]*** *Consider the system* (6.4) *and free-space* (6.5)*. Given a discrete-time reference trajectory $\mathbf{y}_i^{ref}\,\forall i \in \mathbb{Z}^{1:N^h}$ where $N^h \in \mathbb{Z}_+$ is the length of the horizon, convex state constraints $\mathcal{D}^{\mathbf{x}} \subset \mathbb{R}^{n_{\mathbf{x}}}$, convex input constraints $\mathcal{D}^{\mathbf{u}} \subset \mathbb{R}^{n_{\mathbf{u}}}$, and a convex stage cost $L_i : \mathbb{R}^{n_{\mathbf{x}}} \times \mathbb{R}^{n_{\mathbf{u}}} \to \mathbb{R}_{\geq 0}$, a total of N^{obs} spherical obstacles each approximated by a set $\mathcal{O}_k^{pred}$, and risk tolerance $\epsilon \in (0, 1]$, we seek to compute a receding horizon controller $\{\mathbf{u}^*\}_1^{N_h}$ that avoids the unsafe set $\mathcal{O}^{pred} \triangleq \bigcup_{k=1}^{N^{obs}} \mathcal{O}_k^{pred}$ via the following non-convex optimization:*

$$\{\mathbf{u}^*\}_1^{N^h} = \min_{\{\mathbf{u}_k\}_1^{N^h} \in \mathbb{R}^{n_{\mathbf{u}}}} \sum_{i=1}^{N^h} L_i(\mathbf{y}_i^{ref} - \mathbf{y}_i, \mathbf{u}_i) \tag{6.6a}$$

$$s.t. \quad \mathbf{x}_{i+1} = A\mathbf{x}_i + B\mathbf{u}_i \quad \mathbf{y}_{i+1} = G\mathbf{x}_{i+1} \tag{6.6b}$$

$$\mathbf{x}_i \in \mathcal{D}^{\mathbf{x}}, \quad \mathbf{u}_i \in \mathcal{D}^{\mathbf{u}}, \quad \mathbf{x}_1 = \mathbf{x}_{init} \tag{6.6c}$$

$$\mathbb{P}(\mathbf{x}_i \in \mathcal{O}^{pred}) \leq \epsilon, \quad \forall i \in \mathbb{Z}^{1:N^h} \tag{6.6d}$$

6.4 Bootstrap Forecasting

Despite empirical successes in reconstructing and forecasting [58], the theoretical accuracy of SSA is strenuous to obtain, see [3]. Inspired by [45], we use bootstrapping to improve model discovery and to produce probabilistic forecasts.

Our real-time bootstrap forecast, Algorithm 4, assumes time series measurements corrupted by noise. The user-defined parameters N^{train} and N^{step} represent the number of initial training samples, and the number of newly accumulated samples during an initial bootstrap. Further, one must choose parameters δ_t and N^{σ}, where threshold δ_t is used to separate signal from noise, and N^{σ} is the number of steps of progressive relaxation of threshold δ_t.[3] In the desired signal/noise separation (6.1), the unknown theoretical optimal threshold λ must be estimated. Let $Y_N^{\lambda_1:\lambda_d}$ be the Hankelization reconstructed $\hat{\mathbf{y}}$ with the eigenvalues $\{\lambda\}_1^d$ and their corresponding right and left eigenvectors. Note, if $d > n$ where $\lambda_n \leq \lambda \leq \lambda_{n+1}$, then the norm values $\|Y_N^{\lambda_1:\lambda_{d+t}} - Y_N^{\lambda_1:\lambda_{d+t+1}}\|_2 \approx \|Y_N^{\lambda_1:\lambda_{d+t+1}} - Y_N^{\lambda_1:\lambda_{d+t+2}}\|_2$ since they are comprised of the residual measurement noise. We threshold the difference between two consecutive reconstructions with δ_t/N, i.e. finding the smallest $t \in \mathbb{Z}_+$ s.t.:

$$\|Y_N^{\lambda_1:\lambda_t} - Y_N^{\lambda_1:\lambda_{t+1}}\|_2 - \|Y_N^{\lambda_1:\lambda_{t+1}} - Y_N^{\lambda_1:\lambda_{t+2}}\|_2 \leq \frac{\delta_t}{N} \tag{6.7}$$

Since the selection of the threshold δ_t is crucial, we add an additional parameter N^{σ} to ensure no principle components are lost in $Y_N^{\lambda_1:\lambda_d}$ because of bad choice of δ_t, i.e. to avoid $d < n$. We also include the next N^{σ} largest eigenvalues after the first t eigenvalues in the bootstrapping process. Most importantly, the number of bootstraps, N^{strap}, needs to be determined *a priori*, considering the computation capacity, number of obstacles, and the expected noise level[4].

6.5 Bootstrap Planning

This section introduces an MPC-based path planner to solve Problem 4. First, we reformulate the obstacle avoidance constraint (6.6d) in terms of the mean and variance of the bootstrap predictions. Next, we use this constraint in the MPC optimization, and provide probabilistic guarantees of constraint satisfaction. Algorithm 4 produces N^{strap} copies of N^h length predictions of the k^{th} obstacle's location. We

[3] The parameters δ_t and N^{σ} are dictated by measurement noise levels, which can be characterized off-line in a controlled experimental setting.

[4] The effectiveness of Algorithm 4 depends highly on the time delay length L, the number of training measurements N^{train}, the number of bootstraps N^{strap}, and the MPC horizon length, N^h. We recommend that N^{train} be at least $10N^h$ and that $L = \frac{N^{\text{train}}}{4}$. N^{strap} and N^{step} should be as large as allowed by the computing platform and benchmarking them offline.

Algorithm 4 Bootstrap Forecast Algorithms (Per Obstacle)

Data: Obstacle center position measurements $\{\hat{\mathbf{x}}\}_1^N$, $\{\hat{\mathbf{y}}\}_1^N$, $\{\hat{\mathbf{z}}\}_1^N$,
User defined constants: N^{train},N^{step}, δ_t, N^{σ}, N^{strap}
Result: Forecast:$\{{}^j\mathbf{x}\}_{N+1}^{N+N^h}$,$\{{}^j\mathbf{y}\}_{N+1}^{N+N^h}$,$\{{}^j\mathbf{z}\}_{N+1}^{N+N^h}$,$\forall j \in \mathbb{Z}^{1:N^{\text{straps}}}$
Use $\{\hat{\mathbf{x}}_{N+1}, \hat{\mathbf{y}}_{N+1}, \hat{\mathbf{z}}_{N+1}\}$ to update Hankel matrix
while istrap $\leq N^{\text{strap}}$ **do**
  while $N+1 \geq N^{\text{train}}$ **do**
    for states $= x, y, z$ **do**
      while (6.7) holds **do**
        t++
      end
      obtain $(\{\lambda^{\text{istrap}}\}_1^t, \{\mu^{\text{istrap}}\}_1^t, \phi^{\text{istrap}})$ for each states, istrap++
      for $tt = t+1 : t+N^{\sigma}$ **do**
        obtain $(\{\lambda^{\text{istrap}}\}_1^{tt}, \{\mu^{\text{istrap}}\}_1^{tt}, \phi^{\text{istrap}})$ for each states, istrap++
      end
    end
    $N^{\text{train}} = N^{\text{train}} + N^{\text{step}}$
  end
  Back-up Strategy
end
Apply the tuples $(\{{}^j\lambda^{\text{istrap}}\}_1^{t_j}, \{{}^j\mu^{\text{istrap}}\}_1^{t_j}, {}^j\phi^{\text{istrap}})$ $\forall j \in \mathbb{Z}_{1:\text{Nstraps}}$ for x, y, z to the updated Hankel, where t_j denotes number of eigenvalues post truncation for the j^{th} bootstrap. Perform a N^h step forecast using ${}^j\phi^{\text{istrap}}$.

denote the j^{th} copy of the bootstrap prediction as $\{\hat{\mathbf{y}}_k^j\}_1^{N_h} = \{\hat{\mathbf{y}}_{1,k}^j, \hat{\mathbf{y}}_{2,k}^j, \cdots, \hat{\mathbf{y}}_{N^h,k}^j\}$. The collision avoidance set constraint (6.6d) can be reformulated based on the obstacle shape and center as $\|C\mathbf{x}_i - \hat{\mathbf{y}}_{i,k}^j\|_2 \geq \hat{r}_k + r_p \triangleq \overline{r}_k$, for each $i \in \mathbb{Z}^{1:N^h}$, $k \in \mathbb{Z}^{1:N^{\text{obs}}}$, and where r_p is the agent's safety radius (6.4). This collision constraint can be reexpressed as the following concave (in the state $\mathbf{x}_i$) constraint,

$$(C\mathbf{x}_i - \hat{\mathbf{y}}_{i,k}^j)^T(C\mathbf{x}_i - \hat{\mathbf{y}}_{i,k}^j) \geq \overline{r}_k\|(C\mathbf{x}_i - \hat{\mathbf{y}}_{i,k}^j)\|_2. \tag{6.8}$$

We approximate (6.8) as an affine constraint through the use of Sequential Convex Programming (SCP) [13, 95]

$$(C\mathbf{x}_i - \hat{\mathbf{y}}_{i,k}^j)^T(C\overline{\mathbf{x}}_i - \hat{\mathbf{y}}_{i,k}^j) \geq \overline{r}_k\|(C\overline{\mathbf{x}}_i - \hat{\mathbf{y}}_{i,k}^j)\|_2, \tag{6.9}$$

where $\overline{\mathbf{x}}_i$ is approximated with the solution from previous SCP iterations. Note that (6.9) over-approximates constraint (6.8) (see [95] for proof).

Lemma 9 *If we have N^{strap} forecasts of the k^{th} obstacle's position from time $i \in \mathbb{Z}^{1:N^h}$ and the previous SCP trajectory $\{\overline{\mathbf{x}}\}_1^{N_h}$, then we can define the j^{th} bootstrap lumped collision avoidance coefficients $\alpha_{i,k}^j$, $\beta_{i,k}^j$ and the standard deviation of the collision*

avoidance constraint $\Delta_{i,k}$ *as:*

$$\alpha^j_{i,k} \triangleq -C^T(C\overline{\mathbf{x}}_i - \hat{\mathbf{y}}^j_{i,k}) \tag{6.10}$$

$$\beta^j_{i,k} \triangleq \overline{r}_k \|(C\overline{\mathbf{x}}_i - \hat{\mathbf{y}}^j_{i,k})\|_2 - (C\overline{\mathbf{x}}_i)^T(C\overline{\mathbf{x}}_i - \hat{\mathbf{y}}^j_{i,k}) \tag{6.11}$$

$$\Delta_{i,k} \triangleq \sqrt{\mathbf{p}_i^T \Sigma_{\alpha_{i,k}} \mathbf{p}_i + 2\mathbf{p}_k^T \Sigma_{\alpha\beta_{i,k}} + \Sigma_{\beta i,k}}, \tag{6.12}$$

where, $\Sigma_{\alpha_{i,k}} \triangleq cov\left(\alpha^j_{i,k}, \alpha^j_{i,k}\right)$, $\Sigma_{\beta_{i,k}} \triangleq cov\left(\beta^j_{i,k}, \beta^j_{i,k}\right)$, *and* $\Sigma_{\alpha\beta_{i,k}} \triangleq cov\left(\alpha^j_{i,k}, \alpha^j_{i,k}\right)$ *are sample covariance matrices computed using the bootstrapped coefficients* $\{\alpha_{i,k}\}_1^{N^{strap}}$ *and* $\{\beta_{i,k}\}_1^{N^{strap}}$ *and* $\mathbf{p}_i \triangleq C\mathbf{x}_i \in \mathbb{R}^3$. *Let the dimension of the null space of* $\Sigma_{\alpha_{i,k}}$ *be* $n_{i,k} \geq 0$.[5] *The standard deviation* $\Delta_{i,k}$ *has the following upper bound,*

$$\Delta_{i,k} \leq \mathbf{1}^T |\tilde{\Sigma}^{\frac{1}{2}}_{\alpha_{i,k}} \left(\mathbf{p}_i - \mathbf{h}_{i,k}\right)| + \sqrt{3k_{i,k}} \triangleq \zeta_{i,k}, \tag{6.13}$$

where $\tilde{\Sigma}_{\alpha_{i,k}} = \Sigma_{\alpha_{i,k}} + I^{null}_{i,k}$, $I^{null}_{i,k} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & I_{n_{i,k} \times n_{i,k}} \end{bmatrix} \in \mathbb{R}^{3\times 3}$, *and*

$$\begin{bmatrix} \mathbf{h}_{i,k} \\ k_{i,k} \end{bmatrix} \triangleq \begin{bmatrix} -\left(\Sigma_{\alpha_{i,k}} + I^{null}_{i,k}\right)^{-1} \Sigma_{\alpha\beta_{i,k}} \\ \Sigma_{\beta_{i,k}} - \Sigma^T_{\alpha\beta_{i,k}} \left(\Sigma_{\alpha_{i,k}} + I^{null}_{i,k}\right)^{-1} \Sigma_{\alpha\beta_{i,k}} \end{bmatrix}. \tag{6.14}$$

