MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Safe multi-agent motion planning under uncertainty for drones using filtered reinforcement learning

Safaoui, Sleiman; Vinod, Abraham P.; Chakrabarty, Ankush; Quirynen, Rien; Yoshikawa, Nobuyuki; Di Cairano, Stefano

TR2024-048 May 02, 2024

Abstract

We consider the problem of safe multi-agent motion planning for drones in uncertain, cluttered workspaces. For this problem, we present a tractable motion planner that builds upon the strengths of reinforcement learning and constrained- control-based trajectory planning. First, we use single-agent reinforcement learning to learn motion plans from data that reach the target but may not be collision-free. Next, we use a convex optimization, chance constraints, and set-based methods for constrained control to ensure safety, despite the uncertainty in the workspace, agent motion, and sensing. The proposed approach can handle state and control constraints on the agents, and enforce collision avoidance among themselves and with static obstacles in the workspace with high probability. The proposed approach yields a safe, real-time implementable, multi-agent motion planner that is simpler to train than methods based solely on learning. Numerical simulations and experiments show the efficacy of the approach.

IEEE Transactions on Robotics 2024

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Safe multi-agent motion planning under uncertainty for drones using filtered reinforcement learning

Sleiman Safaoui†, Abraham P. Vinod†*, Ankush Chakrabarty, Rien Quirynen, Nobuyuki Yoshikawa, and Stefano Di Cairano

***Abstract*—We consider the problem of safe multi-agent motion planning for drones in uncertain, cluttered workspaces. For this problem, we present a tractable motion planner that builds upon the strengths of reinforcement learning and constrained-control-based trajectory planning. First, we use single-agent reinforcement learning to learn motion plans from data that reach the target but may not be collision-free. Next, we use a convex optimization, chance constraints, and set-based methods for constrained control to ensure safety, despite the uncertainty in the workspace, agent motion, and sensing. The proposed approach can handle state and control constraints on the agents, and enforce collision avoidance among themselves and with static obstacles in the workspace with high probability. The proposed approach yields a safe, real-time implementable, multi-agent motion planner that is simpler to train than methods based solely on learning. Numerical simulations and experiments show the efficacy of the approach.**

***Index Terms*—Safe learning-based control, model predictive control, reinforcement learning, optimization, collision avoidance**

I. Introduction

Multi-agent motion planning in cluttered workspaces under stochastic uncertainty arising from both perception and actuation is a key challenge in designing reliable autonomous systems. The need for such planners, especially for quadrotors, arises in a variety of application areas including transportation, logistics, monitoring, and agriculture. Recently, motion planning using reinforcement learning (RL) has gained attention, due to its ability to leverage data to tackle generic dynamical systems and complex task specifications [1]–[9]. A major challenge for such efforts is the lack of safety guarantees, since most of the existing RL-based approaches enforce safety constraints by soft constraints and are subject to training errors. Additionally, multi-agent RL is known to suffer from non-stationarity and scalability issues, which may prevent the training to converge [9]. We propose *a tractable approach to safe, multi-agent motion planning in stochastic, cluttered workspaces that combines reinforcement learning and set-based methods for constrained control. Our approach yields a safe, real-time implementable, multi-agent motion planner that is simple to train and enforces safety with high probability, by means of chance constraints.*

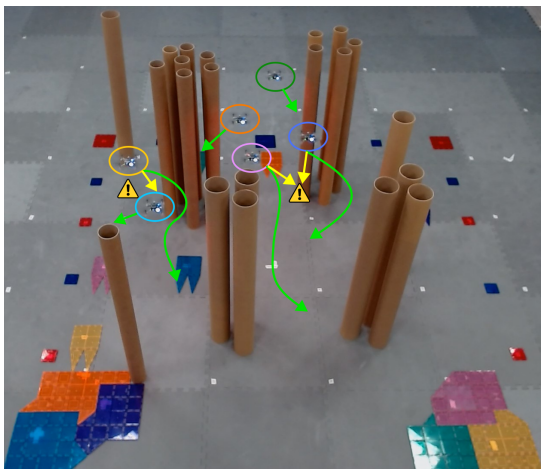

Fig. 1. Existing reinforcement learning-based motion planners can generate unsafe trajectories (yellow arrows), since they treat safety as soft constraints, which is undesirable in safety-critical applications. We propose a constrained-control-based safety filter that renders such motion planners safe (long green arrows) by enforcing safety as hard constraints. See https://youtu.be/QcCSYNwuuo8 for an overview and videos of the experiments.

In the deterministic setting, various approaches have been proposed for multi-agent motion planning such as centralized scheduling and coordination [10], roadmap and discrete search followed by trajectory refinement [11], sampling-based rapidly-exploring random trees [12], adaptive roadmaps [13], buffered Voronoi cells [14], mixed-integer programming [15], sequential convex programming [16], [17], formal methods and finite transition systems [18], and control barrier functions [8], [19], [20]. Recently, RL-based planners have been used in complex environments [1]–[9]. A key advantage of learning-based planners is the ability to leverage past experience in future decision making [21]. Consequently, such planners can tackle *complex* and *high-dimensional* motion planning tasks, while incorporating prior information about the planning task and accommodating uncertainty [9].

Our preliminary work [22] considered *deterministic* safe multi-agent motion planning, where everything was known exactly. We proposed a two-step approach where a single-agent RL algorithm provided a *reference* command which was subsequently *filtered* (or corrected) by a constrained control module. The works closest to [22] are [23]–[26] that also follow a similar two-step process. However, [23], [24] need labelled data for supervised learning, and [25], [26] use multi-agent RL. Multi-agent RL trains multiple agents to collectively

† Equal contributions.

* Corresponding author: A. P. Vinod (abraham.p.vinod@ieee.org)

S. Safaoui is with Eric Jonsson School of Engineering & Computer Science, The University of Texas at Dallas, Richardson, TX. He was can intern at Mitsubishi Electric Research Laboratories during this work. Email: snsafaoui@gmail.com.

A. P. Vinod, A. Chakrabarty, and S. Di Cairano are with Mitsubishi Electric Research Laboratories, Cambridge, MA. Email: {abraham.p.vinod,achakrabarty,dicairano}@ieee.org.

R. Quirynen was with Mitsubishi Electric Research Laboratories at the time of this work. Email: rien.quirynen@gmail.com.

N. Yoshikawa is with Mitsubishi Electric Corporation, Japan. Email: Yoshikawa.Nobuyuki@ak.MitsubishiElectric.co.jp.

complete the task, and as a consequence, is harder to train than single-agent RL. Additionally, [26] relies on solving a two-player game in problems with discrete state and action space. Our approach utilizes single-agent RL that is simpler to train, can accommodate continuous state and action spaces for stabilizable linear dynamics, is real-time implementable, and need not be retrained as the number of agents increase.

In the stochastic setting, the presence of probabilistic constraints makes the motion planning problem more challenging. Here, we must balance the conservativeness of the motion plans with the risk (the probability of violation of a desirable property), while ensuring the existence of a solution that satisfies other problem objectives. The authors of [27], [28] propose single-agent motion planners using chance constrained programming, but these approaches may become prohibitively expensive when extended to multi-agent systems. [29] combined artificial potential fields with stochastic reachability theory to generate motion plans for a single-agent in stochastic, cluttered workspaces. Recently, [30], [31] proposed using a *reference* controller, such as an RL controller trained offline, followed by a constrained-control-based online filtering step to guarantee safety of the control action being applied to the system, which was applied to autonomous racing [32]. However, to the best of our knowledge, these works do not consider a multi-agent setup. Another line of research for multi-agent motion planning in stochastic settings uses buffered Voronoi cells [33], where motion plans are restricted to safety sets computed based on the "best" separating hyperplane between two Gaussian distributions, further tightened by a safety buffer. Such an approach couples planning and safety control, yet it does not guarantee recursive feasibility.

We focus on safe multi-agent motion planning in applications where a *centralized* decision maker coordinates the actions of the agents for safety and performance. Examples of such applications include air traffic control and coordinated traffic control centers [34]. The proposed centralized approach may impose additional communication and computational burden when compared to a decentralized method (for e.g., [33]). However, the ability to enforce coordination helps the proposed approach typically generate safer and more efficient trajectories for the overall system.

Contributions of this work: Since RL has been recently of interest to the robotics community, we propose a solution to address the lack of safety in RL-based planners, specifically in multi-agent motion planning settings. We propose an optimization-based safety filter that, when used in conjunction with RL, provides a safe, multi-agent motion planner in cluttered workspaces under stochastic uncertainty. The proposed safety filter uses convex optimization and set-based control to compute minimum-norm corrections to the RL-based motion plan and guarantee probabilistic collective safety of the multi-agent system. We use single-agent RL to learn from data, while avoiding issues like non-stationarity and scalability that affect multi-agent RL. We also describe how to design terminal state constraints for the constrained-control-based safety filter by using reachability to achieve recursive feasibility. Finally, we demonstrate our approach by both numerical simulations and experiments using quadrotors.

We note that while the proposed solution is discussed in the context of RL, our approach is general and can be used with another planner instead of RL. For instance, the single-agent planner could be sampling-based (e.g. RRT-based planners [35], [36]), and the advantage of our architecture would be in the dimensionality reduction and reduced effort for collision checking with respect to applying the sampling-based planner to the full multi-agent problem.

Relationship with our preliminary work [22]: In [22], we proposed a safe, multi-agent motion planner using reinforcement learning and optimization for deterministic dynamics and a known workspace. We generated continuous-time safety guarantees under the assumption that the safety filter's control input is constant across the entire horizon of the safety filter. In this work, we extend our preliminary work [22] to stochastic workspace, dynamics, and sensing, and provide probabilistic safety guarantees for the overall system without relying on the constant control input assumptions. We also explicitly assess recursive feasibility of the safety filter, and investigate the importance of the RL controller, safety filter, and the terminal constraints in the proposed approach using extensive hardware and simulation experiments.

A. Notation

0_d ($0_{n,m}$) is a vector (matrix) of zeros in $\mathbb{R}^d$ ($\mathbb{R}^{d\times m}$), I_d is the d-dimensional identity matrix, and $\mathbb{N}_{[a,b]}$ is the subset of natural numbers between (and including) $a, b \in \mathbb{N}$, $a \leq b$, and $\mathbb{N}_{[a,b]} = \emptyset$ when $a > b$. $\oplus, \ominus$ are the Minkowski sum and Pontyagrin difference, respectively, and $\|\cdot\|$ is the 2-norm of a vector. The support function of a convex and compact set $\mathcal{C}$ is $S_{\mathcal{C}}(\ell) \triangleq \sup_{x\in\mathcal{C}} \ell \cdot x$ for any $\ell \in \mathbb{R}^d$ [37].

$(\Omega, \mathcal{F}, \mathbb{P})$ is a probability space where Ω is the sample space, $\mathcal{F}$ is a σ-algebra of subsets of Ω, and $\mathbb{P}$ is a probability measure on $\mathcal{F}$. We denote random vectors in bold $\boldsymbol{x} : \Omega \to \mathbb{R}^n$ and their mean $\overline{x} \triangleq \mathbb{E}[\boldsymbol{x}]$, where $\mathbb{E}$ is the expectation operator with respect to $\mathbb{P}$. We use $\hat{x}$ to denote a realization of a random vector $\boldsymbol{x}$. We use $\mathcal{N}(\mu, \Sigma)$ to denote a Gaussian random vector with mean μ and covariance Σ, and refer to $\mathcal{N}(0_n, I_n)$ as the standard Gaussian random vector. For a vector $\boldsymbol{v}(t)$, $\boldsymbol{v}(k|t)$ is the predicted value at $k \geq t$ based on the information available at time t, and we denote $\boldsymbol{v}(t|t) = \boldsymbol{v}(t)$. We use the same notation when referring to the distribution of $\boldsymbol{v}(t)$.

The following abbreviations are used throughout the paper: iid (independent and identically distributed), MPC (model predictive control), QP (quadratic program), and PSD (positive semidefinite).

