Decentralized Probabilistic Multi-Robot Collision Avoidance Using Buffered Uncertainty-Aware Voronoi Cells

Hai Zhu · Bruno Brito · Javier Alonso-Mora

Received: date / Accepted: date

Abstract In this paper, we present a decentralized and communication-free collision avoidance approach for multi-robot systems that accounts for both robot localization and sensing uncertainties. The approach relies on the computation of an uncertainty-aware safe region for each robot to navigate among other robots and static obstacles in the environment, under the assumption of Gaussian-distributed uncertainty. In particular, at each time step, we construct a chance-constrained buffered uncertainty-aware Voronoi cell (B-UAVC) for each robot given a specified collision probability threshold. Probabilistic collision avoidance is achieved by constraining the motion of each robot to be within its corresponding B-UAVC, i.e. the collision probability between the robots and obstacles remains below the specified threshold. The proposed approach is decentralized, communication-free, scalable with the number of robots and robust to robots' localization and sensing uncertainties. We applied the approach to single-integrator, double-integrator, differential-drive robots, and robots with general nonlinear dynamics. Extensive simulations and experiments with a team of ground vehicles, quadrotors, and heterogeneous robot teams are performed to analyze and validate the proposed approach.

Keywords Collision avoidance · Motion planning · Planning under uncertainty · Multi-robot systems

1 Introduction

Multi-robot collision avoidance in cluttered environments is a fundamental problem when deploying a team of autonomous robots for applications such as coverage (Breitenmoser and Martinoli, 2016), target tracking (Zhou et al., 2018), formation flying (Zhu et al., 2019) and multi-view cinematography (Nägeli et al., 2017). Given the robot current states and goal locations, the objective is to plan a local motion for each robot to navigate towards its goal while avoiding collisions with other robots and obstacles in the environment. Most existing algorithms solve the problem in a deterministic manner, where the robot states and obstacle locations are perfectly known. Practically, however, robot states and obstacle locations are generally obtained by an estimator based on sensor measurements that have noise and uncertainty. Taking this uncertainty into consideration is of utmost importance for safe and robust multi-robot collision avoidance.

In this paper, we present a decentralized probabilistic approach for multi-robot collision avoidance under localization and sensing uncertainty that does not rely on communication. Our approach is built on the buffered Voronoi cell (BVC) method developed by Zhou et al. (2017). The BVC method is designed for collision avoidance among multiple single-integrator robots, where each robot only needs to know the positions of neighboring robots. We extend the method into probabilistic scenarios considering robot localization and sensing uncertainties by mathematically formalizing a buffered uncertainty-aware Voronoi cell (B-UAVC). Furthermore, we consider static obstacles with uncertain locations in the environment. We apply our approach to double-integrator dynamics, differential-drive robots, and general high-order dynamical robots.

This work was supported in part by the Netherlands Organization for Scientific Research (NWO) domain Applied Sciences (Veni 15916) and the U.S. Office of Naval Research Global (ONRG) NICOP-grant N62909-19-1-2027. We are grateful for their support.

A video of the experimental results is available at https://youtu.be/5F3fjjgwCSs

Hai Zhu
Department of Cognitive Robotics, Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands
E-mail: h.zhu@tudelft.nl

Bruno Brito
Department of Cognitive Robotics, Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands
E-mail: bruno.debrito@tudelft.nl

Javier Alonso-Mora
Department of Cognitive Robotics, Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands
E-mail: j.alonsomora@tudelft.nl

1.1 Related Works

1.1.1 Multi-robot collision avoidance

The problem of multi-robot collision avoidance has been well studied for deterministic scenarios, where the robots' states are precisely known. One of the state-of-the-art approaches is the reciprocal velocity obstacle (RVO) method (Van den Berg et al., 2008), which builds on the concept of velocity obstacles (VO) (Fiorini and Shiller, 1998). The method models robot interaction pairwise in a distributed manner and estimates future collisions as a function of relative velocity. Based on the basic framework, RVO has been extended towards several revisions: the optimal reciprocal collision-avoidance (ORCA) method (Van Den Berg et al., 2011) casting the problem into a linear programming formulation which can be solved efficiently, the generalized RVO method (Bareiss and van den Berg, 2015) applying for heterogeneous teams of robots, and the ε-cooperative collision avoidance (εCCA) method (Alonso-Mora et al., 2018) accounting for the cooperation of nonholonomic robots. In addition to those RVO-based methods, the model predictive control (MPC) framework has also been widely used for multi-robot collision avoidance, which includes decentralized MPC (Shim et al., 2003), decoupled MPC (Chen et al., 2015), and sequential MPC (Morgan et al., 2016; Luis et al., 2020). While those approaches typically require the robots position and velocity, or more detailed future trajectory information to be known among neighboring robots, the recent developed buffered Voronoi cell (BVC) method (Zhou et al., 2017; Pierson et al., 2020) only requires the robots to know the positions of other robots. In this paper, we build upon the concept of BVC and extend it to probabilistic scenarios, where each robot only needs to estimate the positions of its neighboring robots.

1.1.2 Collision avoidance under uncertainty

Some of the above deterministic collision avoidance approaches have been extended to scenarios where robot localization or sensing uncertainty is considered. Based on RVO, the COCALU method (Claes et al., 2012) takes into account bounded localization uncertainty of the robots by constructing an error-bounded convex hull of the VO of each robot. Gopalakrishnan et al. (2017) presents a probabilistic RVO method for single-integrator robots. Kamel et al. (2017) presents a decentralized MPC where robot motion uncertainty is taken into account by enlarging the robots with their 3-σ confidence ellipsoids. A chance constrained MPC problem was formulated by Lyons et al. (2012) for planar robots, where rectangular regions were computed and inter-robot collision avoidance was transformed to avoid overlaps of those regions. Using local linearization, Zhu and Alonso-Mora (2019b) proposed a chance constrained nonlinear MPC (CCNMPC) method to ensure that the probability of inter-robot collision is below a specified threshold.

Among these attempts to incorporate uncertainty into multi-robot collision avoidance, several limitations are observed. Probabilistic VO-based methods are limited to systems with simple first-order dynamics, or limited to homogeneous teams of robots. Probabilistic MPC-based methods typically demand communication of the planned trajectory of each robot to guarantee collision avoidance, which does not scale well with the number of robots in the system. An alternative to communicating trajectories is to assume that all other robots move with constant velocity (Kamel et al., 2017), which has been shown to lead to collisions in cluttered environments (Zhu and Alonso-Mora, 2019b). Recently, Luo et al. (2020) proposes probabilistic safety barrier certificates (PrSBC) to define the space of admissible control actions that are probabilistic safe, but it is only designed for single-integrator robots. In this paper, we define the probabilistic safe region for each robot directly based on the concept of buffered Voronoi cell (BVC).

The BVC method has also been extended to probabilistic scenarios by Wang and Schwager (2019). Taking into account the robot measurement uncertainty of other robots, they present the probabilistic buffered Voronoi cell (PBVC) to assure a safety level given a collision probability threshold. However, since the PBVC of each robot does not have an analytic solution, they employ a sampling-based approach to approximate it. In contrast, our proposed B-UAVC has an explicit and analytical form, which is more efficient to be computed. Moreover, our B-UAVC can be incorporated with MPC to handle general nonlinear systems, while the PBVC method developed by Wang and Schwager (2019) cannot be directly applied within a MPC framework.

1.1.3 Spatial decomposition in motion planning

Our method constructs a set of local safe regions for the robots, which decompose the workspace. Spatial decomposition is broadly used in robot motion planning. Deits and Tedrake (2015a) proposes the IRIS (iterative regional inflation by semi-definite programming) algorithm to compute safe convex regions among obstacles given a set of seed points. The algorithm is then used for UAV path planning (Deits and Tedrake, 2015b) and multi-robot formation control (Zhu et al., 2019). Liu et al. (2017) presents a simpler but more efficient iteratively inflation algorithm to compute a convex polytope around a line segment among obstacles and utilizes it to construct a safe flight corridor for UAV navigation

(Tordesillas et al., 2019). Similar safe flight corridors are constructed for trajectory planning of quadrotor swarms (Hönig et al., 2018), by computing a set of max-margin separating hyperplanes between a line segment and convex polygonal obstacles. The max-margin separating hyperplanes are also used by Arslan and Koditschek (2019) to construct a local robot-centric safe region in convex sphere worlds for sensor-based reactive navigation. While those spatial decomposition methods have shown successful application in robot motion planning, they all assume perfect knowledge on robots and obstacles positions. In this paper, we consider both the robot localization and obstacle position uncertainty and construct a local uncertainty-aware safe region for each robot.

1.2 Contribution

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

An earlier version of this paper was published by Zhu and Alonso-Mora (2019a). In this version, three main additional extensions are developed: a) we further consider static obstacles with uncertain locations in the environment; b) we extend the approach to double-integrator dynamics and differential-drive robots and c) we provide thorough simulation and experimental results and analyses.

1.3 Organization

The remaining of this paper is organized as follows. In Section 2 we present the problem statement and briefly summarize the concept of BVC. In Section 3 we formally introduce the buffered uncertainty-aware Voronoi cell (B-UAVC) and its construction method. We then describe how the B-UAVC is used for probabilistic multi-robot collision avoidance in Section 4. Simulation and experimental results are presented in Section 5 and Section 6, respectively. Finally, Section 7 concludes the paper.

2 Preliminaries

Throughout this paper vectors are denoted in bold lowercase letters, $\mathbf{x}$, matrices in plain uppercase M, and sets in calligraphic uppercase, $\mathcal{S}$. I indicates the identity matrix. A superscript $\mathbf{x}^T$ denotes the transpose of $\mathbf{x}$. $\|\mathbf{x}\|$ denotes the Euclidean norm of $\mathbf{x}$ and $\|\mathbf{x}\|_Q^2 = \mathbf{x}^T Q \mathbf{x}$ denotes the weighted square norm. A hat $\hat{\mathbf{x}}$ denotes the mean of a random variable $\mathbf{x}$. $\Pr(\cdot)$ indicates the probability of an event and $p(\cdot)$ indicates the probability density function.