Proof: Let the eigendecomposition of $\Sigma_{\alpha_{i,k}}$ be the following:

$$\Sigma_{\alpha_{i,k}} = \begin{bmatrix} U_r & U_n \end{bmatrix} \begin{bmatrix} \Lambda_r & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_r & U_n \end{bmatrix}^T$$

where, $U_r \in \mathbb{R}^{3\times(3-n_{i,k})}$ is comprised of the eigenvectors of $\Sigma_{\alpha_{i,k}}$ that are orthonormal. The columns of $U_n \in \mathbb{R}^{n_{i,k}}$ are the complementary orthonormal basis that spans the null space of $\Sigma_{\alpha_{i,k}}$. By substituting (6.14) one can verify the following inequality:

$$\Delta_{i,k} \leq \sqrt{(\mathbf{p}_i - \mathbf{h}_{i,k})^T \tilde{\Sigma}_{\alpha_{i,k}} (\mathbf{p}_i - \mathbf{h}_{i,k}) + k_{i,k}} \triangleq \tilde{\Delta}_{i,k} \tag{6.15}$$

where $\tilde{\Sigma}_{\alpha_{i,k}}$ is a positive definite matrix because

$$\tilde{\Sigma}_{\alpha_{i,k}} = \begin{bmatrix} U_r & U_n \end{bmatrix} \left(\begin{bmatrix} \Lambda_r & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & I_{n_{i,k}\times n_{i,k}} \end{bmatrix} \right) \begin{bmatrix} U_r & U_n \end{bmatrix}^T .$$

We further upper bound (6.15) by adding a positive constant, $\iota_{i,k} \triangleq \frac{2}{\sqrt{3}} \mathbf{1}^T |\tilde{\Sigma}^{1/2}_{\alpha_{i,k}} \mathbf{p}_k - \mathbf{h}_k|$, to $\tilde{\Delta}^2_{i,k}$ and obtain

[5] In all numerical simulations, $\Sigma_{\alpha_{i,k}}$ is strictly positive definite. However, when one or more measurable states are noiseless, $\Sigma_{\alpha_{i,k}}$ can be ill-conditioned. Instead of adding $I^{\text{null}}_{i,k}$, which can be numerically expansive to determine, we recommend applying Algorithm 4 only to states with measurement noise, and adapt Theorem 10 with deterministic forecasts for the noiseless states and the DRCC formulation for the noisy ones.

$$\tilde{\Delta}_{i,k}^2 \leq \tilde{\Delta}_{i,k}^2 + \frac{2}{\sqrt{3}}\mathbf{1}^T|\tilde{\Sigma}_{\alpha_{i,k}}^{1/2}(\mathbf{p}_k - \mathbf{h}_k)| \leq \sqrt{\xi_{i,k}^T \xi_{i,k}}$$

where $\xi_{i,k} \triangleq |\tilde{\Sigma}_{\alpha_{i,k}}^{1/2}(\mathbf{p}_k - \mathbf{h}_{i,k})| + \mathbf{1}\sqrt{\frac{k_{i,k}}{3}} \in \mathbb{R}^3$. For the inequality to hold, the expression $\iota_{i,k}$ must always be non-negative which is true by construction. Further, let $\zeta_{i,k} = \mathbf{1}^T\xi_{i,k} \in \mathbb{R}$, then $\zeta_{i,k}^2 = (\xi_{i,k}^T\mathbf{1})(\mathbf{1}^T\xi_{i,k}) = \xi_{i,k}^T\xi_{i,k} + 2\epsilon_\xi$. If $\epsilon_\xi \geq 0$, we can then state $\Delta_{i,k} \leq \zeta_{i,k}$ which completes the proof (since $\xi_{i,k} = [\xi_{i,k}^x, \xi_{i,k}^y, \xi_{i,k}^z] \in \mathbb{R}^3$, then $\epsilon_\xi = \xi_{i,k}^x\xi_{i,k}^y + \xi_{i,k}^x\xi_{i,k}^z + \xi_{i,k}^y\xi_{i,k}^z > 0$ because $\xi_{i,k}^x, \xi_{i,k}^y, \xi_{i,k}^z \in \mathbb{R}_+$). ■

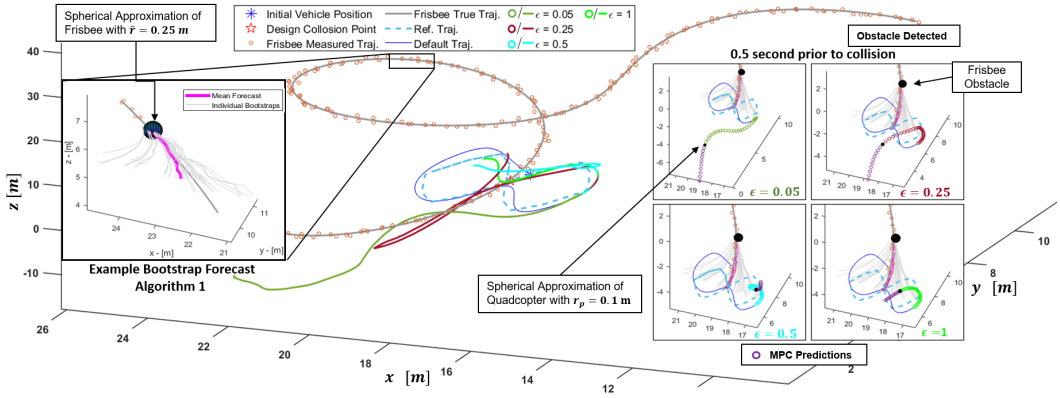


Figure 6.2: Four Monte-Carlo simulations with agent dynamics (6.19) and a Frisbee obstacle (see Figure 6.1) are compared. The same obstacle behaviors are simulated while the agent tracks the same figure '8' reference trajectory with four risk levels $\epsilon = \{0.05, 0.25, 0.5, 1\}$. The simulation is designed to be difficult: the vehicle must deviate from its reference trajectory as the obstacle trajectory is designed to intersect the agent's reference trajectory with noise obstacle trajectory measurements. All measurement noises are sampled uniformly between $[-0.125, 0.125]$ meters. The bootstrap obstacle forecast uses the parameters: $L = 24$, $N^{\text{train}} = 100$, $N^{\text{step}} = 5$, $\delta_t = 20$, $N^\sigma = 8$, $N^{\text{strap}} = 40$. SSA-MPC uses the constants $N^h = 10$, $\chi = 50$ and $\tau = 0.25$ with fixed 4-step SCP iterations. The tuple $(\{\lambda^j\}_1^{t_j}, \{\mu^j\}_1^{t_j}, \phi^j), \forall j \in \mathbb{Z}^{1:40}$ in Algorithm 4 is computed with observables measured at 20 Hz. The four sub-diagrams show the planned trajectory at 4 risk levels; the planner is more conservative as $\epsilon \to 0$, and aligns with the results shown in Table 7.1 and 6.2.

It is costly to incorporate each boostrap as a separate obstacle constraint, as the number of constraints grow linearly with N^{strap}. Instead, we estimate the ensemble mean and covariance of the distance from the obstacle. A DRCC accounts for *all bootstrap distributions* that can have this mean and covariance. This approach results in a significantly fewer obstacle constraints, whose cardinality remains fixed regardless of the number of bootstrap predictions N^{strap}.

Theorem 10 *(SSA-MPC) Consider Problem 4 under Assumptions 5-7 with system*

dynamics (6.4) *and bootstrap SSA forecasts of all obstacles' center positions. Given a risk tolerance* ϵ*, the solution to the following optimal control problem is a feasible solution of Problem 4 as* $w \to \infty$*. The SCP optimization problem at iteration* w *is:*

$$\{\mathbf{u}^*\}_1^{N_h} = \min_{\substack{\mathbf{u}_i \in \mathbb{R}^{n_\mathbf{u}} \\ \mathbf{s}_{i,k} \in \mathbb{R}^3}} \sum_{i=1}^{N_h} L_i(\mathbf{y}_i^{ref} - G\mathbf{x}_i, \mathbf{u}_i) \tag{6.16a}$$

$$s.t. \quad \mathbf{x}_{i+1} = A\mathbf{x}_i + B\mathbf{u}_i \tag{6.16b}$$

$$\mathbf{x}_i \in \mathcal{D}^\mathbf{x}, \quad \mathbf{u}_i \in \mathcal{D}^\mathbf{u}, \quad \mathbf{x}_1 = \mathbf{x}_{init} \tag{6.16c}$$

$$\Lambda_{i,k} \begin{bmatrix} \mathbf{x}_i & \mathbf{s}_{i,k} \end{bmatrix}^T \leq \Gamma_{i,k} \tag{6.16d}$$

$$\|\mathbf{x}_i - \bar{\mathbf{x}}_i\| \leq \chi\tau^w \ \ \forall (i,k) \in \mathbb{Z}^{1:N^h} \times \mathbb{Z}^{1:N^{obs}} \tag{6.16e}$$

where $\{\bar{\mathbf{x}}\}_1^{N^h}$ *is the solution to the* $(w-1)^{th}$ *iteration of the SCP optimization,* $\Lambda_{i,k} \in \mathbb{R}^{7\times 11}$ *and* $\Gamma_{i,k} \in \mathbb{R}^7$ *encode the risk-based collision avoidance relationships,*

$$\Lambda_{i,k} = \begin{bmatrix} \mathbb{E}[\alpha_{i,k}]^T C & \mathbf{1}^T \nu_{\epsilon_n} \\ \tilde{\Sigma}_{\alpha_{i,k}}^{1/2} C & -I_{3\times 3} \\ -\tilde{\Sigma}_{\alpha_{i,k}}^{1/2} C & -I_{3\times 3} \end{bmatrix}, \quad \Gamma_{i,k} = \begin{bmatrix} -\mathbb{E}[\beta_{i,k}] - \nu_{\epsilon_n}\sqrt{3k_{i,k}} \\ \tilde{\Sigma}_{\alpha_{i,k}}^{1/2} \mathbf{h}_{i,k} \\ -\tilde{\Sigma}_{\alpha_{i,k}}^{1/2} \mathbf{h}_{i,k} \end{bmatrix},$$

where $\epsilon_n \triangleq \frac{\epsilon}{N^{obs}}$ *and* $\nu_{\epsilon_n} \triangleq \sqrt{\frac{1-\epsilon_n}{\epsilon_n}}$*. Lastly,* $\chi \geq 0$ *is the initial trust region and* $\tau \in (0,1)$ *the worst-case rate of convergence.*

Proof: Denote the j^{th} random bootstrapped obstacle forecastsas $\mathbf{z}_{i,k}^j \triangleq (\alpha_{i,k}^j)^T \mathbf{x}_i + \beta_{i,k}^j$, where $\alpha_{i,k}^j$ and $\beta_{i,k}^j$ are defined in (6.10) and (6.11). The obstacle avoidance constraint (6.8) has an affine over approximation (6.9), which is equivalently given by $\mathbf{z}_{i,k}^j < 0$. Hence, the chance constraint (6.6d) is,

$$\mathbb{P}(\mathbf{x}_i \in O_{\text{pred}}) = \mathbb{P}\Big(\bigcup_{k=1}^{N^{\text{obs}}} \{\mathbf{z}_{i,k} \geq 0\}\Big) \leq \sum_{k=1}^{N^{\text{obs}}} \mathbb{P}(\mathbf{z}_{i,k} \geq 0).$$

Enforcing $\mathbb{P}(\mathbf{z}_{i,k} \geq 0) \leq \epsilon_n$, $\forall k \in \mathbb{Z}^{1:N^{\text{obs}}}$ also satisfies (6.6d). We can satisfy this constraint as a DRCC:

$$\sup_{\kappa \sim \left(\mathbb{E}[\mathbf{z}_{i,k}], \Sigma_{\mathbf{z}_{i,k}}\right)} \mathbb{P}\{\kappa \geq 0\} \leq \epsilon_n, \ \forall i,k \in \mathbb{Z}^{1:N^h} \times \mathbb{Z}^{1:N^{\text{obs}}},$$

where $\mathbb{E}[\mathbf{z}_{i,k}]$ and $\Sigma_{\mathbf{z}_{i,k}}$ are the sample mean and covariance of the bootstrapped $\{\mathbf{z}_{i,k}\}_1^{N^{\text{strap}}}$. We reformulate the above statement as a deterministic constraint as shown in [51],

$$\underbrace{\mathbb{E}[\mathbf{z}_{i,k}]}_{\mathbb{E}[\alpha_{i,k}]^T C\mathbf{x}_i + \mathbb{E}[\beta_{i,k}]} + \nu_{\epsilon_n} \underbrace{\sqrt{\Sigma_{\mathbf{z}_{i,k}}}}_{\Delta_{i,k}} \leq 0, \ \forall i \in \mathbb{Z}^{1:N^h}, k \in \mathbb{Z}^{1:N^{\text{obs}}}. \tag{6.17}$$

Cases	ϵ	0.05	0.1	0.25	0.5	0.75	1
Const. Speed	%Feas.	97.5	98.2	98.9	99.6	99.9	100
	%Succ.	100	100	100	100	100	59.0
	$\overline{d}_{min}$	2.26	1.85	1.41	1.12	0.94	0.64
	$\sigma(d_{min})$	0.42	0.33	0.25	0.22	0.24	0.35
Ball w/drag	%Feas.	99.5	99.6	99.9	100	100	100
	%Succ.	100	100	100	100	100	79.3
	$\overline{d}_{min}$	2.60	2.14	1.63	1.27	1.04	0.64
	$\sigma(d_{min})$	1.08	0.93	0.70	0.50	0.37	0.27
Frisbee w/drag	%Feas.	90.3	97.4	98.3	98.6	97.5	97.8
	%Succ.	100	100	100	100	99.7	58.0
	$\overline{d}_{min}$	4.97	3.97	2.85	2.01	1.44	0.78
	$\sigma(d_{min})$	1.97	1.53	1.15	0.91	0.76	0.77

Table 6.1: Summary of results from MC simulations of system (6.19)

Constraint (6.17) is not affine in the optimization variable, as is desirable for real-time application. By Lemma 9, $\Delta_{i,k} \leq \zeta_{i,k}$, and we deduce the following tighter inequality constraint as a numerically appealing alternative to (6.17),

$$\mathbb{E}[\alpha_{i,k}]^T C\mathbf{x}_i + \mathbb{E}[\beta_{i,k}] + \nu_{\epsilon_n}\left(\mathbf{1}^T|\Sigma^{1/2}_{\alpha_{i,k}}\mathbf{p}_i - \mathbf{h}_{i,k}| + \sqrt{3k_{i,k}}\right) \leq 0. \tag{6.18}$$

To account for the absolute value term, we introduce auxiliary optimization variables $\mathbf{s}_{i,k}$ that satisfy the following:

$$\Sigma^{1/2}_{\alpha_{i,k}}\mathbf{p}_i - \mathbf{h}_{i,k} \leq \mathbf{s}_{i,k}, \quad -\Sigma^{1/2}_{\alpha_{i,k}}\mathbf{p}_i + \mathbf{h}_{i,k} \leq \mathbf{s}_{i,k}.$$

Therefore, satisfying (6.16d) is equivalent to satisfying (6.18).