II. Problem formulation

Dynamics: Consider $N \in \mathbb{N}$ homogeneous agents with the discrete-time linear dynamics at time t,

$$\boldsymbol{x}_i(t+1) = A\boldsymbol{x}_i(t) + Bu_i(t) + \boldsymbol{w}_i(t), \tag{1}$$

with state $\boldsymbol{x}_i \in \mathbb{R}^n$, input $u_i \in \mathcal{U} \subset \mathbb{R}^m$, process noise $\boldsymbol{w}_i \in \mathbb{R}^n$, state update matrix $A \in \mathbb{R}^{n\times n}$, and input matrix $B \in \mathbb{R}^{n\times m}$ for each agent $i \in \mathbb{N}_{[1,N]}$. The input set $\mathcal{U}$ is a convex and compact polytope, and the process noise is iid, zero-mean

Gaussian $\boldsymbol{w}_i \sim \mathcal{N}(0_n, \Sigma_w)$ for some known PSD matrix $\Sigma_w \in \mathbb{R}^{n\times n}$. The position of the agent i at time t is given by,

$$\boldsymbol{p}_i(t) = C\boldsymbol{x}_i(t), \tag{2}$$

for some $C \in \mathbb{R}^{d\times n}$, $d < m$. For $k \geq t$, the mean state of the agents is predicted according to the nominal dynamics,

$$\overline{x}_i(k+1|t) = A\overline{x}_i(k|t) + Bu_i(k|t) \tag{3a}$$
$$\overline{p}_i(k|t) = C\overline{x}_i(k|t). \tag{3b}$$

We assume that the nominal dynamics (3) are stabilizable, i.e., there exists a stabilizing gain matrix $K \in \mathbb{R}^{m\times n}$ that ensures that all eigenvalues of $(A + BK)$ lie in the unit circle.

Measurement model: We assume that the initial state of each agent $\boldsymbol{x}_i(0) = x_i(0)$ is known, i.e., deterministic. However, for $t > 0$, we have access only to a noisy measurement of the true state $\boldsymbol{x}_i(t)$. Specifically, we assume that the measurements $\hat{y}_i(t)$ are a realization of a random vector $\boldsymbol{y}(t)$,

$$\boldsymbol{y}_i(t) = \boldsymbol{x}_i(t) + \boldsymbol{\eta}_i(t), \tag{4}$$

where $\boldsymbol{\eta} \sim \mathcal{N}(0_n, \Sigma_\eta)$ is a zero-mean Gaussian noise with a known PSD matrix $\Sigma_\eta \in \mathbb{R}^{n\times n}$.

Agent Representation: We consider agents with identical convex and compact rigid bodies, denoted by $\mathcal{A} \subset \mathbb{R}^d$, such that $0_d \in \mathcal{A}$. The rigid bodies of the agents are rotation-invariant [38]. So, we only consider translations.

Remark 1. *We can generalize all the presented results to account for heterogeneous agents with heterogeneous linear dynamics, measurement models, and rigid bodies. We consider homogeneity in all of these aspects to simplify the presentation.*

Workspace Representation: We represent the workspace using a convex and compact polytope $\mathcal{K} \subset \mathbb{R}^d$.

Obstacle Representation: The workspace has N_O static obstacles, each with a convex and compact rigid body $\mathcal{O}_j \subset \mathbb{R}^d$ and $0_d \in \mathcal{O}_j$ ($j \in \mathbb{N}_{[1,N_O]}$). The obstacle shapes are known *a priori*, but their positions are available only via a noisy measurement. Specifically, for each obstacle $j \in \mathbb{N}_{[1,N_O]}$, the position of a representative point of the obstacle (e.g. the *center*) is denoted by $\boldsymbol{c}_j \in \mathbb{R}^d$ where $\boldsymbol{c}_j$ is an iid Gaussian random vector $\boldsymbol{c}_j \sim \mathcal{N}(\overline{c}_j, \Sigma_{c_j})$ with nominal position $\overline{c}_j$ and covariance matrix $\Sigma_{c_j} \in \mathbb{R}^{d\times d}$.

A. Safe multi-agent motion planning under uncertainty

Given target positions $q_i \in \mathbb{R}^d$, we want to design a motion planner that drives the agents towards their respective target positions, while ensuring safety of the agents at all times, despite the uncertainty in the dynamics and the noisy estimates of the agent and obstacle positions. Here, we formalize the required features of safety in the multi-agent motion planning problem by introducing the notion of *probabilistic collective safety*, inspired by existing literature [22], [38].

Definition 1 (PROBABILISTIC COLLECTIVE SAFETY)**.** *The agents are said to be* probabilistically collectively safe *at time t when all the following criteria are met:*

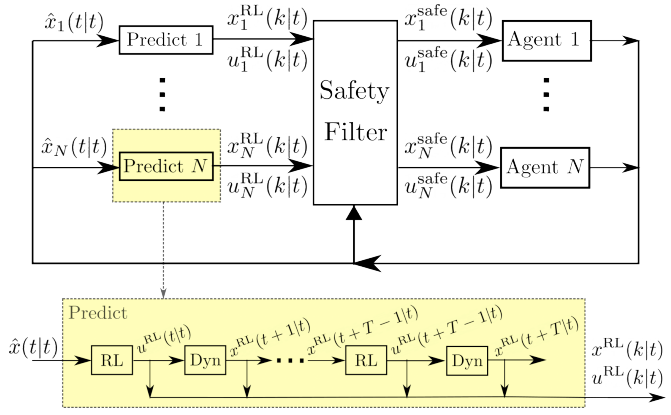

Fig. 2. The proposed solution combines single-agent RL-based motion planning with a constrained-control-based safety filter for safe multi-agent motion planning. It computes a sequence of RL states and controls for the horizon T using a predict block. Next, it uses a safety filter to render these controls safe for each agent. The predict block uses a policy network (trained offline) and the nominal dynamics (3) to compute the RL controls and states.

1) Static obstacle avoidance constraints*: The probability of collision of agent $i \in \mathbb{N}_{[1,N]}$ with obstacle $j \in \mathbb{N}_{[1,N_O]}$ is less than a pre-specified risk bound $\alpha_{i,j,t} \in (0,1)$,*

$$\mathbb{P}((\boldsymbol{p}_i(t) \oplus \mathcal{A}) \cap (\boldsymbol{c}_j(t) \oplus \mathcal{O}_j) \neq \emptyset) \leq \alpha_{i,j,t}. \tag{5}$$

2) Inter-agent collision avoidance*: The probability of collision between agents $i, i' \in \mathbb{N}_{[1,N]}$, $i \neq i'$ is less than a pre-specified risk bound $\beta_{i,i',k} \in (0,1)$,*

$$\mathbb{P}((\boldsymbol{p}_i(t) \oplus \mathcal{A}) \cap (\boldsymbol{p}_{i'}(t) \oplus \mathcal{A}) \neq \emptyset) \leq \beta_{i,i',t}. \tag{6}$$

3) Keep-in constraints*: The probability of agent $i \in \mathbb{N}_{[1,N]}$ exiting the keep-in set $\mathcal{K}$ is less than a pre-specified risk bound $\kappa_{i,k} \in (0,1)$,*

$$\mathbb{P}(\boldsymbol{p}_i(t) \oplus \mathcal{A} \not\subseteq \mathcal{K}) \leq \kappa_{i,k}. \tag{7}$$

Next, we formulate the problem tackled in this paper.

Problem 1 (SAFE MULTI-AGENT PLANNING)**.** *Given user-specified risk bounds $\alpha_{i,j,t}$, $\beta_{i,i',t}$, $\kappa_{i,t}$, for every $i \in \mathbb{N}_{[1,N]}$, $i' \in \mathbb{N}_{[1,i-1]}$, $j \in \mathbb{N}_{[1,N_O]}$ and $t \in \mathbb{N}$, design a multi-agent motion planner that navigates the agents with dynamics* (1) *to their respective targets such that the agents are probabilistically collectively safe at all times t.*

In the statement of Problem 1, we specified a risk bound for every time step ($\alpha_{i,j,t}, \beta_{i,i',t}, \kappa_{i,t}$). On the other hand, when a risk bound for the entire planned trajectory (e.g. $\alpha_{i,j}$) is given over a planning horizon $T \in \mathbb{N}$, one can arrive at $\alpha_{i,j,t}$ via *risk allocation* [27] — divide the risk equally across time steps with $\alpha_{i,j,t} = \alpha_{i,j}/T$ for every $t \in \mathbb{N}_{[1,T]}$.

III. PROPOSED SOLUTION

We solve Problem 1 by the following two steps.

1) *RL training (Offline):* We train a neural network to drive a *single* agent with *nominal* (deterministic) dynamics (3) from any initial state in the workspace to a final desired state. The resulting policy learns to perform static obstacle collision avoidance and remain within

the workspace while transferring a single agent from its initial state to the final state.

2) *Safety Filter (Online):* We use a constrained-control-based safety filter that uses an online evaluation of the RL-based motion planner. The safety filter suitably modifies the motion plan to enforce probabilistic collective safety at all time steps. The safety filter solves a real-time implementable, convex, quadratic program to determine the modifications.

Figure 2 depicts the proposed solution. In the following, we describe the RL-based motion planner, provide details of constructing the safety filter to enforce probabilistic collective safety, and discuss various aspects of the proposed solution.

A. Reinforcement learning-based single-agent motion planner

We design a RL-based motion planner that drives the agent with nominal dynamics (3) to a specified target position $q \in \mathbb{R}^d$ in presence of N_O static obstacles located at nominal positions $\bar{c}_j \ \forall j \in \mathbb{N}_{[1,N_O]}$. Here, we train a *single-agent RL-based* planner in an environment devoid of other agents. After briefly discussing the motivations for such an approximation, we set up the Markov decision process used for training and characterize the neural policy obtained via single-agent RL training.

The advantage of using a single-agent RL-based planner instead of the full multi-agent RL-based planner is in the ease of training. Specifically, a single-agent RL-based planner avoids some issues of multi-agent RL such as non-stationarity, scalability, and the diminished ability to accommodate potential changes in team size post training. Recall that in multi-agent RL, all agents learn concurrently and thus an action taken by an individual agent affects both the reward of the other agents and the evolution of the state of the system. From the agent's perspective the environment is non-stationary [9]. By approximating the problem and eliminating the other agents, training a single-agent RL is a stationary problem which is key for convergence results of RL training and for the reduced training effort [9]. Moreover, compared to multi-agent RL training, whose joint state space and joint action space grow rapidly in dimension with the number of agents, the state space and the action space dimensions are fixed and independent of the team size in the single-agent RL training. Finally, if the team size changes post training, the proposed single-agent RL-based planner in Figure 2 can still be used without any modifications as compared to a complete multi-agent RL-based planner, which may require re-training to handle changes in the team size.

Consider a feedforward-feedback controller $\pi : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^m$ with

$$u = \pi(x, r) = Kx + Fr, \tag{8}$$

where K is a stabilizing gain matrix, $F \in \mathbb{R}^{m\times d}$ provides closed loop unitary gain with the nominal dynamics (3), i.e., $C(I-(A+BK))^{-1}BF = I_d$, and $r \in \mathbb{R}^d$ is the reference position command. We obtain the following stabilized, nominal, prediction model for the agent at any time $k \geq t$,

$$\overline{x}(k+1|t) = (A+BK)\overline{x}(k|t) + BFr(k|t), \tag{9}$$

by closing the loop of the dynamics (3) with the controller (8). For the measurement model (4), the predicted measurements are $\hat{y}(k|t) = \overline{x}(k|t)$ for every $k \geq t$. By construction, the mean predicted position $\overline{p}(k|t) \to r$ as $k \to \infty$ for a constant reference command $r(k|t) = r$.