2.1 Problem Statement

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

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

Definition 1 (Probabilistic Collision-Free) A robot i at position $\mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$ is probabilistic collision-free with a robot j at position $\mathbf{p}_j \sim \mathcal{N}(\hat{\mathbf{p}}_j, \Sigma_j)$ and an obstacle o at position $\mathbf{p}_o \sim \mathcal{N}(\hat{\mathbf{p}}_o, \Sigma_o)$ if

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

$$\Pr(\text{dis}(\mathbf{p}_i, \mathcal{O}_o) \geq r_s) \geq 1 - \delta, \quad \forall o \in \mathcal{I}_o, \tag{2}$$

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

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

$$\dot{\mathbf{p}}_i = \mathbf{u}_i, \tag{3}$$

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

2.2 Buffered Voronoi Cell

The key idea of our proposed method is to compute an uncertainty-aware collision-free region for each robot in the system, which is a major extension of the deterministic buffered Voronoi cell (BVC) method (Zhou et al., 2017; Pierson et al., 2020). In this section, we briefly describe the concept of BVC.

For a set of deterministic points $(\mathbf{p}_1, \dots, \mathbf{p}_n) \in \mathbb{R}^d$, the standard Voronoi cell (VC) of each point $i \in \mathcal{I}$ is defined as (Okabe et al., 2009)

$$\mathcal{V}_i = \{\mathbf{p} \in \mathbb{R}^d : \|\mathbf{p} - \mathbf{p}_i\| \leq \|\mathbf{p} - \mathbf{p}_j\|, \forall j \neq i\}, \tag{4}$$

which can also be written as

$$\mathcal{V}_i = \{\mathbf{p} \in \mathbb{R}^d : \mathbf{p}_{ij}^T \mathbf{p} \leq \mathbf{p}_{ij}^T \frac{\mathbf{p}_i + \mathbf{p}_j}{2}, \forall j \neq i\}, \tag{5}$$

where $\mathbf{p}_{ij} = \mathbf{p}_j - \mathbf{p}_i$. It can be observed that $\mathcal{V}_i$ is the intersection of a set of hyperplanes which separate point i with any other point j in the group, as shown in Fig. 1a. Hence, VC can be obtained by computing the separating hyperplanes between each pair of points.

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

$$\mathcal{V}_i^b = \{\mathbf{p} \in \mathbb{R}^d : \mathbf{p}_{ij}^T \mathbf{p} \leq \mathbf{p}_{ij}^T \frac{\mathbf{p}_i + \mathbf{p}_j}{2} - r_s \|\mathbf{p}_{ij}\|, \forall j \neq i\}, \tag{6}$$

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

In deterministic scenarios, if the robots are mutually collision-free, then the BVC of each robot is a non-empty set (Zhou et al., 2017). It is also trivial to prove that the BVCs are disjoint and if the robots are within their corresponding BVCs individually, they are collision free with each other. Using the concept of BVC, Zhou et al. (2017) proposed a control policy for a group of single-integrator robots whose control inputs are velocities. Each robot can safely and continuously navigate in its BVC, given that other robots in the system also follow the same rule. However, the guarantee does not hold for double-integrator dynamics or non-holonomic robots such as differential-drive robots.

2.3 Shadows of Uncertain Obstacles

To account for uncertain obstacles in the environment, we rely on the concept of obstacle shadows introduced by Axelrod et al. (2018). The ϵ-shadow is defined as follows:

Definition 2 (ϵ-Shadow) A set $\mathcal{S}_o \subseteq \mathbb{R}^d$ is an ϵ-shadow of an uncertain obstacle $\mathcal{O}_o$ if the probability $\Pr(\mathcal{O}_o \subseteq \mathcal{S}_o) \geq 1 - \epsilon$.

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

Definition 3 (Maximal ϵ-Shadow) A set $\mathcal{S}_o \subseteq \mathbb{R}^d$ is a maximal ϵ-shadow of an uncertain obstacle $\mathcal{O}_o$ if the probability $\Pr(\mathcal{O}_o \subseteq \mathcal{S}_o) = 1 - \epsilon$.

The above definition ensures that if there exists a maximal ϵ-shadow $\mathcal{S}_o$ of the uncertain obstacle $\mathcal{O}_o$ that does not intersect the robot, i.e. $\text{dis}(\mathbf{p}_i, \mathcal{S}_o) \geq r_s$, then the collision probability between the robot and obstacle is below ϵ, i.e. $\Pr(\text{dis}(\mathbf{p}_i, \mathcal{O}_o) \geq r_s) \geq 1 - \epsilon$. Note that the maximal ϵ-shadow may also be non-unique. In this paper, we employ the method proposed by Dawson et al. (2020) to construct such shadows. Recall that the uncertain obstacle $\mathcal{O}_o$ is related to the nominal geometry $\hat{\mathcal{O}}_o$ by $\mathcal{O}_o = \{\mathbf{x} + \mathbf{d}_o \mid \mathbf{x} \in \hat{\mathcal{O}}_o, \mathbf{d}_o \sim \mathcal{N}(0, \Sigma_o)\}$. To construct the maximal ϵ-shadow, we first define the following ellipsoidal set

$$\mathcal{D}_o = \{\mathbf{d} : \mathbf{d}^T \Sigma_o^{-1} \mathbf{d} \leq F^{-1}(1 - \epsilon)\}, \tag{7}$$

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

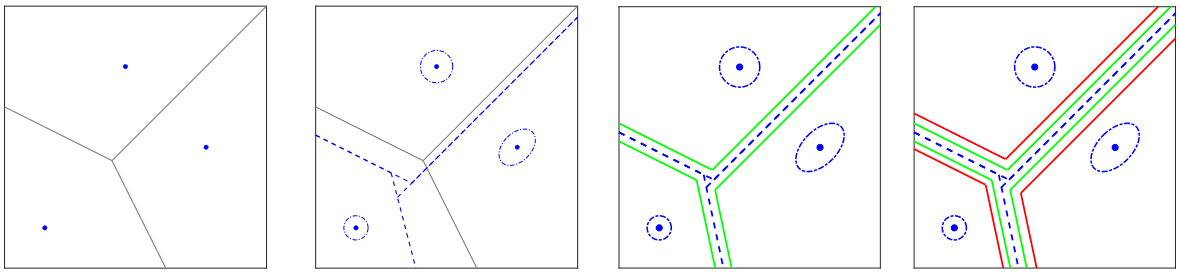
$$\mathcal{S}_o = \hat{\mathcal{O}}_o + \mathcal{D}_o = \{\mathbf{x} + \mathbf{d} \mid \mathbf{x} \in \hat{\mathcal{O}}_o, \mathbf{d} \in \mathcal{D}_o\}, \tag{8}$$

be the Minkowski sum of the nominal obstacle shape $\hat{\mathcal{O}}_o$ and the ellipsoidal set $\mathcal{D}_o$. Then, we have the following lemma (Axelrod et al., 2018) and theorem (Dawson et al., 2020):

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

Theorem 1 *$\mathcal{S}_o$ is a maximal ϵ-shadow of $\mathcal{O}_o$.*

Proofs of the above lemma and theorem are given in Appendix A.1 and A.2.



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

Fig. 1 Example of buffered uncertainty-aware Voronoi cells (B-UAVC). Blue dots are robots; blue dash-dot ellipses indicate the 3-σ confidence ellipsoid of the position uncertainty. (a) Deterministic Voronoi cell (VC, the boundary in gray solid line). (b) Uncertainty-aware Voronoi cell based on the best linear separators (UAVC, the boundary in blue dashed line). (c) UAVC with robot raidus buffer (the boundary in green solid line). (d) Final B-UAVC with robot radius and collision probability buffer (the boundary in red solid line).

3 Buffered Uncertainty-Aware Voronoi Cells

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

3.1 Definition of B-UAVC

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

Definition 4 (Buffered Uncertainty-Aware Voronoi Cell) Given a team of robots $i \in \{1,\dots,n\}$ with positions mean $\hat{\mathbf{p}}_i \in \mathbb{R}^d$ and covariance $\Sigma_i \in \mathbb{R}^{d\times d}$, and a set of convex polytope obstacles $o \in \{1,\dots,m\}$ with known shapes and locations mean $\hat{\mathbf{p}}_o \in \mathbb{R}^d$ and covariance $\Sigma_o \in \mathbb{R}^{d\times d}$, the buffered uncertainty-aware Voronoi cell (B-UAVC) of each robot is defined as a convex polytope region:

$$\mathcal{V}_i^{u,b} = \{\mathbf{p} \in \mathbb{R}^d : \mathbf{a}_{ij}^T\mathbf{p} \le b_{ij} - \beta_{ij}, \forall j \neq i, j \in \mathcal{I}, \tag{9}$$

$$\text{and} \quad \mathbf{a}_{io}^T\mathbf{p} \le b_{io} - \beta_{io}, \forall o \in \mathcal{I}_o\}, \tag{10}$$

such that the probabilistic collision free constraints in Definition 1 are satisfied.

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

$$\mathcal{V}_i^{u} = \{\mathbf{p} \in \mathbb{R}^d : \mathbf{a}_{ij}^T\mathbf{p} \le b_{ij}, \forall j \neq i, j \in \mathcal{I}, \tag{11}$$

$$\text{and} \quad \mathbf{a}_{io}^T\mathbf{p} \le b_{io}, \forall o \in \mathcal{I}_o\}, \tag{12}$$

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

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

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

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

3.2 Inter-Robot Separating Hyperplane

In contrast to only separating two deterministic points in Voronoi cells, we separate two uncertain robots with known positions mean and covariance. To achieve that, we rely on the concept of the best linear separator between two Gaussian distributions (Anderson and Bahadur, 1962).

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

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

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

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

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

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

$$(\mathbf{a}_{ij}, b_{ij}) = \arg\min_{\mathbf{a}_{ij}\in\mathbb{R}^d, b_{ij}\in\mathbb{R}} \max(\Pr_i, \Pr_j), \tag{13}$$

which can be solved using a fast minimax procedure. In this paper, we employ the procedure developed by Anderson and Bahadur (1962) to compute the best linear separator parameters $\mathbf{a}_{ij}$ and b_{ij}. A brief summary of the procedure is presented in Appendix B.