Convergence of the SCP is proven in [47] which is based on implementing a trust region via second-order cone constraints (6.16e). The authors also show the solution to the SCP as $w \to \infty$ is a feasible solution to Problem 4.[6] ■

6.6 Numerical Examples

We consider a quadcopter that follows a reference trajectory $\mathbf{y}^{\text{ref}}$ while avoiding unknown moving obstacles and respecting state and control constraints. Its position and Euler angles (roll, pitch, yaw) in frame E are denoted $x, y, z, \varphi, \theta, \psi$.

Example 1: Fully-actuated Multirotor with attitude controller

Assume there exists a low level attitude controller that tracks given attitude commands within 20 Hz. As a result, we use the following linear dynamic model to

[6] To be numerically feasible, w is usually upper bounded by a finite integer, resulting in a sub-optimal but still feasible solution.

Cases	ϵ	0.05	0.1	0.25	0.5	0.75	1
Const. Speed	%Feas.	83.9	84.2	85.6	86.5	88.6	99.8
	%Succ.	100	100	100	100	100	61.2
	$\overline{d}_{min}$	10.42	8.67	6.59	4.88	3.57	0.46
	$\sigma(d_{min})$	2.47	2.28	1.86	1.43	1.16	0.14
Ball w/drag	%Feas.	92.8	91.8	90.9	91.1	90.7	82.8
	%Succ.	100	100	100	100	100	0.1
	$\overline{d}_{min}$	6.95	5.82	4.25	3.04	2.13	1.82
	$\sigma(d_{min})$	2.52	2.55	2.27	1.98	1.59	N/A
Frisbee w/drag	%Feas.	92.2	94.5	92.9	92.3	86.9	100
	%Succ.	100	100	100	100	100	40.5
	$\overline{d}_{min}$	12.5	11.3	9.10	6.59	4.24	0.34
	$\sigma(d_{min})$	2.47	2.84	3.00	2.81	2.27	0.05

Table 6.2: Summary of results from MC simulations of system (7.10)

extract a high-level motion planner that outputs attitude and thrust inputs:

$$\ddot{x} = -g\theta,\ \ddot{y} = g\varphi,\ \ddot{z} = u_1 - g,\ \ddot{\psi} = u_4, \tag{6.19}$$

where the planner control inputs are given by $u_1, \theta, \varphi, u_4$ which are thrust, roll angle, pitch angle, and yaw rate, and where $g = 9.81 m/s^2$ is the gravitational acceleration.

Example 2: Multirotor operating in small angle regime

A mixer maps thrust and moment inputs into electronic speed controller PWM commands at 8kHz. Since the multirotor is constrained to operate within the state constraints $\theta \in [-0.45, 0.45]$ radians and $\varphi \in [-0.45, 0.45]$ radians, we use the following standard multirotor linear dynamics,

$$\ddot{x} = -g\theta,\ \ddot{y} = g\varphi,\ \ddot{z} = u_1 - g, \tag{6.20}$$

$$\ddot{\varphi} = \frac{u_2}{I_{xx}},\ \ddot{\theta} = \frac{u_3}{I_{yy}}, \ddot{\psi} = \frac{u_4}{I_{zz}}, \tag{6.21}$$

where the planner control inputs u_1, u_2, u_3, u_4 correspond to the thrust force in the body frame and three moments. The vehicle's moments of inertia are $I_{xx} = 0.0075 kgm^2, I_{yy} = 0.0075 kgm^2, I_{zz} = 0.013 kgm^2$. For both examples, the desire reference trajectory consists of positions, $\{x^{\text{ref}}\}, \{y^{\text{ref}}\}, \{z^{\text{ref}}\}$ and yaw angles $\{\psi^{\text{ref}}\}$.

We conducted Monte-Carlo (MC) simulations of our planner as it avoids three differently behaved obstacles which are introduced once in each run for both examples. See code.[7] Case 1 uses a constant speed spherical obstacle without drag. Case 2 is a thrown spherical obstacle with drag. In Case 3, a Frisbee is thrown at various initial

[7] `https://github.com/skylarXwei/Riskaware_MPC_SSA_Sim.git`

positions, velocities, and rotation speed. The sphere dynamics are captured by a 6-state ODE with drag penalties proportional to its velocities. The Frisbee is modeled following [68], using a 12-state model identical to Figure 6.1 with aerodynamic drag. The Frisbee is modeled as a sphere with the same radius as the Frisbee disk. We also benchmark our method against an artificial potential field alternative. A supplementary video (`https://youtu.be/6s8pfRZ171Q`) provides more details.

We conducted 1000 MC simulations per ϵ level to compare the numerical feasibility, percent success in obstacle avoidance (if the MPC planner is feasible), and the planner's conservativeness, as measured by the minimum distance between the obstacle and agent centers. For the three cases, the obstacle speed ranges are [0.41, 8.43], [3.41, 6.37], and [5.76, 6.68] m/s, respectively. The MPC planning and measurement rates are fixed at 20 Hz. With a 10 step horizon and 40 bootstraps, the average per planner update rate is 0.030 ± 0.0014 sec, using Gurobi [60] on an Intel i7-9700K CPU @3.6GHz processor, using a dynamic simulation written in MATLAB. The results in Tables 7.1 and 6.2 show the applicability of our SSA-MPC algorithm, despite vast differences in obstacle behavior. Further, as the risk tolerance ϵ shrinks, the percentage success in obstacle avoidance (when the solution is feasible) increases, with a trade-off in the feasibility of optimization (6.16). Parameters d_{min} and $\sigma(d_{min})$ are the average minimum distance between the agent and the obstacle and the standard deviation of this minimum distance across 1000 MC simulations, respectively. Based on Table 7.1 and 6.2, the risk tolerance ϵ can also viewed as a robustness parameter which is inversely proportional to the distance between the agent and obstacles. However, the cost of more robustly avoiding the obstacles is reflected in the numerical feasibility, a parameter describing the chances of the SCP formulation (6.16) being feasible for the entire simulation. The feasible set of the polytopic collision avoidance constraints (6.16d) shrinks as ϵ (and hence ϵ_n) decreases.

Chapter 7

DATA-DRIVEN MPC USING ADAPTIVE CONFORMAL PREDICTION

So far, we have considered uncertainty quantification techniques that are heuristic. For example in the previous chapter, the bootstrapping technique was an approximate technique to quantify the variations in the linear predictor based on the dataset used. However, we do not know if the risk-level, ϵ, that we used for the MPC is truly accounting for at least ϵ coverage for the real system. In this chapter, we provide an online algorithm that uses the data collected online to obtain uncertainty sets for multistep-ahead predictions with probabilistic coverage. These uncertainty sets are used within a model predictive controller to safely navigate among dynamic agents. While most existing data-driven prediction approaches (and the ones we've considered so far) quantify prediction uncertainty heuristically, we quantify the true prediction uncertainty in a distribution-free, adaptive manner that allows to capture changes in prediction quality and the agents' motion. We empirically evaluate our algorithm on a case study where a drone avoids a flying frisbee and compare with the results from the previous chapter.

This chapter was adapted from:

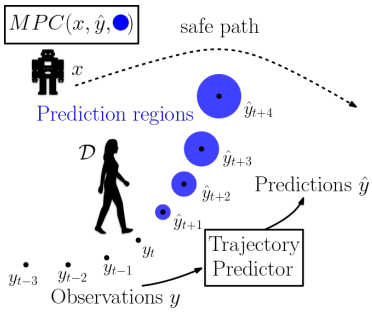
Anushri Dixit*, Lars Lindemann*, Skylar Wei, Matthew Cleveland, George J. Pappas, and Joel W. Burdick. Adaptive conformal prediction for motion planning among dynamic agents. In Submitted to Learning for Dynamics and Control (L4DC) Conference, 2022. URL `https://arxiv.org/pdf/2212.00278.pdf`.

7.1 Introduction

Motion planning of autonomous systems in dynamic environments requires the system to reason about uncertainty in its environment, e.g., a self-driving car needs to reason about uncertainty in the motion of other vehicles, and a mobile robot navigating a crowded space needs to assess uncertainty of nearby pedestrians. These applications are safety critical, as the agents' intentions are unknown, and systems must be able to plan reactive behaviors in response to an increase in uncertainty.

Existing works include predictive and reactive approaches, e.g., multi-agent navigation via the dynamic window approach [48, 94] or navigation functions [128, 129].

Reactive approaches typically consider simplified dynamics and do not optimize performance. Predictive approaches incorporate predictions of the agents' future motion and can optimize performance. Interactive approaches take inter-agent interaction into account [40, 79], while non-interactive approaches ignore potential interactions [38, 135].



In this chapter, we use trajectory predictors to predict the agents' future motion, and quantify prediction uncertainty in an adaptive and online manner from past agent observations of a single trajectory. Particularly, we use tools from the adaptive conformal prediction (ACP) literature [15, 54, 55, 146] to construct prediction regions that quantify multistep-ahead prediction uncertainty. Based on this quantification, we formulate an uncertainty-informed motion planner. Our contributions are as follows:

- We propose an algorithm that adaptively quantifies uncertainty of trajectory predictors using ACP. Our algorithm is distribution-free and applies to a broad class of trajectory predictors, providing average probabilistic coverage.
- We propose a model predictive controller (MPC) that leverages uncertainty quantifications to plan probabilistically safe paths around dynamic obstacles. Importantly, our adaptive algorithm enables us to capture and react to changes in prediction quality and the agents' motion.
- We provide empirical evaluations of a drone avoiding a flying frisbee.

Related Work

Planning in dynamic environments has found broad interest, and non-interactive sampling-based motion planner were presented in [10, 75, 91, 107, 111], while [38, 130, 141, 142] propose non-interactive receding horizon planning algorithms. However, accounting for uncertainty in the agent motion is challenging.

Intent-driven models for planning among human agents have estimated agent uncertainty using Bayesian inference [14, 46, 49, 99]. Model predictive control was also used in a stochastic setting to account for uncertainty under the assumption of bounded or Gaussian uncertainty [41, 98, 145]. Data-driven trajectory predictors can provide mean and variance information of the predictions, which can be approximated as a Gaussian distribution [27] and used within stochastic planning

frameworks [32, 50, 102]. These approaches quantify prediction uncertainty in a heuristic manner for real systems as the authors make certain assumptions on prediction algorithms and agent models and its distribution, e.g., being Gaussian. Distributionally robust approaches such as [142] are distribution free and can ensure safety at the cost of conservatism.

Data-driven trajectory predictors, such as RNNs or LSTMs, provide no information about prediction uncertainty which can lead to unsafe decisions. For this reason, prediction monitors were recently presented in [44, 90] to monitor prediction quality. Especially [90] used conformal prediction to obtain guarantees on the predictor's false negative rate. Conformal prediction was further used to obtain estimates on constraint satisfaction via neural network predictors [22, 36, 84, 108]. Conceptually closest to our work are [29, 83] where prediction uncertainty quantifications are obtained using conformal prediction, and then utilized to design model predictive controllers. While the algorithm in [29] can not provide end-to-end safety guarantees, [83] can provide probabilistic safety guarantees for the planner. However, changes in the distribution that describes the agents' motion can not be accounted for, e.g., when the agents' motion changes depending on the motion of the control system. Another distinct difference is that offline trajectory data is needed, while we obtain uncertainty quantifications in an adaptive manner from past agent observations of a single trajectory.

7.2 Problem Formulation and ACP Preliminaries

The dynamics of our autonomous system are governed by the discrete-time dynamical system,

$$x_{t+1} = f(x_t, u_t), \quad x_0 := \zeta \tag{7.1}$$

where $x_t \in \mathcal{X} \subseteq \mathbb{R}^n$ and $u_t \in \mathcal{U} \subseteq \mathbb{R}^m$ denote the state and the control input at time $t \in \mathbb{N} \cup \{0\}$, respectively. The sets $\mathcal{U}$ and $\mathcal{X}$ denote the set of permissible control inputs and the workspace of the system, respectively. The measurable function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ describes the system dynamics and $\zeta \in \mathbb{R}^n$ is the initial condition of the system. For brevity, let $x := (x_0, x_1, \ldots)$ denote the trajectory of (7.1) under a given control sequence $u := (u_0, u_1, \ldots)$.