We use the following Markov decision process for training:

- *Observation space*: We define the observation vector $o \in \mathbb{R}^{n+d+N_O d}$ as the concatenated vector containing the current measurement of the agent $\hat{y} \in \mathbb{R}^n$, the displacement of the agent's current measured position to the target $(p-q) \in \mathbb{R}^d$ and to the N_O static obstacles $(p-\bar{c}_j) \in \mathbb{R}^d$, for all $j \in \mathbb{N}_{[1,N_O]}$, where $p = C\hat{y}$.
- *Action space*: The action $a \in \mathscr{A} \subset \mathbb{R}^d$ determines the reference position as a perturbation a to the target q, $r = q + a$. The set $\mathscr{A}$ is compact.
- *Step function*: The next predicted measurement $\hat{y}(t+1|t) = \overline{x}(t+1|t)$ is given by (9).
- *Reward function*: The instantaneous reward function is

$$R(o) = \zeta_{\text{obs}} \sum_{j=1}^{N_O} \frac{1}{\|p-\bar{c}_j\|^2 - \gamma_j^2} + \zeta_{\text{tgt}}\|p-q\|, \tag{10}$$

with reward parameters $\zeta_{\text{obs}}, \zeta_{\text{tgt}} \leq 0$ and $\gamma_j \geq 0$. Here, γ_j is the radius of the smallest volume ball that covers the set $\mathcal{O}_j \oplus (-\mathcal{A})$ for each $j \in \mathbb{N}_{[1,N_O]}$. We terminate an episode when the agent either reaches the target or violates the *nominal single-agent* safety conditions, namely the static obstacle avoidance and keep-in constraints. These constraint violations are given by:

$$(\overline{p}_i(t) \oplus \mathcal{A}) \cap (\bar{c}_j(t) \oplus \mathcal{O}_j) \neq \emptyset \text{ and } \overline{p}_i(t) \oplus \mathcal{A} \not\subseteq \mathcal{K}.$$

When the episode terminates, we add a terminal reward or penalty as follows:

$$R(o_\infty) = \begin{cases} R_{\text{target}}, & \text{if } \|p_\infty - q\| \leq d \ , \\ P_{\text{keep-in}}, & \text{if } p_\infty \oplus \mathcal{A} \not\subseteq \mathcal{K}, \\ P_{\text{obstacle}}, & \text{if agent hits an obstacle,} \end{cases}$$

with o_∞ and p_∞ denoting as the observation and position vectors upon termination respectively, $R_{\text{target}} \geq 0$, and $P_{\text{keep-in}}, P_{\text{obstacle}} \leq 0$.

We have set up the Markov decision process to consider deterministic nominal dynamics (3) instead of the original stochastic dynamics (1) in order to simplify the RL training. Our numerical and hardware experiments show that the restriction to deterministic nominal dynamics does not affect the proposed solution severely.

Remark 2. *Our approach can also accommodate a known, time-varying target and biased measurement models $\overline{\boldsymbol{\eta}} \neq 0$. We have considered a time-invariant target q and an unbiased measurement model here to simplify the presentation. Additionally, we use the minimum volume balls with radius γ_j in* (10) *instead of $\mathcal{O}_j \oplus (-\mathcal{A})$ to simplify the collision detection while training the RL-based motion planner.*

Upon completion of training, most of the existing RL algorithms return a policy $\nu : \mathbb{R}^{n+d+N_O d} \to \mathscr{A}$ that provides the action to apply given an observation vector, e.g., by a

neural network [39]. Additionally, we can "rollout" the policy network ν to obtain a trajectory based on the RL motion planner for a planning horizon $T \in \mathbb{N}$. Consider any agent $i \in \mathbb{N}_{[1,N]}$ that starts with the measurement $\hat{y}_i(t)$. We compute the RL motion plan $\{x_i^{\mathrm{RL}}(k|t)\}_{k=t}^{t+T}$, where $x_i^{\mathrm{RL}}(t|t) = \hat{y}_i(t)$, by alternating between finding the control $u_i^{\mathrm{RL}}(k|t)$ given the predicted RL state $x_i^{\mathrm{RL}}(k|t)$ and predicted observation vector $o_i(k|t)$ at some time $k \geq t$ using π,

$$u_i^{\mathrm{RL}}(k|t) = \pi\left(x_i^{\mathrm{RL}}(k|t), q + \nu(o_i(k|t))\right), \tag{11}$$

and predicting the next RL state $x_i^{\mathrm{RL}}(k+1|t)$ using (9) and the corresponding predicted observation vector $o_i(k+1|t)$.

The generated motion plan $\{x_i^{\mathrm{RL}}(k|t)\}_{k=t}^{t+T}$ does not satisfy probabilistic collective safety, since RL cannot guarantee collision-free trajectories (it only penalizes collisions and is subject to training errors), and the generated RL motion plan completely ignores inter-agent collision avoidance and the effect of the process and measurement noises.

B. Safety Filter

We now generate corrections to the RL-based motion plan $\{x_i^{\mathrm{RL}}(k|t)\}_{k=t}^{t+T}$ using a constrained-control-based safety filter that ensures the satisfaction of probabilistic collective safety at all times. Consider the following optimization problem with the stochastic information,

$$\min_{\{U_i^{\mathrm{s}}(t)\}_{i=1}^{N}} \sum_{k\in\mathbb{N}_{[t,t+T-1]}} \sum_{i\in\mathbb{N}_{[1,N]}} \lambda_{i,k} \left\| u_i^{\mathrm{RL}}(k|t) - u_i^{\mathrm{safe}}(k|t) \right\|^2 \tag{12a}$$

$$\text{s.t. Dynamics (1) and (2) with } u = u^{\mathrm{safe}}, \tag{12b}$$

$$\boldsymbol{x}_i(t|t) = \hat{y}_i(t) - \boldsymbol{\eta}(t), \qquad \forall i \in \mathbb{N}_{[1,N]}, \tag{12c}$$

$$u_i^{\mathrm{safe}}(k|t) \in \mathcal{U}, \quad \forall k \in \mathbb{N}_{[t,t+T-1]}, \ \forall i \in \mathbb{N}_{[1,N]}, \tag{12d}$$

$$\text{Probabilistic collective safety at } k, \quad \forall k \in \mathbb{N}_{[t,t+T]}, \tag{12e}$$

$$\text{Terminal constraints for recursive feasibility}, \tag{12f}$$

where $U_i^{\mathrm{s}}(t) = \{u_i^{\mathrm{safe}}(k|t)\}_{k=t}^{t+T-1}$ for each $i \in \mathbb{N}_{[1,N]}$, and $\lambda_{i,k} \geq 0$ are pre-specified weights on the deviations $\left\| u_i^{\mathrm{RL}}(k|t) - u_i^{\mathrm{safe}}(k|t) \right\|^2$ for $i \in \mathbb{N}_{[1,N]}$ and $k \in \mathbb{N}_{[t,t+T-1]}$.

The safety filter (12) takes the RL control sequence $\{u_i^{\mathrm{RL}}(k|t)\}_{k=t}^{t+T-1}$ that is generated using the RL-based single-agent motion planner, and computes safe control inputs $\{u_i^{\mathrm{safe}}(k|t)\}_{k=t}^{t+T-1}$ within the control set (12d) that minimally deviate from the corresponding RL control inputs (12a), while enforcing probabilistic collective safety constraints (12e). The constraint (12c) defines the distribution of the noisy current state $\boldsymbol{x}(t|t)$ from the current measurement $\hat{y}(t)$ and the measurement model (4). Additionally, to avoid computing control actions u_i^{safe} that may render the optimization problem (12) in the safety filter infeasible in the future, we include terminal state constraints (12f) that, when designed as explained later, provide recursive feasibility. Only the first safe control $u_i^{\mathrm{safe}}(t|t)$ is applied for each agent i, and then (12) is solved again at time $t+1$ in an MPC-like fashion [40].

The safety filter (12) is a nonlinear, non-convex, and stochastic optimization problem due to (12e) and (12f), and as a consequence in general not real-time implementable. Therefore, we reformulate (12) by convexifying the constraints and replacing the chance constraints by deterministic risk-tightened constraints, which we describe next.

C. Convexified constraints for probabilistic collective safety

We now present a convex, deterministic reformulation of (12e) that relies on well-known properties of Gaussian random vectors and the generated motion plan $\{x_i^{\mathrm{RL}}(k|t)\}_{k=t}^{t+T}$.

Lemma 1 (GAUSSIAN RANDOM VECTORS [41, SEC. 4.4.2])**.** *1) Let $N_I \in \mathbb{N}$. Given n-dimensional Gaussian random vectors $\boldsymbol{y}_i \sim \mathcal{N}(\overline{y}_i, \Sigma_{y_i})$ with $\overline{y}_i \in \mathbb{R}^n$, $\Sigma_{y_i} \in \mathbb{R}^{n\times n}$, and matrices $Y_i \in \mathbb{R}^{m\times n}$ for each $i \in \mathbb{N}_{[1,N_I]}$, then the random vector $\boldsymbol{y} = \sum_{i=1}^{N_I} Y_i \boldsymbol{y}_i$ is also Gaussian, with*

$$\boldsymbol{y} \sim \mathcal{N}\left(\sum\nolimits_{i=1}^{N_I} Y_i \overline{y}_i, \sum\nolimits_{i=1}^{N_I} Y_i \Sigma_{y_i} Y_i^\top\right). \tag{13}$$

2) Given $\boldsymbol{y} \sim \mathcal{N}(\overline{y}, \Sigma_y)$ with $\overline{y} \in \mathbb{R}^n, \Sigma_y \in \mathbb{R}^{n\times n}$, $a \in \mathbb{R}^n$, $b \in \mathbb{R}$, and the risk bound α, then

$$\mathbb{P}(a \cdot \boldsymbol{y} \leq b) \leq \alpha \iff a \cdot \overline{y} \geq b - \|\Sigma_y^{1/2} a\| \Phi^{-1}(\alpha), \tag{14a}$$

$$\mathbb{P}(a \cdot \boldsymbol{y} \geq b) \leq \alpha \iff a \cdot \overline{y} \leq b - \|\Sigma_y^{1/2} a\| \Phi^{-1}(1-\alpha), \tag{14b}$$

where Φ^{-1} is the inverse cumulative distribution function of a standard Gaussian random variable.

From (12c) and (4), $\boldsymbol{x}(t) \sim \mathcal{N}(\hat{y}_i(t), \Sigma_\eta)$. For every agent $i \in \mathbb{N}_{[1,N]}$, the predicted state and position at any time $k > t$,

$$\boldsymbol{x}_i(k|t) \sim \mathcal{N}(\overline{x}_i(k|t), \Sigma_{x_i}(k|t)), \tag{15a}$$

$$\boldsymbol{p}_i(k|t) \sim \mathcal{N}(\overline{p}_i(k|t), \Sigma_{p_i}(k|t)), \tag{15b}$$

$$\overline{x}_i(k|t) = A^{k-t}\hat{y}_i(t) + \sum_{j=t}^{k-1} A^{k-(j+1)} B u_i^{\mathrm{safe}}(j|t), \tag{15c}$$

$$\overline{p}_i(k|t) = C\overline{x}_i(k|t), \tag{15d}$$

$$\Sigma_{x_i}(k+1|t) = A\Sigma_{x_i}(k|t)A^\top + \Sigma_w, \tag{15e}$$

$$\Sigma_{p_i}(k|t) = C\Sigma_{x_i}(k|t)C^\top. \tag{15f}$$

using the stochastic dynamics (1), (12c), and (14a) in Lemma 1. We observe that $\overline{x}_i(k|t)$ and $\overline{p}_i(k|t)$ depend on the decision variables u_i^{safe}, but $\Sigma_{x_i}(k|t)$ and $\Sigma_{p_i}(k|t)$ do not. Thus, $\Sigma_{x_i}(k|t)$ and $\Sigma_{p_i}(k|t)$ may be computed offline.

Proposition 1 (RISK-TIGHTENED SUFFICIENT SAFETY CONSTRAINTS)**.** *Given a polytope $\mathcal{K} = \cap_{i\in\mathbb{N}_{[1,N_\mathcal{K}]}} \{p \in \mathbb{R}^d : h_i \cdot p \leq g_i\}$ with $N_\mathcal{K} \in \mathbb{N}$ halfspaces characterized by $\{h_i, g_i\}_{i=1}^{N_\mathcal{K}}$, $h_i \in \mathbb{R}^d$ and $g_i \in \mathbb{R}$, and user-defined unit vectors $z_{ij}^{obs}, z_{ij}^{agt} \in \mathbb{R}^d$. Then, for every $i \in \mathbb{N}_{[1,N]}$ and $k \in \mathbb{N}_{[t,t+T]}$,* (16) *is sufficient for* (5)*,* (6)*, and* (7) *to hold.*

We provide the proof of Proposition 1 in Appendix A.