Remark 1 The best linear separator coincides with the separating hyperplane of Eq. (5) when $\Sigma_i = \Sigma_j = \sigma^2 I$. In this case, $\mathbf{a}_{ij} = \frac{2}{\sigma^2}(\hat{\mathbf{p}}_j - \hat{\mathbf{p}}_i)$ and $b_{ij} = \frac{1}{\sigma^2}(\hat{\mathbf{p}}_j - \hat{\mathbf{p}}_i)^T(\hat{\mathbf{p}}_i + \hat{\mathbf{p}}_j)$.

Remark 2 $\forall i \neq j \in \mathcal{I}, \mathbf{a}_{ji} = -\mathbf{a}_{ij}, b_{ji} = -b_{ij}$. This can be obtained according to the definition of the best linear separator.

Remark 3 In contrast to deterministic Voronoi cells, the UAVCs constructed from the best linear separators generally do not constitute a full tessellation of the workspace, i.e. $\bigcup_1^n \mathcal{V}_i^u \subseteq \mathcal{W}$, as shown in Fig. 1b.

3.3 Robot-Obstacle Separating Hyperplane

Our method to calculate the uncertainty-aware separating hyperplane between a robot and a convex polytope obstacle with uncertain location is illustrated in Fig. 2. Given the mean position of the robot $\hat{\mathbf{p}}_i$ and the uncertain obstacle $\mathcal{O}_o = \{\mathbf{x} + \mathbf{d}_o \mid \mathbf{x} \in \hat{\mathcal{O}}_o, \mathbf{d}_o \sim \mathcal{N}(0, \Sigma_o)\}$, we first perform a linear coordinate transformation:

$$W = (\sqrt{\Sigma_o})^{-1}, \tag{14}$$

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

$$\hat{\mathbf{p}}_i^W = W\hat{\mathbf{p}}_i, \tag{15}$$
$$\hat{\mathcal{O}}_o^W = W\hat{\mathcal{O}}_o, \tag{16}$$
$$\mathbf{d}_o^W = W\mathbf{d}_o, \tag{17}$$
$$\Sigma_o^W = W\Sigma_o W^T = I^{d\times d}. \tag{18}$$

The transformed uncertain obstacle is then $\mathcal{O}_o^W = \{\mathbf{x}^W + \mathbf{d}_o^W \mid \mathbf{x}^W \in \hat{\mathcal{O}}_o^W, \mathbf{d}_o^W \sim \mathcal{N}(0, I)\}$. Here we use the super-script $\cdot^W$ to indicate variables in the transformed space. Note that the obstacle position uncertainty covariance is normalized to an identity matrix under the transformation, as shown in Fig. 2 (Top right). This coordinate transformation technique to normalize the uncertainty covariance has also been applied to other motion planning under uncertainty works (Hardy and Campbell, 2013).

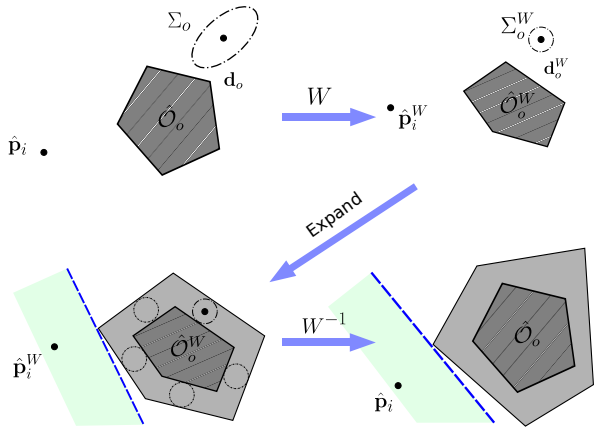


Fig. 2 Depiction of uncertainty-aware separating hyperplane calculation between a point and an arbitrary polytope obstacle with uncertain location. (Top left) A point and a polytope obstacle with uncertain location. (Top right) Effects of the transformation W to normalize the error covariance. (Bottom left) ϵ-shadow of the transformed obstacle and the max-margin separating hyperplane in the transformation space. (Bottom right) Inverse transformation to obtain the uncertainty-aware separating hyperplane.

Then given the collision probability threshold δ, we compute a ϵ-shadow of the transformed uncertain obstacle $\mathcal{O}_o^W$ based on Eqs. (7)-(8):

$$\mathcal{D}_o^W = \{\mathbf{d}^W : \mathbf{d}^{W\,T}\mathbf{d}^W \le F^{-1}(1-\epsilon)\}, \tag{19}$$
$$\mathcal{S}_o^W = \hat{\mathcal{O}}_o^W + \mathcal{D}_o^W, \tag{20}$$

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

Note that we assume $\hat{\mathcal{O}}_o$ is a convex polytope. Hence, the transformed $\hat{\mathcal{O}}_o^W$ is also a polytope. In addition, it can be observed the set $\mathcal{D}_o^W$ defined in Eq. (19) is a circular (sphere in 3D) set with radius $\sqrt{F^{-1}(1-\epsilon)}$. Hence, we can compute the ϵ-shadow in Eq. (20) of the transformed uncertain obstacle by dilating its nominal shape by the diameter of the set $\mathcal{D}_o^W$, which results in an inflated convex polytope. Note that the resulted convex polytope is slightly larger than the exact Minkowski sum $\mathcal{S}_o^W$ which has smaller round corners. This introduces some conservativeness. For simplicity, we use the same notation $\mathcal{S}_o^W$ for the resulted inflated convex polytope and thus there is $\Pr(\mathcal{O}_o^W \subseteq \mathcal{S}_o^W) > \sqrt{1-\delta}$.

Next, we separate $\hat{\mathbf{p}}_i^W$ from $\mathcal{S}_o^W$ by finding a max-margin separating hyperplane between them. Note that

$\mathcal{S}_o^W$ is a bounded convex polytope that can be described by a list of vertices $(\psi_1^W, \dots, \psi_{p_o}^W)$. Hence, finding a max-margin hyperplane between $\hat{\mathbf{p}}_i^W$ and $\mathcal{S}_o^W$ can be formulated as a support vector machine (SVM) problem (Hönig et al., 2018), which can be efficiently solved using a quadratic program:

$$\begin{aligned} \min \quad & \mathbf{a}_{io}^{W\,T}\mathbf{a}_{io}^W \\ \text{s.t.} \quad & \mathbf{a}_{io}^{W\,T}\hat{\mathbf{p}}_{io}^W - b_{io}^W \le 1, \\ & \mathbf{a}_{io}^{W\,T}\psi_k^W - b_{io}^W \ge 1, \quad \forall\, k \in 1, \dots, p_o. \end{aligned} \tag{21}$$

The solution of the above quadratic program (21) formulates a max-margin separating hyperplane with parameters $(\mathbf{a}_{io}^W, b_{io}^W)$. We then shift it along its normal vector towards the obstacle shadow, resulting in a separating hyperplane exactly touching the shadow, as shown in Fig. 2 (Bottom left). Finally we perform an inverse coordinate transformation W^{-1} and obtain the uncertainty-aware separating hyperplane between the robot and obstacle in the original workspace:

$$\begin{aligned} \mathbf{a}_{io} &= W^T\mathbf{a}_{io}^W, \\ b_{io} &= b_{io}^W, \end{aligned} \tag{22}$$

as shown in Fig. 2 (Bottom right), in which the ϵ-shadow in the transformed space $\mathcal{S}_o^W$ becomes $\mathcal{S}_o$ in the original space.

Remark 4 The linear coordinate transformation W and its inverse W^{-1} preserves relative geometries of $\mathcal{O}_o$. That is, $\Pr(\mathcal{O}_o \subseteq \mathcal{S}_o) = \Pr(\mathcal{O}_o^W \subseteq \mathcal{S}_o^W) > \sqrt{1-\delta}$.

3.4 Collision Avoidance Buffer and B-UAVC

In Section 3.2 and 3.3 we have described the method to compute the hyperplanes that construct the UAVC. Now we introduce two buffer terms to the UAVC, to account for the robot physical safety radius and the collision probability threshold.

Recall Eq. (11) that the UAVC of robot i can be written as the intersection of a set of separating hyperplanes

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

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

$$\mathcal{V}_i^u = \{\mathbf{p} \in \mathbb{R}^d : \mathbf{a}_{il}^T\mathbf{p} \le b_{il}, \forall l \in \mathcal{I}_l, l \ne i\}. \tag{23}$$

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

3.4.1 Robot safety radius buffer

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

$$\beta_i^r = r_s \left\|\mathbf{a}_{il}\right\|. \tag{24}$$

Figure 1c shows the buffered UAVC of each robot after taking into account their safety radius.

3.4.2 Collision probability buffer

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

$$\beta_i^\delta = \sqrt{2\mathbf{a}_{il}^T\Sigma_i\mathbf{a}_{il}} \cdot \mathrm{erf}^{-1}(2\sqrt{1-\delta} - 1), \tag{25}$$

where $\mathrm{erf}(\cdot)$ is the Gauss error function (Andrews, 1997) defined as $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$ and $\mathrm{erf}^{-1}(\cdot)$ is its inverse. In this paper, we assume the threshold satisfies $0 < \delta < 0.75$, which is reasonable in practice. Hence, $\mathrm{erf}^{-1}(2\sqrt{1-\delta} - 1) > 0, \beta_i^\delta > 0$. This buffer can be obtained by following the proof of forthcoming Theorem 2 and Theorem 3.

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

$$\mathcal{V}_i^{u,b} = \{\mathbf{p} \in \mathbb{R}^d : \mathbf{a}_{il}^T\mathbf{p} \le b_{il} - \beta_i^r - \beta_i^\delta, \forall l \in \mathcal{I}_l, l \ne i\}. \tag{26}$$

Figure 1d shows the final B-UAVC of each robot in the team.

3.5 Properties of B-UAVC

In this subsection, we justify the design of ϵ in Eq. (19) when computing the shadow of uncertain obstacles, and computation of the collision probability buffer β_i^δ in Eq. (25) by presenting the following two theorems.