The system operates in an environment with N dynamic agents whose trajectories are a priori unknown. Let $\mathcal{D}(x)$ be an unknown distribution over agent trajectories, i.e., let $Y := (Y_0, Y_1, \ldots) \sim \mathcal{D}(x)$ describe a random trajectory where the joint agent state $Y_t := (Y_{t,1}, \ldots, Y_{t,N})$ at times $t \in \mathbb{N} \cup \{0\}$ is drawn from $\mathbb{R}^{Nn}$, i.e., $Y_{t,j}$ is the

state of agent j at time t. For instance, Y_t can denote the uncertain two-dimensional positions of N pedestrians at time t. Modeling dynamic agents by a distribution $\mathcal{D}$ provides great flexibility, and $\mathcal{D}$ can generally describe the motion of Markov decision processes. We use lowercase letters y_t when referring to a realization of Y_t, and assume at time t to have access to past observations $(y_0, \ldots, y_t)$. We make no other assumptions on the distribution $\mathcal{D}$, and in our proposed algorithm we will predict states $(y_{t+1}, \ldots, y_{t+H})$ for a prediction horizon of H from $(y_0, \ldots, y_t)$ and quantify prediction uncertainty using ideas from ACP.

Problem 5 *Given the system in* (7.1)*, the unknown random trajectories $Y \sim \mathcal{D}(x)$, and a failure probability $\delta \in (0, 1)$, design the control inputs u_t such that the Lipschitz continuous constraint function $c : \mathbb{R}^n \times \mathbb{R}^{nN} \to \mathbb{R}$ is satisfied[1] with a probability of at least $1 - \delta$ at each time, i.e., that*

$$Prob\big(c(x_t, Y_t) \geq 0\big) \geq 1 - \delta \quad \textit{for all} \quad t \geq 0. \tag{7.2}$$

To address Problem 5, we use trajectory predictors to predict the motion of the agents $(Y_0, Y_1, \ldots)$ to enforce the constraint (7.2) within a MPC framework.[83] assumed the availability of validation data from $\mathcal{D}$ to build prediction regions that quantify uncertainty of trajectory predictors. In this setting, we can collect data online to adapt our uncertainty sets based on past performance of our predictor using ACP without any assumptions on the distribution of the uncertainty and exchangeability of the validation and training dataset.

Remark 7 *By parameterizing the distribution $\mathcal{D}(x)$ by the trajectory x, we model potential interactions between system and agents. This way, we can adapt to cases where the trajectory predictor (introduced next) is trained without information of x, i.e., without taking interactions into account.*

Trajectory Predictors:

Given observations $(y_0, \ldots, y_t)$ at time t, we want to predict future states $(y_{t+1}, \ldots, y_{t+H})$ for a prediction horizon of H. Assume that PREDICT is a function that maps observations $(y_0, \ldots, y_t)$ to predictions $(\hat{y}_t^1, \ldots, \hat{y}_t^H)$ of $(y_{t+1}, \ldots, y_{t+H})$. Note that t in $\hat{y}_t^\tau$ denotes the time at which the prediction is made, while τ indicates how many steps

[1] For an obstacle avoidance constraint, like $c(x, y) := \|x - y\| - 0.5 \geq 0$, the Lipschitz constant is 1. We implicitly assume that the constraint function is initially satisfied, i.e., that $c(x_0, y_0) \geq 0$.

we predict ahead. In principle, PREDICT can be a classical auto-regressive model or a neural network based method.

While our proposed problem solution is compatible with any trajectory predictor PREDICT, we focus in the case studies on real-time updating strategies like sliding linear predictors with extended Kalman filter. Extracting a dynamics model from data is challenging, especially when the available data is limited, noisy, and partial. [127] showed that the method of delays can be used to reconstruct qualitative features of the full-state, phase space from delayed partial observations. By building on our previous work using time delay embedding in dynamic obstacle avoidance presented in the previous chapter ([142]), we employ a linear predictor based on spatio-temporal factorization of the delayed partial observations as the pairing trajectory predictor.

Adaptive Conformal Prediction (ACP):
Conformal prediction is used to obtain prediction regions for predictive models, e.g., neural networks, without making assumptions on the underlying distribution or the predictive model [9, 119, 140]. Let $R_1, \ldots, R_{t+1}$ be $t+1$ independent and identically distributed (i.i.d.) random variables. The goal in conformal prediction is to obtain a prediction region of R_{t+1} based on $R_1, \ldots, R_t$. Formally, given a failure probability $\delta \in (0, 1)$, we want to obtain a prediction region C such that

$$\text{Prob}(R_{t+1} \leq C) \geq 1 - \delta.$$

We refer to R_i also as the nonconformity score. For supervised learning, we can select $R_i := \|Z_i - \mu(X_i)\|$ where μ is the predictor so that a large nonconformity score indicates a poor predictive model. By a quantile argument, see [131, Lemma 1], we can obtain C to be the $(1-\delta)$th quantile of the empirical distribution of the values $R_1, \ldots, R_t$ and ∞. Calculating the $(1-\delta)$th quantile can be done by assuming that $\bar{R}_1, \ldots, \bar{R}_t$ correspond to the values of $R_1, \ldots, R_t$, but instead sorted in non-decreasing order ($\bar{R}$ refers to the order statistic of R), i.e., for each $\bar{R}_i$ there exists exactly one R_j such that $\bar{R}_i = R_j$ and $\bar{R}_{i+1} \geq \bar{R}_i$. By setting $q := \lceil (t+1)(1-\delta) \rceil \leq t$, we obtain the $(1-\delta)$th quantile as $C := \bar{R}_q$, i.e., the q^{th} smallest nonconformity score.

The underlying assumption in conformal prediction is that $R_1, \ldots, R_{t+1}$ are exchangeable (exchangeability includes i.i.d. data). This is an unreasonable assumption for time-series prediction where R_t may denote the nonconformity score at time

t. To address this issue, ACP was introduced in [15, 54, 55, 146]. The idea is now to obtain a prediction region C_{t+1} adaptively so that $\text{Prob}(R_{t+1} \leq C_{t+1}) \geq 1 - \delta$ for each time t. In fact, the prediction region is now obtained as $C_{t+1} := \bar{R}_{q_{t+1}}$ where $q_{t+1} := \lceil (t+1)(1-\delta_{t+1}) \rceil$ depends on the variable δ_{t+1} that is adapted online based on observed data. In this way, the prediction region C_{t+1} becomes a tuneable parameter by the choice of δ_{t+1}. To adaptively obtain the parameter δ_{t+1}, ideas from online learning are used and we update δ_{t+1} as

$$\delta_{t+1} := \delta_t + \gamma(\delta - e_t) \text{ with } e_t := \begin{cases} 0 & \text{if } r_t \leq C_t \\ 1 & \text{otherwise} \end{cases} \tag{7.3}$$

where we denote by r_t the observed realization of R_t and where γ is a learning rate. The idea is to use δ_{t+1} to adapt to changes in the distribution of $R_1, \ldots, R_{t+1}$ over time by using information on how much the prediction region C_t overcovered ($r_t \ll C_t$) or undercovered ($r_t \gg C_t$) in the past.

Remark 8 *One of the main performance enhancers is the proper choice of γ. In [55], the authors present fully adaptive conformal prediction (FACP) where a set of learning rates $\{\gamma_i\}_{1 \leq i \leq k}$ is used in parallel from which the best γ is selected adaptively. Based on past performance (using a reweighting scheme that evaluates which γ_i provided the best coverage), the authors maintain a belief $p_t^{(i)}$ at each time step t for each $\{\delta_t^{(i)}\}_{1 \leq i \leq k}$. The new update laws are*

$$\delta_{t+1}^{(i)} := \delta_t^{(i)} + \gamma_i(\delta - e_t^{(i)}) \text{ with } e_t^{(i)} := \begin{cases} 0 & \text{if } r_t \leq C_t^{(i)} \\ 1 & \text{otherwise} \end{cases}$$

where the individual prediction regions are $C_t^{(i)} := \bar{R}_{q_t^{(i)}}$ with $q_t^{(i)} := \lceil (t+1)(1 - \delta_t^{(i)}) \rceil$, while the best prediction region is $C_t := \bar{R}_{q_t}$ with $q_t := \lceil (t+1)(1 - \sum_{i=1}^k p_t^{(i)} \delta_t^{(i)}) \rceil$.

7.3 Adaptive Conformal Prediction Regions for Trajectory Predictions

Recall that we can obtain predictions $(\hat{y}_t^1, \ldots, \hat{y}_t^H)$ at time t of future agent states $(Y_{t+1}, \ldots, Y_{t+H})$ from past observations $(y_0, \ldots, y_t)$ using the PREDICT function. Note, however, that these point predictions contain no information about prediction uncertainty and can hence not be used to reason about the safety constraint (7.2). To tackle this issue, we aim to construct prediction regions for $(Y_{t+1}, \ldots, Y_{t+H})$ using ideas from ACP.

To obtain prediction regions for $(Y_{t+1}, \ldots, Y_{t+H})$, we could consider the nonconformity score $\|Y_{t+\tau} - \hat{y}_t^\tau\|$ at time t that captures the multistep-ahead prediction error for each $\tau \in \{1, \ldots, H\}$. A large nonconformity score indicates that the prediction $\hat{y}_t^\tau$ of $Y_{t+\tau}$ is not accurate, while a small score indicates an accurate prediction. For each τ, we wish to obtain a prediction region C_t^τ that is again defined by an update variable δ_t^τ. Note, however, that we can not evaluate $\|y_{t+\tau} - \hat{y}_t^\tau\|$ at time t as only measurements $(y_0, \ldots, y_t)$ are known, but not $(y_{t+1}, \ldots, y_{t+H})$. Consequently, we cannot use the update rule (7.3) to update δ_t^τ, as the error e_t^τ would depend on checking if $\|y_{t+\tau} - \hat{y}_t^\tau\| \leq C_t^\tau$. To address this issue, we define the time lagged nonconformity score

$$R_t^\tau := \|Y_t - \hat{y}_{t-\tau}^\tau\|$$

that we can evaluate at time t so that we can use the update rule (7.3). This nonconformity score R_t^τ is time lagged in the sense that, at time t, we evaluate the τ step-ahead prediction error that was made τ time steps ago. We can now update the parameter δ_{t+1}^τ that defines C_{t+1}^τ as

$$\delta_{t+1}^\tau := \delta_t^\tau + \gamma(\delta - e_t^\tau) \text{ with } e_t^\tau := \begin{cases} 0 & \text{if } \|y_t - \hat{y}_{t-\tau}^\tau\| \leq C_t^\tau \\ 1 & \text{otherwise.} \end{cases} \tag{7.4}$$

To compute the prediction region C_{t+1}^τ, note that we can not compute $R_1^\tau, \ldots, R_{\tau-1}^\tau$. Therefore, with minor change, we let C_{t+1}^τ be the $\lceil (t - \tau + 1)(1 - \delta_{t+1}^\tau) \rceil^{\text{th}}$ smallest value of $(R_\tau^\tau, \ldots, R_t^\tau)$[2].

By obtaining a prediction region for R_{t+1}^τ using ACP, we obtain a prediction region for the τ step-ahead prediction error that was made $\tau - 1$ time steps ago, i.e., for $\|Y_{t+1} - \hat{y}_{t+1-\tau}^\tau\|$. Under the assumption that R_{t+1}^τ and $R_{t+\tau}^\tau$ are independent and identically distributed, R_{t+1}^τ serves as a prediction region for τ step-ahead prediction error that was made 0 time steps ago (now at time t), i.e., for $R_{t+\tau}^\tau$ which encodes $\|Y_{t+\tau} - \hat{y}_t^\tau\|$. Naturally, in our setting R_{t+1}^τ and $R_{t+\tau}^\tau$ are not independent and identically distributed, but it still serves as a good measure for the prediction region $R_{t+\tau}^\tau$. We remark that for the theoretical guarantees that we provide in the next section, only the one step-ahead prediction errors are relevant.

[2]Instead of keeping track of all data, we will choose a sliding window of the N most recent data. For all prediction regions, we will then consider $(R_{t-N}^\tau, \ldots, R_t^\tau)$ and compute C_{t+1}^τ as the $\lceil (N + 1)(1 - \delta_{t+1}^\tau) \rceil^{\text{th}}$ smallest value.

Theorem 11 *Let γ be a learning rate, $\delta_0^1 \in (0,1)$ be an initial value for the recursion* (7.4)*, and T be the number of times that we compute the recursion* (7.4)*. Then, for the onestep-ahead prediction errors, it holds that*

$$1-\delta-p_1 \leq \frac{1}{T}\sum_{t=0}^{T-1} Prob(\|Y_{t+1}-\hat{y}_t^1\| \leq C_{t+1}^1) \leq 1-\delta+p_2 \tag{7.5}$$

with constants $p_1 := \frac{\delta_0^1+\gamma}{T\gamma}$, $p_2 := \frac{(1-\delta_0^1)+\gamma}{T\gamma}$ so that $\lim_{T\to\infty} p_1 = 0$ and $\lim_{T\to\infty} p_2 = 0$.