The reformulation in Proposition 1 follows from applying computational geometry arguments to convexify the chance constraints (5)–(7) using supporting hyperplanes defined by user-specified vectors $z_{ij}^{\mathrm{obs}}, z_{ij}^{\mathrm{agt}}$, and then applying Lemma 1 and Boole's law to arrive at (16). From (15c) and (15d), the constraints in Proposition 1 are linear inequalities in the decision variables $\{u_i^{\mathrm{safe}}(k|t)\}_{k=t}^{t+T-1}$ for every $i \in \mathbb{N}_{[1,N]}$. Figure 3 illustrates the reformulated constraints of Proposition 1.

$$\forall j \in \mathbb{N}_{[1,N_O]}, \quad z_{ij}^{\text{obs}} \cdot (\overline{p}_i(k|t) - \overline{c}_j) \geq S_{\mathcal{O}_j}(z_{ij}^{\text{obs}}) + S_{(-\mathcal{A})}(z_{ij}^{\text{obs}}) - \|(\Sigma_{p_i}(k|t) + \Sigma_{c_j})^{1/2} z_{ij}^{\text{obs}}\| \Phi^{-1}(\alpha_{i,j,t}), \tag{16a}$$

$$\forall j \in \mathbb{N}_{[1,i-1]}, \quad z_{ij}^{\text{agt}} \cdot (\overline{p}_i(k|t) - \overline{p}_j(k|t)) \geq S_{\mathcal{A}}(z_{ij}^{\text{agt}}) + S_{(-\mathcal{A})}(z_{ij}^{\text{agt}}) - \|(\Sigma_{p_i}(k|t) + \Sigma_{p_j}(k|t))^{1/2} z_{ij}^{\text{agt}}\| \Phi^{-1}(\beta_{i,j,t}), \tag{16b}$$

$$\forall j \in \mathbb{N}_{[1,N_\mathcal{K}]}, \quad h_j \cdot \overline{p}_i(k|t) \leq g_j - S_{\mathcal{A}}(h_j) - \|\Sigma_{p_i}^{1/2}(k|t) h_j\| \Phi^{-1}\left(1 - (\kappa_{i,t}/N_\mathcal{K})\right). \tag{16c}$$

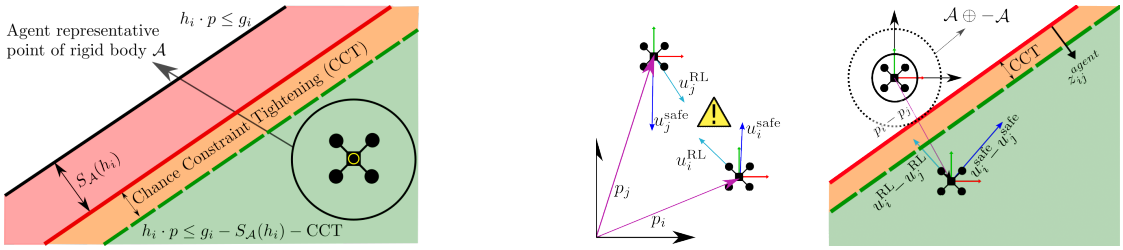

Fig. 3. Probabilistic collective safety constraints (Definition 1) enforced as linear constraints — (Left) Keep-in constraint (black) tightened by the support of $\mathcal{A}$ (red). (Right) Inter-agent collision avoidance constraint uses the support of $\mathcal{A} \oplus -\mathcal{A}$ (red). Both red constraints are tightened by the chance constraint term resulting in a new constraint (dashed green).

D. Ensuring recursive feasibility using reachability

We now turn our attention to (12f) that is designed to ensure that (12) remains feasible in subsequent control time steps. For recursive feasibility (12f), we enforce the existence of a terminal set and a control $u_i^{\text{recurse}}(k) \in \mathcal{U}$ for all $k \geq t+T$ for each agent i such that the following constraints hold for all $k \geq t+T$,

$$\mathbb{P}((\boldsymbol{p}_i(k|t) \oplus \mathcal{A}) \cap (\boldsymbol{c}_j \oplus \mathcal{O}_j) \neq \emptyset) \leq \delta, \tag{17a}$$

$$\mathbb{P}((\boldsymbol{p}_i(k|t) \oplus \mathcal{A}) \cap (\boldsymbol{p}_j(k|t) \oplus \mathcal{A}) \neq \emptyset) \leq \delta, , \tag{17b}$$

$$\mathbb{P}((\boldsymbol{p}_i(k|t) \oplus \mathcal{A}) \not\subseteq \mathcal{K}) \leq \delta, \tag{17c}$$

where $\delta \in (0,1)$ is a (small) user-specified risk threshold.

Existing literature in constrained control typically enforces recursive feasibility using control invariant or positive invariant sets [40], [42]. However, characterization of such sets can be challenging in our setting due to the inherent non-convexity of the probabilistic collective safety constraints. Alternatively, one can approximately enforce these constraints by truncating the recursive feasibility criterion to a finite but long horizon, and then utilizing stochastic reachability [29], [43], [44].

For the sake of tractability, we enforce (17) approximately by imposing chance constraints on the terminal states, while ignoring the stochasticity in the future time steps. We characterize these constraints using appropriately defined *avoid sets* (also known as *inevitable collision states* [29] or *capture sets* [45]) and *viability sets* (also known as *controlled invariant sets*) [40], [46].

Definition 2 (AVOID SET AND VIABILITY SET [40])**.** *For a (bad) set $\mathscr{B} \subset \mathbb{R}^n$, linear dynamics* (3)*, and a control constraint set $\mathcal{U}$, we define an avoid set as follows,*

$$\text{AvoidSet}(\mathscr{B}) = \left\{ \bar{x}(0) \,\middle|\, \begin{array}{c} \exists t \in \mathbb{N},\ \forall u(t) \in \mathcal{U}, \\ \bar{x}(t+1) = A\bar{x}(t) + Bu(t) \in \mathscr{B} \end{array} \right\}$$

For a (good) set $\mathscr{G} \subset \mathbb{R}^n$, we define a viability set as follows,

$$\text{ViabilitySet}(\mathscr{G}) = \left\{ \bar{x}(0) \,\middle|\, \begin{array}{c} \forall t \in \mathbb{N},\ \exists u(t) \in \mathcal{U}, \\ \bar{x}(t+1) = A\bar{x}(t) + Bu(t) \in \mathscr{G} \end{array} \right\}.$$

By construction, we have

$$\mathbb{R}^n \setminus \text{ViabilitySet}(\mathscr{G}) = \text{AvoidSet}(\mathbb{R}^n \setminus \mathscr{G}). \tag{18}$$

Informally, $\text{AvoidSet}(\mathscr{B})$ is the set of mean initial states from which the mean trajectory of (3) enters the bad set $\mathscr{B}$ at some time t, irrespective of the control choices. On the other hand, $\text{ViabilitySet}(\mathscr{G})$ is the set of mean initial states from which the mean trajectory of (3) remains within the good set $\mathscr{G}$ for all time t, by some appropriate choice of control actions.

For any time t, assume that the agents evolve by stochastic dynamics (1) with imperfect measurements according to (4) during the planning interval ($k \in \mathbb{N}_{[t,t+T-1]}$), and they evolve by nominal dynamics (3) with perfect measurements beyond the planning horizon ($k \geq t+T$). Since the safety filter solves (12) at every t based on the new measurement, the impact of this assumption is mild for sufficiently long planning horizon T. Under this assumption, we construct the following approximation of (17) using Definition 2,

$$\mathbb{P}((\boldsymbol{x}_i(k+T|k) - \boldsymbol{c}_j^{\text{lift}}(k+T|k)) \in \text{AvoidSet}(\mathcal{O}_j \oplus (-\mathcal{A}))) \leq \delta, \tag{19a}$$

$$\mathbb{P}((\boldsymbol{x}_i(k+T|k) - \boldsymbol{x}_j(k+T|k)) \in \text{AvoidSet}(\mathcal{A} \oplus (-\mathcal{A}))) \leq \delta, \tag{19b}$$

$$\mathbb{P}(\boldsymbol{x}_i(k+T|k) \notin \text{ViabilitySet}(\mathcal{K} \ominus \mathcal{A})) \leq \delta, \tag{19c}$$

where $\boldsymbol{c}_j^{\text{lift}} \in \mathbb{R}^n$ is obtained by lifting the position to $\mathbb{R}^n$ with added components set to zero, since the obstacles are static. (19c) uses (18) for ease in implementation. Informally, (19a) and (19b) require the agents to be in configurations that lead to collision with static obstacles or each other with at most a probability of δ, and (19c) requires the probability that the agents are in configurations from which they can not remain within the workspace is at most δ.

The constraints (19) are tractable when the sets $\text{AvoidSet}(\mathcal{A} \oplus (-\mathcal{A}))$, $\text{AvoidSet}(\mathcal{A} \oplus (-\mathcal{O}_j))$, and $\text{ViabilitySet}(\mathcal{K} \ominus \mathcal{A})$ are convex. Recall that $\text{AvoidSet}(\mathcal{B})$ is typically non-convex, even when

$$\forall j \in \mathbb{N}_{[1,N_O]}, \quad \ell_{ij}^{\text{obs}} \cdot (\overline{x}_i(t+T|t) - \overline{c}_j^{\text{lift}}) \geq S_{\mathscr{A}_{\mathcal{O}_j}}(\ell_{ij}^{\text{obs}}) - \|(\Sigma_{x_i}(t+T|t) + \Sigma_{c_j^{\text{lift}}})^{1/2} \ell_{ij}^{\text{obs}}\| \Phi^{-1}(\delta), \tag{20a}$$

$$\forall j \in \mathbb{N}_{[1,i-1]}, \quad \ell_{ij}^{\text{agt}} \cdot (\overline{x}_i(t+T|t) - \overline{x}_j(t+T|t)) \geq S_{\mathscr{A}_{\mathcal{A}}} - \|(\Sigma_{x_i}(t+T|t) + \Sigma_{x_j}(t+T|t))^{1/2} \ell_{ij}^{\text{agt}}\| \Phi^{-1}(\delta), \tag{20b}$$

$$\forall j \in \mathbb{N}_{[1,N_{\mathscr{V}}]}, \quad \underline{h}_j \cdot \overline{x}_i(t+T|t) \leq \underline{g}_j - \|\Sigma_{x_i}^{1/2}(t+T|t)\underline{h}_j\| \Phi^{-1}\left(1 - (\delta/N_{\mathscr{V}})\right). \tag{20c}$$

Algorithm 1: Computation of $\text{AvoidSet}^+(\mathscr{B})$ (See [40, Sec. 10.2] for recursion)

Input: Linear dynamics (3), control constraint set $\mathcal{U}$, convex and compact polytope $\mathscr{B}$.
Output: $\text{AvoidSet}^+(\mathscr{B})$.
1: *ListOfSets* $\leftarrow [\mathscr{B}]$, *CurrentSet* $\leftarrow \mathscr{B}$
2: **while** *CurrentSet* is non-empty
3: *CurrentSet* $\leftarrow A^{-1}(\textit{CurrentSet} \ominus B\mathcal{U})$
4: Append *CurrentSet* to *ListOfSets*
5: *ConvexHullOfList* $\leftarrow$ convex hull of *ListOfSets*
6: $\text{AvoidSet}^+(\mathscr{B}) \leftarrow$ minimum volume ellipsoid containing *ConvexHullOfList* (see [41, Sec. 8.4.1])

Algorithm 2: Computation of $\text{ViabilitySet}(\mathscr{G})$ [46]

Input: Linear dynamics (3), control constraint set $\mathcal{U}$, convex and compact polytope $\mathscr{G}$.
Output: $\text{ViabilitySet}(\mathscr{G})$.
1: *CurrentSet* $\leftarrow \mathscr{G}$, *PrevSet* $\leftarrow \emptyset$
2: **while** *CurrentSet* is not equal to *PrevSet*
3: *PrevSet* $\leftarrow$ *CurrentSet*
4: *CurrentSet* $\leftarrow \mathscr{G} \cap A^{-1}(\textit{CurrentSet} \oplus (-B\mathcal{U}))$
5: $\text{ViabilitySet}(\mathscr{G}) \leftarrow$ *CurrentSet*

$\mathscr{B} \in \{\mathcal{A} \oplus (-\mathcal{A}), \mathcal{A} \oplus (-\mathcal{O}_j)\}$ is a convex polytope [40]. This complicates the enforcement of (19a) and (19b). For the sake of tractability and ensuring conservativeness, we propose Algorithm 1 to compute an ellipsoidal outer-approximation of $\text{AvoidSet}(\mathcal{B})$. Outer-approximations of AvoidSet are also sufficient to enforce (19a) and (19b). On the other hand, Algorithm 2 provides an exact approach to compute $\text{ViabilitySet}(\mathcal{K} \ominus \mathcal{A})$ for convex and compact polytopes $\mathcal{K}$ and $\mathcal{A}$. All operations in Algorithms 1 and 2 can be easily accomplished using computational geometry tools and convex optimization, see [40], [41], [46], [47] for more details.