Theorem 2 (Inter-Robot Probabilistic Collision Free) *$\forall \mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$ and $\mathbf{p}_j \sim \mathcal{N}(\hat{\mathbf{p}}_j, \Sigma_j)$, where $\hat{\mathbf{p}}_i \in \mathcal{V}_i^{u,b}$ and $\hat{\mathbf{p}}_j \in \mathcal{V}_j^{u,b}, i \ne j \in \mathcal{I}$, we have*

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

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

Proof We first introduce the following lemma:

Lemma 2 (Linear Chance Constraint (Blackmore et al., 2011)) *A multivariate random variable $\mathbf{x} \sim \mathcal{N}(\hat{\mathbf{x}}, \Sigma)$ satisfies*

$$\Pr(\mathbf{a}^T\mathbf{x} \le b) = \frac{1}{2} + \frac{1}{2}\mathrm{erf}\left(\frac{b - \mathbf{a}^T\hat{\mathbf{x}}}{\sqrt{2\mathbf{a}^T\Sigma\mathbf{a}}}\right). \tag{27}$$

According to Eq. (26), if $\hat{\mathbf{p}}_i \in \mathcal{V}_i^{u,b}$, there is

$$\mathbf{a}_{ij}^T \hat{\mathbf{p}}_i \leq b_{ij} - r_s \|\mathbf{a}_{ij}\| - \sqrt{2\mathbf{a}_{ij}^T \Sigma_i \mathbf{a}_{ij}} \cdot \mathrm{erf}^{-1}(2\sqrt{1-\delta} - 1). \tag{28}$$

Applying Lemma 2 and substituting the above equation, we have

$$\begin{aligned} &\Pr(\mathbf{a}_{ij}\mathbf{p}_i \leq b_{ij} - r_s \|\mathbf{a}_{ij}\|) \\ &= \frac{1}{2} + \frac{1}{2}\mathrm{erf}\left(\frac{b_{ij} - r_s \|\mathbf{a}_{ij}\|) - \mathbf{a}_{ij}^T \hat{\mathbf{p}}_i}{\sqrt{2\mathbf{a}_{ij}^T \mathbf{a}_{ij}}}\right) \\ &\geq \frac{1}{2} + \frac{1}{2}\mathrm{erf}\left(\mathrm{erf}^{-1}(2\sqrt{1-\delta} - 1)\right) \\ &= \frac{1}{2} + \frac{1}{2}(2\sqrt{1-\delta} - 1) \\ &= \sqrt{1-\delta}. \end{aligned} \tag{29}$$

Similarly for robot j, there is

$$\Pr(\mathbf{a}_{ji}\mathbf{p}_j \leq b_{ji} - r_s \|\mathbf{a}_{ji}\|) \geq \sqrt{1-\delta}. \tag{30}$$

Note that $\mathbf{a}_{ij} = -\mathbf{a}_{ji}$, $b_{ij} = -b_{ji}$ (Remark 2). It is trivial to prove that

$$\left.\begin{aligned} \mathbf{a}_{ij}\mathbf{p}_i \leq b_{ij} - r_s \|\mathbf{a}_{ij}\| \\ \mathbf{a}_{ji}\mathbf{p}_j \leq b_{ji} - r_s \|\mathbf{a}_{ji}\| \end{aligned}\right\} \implies \|\mathbf{p}_i - \mathbf{p}_j\| \geq 2r_s. \tag{31}$$

Hence, we have

$$\begin{aligned} &\Pr(\mathrm{dis}(\mathbf{p}_i, \mathbf{p}_j) \geq 2r_s) = \Pr(\|\mathbf{p}_i - \mathbf{p}_j\| \geq 2r_s) \\ &\geq \Pr(\mathbf{a}_{ij}\mathbf{p}_i \leq b_{ij} - r_s \|\mathbf{a}_{ij}\|) \cdot \Pr(\mathbf{a}_{ji}\mathbf{p}_j \leq b_{ji} - r_s \|\mathbf{a}_{ji}\|) \\ &\geq \sqrt{1-\delta} \cdot \sqrt{1-\delta} \\ &= 1 - \delta. \end{aligned} \tag{32}$$

This completes the proof. □

Theorem 3 (Robot-Obstacle Probabilistic Collision Free) *$\forall \mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$, where $\hat{\mathbf{p}}_i \in \mathcal{V}_i^{u,b}$, we have $\Pr(\mathrm{dis}(\mathbf{p}_i, \mathcal{O}_o) \geq r_s) \geq 1 - \delta$, i.e. the probability of collision between robot i and obstacle o is below the threshold δ.*

Proof Similar to Eq. (29), we have

$$\Pr(\mathbf{a}_{io}\mathbf{p}_i \leq b_{io} - r_s \|\mathbf{a}_{io}\|) \geq \sqrt{1-\delta}. \tag{33}$$

Based on the computation of $\mathbf{a}_{io}$ and b_{io} in Eq. (21)-(22), it is straightforward to prove that

$$\mathbf{a}_{io}\mathbf{p}_i \leq b_{io} - r_s \|\mathbf{a}_{io}\| \implies \mathrm{dis}(\mathbf{p}_i, \mathcal{S}_o) \geq r_s. \tag{34}$$

Thus,

$$\Pr(\mathrm{dis}(\mathbf{p}_i, \mathcal{S}_o) \geq r_s) \geq \sqrt{1-\delta}. \tag{35}$$

If $\mathcal{O}_o \subseteq \mathcal{S}_o$ and $\mathrm{dis}(\mathbf{p}_i, \mathcal{S}_o) \geq r_s$, there is $\mathrm{dis}(\mathbf{p}_i, \mathcal{O}_o) \geq r_s$. Hence, by combining with Remark 4, we have

$$\begin{aligned} \Pr(\mathrm{dis}(\mathbf{p}_i, \mathcal{O}_o) \geq r_s) &\geq \Pr(\mathcal{O}_o \subseteq \mathcal{S}_o) \cdot \Pr(\mathrm{dis}(\mathbf{p}_i, \mathcal{S}_o) \geq r_s) \\ &> \sqrt{1-\delta} \cdot \sqrt{1-\delta} \\ &= 1 - \delta, \end{aligned} \tag{36}$$

which completes the proof. □

4 Collision Avoidance Using B-UAVC

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

4.1 Reactive Feedback Control

4.1.1 Single integrator dynamics

Consider robots with single-integrator dynamics $\dot{\mathbf{p}}_i = \mathbf{u}_i$, where $\mathbf{u}_i = \mathbf{v}_i$ is the control input. Similar to Zhou et al. (2017), a fast reactive feedback one-step controller can be designed to make each robot move towards its goal location $\mathbf{g}_i$, as follows:

$$\mathbf{u}_i = v_{i,\max} \cdot \frac{\mathbf{g}_i^* - \hat{\mathbf{p}}_i}{\|\mathbf{g}_i^* - \hat{\mathbf{p}}_i\|}, \tag{37}$$

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

$$\mathbf{g}_i^* := \arg\min_{\mathbf{p} \in \mathcal{V}_i^{u,b}} \|\mathbf{p} - \mathbf{g}_i\|, \tag{38}$$

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

The strategy used in the controller, Eq. (37), is also called the "move-to-projected-goal" strategy (Arslan and Koditschek, 2019). At each time step, each robot in the system first constructs its B-UAVC $\mathcal{V}_i^{u,b}$, then computes the closest point in $\mathcal{V}_i^{u,b}$ to its goal, i.e. the "projected goal", and generates a control input according to Eq. (37). Note that the constructed B-UAVC is a convex polytope represented by the intersection of a set of half-spaces hyperplanes. Hence, finding the closest point, Eq. (38), can be recast as a linearly constrained least-square problem, which can be solved efficiently using quadratic programming in polynomial time (Kozlov et al., 1980).

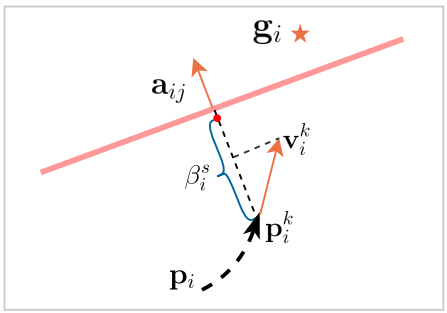


Fig. 3 Additional buffer is added to allow robots with double-integrator dynamics to have enough space to decelerate.

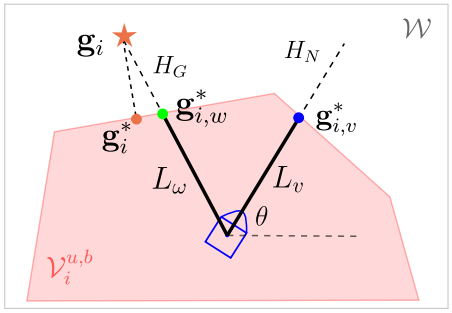


Fig. 4 Reactive feedback control for differential-drive robots.

4.1.2 Double integrator dynamics

For single-integrator robots, the reactive controller Eq. (37) guarantees the robot to be always within its corresponding B-UAVC and thus probabilistic collision free with other robots and obstacles. However, the controller may drive the robot towards to the boundary of its B-UAVC. Consider the double-integrator robot which has a limited acceleration, $\ddot{\mathbf{p}}_i = \mathbf{u}_i$, where $\mathbf{u}_i = \mathbf{acc}_i$ is the control input. It might not be able to continue to stay within its B-UAVC when moving close to the boundary of the B-UAVC. Hence, to enhance safety, as illustrated in Fig. 3 we introduce an additional safety stopping buffer, which is defined as

$$\beta_i^s = \begin{cases} \frac{\|\mathbf{a}_{il}^T \mathbf{v}_i\|^2}{2acc_{i,\max}}, & \text{if } \mathbf{a}_{il}^T \mathbf{v}_i > 0; \\ 0, & \text{otherwise}, \end{cases} \tag{39}$$

where $acc_{i,\max}$ is the maximal acceleration of the robot. This additional stopping buffer heuristically leaves more space for the robot to decelerate in advance before touching the boundaries of the original B-UAVC. Hence, the updated B-UAVC in Eq. (26) with an additional safety stopping buffer now becomes

$$\mathcal{V}_i^{u,b} = \{\mathbf{p} \in \mathbb{R}^d : \mathbf{a}_{il}^T \mathbf{p} \leq b_{il} - \beta_i^r - \beta_i^\delta - \beta_i^s, \quad \forall l \in \mathcal{I}_l, l \neq i\}. \tag{40}$$

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

$$\mathbf{u}_i = acc_{i,\max} \cdot \frac{\mathbf{g}_i^* - \hat{\mathbf{p}}_i}{\|\mathbf{g}_i^* - \hat{\mathbf{p}}_i\|}. \tag{41}$$

4.1.3 Differential-drive robots

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

$$\begin{aligned} \dot{\hat{\mathbf{p}}}_i &= v_i \begin{bmatrix} \cos\theta_i \\ \sin\theta_i \end{bmatrix}, \\ \dot{\theta}_i &= \omega_i, \end{aligned} \tag{42}$$

where $\theta_i \in [-\pi, \pi)$ is the orientation of the robot, and $\mathbf{u}_i = (v_i, \omega_i)^T \in \mathbb{R}^2$ is the vector of robot control inputs in which v_i and ω_i are the linear and angular velocity, respectively. We adopt the control strategy developed by Arslan and Koditschek (2019) and Astolfi (1999) and briefly describe it in the following.