Proof: Since the probability of an event is equivalent to the expected value of the indicator function of that event, it follows by the definition of the error e_{t+1}^1 that

$$\text{Prob}(\|Y_{t+1}-\hat{y}_t^1\| \leq C_{t+1}^1) = \mathbb{E}[1-e_{t+1}^1] = 1-\mathbb{E}[e_{t+1}^1]. \tag{7.6}$$

For a given initialization δ_0^τ and learning rate γ, we know from [54, Proposition 4.1] that the following bound holds (with probability one) for the misclassification errors

$$\frac{-(1-\delta_0^1)+\gamma}{T\gamma} \leq \frac{1}{T}\sum_{t=0}^{T-1} e_{t+1}^1 - \delta \leq \frac{\delta_0^1+\gamma}{T\gamma} \implies \left|\frac{1}{T}\sum_{t=0}^{T-1} e_{t+1}^1 - \quad \delta\right| \leq \frac{\max(\delta_0^1, 1-\delta_0^1)+\gamma}{T\gamma}.$$

Hence, taking the expectation of the above two-sided inequality, we get that

$$\begin{aligned}
&\frac{-(1-\delta_0^1)+\gamma}{T\gamma} \leq \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[e_{t+1}^1] - \delta \leq \frac{\delta_0^1+\gamma}{T\gamma},\\
\overset{(a)}{\Leftrightarrow}\quad &\frac{-(1-\delta_0^1)+\gamma}{T\gamma} \leq \frac{1}{T}\sum_{t=0}^{T-1}\left(1-\text{Prob}(\|Y_{t+1}-\hat{y}_t^1\| \leq C_{t+1}^1)\right) - \delta \leq \frac{\delta_0^1+\gamma}{T\gamma},\\
\Leftrightarrow\quad &1-\delta+\frac{(1-\delta_0^1)+\gamma}{T\gamma} \geq \frac{1}{T}\sum_{t=0}^{T-1}\text{Prob}(\|Y_{t+1}-\hat{y}_t^1\| \leq C_{t+1}^1) \geq 1-\delta-\frac{\delta_0^1+\gamma}{T\gamma},
\end{aligned}$$

where we used equation (7.6) for the equivalence in (a). ■

Remark 9 *The above result can be similarly extended to the FACP case with a set of candidate learning rates, γ, [55, Theorem 3.2].*

Example 1 *To illustrate these multistep-ahead prediction regions, consider a planar double pendulum whose dynamics are governed by chaotic, nonlinear dynamics that are sensitive to the initial condition [122]. We study the predictions made by a linear predictor that uses noisy observations of the position*

of the double pendulum and use ACP to predict the uncertainty in the predictions. Both the trajectory predictor and the uncertainty quantification using ACP use online data from a single trajectory. ACP provides the multi-step errors in the linear predictions with a coverage level of $\delta = 0.1$*, and learning rates* $\gamma = \begin{pmatrix} 0.0008 & 0.0015 & 0.003 & 0.005 & 0.009 & 0.017 & 0.03 & 0.05 & 0.08 \end{pmatrix}$.

Figure 7.1 compares the 1-step and 6-step ahead error prediction regions to the true multi-step errors for two states, the second mass position, x_2, y_2*. The percentages of one-step errors that are incorrectly predicted, i.e.,* $e_t^1 = 1$*, for the positions of each mass,* x_1, x_2, y_1, y_2 *are* $2.36\%, 0.94\%, 1.57\%, 1.73\%$ *respectively. We can see the effects of adaptation as the ACP prediction regions are larger in areas of poor performance of the linear predictor (and consequently higher error in the prediction) and smaller in regions where the linear predictor performs well.*

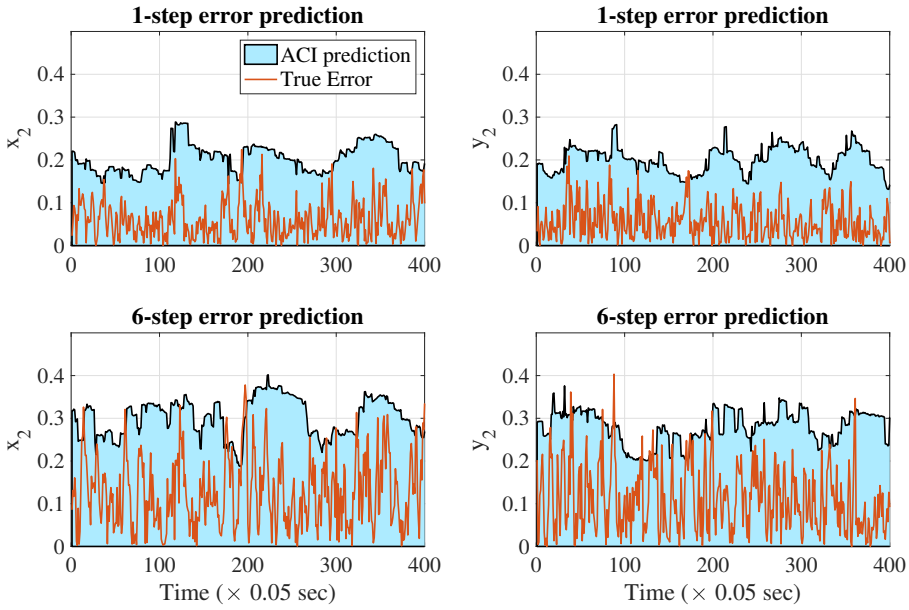


Figure 7.1: The multi-step prediction errors are shown for two of the six states of a double pendulum (x_2, y_2). ACP can correctly predict regions of high and low error (90% coverage regions) by adjusting the prediction quantile using update law (7.3). The orange lines are the true multi-step prediction errors and the blue areas are the error regions predicted by ACP.

7.4 Uncertainty-Informed Model Predictive Control

Based on the obtained uncertainty quantification from the previous section, we propose an uncertainty-informed model predictive controller (MPC) that uses predictions $\hat{y}_t^\tau$ and adaptive prediction regions C_{t+1}^τ. The underlying optimization

problem that is solved at every time step t is:

$$\min_{(u_t,\ldots,u_{t+H-1})} \sum_{k=t}^{t+H-1} J(x_{k+1}, u_k) \tag{7.7a}$$

$$\text{s.t.} \quad x_{k+1} = f(x_k, u_k), \qquad k \in \{t, \ldots, t+H-1\} \tag{7.7b}$$

$$c(x_{t+\tau}, \hat{y}_t^\tau) \geq LC_{t+1}^\tau, \qquad \tau \in \{1, \ldots, H\} \tag{7.7c}$$

$$u_k \in \mathcal{U}, x_{k+1} \in \mathcal{X}, \qquad k \in \{t, \ldots, t+H-1\} \tag{7.7d}$$

where L is the Lipschitz constant of the constraint function c, J is a step-wise cost function, and $u_t, \ldots, u_{t+H-1}$ is the control sequence. The optimization problem in (7.7) is convex if the functions J and f are convex, the function c is convex in its first argument, and the sets $\mathcal{U}$ and $\mathcal{X}$ are convex.

Based on this optimization problem, we propose a receding horizon control strategy in Algorithm 5. In line 1 of Algorithm 5, we initialize the parameter δ_0^t simply to δ. Lines 2-11 present the real-time planning loop by: 1) updating the states x_t and y_t and calculating new predictions $\hat{y}_t^\tau$ (lines 3-4), 2) computing the adaptive nonconformity scores C_{t+1}^τ (lines 5-9), and 3) solving the optimization problem in (7.7) of which we apply only u_t (lines 10-11).

Algorithm 5 MPC with ACP Regions

1: **Input:** Failure probability δ, prediction horizon H, learning rate γ
2: **Output:** Control input $u_t(x_t, y_0, \ldots, y_t)$ at each time t
3: $\delta_0^\tau \leftarrow \delta$ for $\tau \in \{1, \ldots, H\}$
4: **for** t from 0 to ∞ # real-time motion planning loop **do**
5: Update x_t and y_t
6: Obtain predictions $\hat{y}_t^\tau$ for $\tau \in \{1, \ldots, H\}$
7: **for** τ from 1 to H # compute ACP regions **do**
8: $\delta_{t+1}^\tau \leftarrow \delta_t^\tau + \gamma(\delta - e_t^\tau)$
9: $R_t^\tau := \|y_t - \hat{y}_{t-\tau}^\tau\|$
10: $q \leftarrow \lceil (t+1)(1-\delta_{t+1}^\tau) \rceil$
11: Set C_{t+1}^τ as the qth smallest value of $(R_\tau^\tau, \ldots, R_t^\tau)$
12: **end for**
13: Calculate controls $u_t, ..., u_{t+H-1}$ as the solution of (7.7)
14: Apply u_t to (7.1)
15: **end for**

Remark 10 *While Algorithm 5 uses a single learning rate, one can similarly extend the above algorithm to be fully adaptive using a candidate set of* $\{\gamma_i\}_{1 \leq i \leq k}$ *without loss of generality.*

Remark 11 *[54] assume that when $\delta_{t+1} \leq 0$, the prediction region $C_{t+1} \to \infty$. This means that when the algorithm requires robust behavior, the ∞-prediction region ensures that any prediction at the next time-step should be correctly classified. For a physical system, there are limits on how much the dynamic obstacle can accelerate in one time-step which gives us an upper bound $R_{max} < \infty$ on the worst-case error. In practice, we enforce $0 \leq \delta_{t+1} \leq 1$ with $C_{t+1} \leq R_{max}$.*

Theorem 12 *Let γ be a learning rate, $\delta_0^1 \in (0,1)$ be an initial value for the recursion* (7.4)*, and T be the number of times that we compute the recursion* (7.4)*. If the optimization problem* (7.7) *in Algorithm 5 is recursively feasible, then Algorithm 5 will lead to*

$$\frac{1}{T}\sum_{t=0}^{T-1} Prob\big(c(x_{t+1}, Y_{t+1}) \geq 0\big) \geq 1 - \delta - p_1 \tag{7.8}$$

with constant $p_1 := \frac{\delta_0^1 + \gamma}{T\gamma}$ so that $\lim_{T\to\infty} p_1 = 0$.

Proof: By assumption, the optimization problem in (7.7) is feasible at each time $t \in \{0, 1, \ldots\}$. Due to constraint (7.7c) and Lipschitz continuity of c, it hence holds that

$$0 \leq c(x_{t+1}, \hat{y}_t^1) - LC_{t+1}^1 \leq c(x_{t+1}, Y_{t+1}) + L\|Y_{t+1} - \hat{y}_t^1\| - LC_{t+1}^1 \tag{7.9}$$

at each time $t \in \{0, 1, \ldots\}$. Consequently, note that $\|Y_{t+1} - \hat{y}_t^1\| \leq C_{t+1}^1$ is a sufficient condition for $c(x_{t+1}, Y_{t+1}) \geq 0$. In a next step, we can derive that

$$\begin{aligned}
\mathrm{Prob}\big(c(x_{t+1}, Y_{t+1}) \geq 0\big) &\overset{(a)}{=} \mathrm{Prob}\big(c(x_{t+1}, Y_{t+1}) \geq 0 \,\big|\, \|Y_{t+1} - \hat{y}_t^1\| \leq C_t^1\big)\mathrm{Prob}(\|Y_{t+1} - \hat{y}_t^1\| \leq C_t^1) \\
&\quad + \mathrm{Prob}\big(c(x_{t+1}, Y_{t+1}) \geq 0 \,\big|\, \|Y_{t+1} - \hat{y}_t^1\| > C_t^1\big)\mathrm{Prob}(\|Y_{t+1} - \hat{y}_t^1\| > C_t^1) \\
&\overset{(b)}{\geq} \mathrm{Prob}\big(c(x_{t+1}, Y_{t+1}) \geq 0 \,\big|\, \|Y_{t+1} - \hat{y}_t^1\| \leq C_t^1\big)\mathrm{Prob}(\|Y_{t+1} - \hat{y}_t^1\| \leq C_t^1) \\
&\overset{(c)}{=} \mathrm{Prob}(\|Y_{t+1} - \hat{y}_t^1\| \leq C_t^1)
\end{aligned}$$

where the equality in (a) follows from the law of total probability, while the inequality in (b) follows from the nonnegativity of probabilities. The equality in (c) follows as $\mathrm{Prob}(c(x_{t+1}, Y_{t+1}) \geq 0 \,|\, \|Y_{t+1} - \hat{y}_t^1\| \leq C_t^1) = 1$ since $\|Y_{t+1} - \hat{y}_t^1\| \leq C_t^1$ implies $c(x_{t+1}, Y_{t+1}) \geq 0$ according to (7.9). We now use the result from Theorem 11 to complete the proof. ∎

7.5 Case Studies: Multirotor operating in small angle regime dodging a flying frisbee

We compare the performance of the MPC with ACP uncertainty prediction regions with our previous work that uses a distributionally robust approach to uncertainty quantification [142]. We use the same multirotor operating in the presence of a moving obstacle example with a MPC planner. The multirotor is constrained to operate within the state constraints $\theta \in [-0.45, 0.45]$ radians and $\varphi \in [-0.45, 0.45]$ radians. We use the following standard multirotor linear dynamics,

$$\ddot{x} = -g\theta,\ \ddot{y} = g\varphi,\ \ddot{z} = u_1 - g,\ \ddot{\varphi} = \frac{u_2}{I_{xx}},\ \ddot{\theta} = \frac{u_3}{I_{yy}},\ \ddot{\psi} = \frac{u_4}{I_{zz}}, \tag{7.10}$$

where the planner control inputs u_1, u_2, u_3, u_4 correspond to the thrust force in the body frame and three moments. The vehicle's moments of inertia are $I_{xx} = 0.0075kgm^2, I_{yy} = 0.0075kgm^2, I_{zz} = 0.013kgm^2$. The MPC planner has a horizon length of 10 steps and the planner is updated at 20 Hz. It is implemented through a Sequential Convex Programming approach [95].