We conclude this section by characterizing a set of linear constraints that are sufficient to enforce (19). The proof of Proposition 2 uses the same arguments as that seen in Proposition 1.

Proposition 2 (RISK-TIGHTENED SUFFICIENT TERMINAL RECURSIVE FEASIBILITY CONSTRAINTS)**.** *Given a collection of user-defined unit vectors $\ell_{ij}^{obs}, \ell_{ij}^{agt} \in \mathbb{R}^n$, let $\mathscr{A}_{\mathcal{O}_j} \triangleq \text{AvoidSet}^+(\mathcal{O}_j \oplus (-\mathcal{A}))$ and $\mathscr{A}_{\mathcal{A}} \triangleq \text{AvoidSet}^+(\mathcal{A} \oplus (-\mathcal{A}))$ denote ellipsoidal outer-approximations of the corresponding avoid sets, and $\mathscr{V} \triangleq \text{ViabilitySet}(\mathcal{K} \ominus \mathcal{A})$ denote a polytope with $N_{\mathscr{V}}$ halfspace constraints, $\mathscr{V} = \cap_{i \in \mathbb{N}_{[1,N_{\mathscr{V}}]}} \left\{x \in \mathbb{R}^n : \underline{h}_i \cdot x \leq \underline{g}_i\right\}$. Then, for every $i \in \mathbb{N}_{[1,N]}$,* (20) *is sufficient for* (19) *to hold.*

Proposition 2 characterizes linear constraints that are sufficient to enforce (19). For (19a) and (19b), the sufficient condition is obtained by tightening the complement of a supporting halfspace of the convex outer-approximations of the avoid set. For (19c), the sufficient condition utilizes Boole's inequality and uniform risk allocation.

E. Reformulated Risk-Tightened MPC Safety Filter

Following the reformulations discussed in Sections III-C and III-D, we obtain a quadratic program (21),

$$\begin{aligned} \underset{\{U_i^{\text{s}}(t)\}_{i=1}^N}{\text{minimize}} \quad & \text{(12a)} \\ \text{subject to} \quad & \text{(12d), (15c), (15d), (15e), (15f),} \\ & \text{(16) for every } i \in \mathbb{N}_{[1,N]},\ k \in \mathbb{N}_{[t,t+T]}, \\ & \text{(20) for every } i \in \mathbb{N}_{[1,N]}, \end{aligned} \tag{21}$$

where $U_i^{\text{s}}(t) = \{u_i^{\text{safe}}(k|t)\}_{k=t}^{t+T-1}$ for each $i \in \mathbb{N}_{[1,N]}$. (21) uses the mean states and positions of the agents, and the deterministic linear constraints characterized in Propositions 1 and 2 for probabilistic collective safety and recursive feasibility. A solution of (21) is a feasible (but not necessarily optimal) solution of (12).

From the setting of (21), it is evident that the specialization of the RL-based motion planner to the deterministic nominal dynamics (3) does not affect the proposed solution adversely. Motivated by the superposition principle, we have used a simpler RL-based motion planner that considers deterministic dynamics (3) instead of the stochastic dynamics (1), and delegated the responsibility of probabilistic collective safety under (1) to the safety filter.

F. Discussion

1) Choice of Dynamics: Our primary targets are quadrotors as we describe in Section IV. Waypoint tracking for quadrotors using on-board controllers is now well-known [17], [48]. Consequently, assuming linear dynamics (1) is appropriate since the safety filter can generate safe waypoints that deviates minimally from the RL-based motion plan.

Theoretically, it is possible to apply the proposed solution to nonlinear dynamics. However, the construction of terminal sets for collision avoidance and recursive feasibility, similar to the sets proposed in Section III-D, become more challenging [42] On the other hand, our approach achieves recursive feasibility in the presence of stochastic process noises. Using process noise to (conservatively) model the linearization error when using linear models for nonlinear dynamics, we can use the proposed approach to provide (conservative) safety guarantees.

2) Gaussian Noise Assumption: In our problem statement, we assumed Gaussian noise and imposed chance constraints. Alternatively, we can use other risk metrics based on axiomatic risk theory and more generalized noise distributions [49]–[51]. However, most of these approaches either do not admit closed-form deterministic reformulations resulting in high computational costs, or are overly conservative. For example, our assumptions on $\boldsymbol{w}$ and $\boldsymbol{\eta}$ having a Gaussian distribution can be relaxed to any probability distribution that has a pre-specified mean and covariance. In this case, the reformulated constraints are similar to (16) and (20) but with $\Phi^{-1}(\alpha)$ terms replaced by the Chebychev bound $\sqrt{\frac{1-\alpha}{\alpha}}$ [52]. However, the resulting deterministic sufficient conditions are far more conservative than those in Propositions 1 and 2 [53, Fig. 2].

3) Ellipsoidal Convex Set Usage: We recommend using ellipsoidal representations (or outer-approximations) for various convex sets, primarily due to the convexification step presented in Section III-D. Ellipsoids $\mathcal{E}(c,Q) = \{x|(x-c)\cdot(Q^{-1}(x-c)) \leq 1\}$ admit a closed form solution for the support function $\rho_{\mathcal{E}(c,Q)}(\ell) = \ell \cdot c + \sqrt{\ell \cdot (Q\ell)}$, and the supporting hyperplane changes smoothly along the set boundary with changing ℓ. Compared to that, the support function of a polytope requires solving a linear program, and may change abruptly when changing ℓ.

4) Safety Filtering with Other Motion Planners: We use the proposed safety filter (12) in conjunction with single-agent RL motion planning, since RL-based planners have become popular in recent literature (see discussion in Section I) but lack safety guarantees especially in terms of enforcing (hard) constraints. While the proposed combination of RL and safety filter can provide hard constraint satisfaction guarantees, the safety filter's applicability is not limited to single-agent RL-based planners. As illustrated in Figure 2 and as seen from the derivations, the safety filter only requires the agents' reference state and control trajectories. These inputs may also be obtained from many other planners, including more traditional ones such as sampling-based. For example, RRT-based planners [35], [36] can be used for single-agent motion planning while avoiding static obstacles in the environment. The multi-agent plans can then be obtained by combining separate single-agent RRT-based plans using the proposed safety filter to guarantee inter-agent collision avoidance. This allows more efficient computations and memory reduction with respect to applying sampling-based planning to the multi-agent problem due to the smaller dimension and reduced number of collisions to be checked.

5) Intermediate multi-agent RL-based planners: The proposed approach can also be applied to the intermediate case of a planner for multiple agents N_{few}, but less than the total number N. In this case, multiple planners generate plans each for N_{few} agents up to the total number N. Each group of N_{few} may be collision-free, but the safety filter is applied to ensure safety between agents in different groups. Overall, the fundamental idea behind our approach is to take a challenging motion planning problem, approximate it by a problem that is significantly simpler to solve, at the price of losing safety due to the approximation, and then recovering it by the safety filter. In the case of multi-agent planning, approximation is done by reducing the amount of agents, hence here we discussed the largest possible reduction that provides the largest simplification, that is only one agent is considered in planning, but the approach will also work for any intermediate case.

IV. Implementation Details and Experiment Setup

Dynamics: We used the Crazyflie 2.1 quadrotors [54] as our target platform. We flew all the quadrotors at the same height of 0.95 m to make the collision avoidance problem more challenging. While it would be possible to resolve collisions by flying the drones at different heights, this solution does not generalize to other systems where more spatial dimensions do not exist (e.g. ground robots), and would not scale well to increasing number of robots or physically constrained environments.

We approximated the 2D motion of the quadrotors using 2D double integrator dynamics, and thus, A, B, C are given by

$$A = \begin{bmatrix} I_2 & T_s I_2 \\ 0_{2,2} & I_2 \end{bmatrix},\ B = \begin{bmatrix} \frac{T_s^2}{2} I_2 \\ T_s I_2 \end{bmatrix},\ C = \begin{bmatrix} I_2 & 0_{2,2} \end{bmatrix}, \tag{22}$$

with sampling time $T_s = 0.1$. We model the quadrotors as circles ($\mathcal{A}$ is a circle of radius $r_A = 0.1$) to include the 0.092 m Crazyflie diameter as well as leave extra margin for aerodynamic effects and a safety padding.

Hardware setup: We used six quadrotors ($N = 6$) in our experiments. We relied on the Crazyswarm platform [48] to communicate and control the quadrotors at 10Hz. The drones are equipped with IR-reflective markers detected by an OptiTrack motion capture system running at 120Hz. The Crazyswarm package tracked the Crazyflies using the raw point-cloud data from the OptiTrack motion capture system, and it issued desired waypoints at a nominal 10 Hz update frequency over radio. The Crazyflies tracked those waypoints using their standard on-board controllers. In addition to the uncertainty in the Crazyflie position estimate induced by the Crazyswarm tracking algorithm, we added a position estimation noise $\boldsymbol{\eta}$ defined in (4). Such measurement noises affects the safety filter, but is not visualized in the plotted physical experiment trajectories.

Workspace: We considered a 3×3 meter workspace with seven circular obstacles and two goal regions. The obstacles are depicted by black circles and the goal regions are depicted by transparent circles with a star at the center (see Figure 7). We also added position estimation noise to the nominal obstacle locations.

Safety filter parameters: We used Gaussian noise with the following covariances: $\Sigma_w = \Sigma_\eta = \text{diag}(10^{-4}, 0, 10^{-4}, 0)$ and $\Sigma_{c_j} = \text{diag}(10^{-4}, 10^{-4})\ \forall j \in \mathbb{N}_{[1,N_O]}$. As for the risk bounds, we used $\kappa_i = \alpha_{i,j} = \beta_{i,i'} = 0.01$ and divided them equally across the planning horizon $T = 10$. We used $\delta = 0.1$ for the terminal constraints. For the purposes of constructing the terminal sets, we select velocity bounds of 1 m/s in the simulation, and 0.2 m/s in the experiments.

Computer setup: We used an Ubuntu 20.04 LTS workstation with an AMD Ryzen 9 9590X 16-core CPU, a Nvidia

GeForce GTX TITAN Black GPU, and 128GB of RAM for all training, simulation, and hardware experiments.

RL training: We used `Stable-Baselines3`'s implementation of the PPO (proximal policy optimization) algorithm [39] to train the RL agents. We ran two training sessions, one for each goal, for 10 million time steps each. We used the default parameters of `Stable-Baselines3` with the following modifications: 0.01 entropy coefficient, 2021 seed, and `cpu` device. We use $\zeta_{\text{obs}} = -0.001$, $\zeta_{\text{tgt}} = -0.1$, $R_{\text{target}} = 10^4$, $P_{\text{keep-in}} = -10^4$, and $P_{\text{obstacle}} = -500$ for the reward function parameters.

After training, we selected the trained policy at about 9.7 million steps and 9.44 million steps for the two targets respectively. Each training session took just over 11 hours.