As shown in Fig. 4, firstly, two line segments

$$L_v = \mathcal{V}_i^{u,b} \cap H_N, \tag{43}$$

$$L_\omega = \mathcal{V}_i^{u,b} \cap H_G, \tag{44}$$

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

$$\begin{aligned} v_i &= -k \cdot [\cos(\theta) \ \ \sin\theta](\hat{\mathbf{p}}_i - \mathbf{g}_{i,v}^*), \\ \omega_i &= k \cdot \operatorname{atan}\left(\frac{[-\sin(\theta) \ \ \cos\theta](\hat{\mathbf{p}}_i - (\mathbf{g}_i^* + \mathbf{g}_{i,\omega}^*)/2)}{[\cos(\theta) \ \ \sin\theta](\hat{\mathbf{p}}_i - (\mathbf{g}_i^* + \mathbf{g}_{i,\omega}^*)/2)} \right), \end{aligned} \tag{45}$$

where $k > 0$ is the fixed control gain. It is proved by Arslan and Koditschek (2019) that if the local safe region is convex, then the robot will stay within the convex safe region under the control law of Eq. (45).

4.2 Receding Horizon Planning

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

Problem 1 (Receding Horizon Trajectory Planning)

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

$$\text{s.t.} \quad \mathbf{x}_i^0 = \hat{\mathbf{x}}_i, \tag{46a}$$

$$\hat{\mathbf{x}}_i^k = \mathbf{f}_i(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}), \tag{46b}$$

$$\hat{\mathbf{p}}_i^k \in \mathcal{V}_i^{u,b}, \tag{46c}$$

$$\mathbf{u}_i^{k-1} \in \mathcal{U}_i, \tag{46d}$$

$$\forall i \in \mathcal{I}, \forall k \in \{1, \dots, N\}. \tag{46e}$$

In Problem 1, $\mathcal{U}_i \in \mathbb{R}^{n_u}$ is the admissible control space; $R \in \mathbb{R}^{n_u \times n_u}$, $Q_N \in \mathbb{R}^{d \times d}$ are positive semi-definite symmetric matrices. The constraint (46c) restrains the planned trajectory to be within the robot's B-UAVC $\mathcal{V}_i^{u,b}$. According to the definition of $\mathcal{V}_i^{u,b}$ in Eq. (40), the constraint can be formulated as a set of linear inequality constraints:

$$\mathbf{a}_{il}^T \hat{\mathbf{p}}_i^k \leq b_{il} - \beta_i^r - \beta_i^\delta - \beta_i^s, \ \forall l \in \mathcal{I}_l, l \neq i. \tag{47}$$

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

Remark 5 (Probability of collision for the planned trajectory) From Theorem 2 and 3, constraint (46c) guarantees that at each stage within the planning horizon, the collision probability of robot i with any other robot or obstacle is below the specified threshold δ. Hence, the probability of collision for the entire planning trajectory of robot i with respect to each other robot and obstacle can be bounded by $\Pr(\cup_{k=1}^N \hat{\mathbf{p}}_i^k \notin \mathcal{V}_i^{u,b}) \leq \sum_{k=1}^N \Pr(\hat{\mathbf{p}}_i^k \notin \mathcal{V}_i^{u,b}) = N\delta$. Nevertheless, this bound is over conservative in practice. The real collision probability of the planned trajectory is much smaller than $N\delta$ (Schmerling and Pavone, 2017). Hence, we impose the collision probability threshold δ for each individual stage in the context of receding horizon planning, thanks to the fast re-planning and relatively small displacement between stages (Luo et al., 2020).

Algorithm 1 Collision Avoidance Using B-UAVC for Each Robot $i \in \mathcal{I}$ in a Multi-robot Team

Construction of B-UAVC

Obtain $\mathbf{p}_i \sim \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$ via state estimation
for Each other robot $j \in \mathcal{I}, j \neq i$ **do**
  Estimate $\mathbf{p}_j \sim \mathcal{N}(\hat{\mathbf{p}}_j, \Sigma_j)$
  Compute the best linear separator parameters $(\mathbf{a}_{ij}, b_{ij})$ via Eq. (13)
end for
for Each static obstacle $o \in \mathcal{I}_o$ **do**
  Estimate $\mathbf{d}_o \sim \mathcal{N}(0, \Sigma_o)$ with known $\hat{\mathcal{O}}_o$
  Compute the separating hyperplane parameters $(\mathbf{a}_{io}, b_{io})$ via Eqs. (14)-(22)
end for
for Each separating hyperplane $l \in \mathcal{I}_l, l \neq i$ **do**
  Compute the safety radius buffer via Eq. (24): $\beta_i^r = r_s \|\mathbf{a}_{il}\|$
  Compute the collision probability buffer via Eq. (25): $\beta_i^\delta = \sqrt{2\mathbf{a}_{il}^T \Sigma_i \mathbf{a}_{il}} \cdot \text{erf}^{-1}(2\sqrt{1-\delta} - 1)$
  Construct the B-UAVC via Eq. (26)
end for

Collision Avoidance Action

if i is single-integrator **then**
  Compute control input via Eqs. (37)-(38)
else if i is double-integrator **then**
  Compute control input via Eqs. (39)-(41)
else if i is differential-drive **then**
  Compute control input via Eqs. (43)-(45)
else
  Compute control input by solving Problem 1
end if

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

4.3 Discussion

4.3.1 Uncertainty estimation

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

practice since the robot generally has more accurate measurements of its own position than other robots in the environment. Then robot i computes its B-UAVC using $\hat{\mathbf{p}}_i, \Sigma_i, \tilde{\mathbf{p}}_j$, and $\tilde{\Sigma}_j$. According to the properties of the best linear separator, this assumption leads that each robot i always partitions a smaller space when computing the separating hyperplane with another robot j, which results in a more conservative B-UAVC to ensure safety for robot i itself.

4.3.2 Empty B-UAVCs

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

5 Simulation Results

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

5.1 Comparison to the BVC Method

We first compare our proposed B-UAVC method with the BVC approach (Zhou et al., 2017) that we extend in two-dimensional obstacle-free environments with single-integrator robots. Both the B-UAVC and BVC methods only need robot position information to achieve collision avoidance, in contrast to the well-known reciprocal velocity obstacle (RVO) method (Van Den Berg et al., 2011) which also requires robot velocity information to be communicated or sensed. Comparison between BVC and RVO has been demonstrated by Zhou et al. (2017) in 2D scenarios, hence in this paper we focus on comparing the proposed B-UAVC with BVC.

We deploy the B-UAVC and BVC in a 10×10m environment with 2, 4, 8, 16 and 32 robots forming an *antipodal circle swapping* scenario (Van Den Berg et al. (2011)). In this scenario, the robots are initially placed on a circle (equally spaced) and their goals are located at the antipodal points of the circle. We use a circle with a radius of 4.0 m in simulation. Each robot has a radius of 0.2 m, a local sensing range of 2.0 m and a maximum allowed speed of 0.4 m/s. The goal is assumed to be reached for each robot when the distance between its center and goal location is smaller than 0.1 m. To simulate collision avoidance under uncertainty, two different levels of noise, $\Sigma_1 = \text{diag}(0.04\text{ m}, 0.04\text{ m})^2$ and $\Sigma_2 = \text{diag}(0.06\text{ m}, 0.06\text{ m})^2$, are added to the robot position measurements. Particularly, each robot's localization uncertainty covariance is Σ_1 and its estimation of other robots' position uncertainty covariance is Σ_2. The time step used in simulation is $\Delta t = 0.1$ s.

In the basic BVC implementation, an extra 10% or 100% radius buffer is added to the robot's real physical radius to account for measurement uncertainty for comparison (Wang and Schwager, 2019). In the B-UAVC implementation, the collision probability threshold is set as $\delta = 0.05$. Any robot will stop moving when it arrives at its goal or is involved in a collision. Both the B-UAVC and BVC methods use the same deadlock resolution techniques proposed in this paper (Appendix C). We set a maximum simulation step $K = 800$ and the collision-free robots that do not reach their goals within K steps are regarded to be in deadlocks/livelocks.

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

The simulation results are presented in Fig. 5. In all runs, no deadlocks are observed. In terms of collision avoidance, both the B-UAVC approach and BVC with additional 100% robot radius achieve zero collision in all runs. The BVC with only 10% robot radius leads to collisions when the total number of robots gets larger. In particular, when there are 32 robots an average of 28% robots collide, as shown in Fig. 5a. While the BVC with 100% additional robot radius can also achieve zero collision rate as our proposed B-UAVC, it is more conservative and less efficient. In average, the B-UAVC saves 10.1% robot travelled distance (Fig. 5c) and 14.4% time for completing a single run (Fig. (d)) comparing to the BVC with additional 100% robot radius.

Remark 6 The "BVC + X%" is a heuristic way to handle uncertainty. The above simulation results show that if X is too small, then it cannot ensure safety; while if X is too large, the results will be very conservative and less efficient. So generally reasoning about individual uncertainties using the proposed B-UAVC method will perform better than determining an extra X% buffer.

Remark 7 In some cases we can design such an X that it will have the same results as the B-UAVC method.