Numerical simulations of the proposed MPC planner with ACP regions and dynamics (7.10) are presented as it avoids a Frisbee that is thrown at the drone from various initial positions, velocities, and rotation speed. The Frisbee is modeled following [68], and we implement linear predictions of the trajectory arising from its nonlinear dynamics.

We conducted 1000 Monte Carlo simulations per allowed failure probability level δ to compare the numerical feasibility, percentage of success in obstacle avoidance (if the MPC planner is feasible), and the planner's conservativeness, as measured by the minimum distance between the obstacle and agent centers, i.e., $\bar{d}_{min}$ and $\sigma(d_{min})$ describe the average and standard deviation of this minimum distance across simulations, respectively. We compare three uncertainty quantification techniques in Table 7.1, (1) The proposed ACP method (Algorithm 5), (2) empirical bootstrap prediction that accounts for the uncertainty in the predictions using the empirical bootstrap variance [142], and (3) the sliding linear predictor with an Extended Kalman Filter (EKF) that approximates the uncertainty in the obstacle predictions as a Gaussian distribution.

Discussion: Table 7.1 shows that our proposed method can successfully avoid the Frisbee, while using a significantly smaller average divergence distance (d_{min}, $\sigma(d_{min})$) from the Frisbee. I.e., our approach avoids the conservatism of other approaches due to the adaptivity of the uncertainty sets. Our method can usefully adjust the

Case	δ	0.025			0.05		
	UQ method	Proposed	[142]	w/EKF	Proposed	[142]	w/EKF
Frisbee w/drag	%Feas.	83.8	87.4	97.1	80.9	90.3	97.6
	%Succ.	99.2	100	100	100	100	100
	$\overline{d}_{min}$	2.91	14.2	5.27	2.74	4.97	4.25
	$\sigma(d_{min})$	1.25	2.04	1.28	1.3	1.97	1.11

Table 7.1: Summary of results from MC simulations of system (7.10). We used FACP for predicting uncertainty sets with learning rates $\gamma = \{0.0008, 0.0015, 0.003, 0.005, 0.009, 0.017, 0.03, 0.05, 0.08, 0.13\}$ and using the last 30 measurements of the obstacle.

prediction sets when the underlying uncertainty distribution is shifting (due to discrepancy in the linear dynamic predicted and the true nonlinear obstacle motion). We also note that the feasibility of the MPC optimization is worse for our method compared to [142] and the EKF predictor. This issue arises during sudden changes in the size of the uncertainty sets when the learning rate γ is chosen too large. We will investigate this issue in future work by considering tools to ensure recursive feasibility [65] or by providing backup controllers [124, 133] when the MPC is infeasible.

Chapter 8

CONCLUSION

This thesis considered ways to intelligently quantify and incorporate the uncertainty in sensing unstructured environments into motion planning algorithms for autonomous robots. Towards this end, my work developed data-driven uncertainty quantification techniques in increasingly complex environments and developed a theoretical framework to account for this uncertainty in a principled manner.

State-of-the-art motion planning techniques make many simplifying assumptions on what the nature of the uncertainty affecting the robotic system looks like. For example - we can assume the process noise is bounded and that this bound is known. In the real world, this robust bound may be too conservative and result in overly cautious robot motion that does not leverage the true range of motion autonomous robots can have. Another example of a simplifying assumption in designing controllers for autonomous systems is that agents move in the environment with a Gaussian uncertainty behavior. This assumption is too optimistic and can result in failures for safety-critical systems when the real-world is unstructured and does not have Gaussian behavior. In this thesis, I looked at risk-aware methodologies that don't make simplifying assumptions on the environment or the robot models, and hence provide agile behaviors without sacrificing safety. These risk-averse strategies were demonstrated on wheeled and legged ground robots and on drones.

Chapter 3 considered traversability risk, and how it can be incorporated into the geometric planning and a model predictive control planning framework. This risk incorporated information about not only which parts of the unknown environments were obstacles, but also about which areas of the environment are "riskier" to travel over than others. We considered geometric, semantic, confidence-based traversability risk sources. This planning framework was deployed in the DARPA Subterranean Challenge and verified with extensive field experiments.

Chapter 4 considered distributionally robust chance constraints instead of risk constraints, and reformulated the Total Variation Distance-based DRCC using CVaR. We compared this method against risk-aware MPC without DRCCs as well as Stochastic MPC, and showed that the use of DRCCs provides robustness to the variation in the uncertainty distribution not just in the cost but also in the constraints. The

computational complexity of this reformulation is comparable to that of Stochastic MPC while providing more robustness.

We then considered risk-aware MPC in an environment with dynamic obstacles. First, in Chapter 5, we assumed that the motion of the dynamic obstacles and the uncertainty associated with their movement was known apriori. We showed that traditional stochastic and robust MPC frameworks can be generalized into a single framework with coherent risk measure-based cost and constraints. This framework provides a convex, mixed-integer optimization reformulation of the MPC problem that can account for a large class of coherent risk measures within the cost and constraints of the MPC.

The approaches presented in Chapters 4 and 5 assumed that in the description of the ambiguity set, the nominal uncertainty is given a priori and that the risk-level, or the size of the ambiguity set, is also known. In Chapters 6 and 7, these assumptions on a prioir knowledge are relaxed, while still maintaining a risk-averse result. In Chapter 6, the trajectory of the dynamic obstacle or its uncertainty distribution was not assumed to be known a priori. We provide a technique to make data-driven predictions of the moving obstacles using Singular Spectrum Analysis. Given these predictions, we quantify their uncertainty using a Bootstrapping technique. We use the bootstrapped predictions in the MPC optimization to provide a distributionally robust reformulation of the obstacle avoidance safety constraint.

Chapter 7 further provides a calibration technique to adjust the size of the uncertainty set of the moving obstacle predictions using Adaptive Conformal Predictions. This technique dynamically quantifies prediction uncertainty from an online data stream and provide an uncertainty-informed model predictive controller to safely navigate in the presence of dynamic obstacless. In contrast to other the data-driven prediction models that quantify prediction uncertainty in a heuristic manner, like in Chapter 6, the true prediction uncertainty is quantified in a distribution-free, adaptive manner that even allows to capture changes in prediction quality and the obstacle's motion.

This work is just a first step towards providing safety without sacrificing performance in unstructured environments. My work so far allows for a rich understanding of a static environment and allows for prediction of the trajectories of dynamic obstacles. We saw that data-driven models of the environment will encounter distribution shifts when deployed in an unknown, dynamic environment and that planning algorithms must be able to adapt to these distribution shifts in an online setting. Some examples of this distribution shift include a pedestrian at a crosswalk reacting to a speeding

autonomous vehicle by suddenly stopping or running across the crosswalk based on where they are on the crosswalk.

The usual assumption of identically and independently distributed (i.i.d.) data no longer holds and the trajectory data we collect online in a dynamic environment is nonstationary. The following question arises: can we provide certificates to detect when the learned, data-driven, environment models might fail? In an environment with distribution shifts, we can reduce the number of predicted failures by incorporating distributional robustness in the learning framework. This can be achieved by constructing sets of distributions online, using dependent data, that contain the true underlying distribution that describes the real world and allowing the risk-level (or the size of the set of distributions) to change dynamically based on the environment.

Furthermore, in order to obtain risk-averse behavior for autonomous systems that interact with other moving agents and even humans, we must be able incorporate the dynamic ambiguity sets within a planning framework, while retaining performance and providing safety guarantees. For example - is it possible to estimate the risk-level that is acceptable to different users of an autonomous taxi in order to give them the best driving experience? Given the goal of developing controllers that can adapt in increasingly dynamic environments with distribution shifts, investigating the modeling of multi-agent interactions with the goal of incorporating risk-aware prediction and planning into a game-theoretic framework is an exciting avenue of future work. These advances will help us develop intelligent systems that can interact with and convey intent to other dynamic agents in the real-world.

Beyond the risk-aware control and uncertainty quantification discussed in this thesis, it will be important to also consider risk-aware high-level planning methodologies for robotic applications. In [6], we took a first step towards addressing this by considering the stochastic shortest path planning problem in Markov Decision Processes (MDPs). In order to account for rare but important realizations of the system, we consider a nested dynamic coherent risk total cost functional rather than the conventional risk-neutral total expected cost. Under some assumptions, we show that optimal, stationary, Markovian policies exist and can be found via a special Bellman's equation. Future research will explore risk-averse polices for POMDPs that maximize the satisfaction probability of a set of high-level mission specifications in terms of temporal logic formulae [4, 121].

While there exist many control synthesis techniques for risk-sensitive settings, there are few verification techniques - especially for arbitrary risk measures - that account

for unstructured uncertainty. The inability to obtain risk-aware verification through the same risk measures used for policy development stems primarily from the inability to calculate these measures for unknown probability distributions. We develop sample-based bounds for arbitrary (coherent) risk measures using ideas from scenario optimization and provide a fundamental requirement on the number of samples required to generate our bounds and use these bounds to provide high confidence statements on system performance in a risk-aware setting in [3]. Next steps involve closely analyzing the tightness of these bounds and their applicability controller synthesis that works well not only in simulation but also on hardware.

From richer uncertainty quantification to a deeper look at risk-aware control, planning, and verification, I have listed just a few avenues of future work in the area of risk-aware autonomy above. I think there are many exciting questions yet to be answered before safety-critical autonomous systems can attain human-like behavior with performance and generalization guarantees.

Bibliography

[1] A. Agha-mohammadi, et al. NeBula: Quest for robotic autonomy in challenging environments; TEAM CoSTAR at the DARPA Subterranean Challenge. Field Robotics, 2021.

[2] H. Adbi and L.J. Williams. Principal component analysis. Wiley interdisc. reviews: comp. statistics, 2(4):433–459, 2010.

[3] A. Agarwal, A. Alomar, and D. Shah. On multivariate singular spectrum analysis. arXiv preprint arXiv:2006.13448, 06 2020.

[4] M. Ahmadi, R. Sharan, and J. W. Burdick. Stochastic finite state control of POMDPs with LTL specifications. arXiv preprint arXiv:2001.07679, 2020.

[5] M. Ahmadi, X. Xiong, and A. D. Ames. Risk-averse control via CVaR barrier functions: Application to bipedal robot locomotion. IEEE Control Systems Letters, 6:878–883, 2021.

[6] Mohamadreza Ahmadi, Anushri Dixit, Joel W. Burdick, and Aaron D. Ames. Risk-averse stochastic shortest path planning. arXiv:2103.14727, 2021.

[7] A. Ahmadi-Javid. Entropic value-at-risk: A new coherent risk measure. Journal of Optimization Theory and Applications, 155(3):1105–1123, 2012.

[8] 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. doi: 10.1109/TAC.2016.2638961.

[9] Anastasios N Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. arXiv preprint arXiv:2107.07511, 2021.

[10] Georges S Aoude, Brandon D Luders, Joshua M Joseph, Nicholas Roy, and Jonathan P How. Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns. Autonomous Robots, 35(1):51–76, 2013.

[11] MOSEK ApS. The MOSEK optimization toolbox for MATLAB manual. Version 9.0., 2019. URL `http://docs.mosek.com/9.0/toolbox/index.html`.

[12] P. Artzner, F. Delbaen, Jean-Marc Eber, and D. Heath. Coherent measures of risk. Mathematical finance, 9(3):203–228, 1999.

[13] F. Augugliaro, A.P Schoellig, and R. D'Andrea. Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach. Int. Conf Intell. Robots Syst., 2012.

[14] Somil Bansal, Andrea Bajcsy, Ellis Ratner, Anca D Dragan, and Claire J Tomlin. A hamilton-jacobi reachability-based framework for predicting and analyzing human motion for safe planning. In 2020 IEEE International Conference on Robotics and Automation (ICRA), pages 7149–7155. IEEE, 2020.

[15] Osbert Bastani, Varun Gupta, Christopher Jung, Georgy Noarov, Ramya Ramalingam, and Aaron Roth. Practical adversarial multivalid conformal prediction. arXiv preprint arXiv:2206.01067, 2022.

[16] Alberto Bemporad and Manfred Morari. Robust model predictive control: A survey. In Robustness in identification and control, pages 207–226. Springer, 1999.

[17] Dimitri Bertsekas. Convex optimization theory, volume 1. Athena Scientific, 2009.

[18] L. Blackmore, M. Ono, A. Bektassov, and B.C. Williams. A probabilistic particle-control approximation of chance-constrained stochastic predictive control. IEEE Trans. Robotics, 26(3):502–517, 2010. doi: 10.1109/TRO.2010.2044948.

[19] Lars Blackmore, Masahiro Ono, and Brian C. Williams. Chance-constrained optimal path planning with obstacles. IEEE Transactions on Robotics, 27(6): 1080–1094, 2011. doi: 10.1109/TRO.2011.2161160.

[20] Paul T Boggs and Jon W Tolle. Sequential quadratic programming. Acta numerica, pages 529–562, 1995.

[21] Francesco Borrelli. Constrained optimal control of linear and hybrid systems, volume 290. Springer, 2003.