Choice of unit vectors in Proposition 1, 2: Inspired by [17], we used the following unit vectors:

$$z_{ij}^{\text{obs}}(k|t) \triangleq \frac{p_i^{\text{RL}}(k|t) - \bar{c}_j}{\|p_i^{\text{RL}}(k|t) - \bar{c}_j\|}, z_{ij}^{\text{agt}}(k|t) \triangleq \frac{p_i^{\text{RL}}(k|t) - p_j^{\text{RL}}(k|t)}{\|p_i^{\text{RL}}(k|t) - p_j^{\text{RL}}(k|t)\|} \quad (23a)$$

$$\ell_{ij}^{\text{obs}}(k|t) \triangleq \frac{x_i^{\text{RL}}(k|t) - \bar{c}_j^{\text{lift}}}{\|x_i^{\text{RL}}(k|t) - \bar{c}_j^{\text{lift}}\|}, \ell_{ij}^{\text{agt}}(k|t) \triangleq \frac{x_i^{\text{RL}}(k|t) - x_j^{\text{RL}}(k|t)}{\|x_i^{\text{RL}}(k|t) - x_j^{\text{RL}}(k|t)\|} \quad (23b)$$

where $p_i^{\text{RL}}(k|t) = Cx_i^{\text{RL}}(k|t)$. Such a choice used the predicted RL states and positions and the nominal obstacle locations to produce a heuristic for the computation of the safe halfspace polytopes. For the 2D double integrator dynamics (22), the lifted state is the position vector with zeros appended for the velocity components, i.e. $\bar{c}_j^{\text{lift}} = [\bar{c}_j^\top \ 0_2^\top]^\top$.

Solving the QP: We modeled the QP associated with the safety filter in Python 3.7 using CVXPY [55], utilizing parameters for values in (21) that change at every control time step, and solved it using ECOS [56] in experiments, and `GUROBI` [57] in simulations.

V. Experiments

We present results of the experimental validation of our approach on a quadrotor testbed. We show that the trained, single-agent RL-based motion planner generalizes well when used with the proposed safety filter. We also compare the proposed approach with a MPC-based multi-agent motion planner in simulation to emphasize the benefits of the RL step as well as the effects of the terminal constraints. We conclude with a demonstration of the scalability of our approach.

A. Experimental validation

Figure 4 shows snapshots of two experiments and their reconstructed plots. In these experiments, we compare the proposed solution with a safety-filtered baseline controller. Here, the baseline controller is a proportional controller that regulates the drones to the target while ignoring all static and dynamic obstacles, which are handled by the safety filter (21).

In Figure 4, the top two rows are for the proposed solution with the RL controller and the proposed safety filter (21) while the bottom two rows use the baseline controller instead of the RL controller. In both cases, the proposed safety filter ensures that the agents remain safe. In the RL case, the agents manage to reach their goals more rapidly, while the baseline controller case, the agents take significantly longer to reach their goals. In fact, when using the baseline controller instead of the RL controller, we found that the pink agent typically gets stuck between two obstacles and fails to reach its goal (see the bottom two rows of Figure 4).

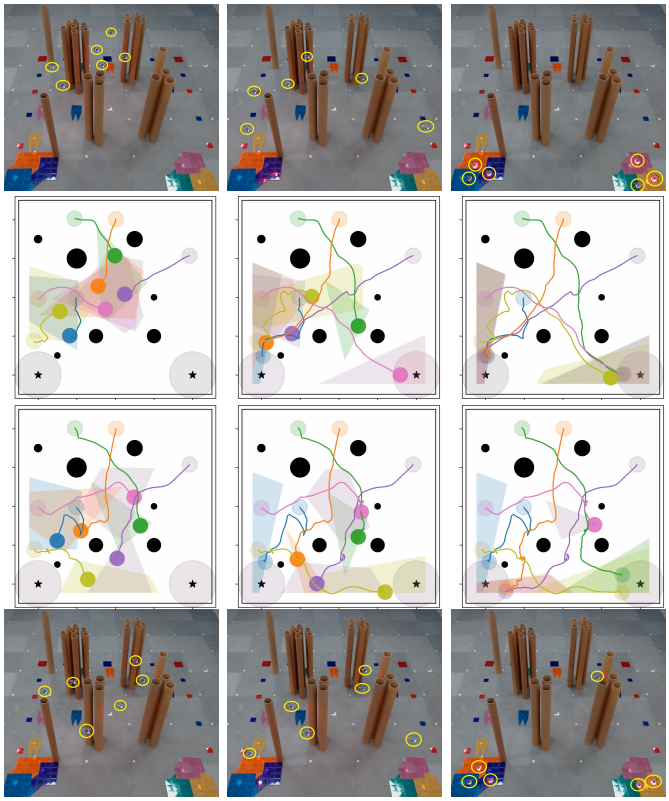

Fig. 4. Safe multi-agent motion planning using the proposed safety filter in conjunction with the RL-based controller and a classical proportional controller (baseline). (Top two rows) Snapshots and reconstructed illustrations of the hardware experiment's trajectories when using the RL-based controller with the safety filter at 7, 14, and 21 seconds. (Bottom two rows) Trajectories of the hardware experiment when using the baseline controller with the safety filter at times 13, 21, and 70 seconds. The black circles and boundary are the obstacles and keep-in set. Transparent starred circles depict the targets. Colored circles denote the agents' starting and goal positions. The colored paths indicate the trajectory and the shaded regions are the static obstacle-free positions at the current control time step (determined via convexification).

Figure 5 shows the clearances between each agent (15 pairs for the six agents) during the physical experiment. Specifically, it plots the inter-agent distances *minus* twice the agent radius, i.e. $\|p_i(t) - p_j(t)\| - 2r \ \forall i, j, i \neq j$. Thus, a negative distance indicates a collision. Due to the use of probabilistic constraints, the distances are always positive, which shows that the system is collectively safe.

Figure 6 shows the QP setup time (blue) and the total time for setting up and solving the QP (orange) over the RL experiment's duration. The total time spent setting up and solving (21) for six agents was on average 0.05 seconds. Since the time spent was always less than 0.06 seconds, we had a sufficient margin to the 0.1 control sampling period.

Figure 7 shows the reconstruction of the agent trajectories for both the RL and baseline controllers based on the data collected during the experiments. As expected, the final trajectories for both RL and baseline controllers remain sufficiently far from the obstacles and the keep-in set bounds. While

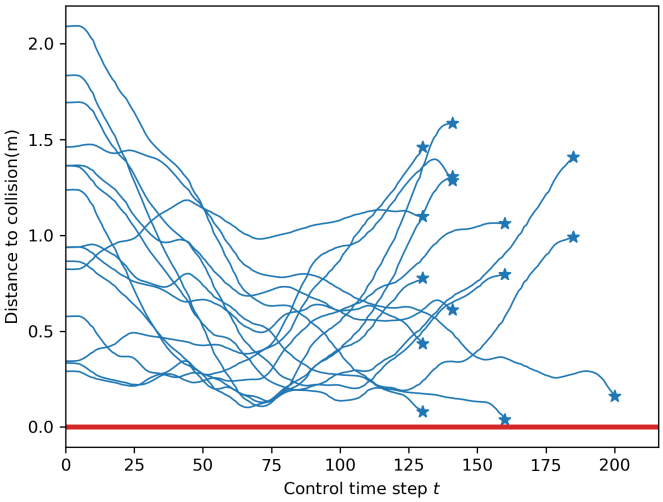

Fig. 5. Clearance between the agents during the physical experiment with RL controller, where a clearance (distance to collision) accounts for the physical dimensions of the agents. A negative clearance indicates a collision. Stars indicate one of the two agents reaching the target.

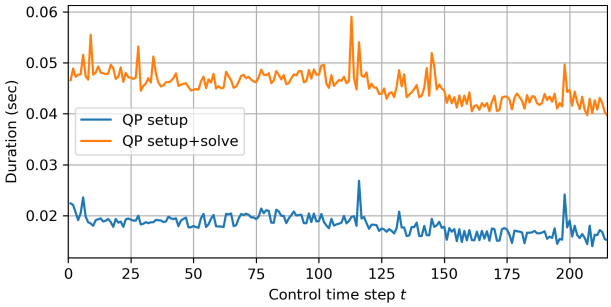

Fig. 6. Problem setup and solution durations to solve the quadratic program (21) in the experiment using CVXPY [55] and ECOS [56].

avoiding the red padding, which represents the enlargement of the obstacle rigid body by the agent's radius, is sufficient for collision avoidance, the chance constraints prevent the trajectories from getting too close and hence result in the additional virtual padding around the obstacles.

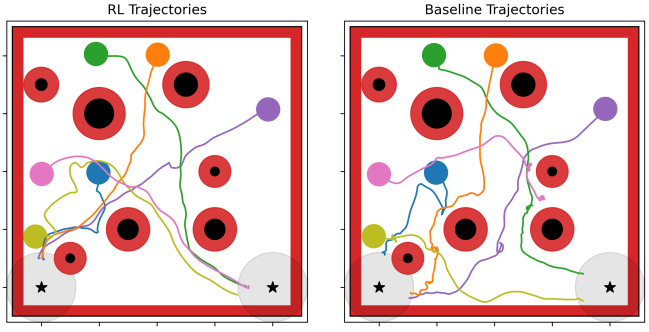

Fig. 7. Reconstruction of the RL (left) and baseline (right) trajectories from the experiments. The red padding around the keep-in set and obstacles, representing the agent radius, is never crossed and hence all trajectories are safe.

B. Evaluation of the RL motion planner

The deterministic evaluation of the learned policy over a 100×100 grid is presented in Figure 8 (top row).

We observe that the RL agents learned to navigate to the goal starting from most initial conditions. As expected, the learned policy is not perfect and sometimes results in collisions with the obstacles or the workspace (Rows 1 and 3). Nevertheless, the combination of RL and the safety filter ensures safe motion planning (Rows 2 and 4). For less than 2% of the initial conditions, the RL policy did not reach the target within 800 time steps (80 s), which we mark as "loiter", i.e., static/dynamic deadlock, but safety was still guaranteed. In practice, it is usually possible to recover from such deadlock conditions by small state perturbations. We plan to investigate formal methods for avoiding and recovering from such deadlocks in future studies.

C. Simulation study: Impact of RL and terminal constraints

Next, we compare our approach with a pure MPC-based motion planner in simulation. Specifically, we solved (21), where the objective (12a) is replaced with a set point regulation cost, which results in the optimization problem,

$$\begin{aligned} \min_{\{U_i^{\mathrm{s}}(t)\}_{i=1}^N} \quad & \textstyle\sum_{k=t}^{t+T} \sum_{i=1}^{N} \lambda_{i,k} \|\overline{p}_i(k|t) - q_i\|^2 + \varepsilon \|u_i^{\mathrm{safe}}(k|t)\|^2, \\ \text{s. t.} \quad & \text{Constraints of (21)}, \end{aligned} \tag{24}$$

with $U_i^{\mathrm{s}}(t) = \{u_i^{\mathrm{safe}}(k|t)\}_{k=t}^{t+T-1}$ for each $i \in \mathbb{N}_{[1,N]}$, pre-specified weights $\lambda_{i,t} \geq 0$ on the deviations $\|\overline{p}_i(k|t) - q_i\|^2$, and a penalty for inputs $\varepsilon > 0$.

Problem (24) is a convex quadratic program, thanks to the convexification step (Propositions 1 and 2) that uses a modified version of (23). Recall that the constraints of (21) included constraints (16) and (20) that required user-specified unit vectors $z_{ij}^{\mathrm{obs}}, z_{ij}^{\mathrm{agt}}, \ell_{ij}^{\mathrm{obs}}$, and ℓ_{ij}^{agt}, which were defined using the RL trajectory in (23). When formulating (24), we defined these vectors using the baseline controller trajectory instead of the RL trajectory for a fair comparison. One can view (24) as an extension of existing single-agent motion planners under uncertainty (for example, [27], [28]) for multi-agent motion planning, with the addition of terminal constraints for recursive feasibility proposed in Section III-D. Note that (24) enables explicit coordination between agents as they move towards their goal, while the proposed safety filter (21) only minimizes deviations from RL-based single-agent motion planners. We now study the RL block and the terminal constraints (Proposition 2) by comparing the performances of the proposed approach and a pure MPC approach (24), with and without terminal constraints (Proposition 2).

Table I summarizes the performance of both the approaches in 100 Monte-Carlo simulations. We observe that the proposed approach (21) completed the motion planning task for a significantly larger number of simulations than a pure MPC approach (24) (99% vs 55% success), illustrating the benefits of including RL. The sources of failure in these simulations include collisions with static or dynamic obstacles (safety is enforced in probability) as well as numerical issues for the

TABLE I

COMPARISON OF THE PROPOSED SAFETY FILTER (21) WITH A PURE MPC-BASED MOTION PLANNER (24). THE PROPOSED APPROACH COMPLETES THE MOTION PLANNING TASK FOR MORE PERCENTAGE OF TRIALS. WE REPORT THE $(5, 50, 95)$-PERCENTILES OF THE RESULTS OF THE SUBSET OF 100 MONTE-CARLO SIMULATIONS THAT COMPLETED THE TASK SUCCESSFULLY.