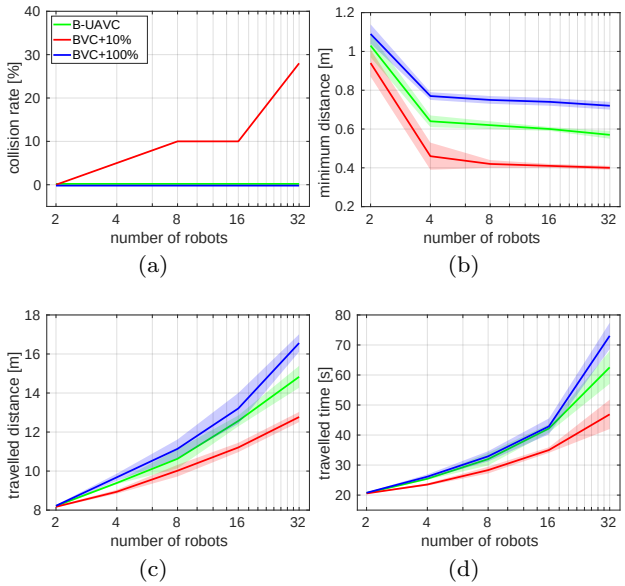


Fig. 5 Evaluation of the antipodal circle scenario with varying numbers of single-integrator robots. The (a) collision rate, (b) minimum distance, (c) travelled distance and (d) complete time are shown. Lines denote mean values and shaded areas around the lines denote standard deviations over 10 repetitions for each scenario.

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

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

If we choose an extra safety buffer $X\%$ such that

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

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

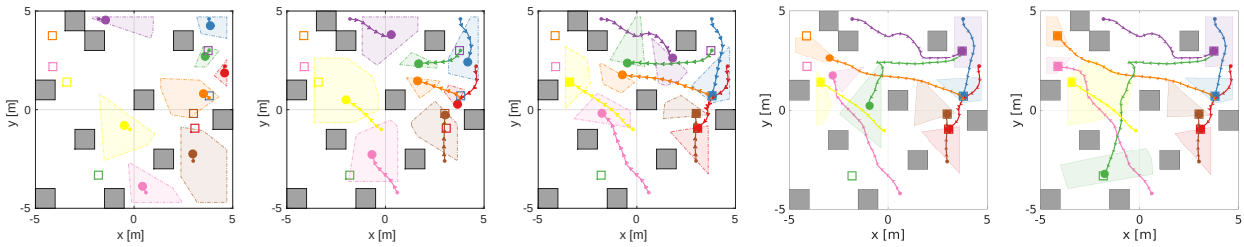
5.2 Performance Analysis

We then study the effect of collision probability threshold on the performance of the proposed B-UAVC method. Similarly, we deploy the B-UAVC in a 10×10m environment with 2, 4, 8, 16 and 32 robots in obstacle-free and cluttered environments with 10% obstacle density. In the obstacle-free case for each number of robots n, 10 scenarios are randomly generated to form a challenging *asymmetric swapping* scenario (Serra-Gómez et al., 2020), indicating that the environment is split into n sections around the center and each robot is initially randomly placed in one of them while required to navigate to its opposite section around the center. In the obstacle-cluttered case, 10 *random moving* scenarios are simulated for each different number of robots in which robot initial positions and goal locations are randomly generated. Fig. 6 shows a sample run of the scenario with 8 robots and 10 obstacles. We then run each generated scenario 5 times given a parameter setting (collision probability threshold). The robots have the same radius and maximal speed as in Section 5.1. Localization noise with zero mean and covariance $\Sigma = \mathrm{diag}(0.06\ \mathrm{m}, 0.06\ \mathrm{m})^2$ is added. For evaluation of performance, we focus on the robot collision rate, the robot deadlock rate, and the minimum distance among successful robots.

We evaluate the performance of B-UAVC with different levels of collision probability threshold: $\delta = 0.05$, 0.10, 0.20 and 0.30. The simulation results are presented in Fig. 7. In the top row of the figure, we consider the collision rate among robots. The result shows that with a roughly small collision probability threshold $\delta = 0.05, 0.10, 0.20$, no collisions are observed in both obstacle-free asymmetric swapping and obstacle-cluttered random moving scenarios, indicating that the B-UAVC method maintains a high level of safety. However, when δ is set to 0.3, the collision rate among robots increase dramatically, in particular when the number of robots is large. For example, in the asymmetric swapping scenario with 32 robots, there are 68.75% robots involve in collisions in average. In the bottom row of the figure, the minimum distance among robots are compared. The result shows that with smaller threshold, the minimum distance will be a little bit larger. The reason is that robots with a smaller threshold will have more conservative behavior and have smaller B-UAVCs during navigation.

5.3 Simulations with Quadrotors in 3D Space

We evaluate our receding horizon planning algorithm with quadrotors in 3D space and compare our method with one of the state-of-the-art quadrotor collision avoidance methods: the chance constrained nonlinear MPC (CCNMPC) with sequential planning (Zhu and Alonso-Mora, 2019b), which requires communication of future planned trajectories among robots. For both methods, we adopt the same quadrotor dynamics model for planning. The quadrotor radius is set as $r = 0.3$ m and the collision probability threshold is set to $\delta = 0.03$. The



(a) $t = 1$ s. (b) $t = 6$ s. (c) $t = 12$ s. (d) $t = 18$ s. (e) $t = 27$ s.

Fig. 6 A sample simulation run of the random moving scenario with 8 robots and 10% obstacle density. The robot initial and goal locations are marked in circle disks and solid squares. Grey boxes are static obstacles. The B-UAVCs are shown in shaded patches with dashed boundaries.

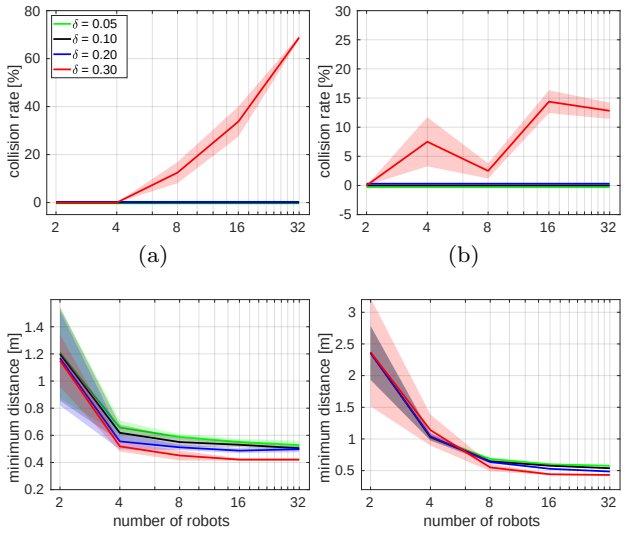


Fig. 7 Effect of the collision probability threshold on the method performance. The (a)-(b) collision rate, and (c)-(d) minimum distance among robots are shown. The evaluation has 2, 4, 8, 16, and 32 robot cases with 10 instances each. The left column shows results of the asymmetric swapping scenario and the right column shows results of the random moving scenario with 10% obstacle density. Lines denote mean values and shaded areas around the lines denote standard deviations over 50 runs.

time step is $\Delta t = 0.05$ s and the total number of steps is $N = 20$ resulting in a planing horizon of one second.

As shown in Fig. 8, we simulate with six quadrotors exchanging their initial positions in an obstacle-free 3D space. Each quadrotor is under localization uncertainty $\Sigma = \text{diag}(0.04 \text{ m}, \ 0.04 \text{ m}, \ 0.04 \text{ m})^2$. For each method, we run the simulation 10 times and calculate the minimum distance among robots. Both our B-UAVC method and the CCNMPC method successfully navigates all robots without collision. An average minimum distance of 0.72 m is observed in our B-UAVC method, while the one of CCNMPC is 0.62 m, which indicates our method is more conservative than the CCNMPC. However, the CCNMPC is centralized and requires robots

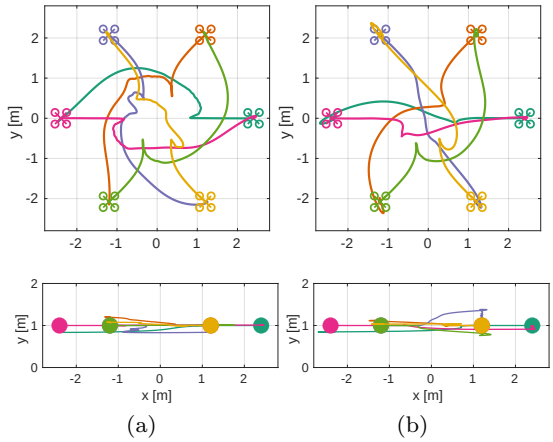


Fig. 8 Simulation with six quadrotors exchanging positions in 3D space. Solid lines represent executed trajectories of the robots. (a) Results of our B-UAVC method. (b) Results of the CCNMPC method (Zhu and Alonso-Mora, 2019b).

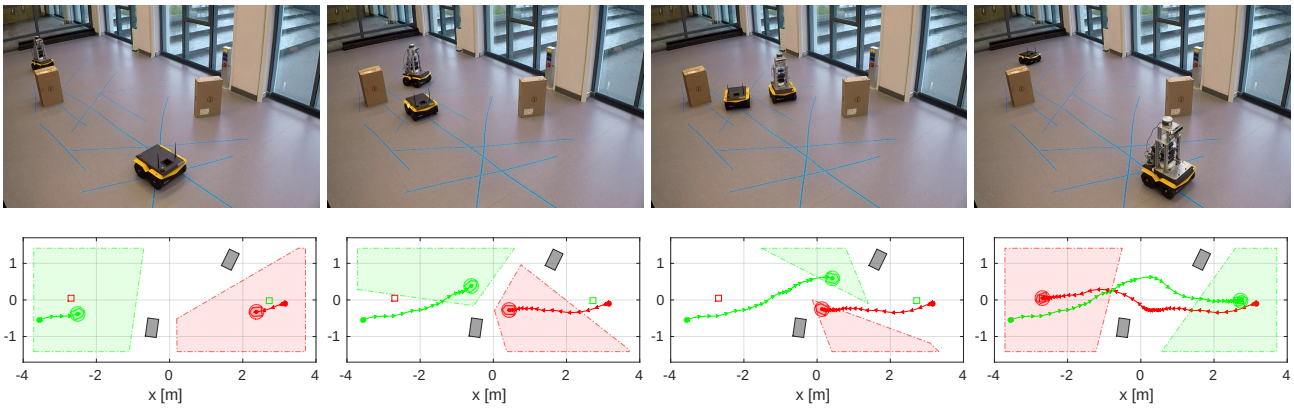
to communicate their future planned trajectories with each other, while the B-UAVC method only needs robot positions to be shared or sensed.