[22] Luca Bortolussi, Francesca Cairoli, Nicola Paoletti, Scott A Smolka, and Scott D Stoller. Neural predictive monitoring. In International Conference on Runtime Verification, pages 129–147. Springer, 2019.

[23] Amanda Bouman, Muhammad Fadhil Ginting, Nikhilesh Alatur, Matteo Palieri, David D Fan, Thomas Touma, Torkom Pailevanian, Sung-Kyun Kim, Kyohei Otsu, Joel Burdick, et al. Autonomous Spot: Long-Range Autonomous Exploration of Extreme Environments with Legged Locomotion. arXiv preprint arXiv:2010.09259, 2020.

[24] S. Boyd and L. Vandenberghe. Convex optimization. Cambridge university press, 2004.

[25] Stephen Boyd and Lieven Vandenberghe. Convex Optimization. Cambridge University Press, USA, 2004. ISBN 0521833787.

[26] S.L. Brunton, B.W. Brunton, J.L. Proctor, E. Kaiser, and J. Kutz. Chaos as an intermittently forced linear system. Nat. commun., 2017.

[27] Finn Lukas Busch, Jake Johnson, Edward L. Zhu, and Francesco Borrelli. A gaussian process model for opponent prediction in autonomous racing, 2022. URL https://arxiv.org/abs/2204.12533.

[28] J. Canny. The complexity of robot motion planning. MIT press, 1988.

[29] Yuxiao Chen, Ugo Rosolia, Chuchu Fan, Aaron D Ames, and Richard Murray. Reactive motion planning with probabilistic safety guarantees. arXiv preprint arXiv:2011.03590, 2020.

[30] Yuxiao Chen, Jip Kim, and James Anderson. Distributionally robust decision making leveraging conditional distributions. arXiv preprint arXiv:2204.00138, 2022.

[31] Yuxiao Chen, Ugo Rosolia, Wyatt Ubellacker, Noel Csomay-Shanklin, and Aaron D Ames. Interactive multi-modal motion planning with branch model predictive control. IEEE Robotics and Automation Letters, 7(2):5365–5372, 2022.

[32] Sungjoon Choi, Eunwoo Kim, Kyungjae Lee, and Songhwai Oh. Real-time nonparametric reactive navigation of mobile robots in dynamic environments. Robotics and Autonomous Systems, 91:11–24, 2017.

[33] M. Christoph and S. Liu. Stochastic mpc with distributionally robust chance constraints. IFAC-PapersOnLine, 53(2):7136–7141, 2020. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2020.12.521. URL https://www.sciencedirect.com/science/article/pii/S2405896320308181.

[34] Shreyansh Daftry, Neil Abcouwer, Tyler Del Sesto, Siddarth Venkatraman, Jialin Song, Lucas Igel, Amos Byon, Ugo Rosolia, Yisong Yue, and Masahiro Ono. Mlnav: Learning to safely navigate on martian terrains. IEEE Robotics and Automation Letters, 7(2):5461–5468, 2022. doi: 10.1109/LRA.2022.3156654.

[35] Sharmita Dey, David Fan, Robin Schmid, Anushri Dixit, Kyohei Otsu, Thomas Touma, Arndt F Schilling, and Ali-akbar Agha-mohammadi. Prepare: Predictive proprioception for agile failure event detection in robotic exploration of extreme terrains. arXiv preprint arXiv:2208.00322, 2022.

[36] Thomas G Dietterich and Jesse Hostetler. Conformal prediction intervals for markov decision process trajectories. arXiv preprint arXiv:2206.04860, 2022.

[37] Anushri Dixit, Mohamadreza Ahmadi, and Joel W. Burdick. Risk-sensitive motion planning using entropic value-at-risk. In European Control Conference, 2021.

[38] Noel E Du Toit and Joel W Burdick. Robot motion planning in dynamic, uncertain environments. IEEE Transactions on Robotics, 28(1):101–115, 2011.

[39] K Ebadi, Y Chang, M Palieri, A Stephens, A Hatteland, E Heiden, A Thakur, B Morrell, L Carlone, and A Agha-mohammadi. LAMP: Large-scale autonomous mapping and positioning for exploration of perceptually-degraded subterranean environments. In IEEE International Conference on Robotics and Automation, pages 80–86, 2020.

[40] Michael Everett, Yu Fan Chen, and Jonathan P How. Collision avoidance in pedestrian-rich environments with deep reinforcement learning. IEEE Access, 9:10357–10377, 2021.

[41] David D Fan, Kyohei Otsu, Yuki Kubo, Anushri Dixit, Joel Burdick, and Ali-Akbar Agha-Mohammadi. Step: Stochastic traversability evaluation and planning for risk-aware off-road navigation. In Robotics: Science and Systems, pages 1–21. RSS Foundation, 2021.

[42] Péter Fankhauser and Marco Hutter. A universal grid map library: Implementation and use case for rough terrain navigation. Robot Operating System (ROS), The Complete Reference, pages 99–120, 2016.

[43] Péter Fankhauser, Michael Bloesch, and Marco Hutter. Probabilistic terrain mapping for mobile robots with uncertain localization. IEEE Robotics and Automation Letters, 3(4):3019–3026, 2018.

[44] Alec Farid, Sushant Veer, Boris Ivanovic, Karen Leung, and Marco Pavone. Task-relevant failure detection for trajectory predictors in autonomous vehicles. arXiv preprint arXiv:2207.12380, 2022.

[45] Urban Fasel, J Nathan Kutz, Bingni W Brunton, and Steven L Brunton. Ensemble-sindy: Robust sparse model discovery in the low-data, high-noise limit, with active learning and control. Proceedings of the Royal Society A, 478(2260):20210904, 2022.

[46] Jaime F Fisac, Andrea Bajcsy, Sylvia L Herbert, David Fridovich-Keil, Steven Wang, Claire J Tomlin, and Anca D Dragan. Probabilistically safe robot planning with confidence-based human predictions. In 14th Robotics: Science and Systems, RSS 2018. MIT Press Journals, 2018.

[47] R. Foust, S.J. Chung, and F.Y. Hadaegh. Optimal guidance and control with nonlinear dynamics using sequential convex programming. J. of Guid., Control, and Dyn., 43(4):633–644, 2020.

[48] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. IEEE Robotics & Automation Magazine, 4 (1):23–33, 1997.

[49] 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 avoidance1. The International Journal of Robotics Research, 39(2-3):250–265, 2020.

[50] Chiara Fulgenzi, Christopher Tay, Anne Spalanzani, and Christian Laugier. Probabilistic navigation in dynamic environment using rapidly-exploring random trees and gaussian processes. In 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 1056–1062. IEEE, 2008.

[51] L. Ghaoui, M. Oks, and F. Oustry. Worst-case value-at-risk and robust portfolio optimization: A conic programming approach. Operations Research, 51:543–556, 08 2003. doi: 10.1287/opre.51.4.543.16101.

[52] S. Ghosh, K. Otsu, and M. Ono. Probabilistic kinematic state estimation for motion planning of planetary rovers. In IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 5148–5154, 2018.

[53] Alison L. Gibbs and Francis Edward Su. On choosing and bounding probability metrics. INTERNAT. STATIST. REV., pages 419–435, 2002.

[54] Isaac Gibbs and Emmanuel Candes. Adaptive conformal inference under distribution shift. Advances in Neural Information Processing Systems, 34: 1660–1672, 2021.

[55] Isaac Gibbs and Emmanuel Candès. Conformal inference for online prediction with arbitrary distribution shifts. arXiv preprint arXiv:2208.08401, 2022.

[56] Steven B Goldberg, Mark W Maimone, and Larry Matthies. Stereo vision and rover navigation software for planetary exploration. In IEEE Aerospace Conference. IEEE, 2002.

[57] N.E. Goljandina and A.A. Zigljavskij. Singular spectrum analysis for time series. Springer, 2020.

[58] N. Golyandina and A. Korobeynikov. Basic singular spectrum analysis and forecasting with r. Comp. Stat. & Data Anal.

[59] N. Golyandina, V. Nekrutkin, and A.A. Zhigljavsky. Analysis of time series structure - ssa and related techniques. In Monographs on statistics and applied probability, 2001.

[60] Gurobi Optimization, LLC. Gurobi Optimizer Ref. Manual, 2022. URL `https://www.gurobi.com`.

[61] Alain Haït, Thierry Simeon, and Michel Taïx. Algorithms for rough terrain trajectory planning. Advanced Robotics, 16(8):673–699, 2002.

[62] A. Hakobyan, G. C. Kim, and I. Yang. Risk-aware motion planning and control using cvar-constrained optimization. IEEE Robotics and Automation Letters, 4(4):3924–3931, 2019.

[63] 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.

[64] Weiqiao Han, Ashkan Jasour, and Brian Williams. Non-gaussian risk bounded trajectory optimization for stochastic nonlinear systems in uncertain environments. arXiv preprint arXiv:2203.03038, 2022.

[65] Lukas Hewing, Kim P. Wabersich, and Melanie N. Zeilinger. Recursively feasible stochastic model predictive control using indirect feedback. Automatica, 119:109095, 2020. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2020.109095. URL https://www.sciencedirect.com/science/article/pii/S0005109820302934.

[66] Michael Himmelsbach, Felix V Hundelshausen, and H-J Wuensche. Fast segmentation of 3d point clouds for ground vehicles. In Intelligent Vehicles Symposium (IV), 2010 IEEE, pages 560–565. IEEE, 2010.

[67] Patrick D. Holmes, Shreyas Kousik, Bohao Zhang, Daphna Raz, Corina Barbalata, Matthew Johnson-Roberson, and Ram Vasudevan. Reachable sets for safe, real-time manipulator trajectory design. ArXiv, abs/2002.01591, 2020.

[68] S.A. Hummel. Frisbee flight simulation and throw biomechanics. University of California, Davis, 2003.

[69] I. Hwang and C.E. Seah. Intent-based probabilistic conflict detection for the next generation air transportation system. Proceedings of the IEEE, 96(12): 2040–2059, 2008. doi: 10.1109/JPROC.2008.2006138.

[70] A Jasour. Risk aware and robust nonlinear planning (rarnop). Course Notes for MIT, 16:S498, 2019.

[71] Ashkan M Jasour and Brian C Williams. Risk contours map for risk bounded motion planning under perception uncertainties. Robotics: Science and Systems.

[72] J.N. Juang and R.S. Pappa. An eigensystem realization algorithm for modal parameter identification and model reduction. J. Guidance, Control, and Dynamics, 8(5):620–627, 1985.

[73] Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, Peter Pastor, and Stefan Schaal. STOMP: Stochastic trajectory optimization for motion planning. In IEEE International Conference on Robotics and Automation, pages 4569–4574, 2011.

[74] Himangshu Kalita, Steven Morad, Aaditya Ravindran, and Jekan Thangavelautham. Path planning and navigation inside off-world lava tubes and caves. In IEEE/ION Position, Location and Navigation Symposium, pages 1311–1318, 2018.

[75] Samarth Kalluraya, George J. Pappas, and Yiannis Kantaros. Multi-robot mission planning in dynamic semantic environments. arXiv preprint arXiv:2209.06323, 2022.

[76] Sung-Kyun Kim, Amanda Bouman, Gautam Salhotra, David D. Fan, Kyohei Otsu, Joel Burdick, and A. Agha-mohammadi. PLGRIM: Hierarchical value learning for large-scale exploration in unknown environments. In International Conference on Automated Planning and Scheduling (ICAPS), volume 31, 2021.

[77] Anton J Kleywegt, Alexander Shapiro, and Tito Homem-de Mello. The sample average approximation method for stochastic discrete optimization. SIAM Journal on Optimization, 12(2):479–502, 2002.

[78] Sven Koenig and Reid G Simmons. Risk-sensitive planning with probabilistic decision graphs. In Principles of Knowledge Representation and Reasoning, pages 363–373. Elsevier, 1994.

[79] Henrik Kretzschmar, Markus Spies, Christoph Sprunk, and Wolfram Burgard. Socially compliant mobile robot navigation via inverse reinforcement learning. The International Journal of Robotics Research, 35(11):1289–1307, 2016.

[80] C.P. Lam, C.T. Chouand K.H. Chiang, and L.C. Fu. Human-centered robot navigation—towards a harmoniously human–robot coexisting environment. IEEE Trans. Robotics, 27(1):99–112, 2011.

[81] J.C. Latombe. Robot motion planning. Kluwer, 1996.

[82] Thomas Lew, Riccardo Bonalli, and Marco Pavone. Chance-constrained sequential convex programming for robust trajectory optimization. In 2020 European Control Conference (ECC), pages 1871–1878. IEEE, 2020.

[83] Lars Lindemann, Matthew Cleaveland, Gihyun Shim, and George J Pappas. Safe planning in dynamic environments using conformal prediction. arXiv preprint arXiv:2210.10254, 2022.

[84] Lars Lindemann, Xin Qin, Jyotirmoy V Deshmukh, and George J Pappas. Conformal prediction for stl runtime verification. arXiv preprint arXiv:2211.01539, 2022.

[85] B. Lindqvist, S.S. Mansouri, A. Agha, and G. Nikolakopoulos. Nonlinear mpc for collision avoidance and control of uavs with dynamic obstacles. IEEE Robot. & Autom. Lett, 2020.

[86] Johan Lofberg. Yalmip: A toolbox for modeling and optimization in matlab. In Computer Aided Control Systems Design, 2004 IEEE International Symposium on, pages 284–289. IEEE, 2004.