Safe controller	Term. const.	% Success	Task completion time	Min. obstacle separation	Min. agent separation
Proposed approach	Yes	99	(206, 236, 401)	(0.15, 0.21, 0.26)	(0.31, 0.32, 0.36)
RL + Safety filter (21)	No	92	(229, 325, 648)	(0.15, 0.19, 0.24)	(0.25, 0.27, 0.28)
Pure MPC (24)	Yes	Failed at control time step $(10, 10, 10)$			
	No	55	(110, 150, 194)	(0.15, 0.16, 0.17)	(0.23, 0.23, 0.23)

solver. For the proposed approach (21), the use of terminal constraints for recursive feasibility (Proposition 2) typically resulted in a larger minimum separation between agents and obstacles, and among agents. The use of terminal constraints also led to smaller task completion time, possibly due to the larger minimum separations. On the other hand, the use of similar terminal constraints in the MPC approach (24) made the problem considerably harder and led to numerical issues in all trials, possibly because the trajectory of the baseline controller may not be as informative as the RL trajectory for the convexification step. Finally, we observe that the proposed approach takes longer to complete the motion planning task than the pure MPC approach without the terminal constraints, when the latter does not result in safety violations. This is expected since the terminal constraints impose additional restriction on the generated trajectory to achieve recursive feasibility. The single agent motion planner combined with safety filter is suboptimal when applied to a multi-agent motion planning problem and is more conservative due to the terminal constraints, but guarantees safety. Thus, there is a trade-off between safety and performance.

D. Scalability study of the proposed approach

To perform scalability analysis of the safety filter, we reduced r_A to 0.01, reduced the noise covariance from 10^{-4} to 10^{-6}, and collected computational times for the safety filter for 1000 control time steps starting from randomly initialized locations for the agents in simulation.

Figure 9 shows the computation time to solve (21), where the number of agents ranges from 2 to 24. The compute time of the safety filter increases only moderately with the number of agents, thanks to the convex quadratic program structure of (21). Compared to our preliminary work in the deterministic setting [22], (21) needs a larger computational effort, possibly due to the larger number of decision variables and larger number of constraints. Specifically, (21) computes time-varying control commands over the planning horizon compared to constant input approach used in [22], and (21) includes additional constraints for recursive feasibility (20).

VI. CONCLUSION

We presented a solution for the multi-agent motion planning problem that combines reinforcement learning and constrained control. We utilize single-agent RL to train a policy for traversing a cluttered workspace while ignoring inter-agent collision avoidance, and use a real-time implementable, constrained-control-based safety filter to account for inter-agent collision avoidance and ensure probabilistic collective safety of the agents. The formulated QP includes chance constraints to achieve safety under process and measurement noise as well as probabilistic recursive feasibility constraints. We demonstrated the efficacy of our approach via numerical simulations, and validated our approach on a hardware testbed using quadrotors.

In our future work, we will investigate the application of the proposed approach in a decentralized setting, consider safe multi-agent motion planning for agents with nonlinear dynamics, and evaluate RL-based planning with a subset of the multiple agents larger than one.

APPENDIX

PROOF OF PROPOSITION 1

Static obstacle collision avoidance ((16a) $\Rightarrow$ (5))*:* Using computational geometry arguments, (5) is a non-convex chance constraint, and is equivalent to

$$\mathbb{P}(\boldsymbol{p}_i(k|t) - \boldsymbol{c}_j(t) \notin \mathcal{O}_j \oplus (-\mathcal{A})) \geq 1 - \alpha_{i,j,t}. \tag{25}$$

To convexify it, we use a separating hyperplane for $(\overline{p}_i(k|t) - \overline{c}_j)$ and $\mathcal{O}_j \oplus (-\mathcal{A})$ along the direction of a user-specified direction z_{ij}^{obs} [41]. Thus,

$$\begin{aligned}&\mathbb{P}(z_{ij}^{\text{obs}} \cdot (\boldsymbol{p}_i(k|t) - \boldsymbol{c}_j) \geq S_{\mathcal{O}_j}(z_{ij}^{\text{obs}}) + S_{-\mathcal{A}}(z_{ij}^{\text{obs}})) \geq 1 - \alpha_{i,j,t}\\ &\iff \mathbb{P}(z_{ij}^{\text{obs}} \cdot (\boldsymbol{p}_i(k|t) - \boldsymbol{c}_j) \leq S_{\mathcal{O}_j}(z_{ij}^{\text{obs}}) + S_{-\mathcal{A}}(z_{ij}^{\text{obs}})) \leq \alpha_{i,j,t}\\ &\implies (25).\end{aligned}$$

We use (14a) in Lemma 1 to reformulate the left hand side of the above implication to arrive at (16a). Thus, (5) holds, if (16a) holds.

Inter-agent collision avoidance ((16b) $\Rightarrow$ (6))*:* Using arguments similar to the above with $z_{ij}^{\text{agt}}, \overline{p}_j(k|t), \Sigma_{p_j}(k|t)$ instead of $z_{ij}^{\text{obs}}, \overline{c}_j, \Sigma_{c_j}$, we can show that (6) holds, if (16b) holds.

Keep-in constraint ((16c) $\Rightarrow$ (7))*:* From the definition of Pontryagin difference, (7) is equivalent to

$$\mathbb{P}(\boldsymbol{p}_i(k|t) \notin \mathcal{K} \ominus \mathcal{A}) \leq \kappa_{i,t}. \tag{26}$$

Here, $\mathcal{K} \ominus \mathcal{A}$ is easy to compute [37, Thm 2.3]. Specifically, $\mathcal{K} \ominus \mathcal{A} = \cap_{i \in \mathbb{N}_{[1,N_{\mathcal{K}}]}} \{p : h_i \cdot p \leq g_i - S_{\mathcal{A}}(h_i)\}$. Using Boole's inequality and assuming that the risk bound is divided equally across all halfspaces, we have

$$\mathbb{P}(h_j \cdot \boldsymbol{p}_i(k|t) > g_j - S_{\mathcal{A}}(h_j)) \leq \frac{\kappa_{i,t}}{N_{\mathcal{K}}},\ \forall j \in \mathbb{N}_{[1,N_{\mathcal{K}}]} \Rightarrow (26).$$

We use (14b) in Lemma 1 to reformulate the left hand side of the above implication to arrive at (16c). Thus, (7) holds, if (16c) holds. □

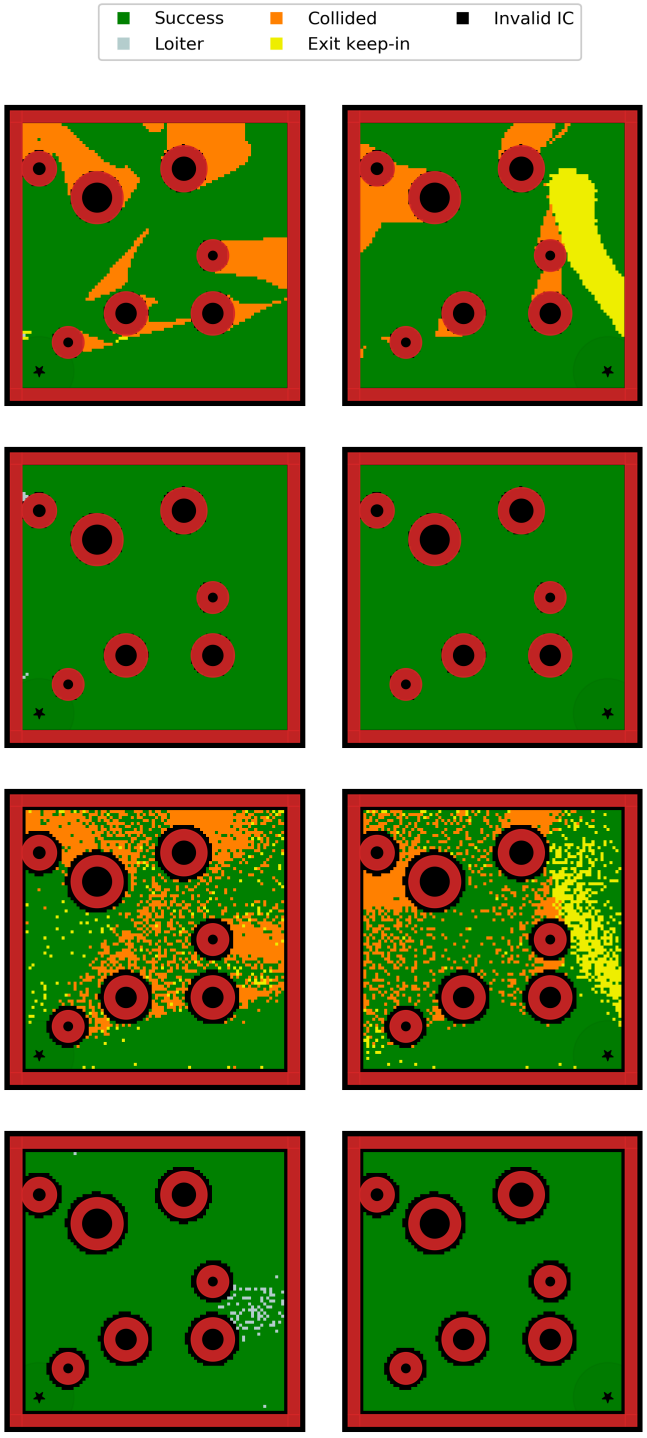

Fig. 8. Evaluation of the learned policy for a single agent over a 100×100 grid of initial positions. (Left column) policy for target 1. (Right column) policy for target 2. From top to bottom: (Row 1) RL policy, no noise; (Row 2) RL+Filter, no noise; (Row 3) RL policy, with noise; (Row 4) RL+Filter, with noise. We observe that the combination of safety filter and single-agent RL controller achieves the highest generalization, with and without noise.

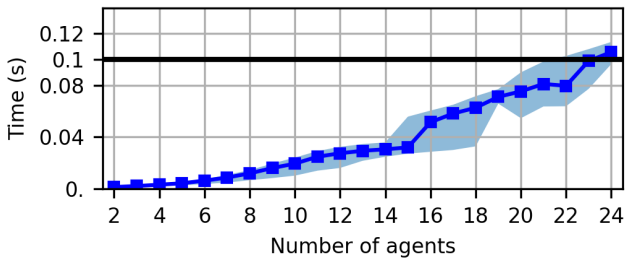

Fig. 9. Computation times (in $(5, 50, 95)$ percentiles) of the safety filter show a modest increase as the number of agents increases. The computation times were collected from 1000 control time steps of the simulated workspace. We used `GUROBI` [57] to solve the quadratic program (21).

References

[1] H. Qie, D. Shi, T. Shen, X. Xu, Y. Li, and L. Wang, "Joint optimization of multi-uav target assignment and path planning based on multi-agent reinforcement learning," *IEEE Access*, pp. 146 264–146 272, 2019.

[2] L. Lv, S. Zhang, D. Ding, and Y. Wang, "Path planning via an improved DQN-based learning policy," *IEEE Access*, pp. 67 319–67 330, 2019.

[3] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6382–6393.

[4] M. Srinivasan, A. Chakrabarty, R. Quirynen, N. Yoshikawa, T. Mariyama, and S. Di Cairano, "Fast multi-robot motion planning via imitation learning of mixed-integer programs," in *Proceedings of IFAC Modeling, Estimation and Control Conference (MECC)*, 2021.

[5] M. Everett, Y. Chen, and J. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," in *IEEE Int. Conf. Intell. Robots Syst (IROS)*, 2018, pp. 3052–3059.

[6] S. Semnani, H. Liu, M. Everett, A. de Ruiter, and J. How, "Multi-agent motion planning for dense and dynamic environments via deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3221–3226, 2020.

[7] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[8] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick, "End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 3387–3395.

[9] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," *Handbook of Reinforcement Learning and Control*, pp. 321–384, 2021.