6 Experimental Validation

In this section we describe the experimental results with a team of real robots. A video demonstrating the results accompanies this paper.

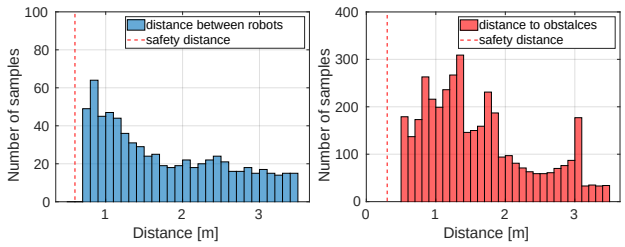
6.1 Experimental Setup

We test our proposed approach on both ground vehicles and aerial vehicles in an indoor environment of 8m (L) × 3.4m (W) × 2.5m (H). Our ground vehicle platform is the Clearpath Jackal robot and our aerial vehicle platform is the Parrot Bebop 2 quadrotor. For ground vehicles, we apply the controller designed for differential-drive robots as shown in Section 4.1.3. For quadrotors, the receding horizon trajectory planner presented in Section 4.2 is employed. The quadrotor dynamics model $\mathbf{f}$ in Problem 1 is given in Appendix D. For solving



(a) $t = 2$ s. (b) $t = 6$ s. (c) $t = 8$ s. (d) $t = 15$ s.

Fig. 9 Collision avoidance with two differential-drive robots and two static obstacles. The two robots are required to swap their positions. Top row: Snapshots of the experiment. Bottom row: Trajectories of the robots. Robot initial and goal positions are marked in circle disks and solid squares, respectively. Grey boxes are static obstacles. The B-UAVCs are shown in shaded patches with dashed boundaries.



(a) (b)

Fig. 10 Experimental results with two differential-drive robots. (a) Histogram of inter-robot distance. (b) Histogram of distance between robots and obstacles.

Problem 1 which is a nonlinear programming problem, we rely on the solver Forces Pro (Zanelli et al., 2020) to generate fast C code to solve it. Both types of robots allow executing control commands sent via ROS. The experiments are conducted in a standard laptop (Quad-core Intel i7 CPU@2.6 GHz) which connects with the robots via WiFi.

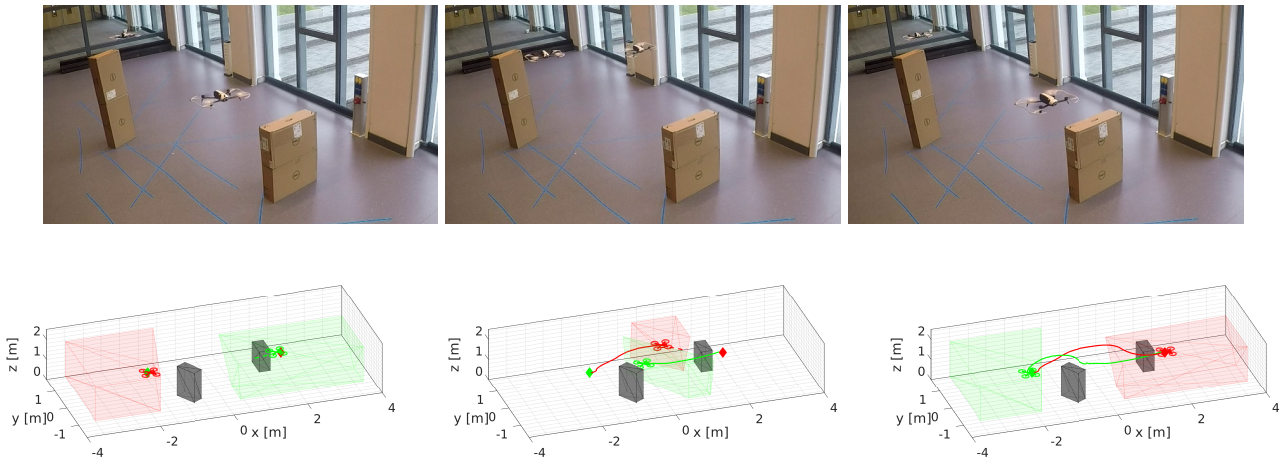
An external motion capture system (OptiTrack) is used to track the pose (position and orientation) of each robot and obstacle in the environment running in real time at 120 Hz, which is regarded as the *real* (ground-truth) pose. To validate collision avoidance under uncertainty, we then manually add Gaussian noise to the real pose data to generate noisy measurements. Taking the noisy measurements as inputs, a standard Kalman filter running at 120 Hz is employed to estimate the states of the robots and obstacles. In all experiments, the added position measurements noise to the robots is zero mean with covariance $\Sigma_i' = \text{diag}(0.06\text{ m}, 0.06\text{ m}, 0.06\text{ m})^2$, which results in an average estimated position uncertainty covariance $\Sigma_i = \text{diag}(0.04\text{ m}, 0.04\text{ m}, 0.04\text{ m})^2$. The added noise to the obstacles is zero mean with covariance $\Sigma_o' = \text{diag}(0.03\text{ m}, 0.03\text{ m}, 0.03\text{ m})^2$ and the resulted estimated position uncertainty covariance is $\Sigma_o = \text{diag}(0.02\text{ m}, 0.02\text{ m}, 0.02\text{ m})^2$. The collision probability threshold is set as $\delta = 0.03$ as in previous works (Zhu and Alonso-Mora, 2019a,b).

6.2 Experimental Results

6.2.1 Experiments with differential-drive robots in 2D

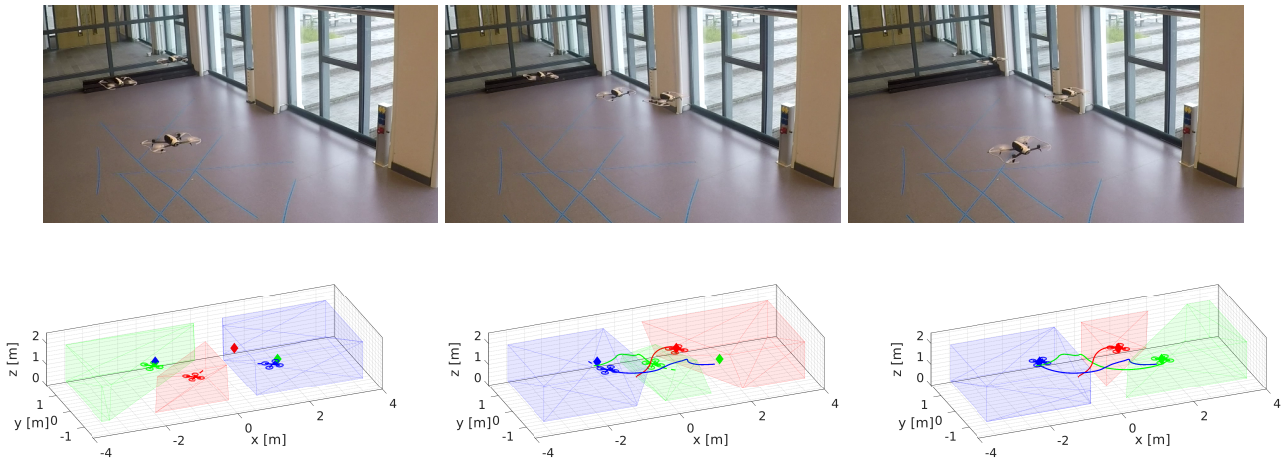
We first validated our proposed approach with two differential-drive robots. In the experiment, two robots are required to swap their positions while avoiding two static obstacles in the environment. The robot safety radius is set as 0.3 m. We run the experiment four times. The two robots successfully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

Fig. 9 presents the results of one run. The top row of the figure shows a series of snapshots during the experiment, while the bottom row shows the robots' travelled trajectories and their corresponding B-UAVCs. It can be seen that each robot always keeps a very safe region (B-UAVC) taking into account its localization and sensing uncertainties. In Fig. 10 we cumulate the distance between the two robots (Fig. 10a) and distance between the robots and obstacles (Fig. 10b) during the whole experiments. It can be seen that a minimum safe inter-robot distance of 0.6 m and a safe robot-obstacle distance of 0.3 m were maintained over all the runs.



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

Fig. 11 Collision avoidance with two quadrotors and two static obstacles. The two quadrotors are required to swap their positions. Top row: Snapshots of the experiment. Bottom row: Trajectories of the robots. Quadrotor initial and goal positions are marked in circles and diamonds. Solid lines represent travelled trajectories and dashed lines represent planned trajectories.



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

Fig. 12 Collision avoidance with three quadrotors in a shared workspace. Top row: Snapshots of the experiment. Bottom row: Trajectories of the robots.

6.2.2 Experiments with quadrotors in 3D

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

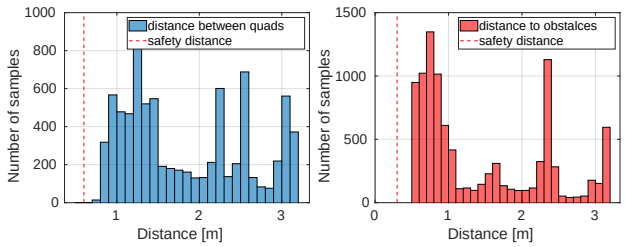
Scenario 1

Two quadrotors swap their positions while avoiding two static obstacles in the environment. We performed the swapping action four times and Fig. 11 presents one run of the results.

Scenario 2

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

Fig. 12 presents a series of snapshots during the experiment. Fig. 13 cumulates the inter-quadrotor distance in the experiments of both scenarios, and the distance between quadrotors and obstacles in Scenario 1. It can



(a) (b)

Fig. 13 Experimental results with two/three quadrotors with/without obstacles. (a) Histogram of inter-robot distance. (b) Histogram of distance between robots and obstacles.

be seen that a minimum safety distance of 0.6 m among quadrotors and that of 0.3 m between quadrotors and obstacles were achieved during the whole experiments.