[87] Johan Löfberg. Oops! i cannot do it again: Testing for recursive feasibility in mpc. Automatica, 48(3):550–555, 2012.

[88] B. T. Lopez and J. P. How. Aggressive collision avoidance with limited field-of-view sensing. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 1358–1365, 2017. doi: 10.1109/IROS.2017.8202314.

[89] James Luedtke, Shabbir Ahmed, and George L Nemhauser. An integer programming approach for linear programs with probabilistic constraints. Mathematical programming, 122(2):247–272, 2010.

[90] Rachel Luo, Shengjia Zhao, Jonathan Kuck, Boris Ivanovic, Silvio Savarese, Edward Schmerling, and Marco Pavone. Sample-efficient safety assurances using conformal prediction. arXiv preprint arXiv:2109.14082, 2021.

[91] Keyvan Majd, Shakiba Yaghoubi, Tomoya Yamaguchi, Bardh Hoxha, Danil Prokhorov, and Georgios Fainekos. Safe navigation in human occupied environments using sampling and control barrier functions. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 5794–5800. IEEE, 2021.

[92] A. Majumdar and M. Pavone. How should a robot assess risk? towards an axiomatic theory of risk in robotics. In Robotics Research, pages 75–84. Springer, 2020.

[93] Ali Mesbah. Stochastic model predictive control: An overview and perspectives for future research. IEEE Control Systems Magazine, 36(6):30–44, 2016. doi: 10.1109/MCS.2016.2602087.

[94] Stefan Mitsch, Khalil Ghorbal, and André Platzer. On provably safe obstacle avoidance for autonomous robotic ground vehicles. In Robotics: Science and Systems IX, Technische Universität Berlin, Berlin, Germany, June 24-June 28, 2013, 2013.

[95] D. Morgan, S.J. Chung, and F. Hadaegh. Model predictive control of swarms of spacecraft using sequential convex programming. Journal of Guidance, Control, and Dynamics, 37:1–16, 04 2014. doi: 10.2514/1.G000218.

[96] Keiji Nagatani, Seiga Kiribayashi, Yoshito Okada, Kazuki Otake, Kazuya Yoshida, Satoshi Tadokoro, Takeshi Nishimura, Tomoaki Yoshida, Eiji Koyanagi, Mineo Fukushima, et al. Emergency response to the nuclear accident at the fukushima daiichi nuclear power plants using mobile rescue robots. Journal of Field Robotics, 30(1):44–63, 2013.

[97] Siddharth H Nair, Vijay Govindarajan, Theresa Lin, Chris Meissen, H Eric Tseng, and Francesco Borrelli. Stochastic mpc with multi-modal predictions for traffic intersections. arXiv preprint arXiv:2109.09792, 2021.

[98] Siddharth H. Nair, Vijay Govindarajan, Theresa Lin, Chris Meissen, H. Eric Tseng, and Francesco Borrelli. Stochastic mpc with multi-modal predictions for traffic intersections. In 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), pages 635–640, 2022. doi: 10.1109/ITSC55140.2022.9921751.

[99] Kensuke Nakamura and Somil Bansal. Online update of safety assurances using confidence-based predictions. arXiv preprint arXiv:2210.01199, 2022.

[100] Arkadi Nemirovski and Alexander Shapiro. Convex approximations of chance constrained programs. SIAM Journal on Optimization, 2007.

[101] Jorge Nocedal and Stephen Wright. Numerical optimization. Springer Science & Business Media, 2006.

[102] Marco Omainska, Junya Yamauchi, Thomas Beckers, Takeshi Hatanaka, Sandra Hirche, and Masayuki Fujita. Gaussian process-based visual pursuit control with unknown target motion learning in three dimensions. SICE Journal of Control, Measurement, and System Integration, 14(1):116–127, 2021.

[103] Masahiro Ono, Marco Pavone, Kuwata Kuwata, and J. (Bob) Balaram. Chance-constrained dynamic programming with application to risk-aware robotic space exploration. Autonomous Robots, 39(4):555–571, 2015.

[104] Kyohei Otsu, Guillaume Matheron, Sourish Ghosh, Olivier Toupet, and Masahiro Ono. Fast approximate clearance evaluation for rovers with articulated suspension systems. Journal of Field Robotics, 37:768–785, 2020.

[105] Matteo Palieri, Benjamin Morrell, Abhishek Thakur, Kamak Ebadi, Jeremy Nash, Arghya Chatterjee, Christoforos Kanellakis, Luca Carlone, Cataldo Guaragnella, and A. Agha-mohammadi. LOCUS: A multi-sensor lidar-centric solution for high-precision odometry and 3D mapping in real-time. IEEE Robotics and Automation Letters, 6(2):421–428, 2020.

[106] Panagiotis Papadakis. Terrain traversability analysis methods for unmanned ground vehicles: A survey. Engineering Applications of Artificial Intelligence, 26(4):1373–1385, 2013.

[107] Mike Phillips and Maxim Likhachev. Sipp: Safe interval path planning for dynamic environments. In 2011 IEEE International Conference on Robotics and Automation, pages 5628–5635. IEEE, 2011.

[108] Xin Qin, Yuan Xian, Aditya Zutshi, Chuchu Fan, and Jyotirmoy V Deshmukh. Statistical verification of cyber-physical systems using surrogate models and conformal inference. In 2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS), pages 116–126. IEEE, 2022.

[109] Nathan Ratliff, Matt Zucker, J Andrew Bagnell, and Siddhartha Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In 2009 IEEE International Conference on Robotics and Automation, pages 489–494. IEEE, 2009.

[110] Venkatraman Renganathan, Iman Shames, and Tyler H Summers. Towards integrated perception and motion planning with distributionally robust risk constraints. IFAC-PapersOnLine, 53(2):15530–15536, 2020.

[111] Venkatraman Renganathan, Sleiman Safaoui, Aadi Kothari, Benjamin Gravell, Iman Shames, and Tyler Summers. Risk bounded nonlinear robot motion planning with integrated perception & control. arXiv preprint arXiv:2201.01483, 2022.

[112] A. Richards and J. P. How. Model predictive control of vehicle maneuvers with guaranteed completion time and robust feasibility. In Proceedings of the 2003 American Control Conference, 2003., volume 5, pages 4034–4040 vol.5, 2003. doi: 10.1109/ACC.2003.1240467.

[113] Ugo Rosolia, Ashwin Carvalho, and Francesco Borrelli. Autonomous racing using learning model predictive control. In 2017 American Control Conference (ACC), pages 5115–5120, 2017. doi: 10.23919/ACC.2017.7963748.

[114] Ugo Rosolia, Andrew Singletary, and Aaron D Ames. Unified multi-rate control: from low level actuation to high level planning. arXiv preprint arXiv:2012.06558, 2020.

[115] Andrzej Ruszczynski. Risk-averse dynamic programming for markov decision processes. Mathematical Programming, 75(2):235–261, 2014.

[116] John Schulman, Yan Duan, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, Jia Pan, Sachin Patil, Ken Goldberg, and Pieter Abbeel. Motion planning with sequential convex optimization and convex collision checking. The International Journal of Robotics Research, 33(9):1251–1270, 2014.

[117] Tobias Schwarze and Zhichao Zhong. Stair detection and tracking from egocentric stereo vision. In 2015 IEEE International Conference on Image Processing (ICIP), pages 2690–2694. IEEE, 2015.

[118] Esmaeil Seraj and Matthew Gombolay. Coordinated control of uavs for human-centered active sensing of wildfires. In 2020 American Control Conference (ACC), pages 1845–1852. IEEE, 2020.

[119] Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. Journal of Machine Learning Research, 9(3), 2008.

[120] Alexander Shapiro. Distributionally robust stochastic programming. SIAM Journal on Optimization, 27(4):2258–2275, 2017.

[121] R. Sharan and J. Burdick. Finite State Control of POMDPs with LTL Specifications. In American Control Conference, 2014.

[122] Troy Shinbrot, Celso A Grebogi, Jack Wisdom, and James A Yorke. Chaos in a double pendulum. In American Journal of Physics, American Association of Physics Teachers. 1992.

[123] S. Singh, Y. Chow, A. Majumdar, and M. Pavone. A framework for time-consistent, risk-sensitive model predictive control: Theory and algorithms. IEEE Transactions on Automatic Control, 2018.

[124] Andrew Singletary, Aiden Swann, Ivan Dario Jimenez Rodriguez, and Aaron D. Ames. Safe drone flight with time-varying backup controllers, 2022. URL https://arxiv.org/abs/2207.05220.

[125] P. Sopasakis, M. Schuurmans, and P. Patrinos. Risk-averse risk-constrained optimal control. In 2019 18th European Control Conference (ECC), pages 375–380, 2019. doi: 10.23919/ECC.2019.8796021.

[126] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. Mathematical Programming Computation, pages 1–36, 2020.

[127] F. Takens. Detecting strange attractors in turbulence. In Dynamical systems and turbulence, pages 366–381. Springer, 1981.

[128] Herbert G Tanner, Savvas G Loizou, and Kostas J Kyriakopoulos. Nonholonomic navigation and control of cooperating mobile manipulators. IEEE Trans. Robotics and Automation, 19(1):53–64, 2003.

[129] Dimos V. Dimarogonas et al. A feedback stabilization and collision avoidance scheme for multiple independent non-point agents. Automatica, 42(2):229–243, 2006.

[130] Antony Thomas, Fulvio Mastrogiovanni, and Marco Baglietto. Probabilistic collision constraint for motion planning in dynamic environments. In International Conference on Intelligent Autonomous Systems, pages 141–154. Springer, 2021.

[131] Ryan J Tibshirani, Rina Foygel Barber, Emmanuel Candes, and Aaditya Ramdas. Conformal prediction under covariate shift. Advances in neural information processing systems, 32, 2019.

[132] C. Tomlin, G.J. Pappas, and S. Sastry. Conflict resolution for air traffic management: a study in multiagent hybrid systems. IEEE Transactions on Automatic Control, 43(4):509–521, 1998. doi: 10.1109/9.664154.

[133] Jesus Tordesillas, Brett T. Lopez, Michael Everett, and Jonathan P. How. Faster: Fast and safe trajectory planner for navigation in unknown environments, 2020. URL https://arxiv.org/abs/2001.04420.

[134] Jesús Tordesillas and Jonathan How. Panther: Perception-aware trajectory planner in dynamic environments. ArXiv, 03 2021.

[135] Peter Trautman and Andreas Krause. Unfreezing the robot: Navigation in dense, interacting crowds. In 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 797–803. IEEE, 2010.

[136] Ioannis Tzortzis, Charalambos D. Charalambous, and Christoforos N. Hadjicostis. A distributionally robust lqr for systems with multiple uncertain players. In 2021 60th IEEE Conf. on Decision and Control (CDC), pages 3972–3977, 2021. doi: 10.1109/CDC45484.2021.9682976.

[137] J. van den Berg, D. Ferguson, and J. Kuffner. Anytime path planning and replanning in dynamic environments. In IEEE Int. Conf. Robotics and Automation, pages 2366–2371, 2006. doi: 10.1109/ROBOT.2006.1642056.

[138] Bart P. G. Van Parys, Daniel Kuhn, Paul J. Goulart, and Manfred Morari. Distributionally robust control of constrained stochastic systems. IEEE Transactions on Automatic Control, 61(2):430–442, 2016. doi: 10.1109/TAC.2015.2444134.

[139] A. Vecchietti, S. Lee, and I.E. Grossman. Modeling of Discrete/Continuous Optimization Problems: Characterization and Formulation of Disjunctions and Their Relaxations. Computers and Chemical Engineering, 27(3):433–448, 2003.

[140] Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. Algorithmic learning in a random world. Springer Science & Business Media, 2005.

[141] Allan Wang, Christoforos Mavrogiannis, and Aaron Steinfeld. Group-based motion prediction for navigation in crowded environments. In Conference on Robot Learning, pages 871–882. PMLR, 2022.

[142] Skylar X. Wei, Anushri Dixit, Shashank Tomar, and Joel W. Burdick. Moving obstacle avoidance: A data-driven risk-aware approach. IEEE Control Systems Letters, 7:289–294, 2022. doi: 10.1109/LCSYS.2022.3181191.

[143] Weijun Xie. On distributionally robust chance constrained programs with wasserstein distance. Math. Program., 186(1):115–155, 2021.

[144] Huan Xu and Shie Mannor. Distributionally robust Markov decision processes. In Advances in Neural Information Processing Systems, pages 2505–2513, 2010.

[145] Youngmin Yoon, Changhee Kim, Jongmin Lee, and Kyongsu Yi. Interaction-aware probabilistic trajectory prediction of cut-in vehicles using gaussian process for proactive control of autonomous vehicles. IEEE Access, 9:63440–63455, 2021. doi: 10.1109/ACCESS.2021.3075677.

[146] Margaux Zaffran, Olivier Féron, Yannig Goude, Julie Josse, and Aymeric Dieuleveut. Adaptive conformal predictions for time series. In International Conference on Machine Learning, pages 25834–25866. PMLR, 2022.

[147] Jingyu Zhang and Toshiyuki Ohtsuka. Stochastic model predictive control using simplified affine disturbance feedback for chance-constrained systems. IEEE Control Systems Letters, 5(5):1633–1638, 2021. doi: 10.1109/LCSYS.2020.3042085.

[148] Alireza Zolanvari and Ashish Cherukuri. Data-driven distributionally robust iterative risk-constrained model predictive control. arXiv preprint arXiv:2111.12977, 2021.