[10] B. Gravell and T. Summers, "Centralized collision-free polynomial trajectories and goal assignment for aerial swarms," *Control Engineering Practice*, vol. 109, p. 104753, 2021.

[11] W. Hönig, J. Preiss, G. Kumar, Tand Sukhatme, and N. Ayanian, "Trajectory planning for quadrotor swarms," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 856–869, 2018.

[12] M. Ragaglia, M. Prandini, and L. Bascetta, "Multi-agent poli-rrt," in *International Workshop on Modelling and Simulation for Autonomous Systems*. Springer, 2016, pp. 261–270.

[13] A. Sud, R. Gayle, E. Andersen, S. Guy, M. Lin, and D. Manocha, "Real-time navigation of independent agents using adaptive roadmaps," in *ACM SIGGRAPH 2008 classes*, 2008, pp. 1–10.

[14] D. Zhou, Z. Wang, S. Bandyopadhyay, and M. Schwager, "Fast, on-line collision avoidance for dynamic vehicles using buffered voronoi cells," *IEEE Robotics and Automation Letters*, vol. 2, pp. 1047–1054, 2017.

[15] M. Radmanesh and M. Kumar, "Flight formation of uavs in presence of moving obstacles using fast-dynamic mixed integer linear programming," *Aerospace Science and Technology*, vol. 50, pp. 149–160, 2016.

[16] Y. Chen, M. Cutler, and J. How, "Decoupled multiagent path planning via incremental sequential convex programming," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2015, pp. 5954–5961.

[17] F. Augugliaro, A. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 1917–1922.

[18] C. Verginis, Z. Xu, and D. V. Dimarogonas, "Decentralized motion planning with collision avoidance for a team of uavs under high level goals," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2017, pp. 781–787.

[19] L. Wang, A. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Trans. Robot.*, vol. 33, no. 3, pp. 661–674, 2017.

[20] M. Srinivasan, S. Coogan, and M. Egerstedt, "Control of multi-agent systems with finite time control barrier certificates and temporal logic," in *IEEE Conf. Decision Control*, 2018, pp. 1991–1996.

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

[22] A. P. Vinod, S. Safaoui, A. Chakrabarty, R. Quirynen, N. Yoshikawa, and S. Di Cairano, "Safe multi-agent motion planning via filtered reinforcement learning," in *2022 IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2022, pp. 7270–7276.

[23] A. Rodionova, Y. V. Pant, K. Jang, H. Abbas, and R. Mangharam, "Learning-to-fly: Learning-based collision avoidance for scalable urban air mobility," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–8.

[24] A. Rodionova, Y. Pant, C. Kurtz, K. Jang, H. Abbas, and R. Mangharam, "Learning-'n-flying: A learning-based, decentralized mission-aware UAS collision avoidance scheme," *ACM Transactions on Cyber-Physical Systems (TCPS)*, vol. 5, no. 4, pp. 1–26, 2021.

[25] Z. Cai, H. Cao, W. Lu, L. Zhang, and H. Xiong, "Safe multi-agent reinforcement learning through decentralized multiple control barrier functions," *arXiv preprint arXiv:2103.12553*, 2021.

[26] I. ElSayed-Aly, S. Bharadwaj, C. Amato, R. Ehlers, U. Topcu, and L. Feng, "Safe multi-agent reinforcement learning via shielding," in *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 2021, pp. 483–491.

[27] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Trans. Robot.*, vol. 27, no. 6, pp. 1080–1094, 2011.

[28] A. Vinod, S. Rice, Y. Mao, M. Oishi, and B. Açıkmeşe, "Stochastic motion planning using successive convexification and probabilistic occupancy functions," in *IEEE Conf. Decision Control*, 2018, pp. 4425–4432.

[29] N. Malone, H.-T. Chiang, K. Lesser, M. Oishi, and L. Tapia, "Hybrid dynamic moving obstacle avoidance using a stochastic reachable set-based potential field," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1124–1138, 2017.

[30] K. P. Wabersich and M. N. Zeilinger, "Linear model predictive safety certification for learning-based control," in *2018 IEEE Conf. Decision Control*. IEEE, 2018, pp. 7130–7135.

[31] ——, "A predictive safety filter for learning-based control of constrained nonlinear dynamical systems," *Automatica*, vol. 129, p. 109597, 2021.

[32] B. Tearle, K. P. Wabersich, A. Carron, and M. N. Zeilinger, "A predictive safety filter for learning-based racing control," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7635–7642, 2021.

[33] H. Zhu, B. Brito, and J. Alonso-Mora, "Decentralized probabilistic multi-robot collision avoidance using buffered uncertainty-aware voronoi cells," *Autonomous Robots*, pp. 1–20, 2022.

[34] R. Firoozi, R. Quirynen, and S. Di Cairano, "Coordination of autonomous vehicles and dynamic traffic rules in mixed automated/manual traffic," in *American Control Conference*, 2022.

[35] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Research Report 9811*, 1998.

[36] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[37] I. Kolmanovsky and E. G. Gilbert, "Theory and computation of disturbance invariant sets for discrete-time linear systems," *Mathematical problems in engineering*, vol. 4, no. 4, pp. 317–367, 1998.

[38] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[39] A. Raffin, A. Hill, M. Ernestus, A. Gleave, A. Kanervisto, and N. Dormann, "Stable baselines3," *GitHub repository*, 2019.

[40] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[41] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[42] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Systems Magazine*, vol. 36, no. 6, pp. 30–44, 2016.

[43] S. Summers and J. Lygeros, "Verification of discrete time stochastic hybrid systems: A stochastic reach-avoid decision problem," *Automatica*, vol. 46, no. 12, pp. 1951–1961, 2010.

[44] A. Vinod and M. Oishi, "Stochastic reachability of a target tube: Theory and computation," *Automatica*, vol. 125, 2021.

[45] J.-P. Aubin, A. M. Bayen, and P. Saint-Pierre, *Viability theory: new directions*. Springer Science & Business Media, 2011.

[46] J. N. Maidens, S. Kaynama, I. M. Mitchell, M. M. Oishi, and G. A. Dumont, "Lagrangian methods for approximating the viability kernel in high-dimensional systems," *Automatica*, vol. 49, no. 7, pp. 2017–2029, 2013.

[47] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *Proc. of the Eur. Control Conf.*, Zürich, Switzerland, July 17–19 2013, pp. 502–510, http://control.ee.ethz.ch/~mpt.

[48] J. A. Preiss, W. Honig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A large nano-quadcopter swarm," in *2017 IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2017, pp. 3299–3304.

[49] A. Majumdar and M. Pavone, "How should a robot assess risk? towards an axiomatic theory of risk in robotics," in *Robotics Research*. Springer, 2020, pp. 75–84.

[50] S. Safaoui, B. J. Gravell, V. Renganathan, and T. H. Summers, "Risk-averse RRT* planning with nonlinear steering and tracking controllers for nonlinear robotic systems under uncertainty," in *IEEE/RSJ International Conference on Intelligent Robots and Systems,*. IEEE, 2021, pp. 3681–3688.

[51] L. Lindemann, G. J. Pappas, and D. V. Dimarogonas, "Reactive and risk-aware control for signal temporal logic," *IEEE Trans. Autom. Control*, 2021.

[52] G. C. Calafiore and L. El Ghaoui, "On distributionally robust chance-constrained linear programs," *Journal of Optimization Theory and Applications*, vol. 130, no. 1, pp. 1–22, 2006.

[53] M. Farina, L. Giulioni, and R. Scattolini, "Stochastic linear model predictive control with chance constraints–a review," *Journal of Process Control*, vol. 44, pp. 53–67, 2016.

[54] "Crazyflie 2.1." [Online]. Available: https://www.bitcraze.io/products/crazyflie-2-1/

[55] S. Diamond and S. Boyd, "CVXPY: A python-embedded modeling language for convex optimization," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2909–2913, 2016.

[56] A. Domahidi, E. Chu, and S. Boyd, "ECOS: An SOCP solver for embedded systems," in *Eur. Control Conf.*, 2013, pp. 3071–3076.

[57] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2021. [Online]. Available: https://www.gurobi.com

Sleiman Safaoui (Member, IEEE) received the B.S. and M.S. degrees in Electrical Engineering from the University of Texas at Dallas, Richardson, TX, USA, in 2019 and 2023, respectively. He completed his Ph.D. degree in Electrical Engineering at the University of Texas at Dallas, Richardson, TX, USA as a Research Assistant with the Control, Optimization, and Networks Lab (CONLab) in 2023. He is currently a Research Associate with CONLab. From Aug 2021 to March 2022, he was an intern at Mitsubishi Electric Research Laboratories (MERL) where he worked on ground and aerial vehicle autonomy research projects. His current research interests include risk-based motion planning and control for robotic systems under uncertainty, autonomous vehicles, and multirobot systems.

Abraham P. Vinod (Member, IEEE) received the B.Tech. and M.Tech. degrees in electrical engineering from the Indian Institute of Technology, Madras (IITM), Chennai, India, in 2014 and the Ph.D. degree in electrical engineering from the University of New Mexico, Albuquerque, NM, USA, in 2018. He is currently working as a Research Scientist at Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA. His research interests are in the areas of optimization, stochastic control, multi-agent systems, and learning. Dr. Vinod was awarded the Best Student Paper Award in the 2017 ACM Hybrid Systems: Computation and Control Conference, and was the finalist for the Best Paper Award in the 2018 ACM Hybrid Systems: Computation and Control Conference.

Ankush Chakrabarty (Senior Member, IEEE) was awarded the Ross Fellowship and received the Ph.D. degree in Electrical and Computer Engineering from Purdue University, West Lafayette, IN, USA, in 2016. He was a Postdoctoral Fellow with Harvard University, Cambridge, MA, USA, from 2016 to 2018, where he worked on the conceptualization and development of an embedded artificial pancreas system. He has been with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, since 2018, where he is currently a Principal Research Scientist. His research lies in the intersection of machine learning and control engineering for digital twins of building energy systems. He has an Erdös number of 4.

Rien Quirynen received the Bachelor's degree in computer science and electrical engineering and the Master's degree in mathematical engineering from KU Leuven, Belgium. He received a four-year Ph.D. Scholarship from the Research Foundation–Flanders (FWO) in 2012-2016, and the joint Ph.D. degree from KU Leuven, Belgium and the University of Freiburg, Germany. He worked as a senior research scientist at Mitsubishi Electric Research Laboratories in Cambridge, MA, USA until 2023. Since 2023, Rien is a staff software engineer at Stack AV. His research focuses on numerical optimization algorithms for decision making, motion planning and predictive control of autonomous systems. He has authored/coauthored more than 75 peer-reviewed papers in journals and conference proceedings and 25 patents. Dr. Quirynen serves as an Associate Editor for the Wiley journal of Optimal Control Applications and Methods and for the IEEE CCTA Editorial Board.

Nobuyuki Yoshikawa received the B.S. in Engineering (2008) and MMSc (2012) from Keio Univ. He joined Mitsubishi Electric Corporation as a researcher in the Information technology R&D center and work for artificial intelligent system development for control and optimization. Since 2022, he also worked in the joint research group for quantum computing at Tohoku University as a visiting associate professor. His main field of research includes the optimization of scheduling and optimal control with machine learning.

Stefano Di Cairano (Senior Member, IEEE) received the Master's (Laurea) and the Ph.D. degrees in information engineering in 2004 and 2008, respectively, from the University of Siena, Italy. During 2008-2011, he was with Powertrain Control R&A, Ford Research and Advanced Engineering, Dearborn, MI, USA. Since 2011, he is with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, where he is currently a Deputy Director, and a Distinguished Research Scientist. His research focuses on optimization-based control and decision-making strategies for complex mechatronic systems, in automotive, factory automation, transportation systems, and aerospace. His research interests include model predictive control, constrained control, path planning, hybrid systems, optimization, and particle filtering. He has authored/coauthored more than 200 peer-reviewed papers in journals and conference proceedings and 80 patents. Dr. Di Cairano was the Chair of the IEEE CSS Technical Committee on Automotive Controls and of the IEEE CSS Standing Committee on Standards. He is the inaugural Chair of the IEEE CCTA Editorial Board and was an Associate Editor of the IEEE TRANS. CONTROL SYSTEMS TECHNOLOGY.