6.2.3 Experiments with heterogeneous teams of robots

We further tested our approach with one ground differential-drive robot and one quadrotor to show that it can be applied to heterogeneous robot teams. In the experiment, the ground robot only considers its motion and the obstacles in 2D (the ground plane) while ignoring the flying quadrotor. In contrast, the quadrotor considers both itself location and the ground robot's location as well as obstacles in 3D, in which it assumes the ground vehicle has a height of 0.6 m. To this end, the B-UAVC of the ground robot is a 2D convex region while that of the quadrotor is a 3D one.

Fig. 14 shows the results of the experiment. It can be seen that the two robots successfully reached their goals while avoiding each other and the static obstacles. Particularly at $t = 4$ s, the quadrotor actively flies upward to avoid the ground robot. In Fig. 15 we cumulate the distance between the two robots and the distance between robots and obstacles, which show that a safe inter-robot clearance of 0.6 m and that of 0.3 m between robots and obstacles were maintained during the experiment.

7 Conclusion

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

For future work, we plan to employ the proposed approach as a low-level robust collision-avoidance controller, and incorporate it with other higher-level multi-robot trajectory planning and coordination methods to achieve more efficient multi-robot navigation.

Appendix

A Proofs of Lemmas and Theorems

A.1 Proof of Lemma 1

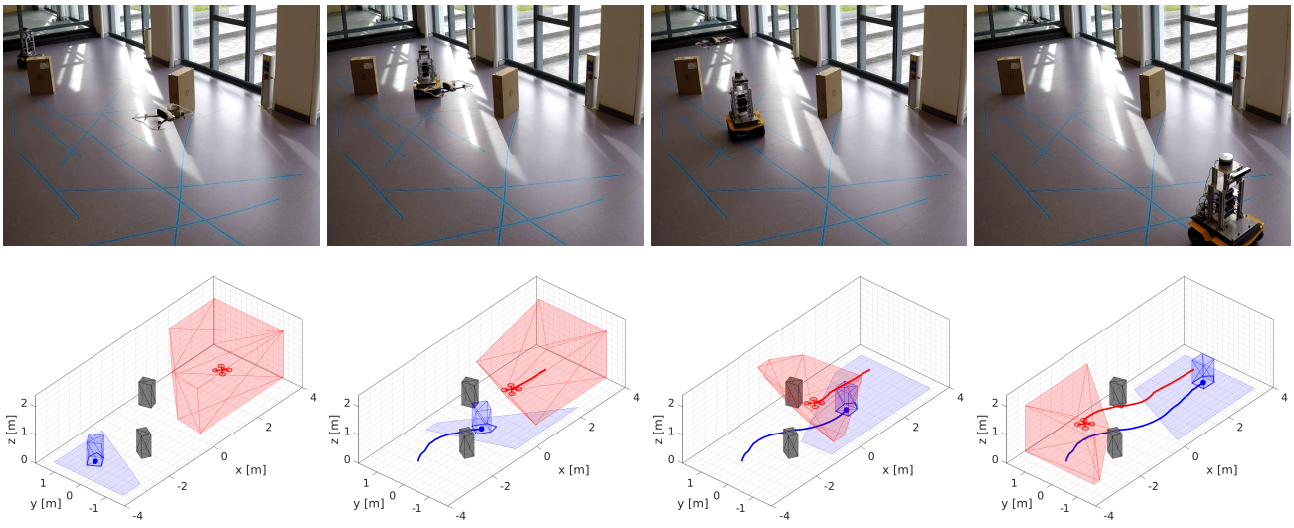
Proof First we can write the random variable $\mathbf{d}_o$ in an equivalent form $\mathbf{d}_o = \Sigma'_o \mathbf{d}'_o$, where $\mathbf{d}'_o \sim \mathcal{N}(0, I) \in \mathbb{R}^d$ and $\Sigma'_o \Sigma'^T_o = \Sigma_o$. Note that $\mathbf{d}'^T_o \mathbf{d}'_o$ is a chi-squared random variable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

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

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

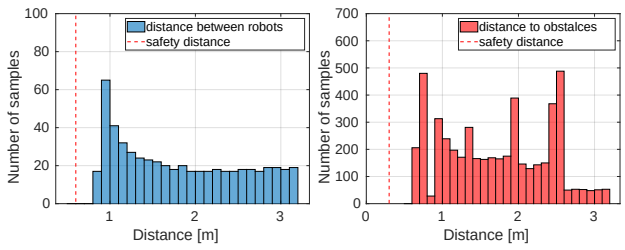
A.2 Proof of Theorem 1

Proof We need to prove that the set $\mathcal{S}_o$ contains the set $\mathcal{O}_o$ with probability $1 - \epsilon$. It is equivalent to that for any point in $\mathcal{O}_o$, the set $\mathcal{S}_o$ contains this point with probability $1 - \epsilon$. Recall the definition of $\mathcal{O}_o$, every $\mathbf{y} \in \mathcal{O}_o$ can be written as $\mathbf{x} + \mathbf{d}_o$ with some $\mathbf{x} \in \hat{\mathcal{O}}_o$. Also note



(a) $t = 0.1$ s. (b) $t = 4$ s. (c) $t = 6$ s. (d) $t = 12$ s.

Fig. 14 Collision avoidance with a heterogeneous team of a differential-drive robot and a quadrotor. Top row: Snapshots of the experiment. Bottom row: Trajectories of the robots.



(a) (b)

Fig. 15 Experimental results with a ground differential-drive robot and a quadrotor. (a) Histogram of inter-robot distance. (b) Histogram of distance between robots and obstacles.

the definition $\mathcal{S}_o = \{\mathbf{x}+\mathbf{d} \mid \mathbf{x} \in \hat{\mathcal{O}}_o, \mathbf{d} \in \mathcal{D}_o\}$. Hence the probability that $\mathcal{S}_o$ contains $\mathbf{y}$ is equal to the probability that $\mathcal{D}_o$ contains $\mathbf{d}_o$. That is, $\Pr(\mathbf{y} \in \mathcal{S}_o) = \Pr(\mathbf{d}_o \in \mathcal{D}_o) = 1-\epsilon, \forall \mathbf{y} \in \mathcal{O}_o$. Thus, $\Pr(\mathcal{O}_o \subseteq \mathcal{S}_o) = 1-\epsilon$. $\mathcal{S}_o$ is a maximal ϵ-shadow of $\mathcal{O}_o$.

B Procedure to Compute the Best Linear Separator Between Two Gaussian Distributions

The objective is to solve the following minimax problem:

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

where

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

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

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

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

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

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

$$\mathbf{a}_{ij} = [t\Sigma_i + (1-t)\Sigma_j]^{-1}\hat{\mathbf{p}}_{ij}, \tag{48}$$

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

$$b_{ij} = \mathbf{a}_{ij}^T\hat{\mathbf{p}}_i + t\mathbf{a}_{ij}^T\Sigma_i\mathbf{a}_{ij} = \mathbf{a}_{ij}^T\hat{\mathbf{p}}_j - (1-t)\mathbf{a}_{ij}^T\Sigma_j\mathbf{a}_{ij}. \tag{49}$$

It is proved that $u_1 = u_2$ must be hold for the solution of the minimax problem (Anderson and Bahadur, 1962), which leads to

$$\mathbf{a}_{ij}^T[t^2\Sigma_i - (1-t)^2\Sigma_j]\mathbf{a}_{ij} = 0. \tag{50}$$

Thus, one can first solve for t by combining Eqs. (48) and (50) via numerical iteration efficiently. Then $\mathbf{a}_{ij}$ and b_{ij} can be computed using Eqs. (48) and (49).

C Deadlock Resolution Heuristic

We detect and resolve deadlocks in a heuristic way in this paper. Let $\|\Delta\mathbf{p}_i\|$ be the position progress between two consecutive time steps of robot i, and $\Delta\mathbf{p}_{\min}$ a predefined minimum allowable progress distance for the robot in n_{dead} time steps. If the robot has not reached its goal and $\Sigma_{n_{\text{dead}}} \|\Delta\mathbf{p}_i\| \leq \Delta\mathbf{p}_{\min}$, we consider the robot as in a deadlock situation. For the one-step controller, each robot must be at the "projected goal" $\mathbf{g}_i^*$ when the system is in a deadlock configuration (Zhou et al., 2017). In this case, each robot chooses one of the nearby edges within its B-UAVC to move along. For receding horizon planning of high-order dynamical systems, the robot may get stuck due to a local minima of the trajectory optimization problem. In this case, we temporarily change the goal location $\mathbf{g}_i$ of each robot by clockwise rotating it along the z axis with 90°, i.e.

$$\mathbf{g}_{i,\text{temp}} = R_Z(-90^\circ)(\mathbf{g}_i - \hat{\mathbf{p}}_i) + \hat{\mathbf{p}}_i, \tag{51}$$

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

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

D Quadrotor Dynamics Model

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

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

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

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

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

The dynamics of the quadrotor position and velocity are

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

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

The attitude dynamics of the quadrotor are

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

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

We obtained the dynamics model parameters $k_{D_x} = 0.25$, $k_{D_y} = 0.33$, $k_{v_z} = 1.2270$, $\tau_{v_z} = 0.3367$, $k_\phi = 1.1260$, $\tau_\phi = 0.2368$, $k_\theta = 1.1075$ and $\tau_\theta = 0.2318$ by collecting real flying data and performing system identification.

References

Alonso-Mora J, Beardsley P, Siegwart R (2018) Cooperative collision avoidance for nonholonomic robots. *IEEE Transactions on Robotics*, 34(2):404–420

Anderson TW, Bahadur RR (1962) Classification into two multivariate normal distributions with different covariance matrices. *The Annals of Mathematical Statistics*, 33(2):420–431

Andrews LC (1997) *Special functions of mathematics for engineers*, vol 49. SPIE press

Arslan O, Koditschek DE (2019) Sensor-based reactive navigation in unknown convex sphere worlds. *International Journal of Robotics Research*, 38(2-3):196–223

Astolfi A (1999) Exponential stabilization of a wheeled mobile robot via discontinuous control. *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, 121(1):121–126

Axelrod B, Kaelbling LP, Lozano-Pérez T (2018) Provably safe robot navigation with obstacle uncertainty. *The International Journal of Robotics Research*, 37(13-14):1760–1774

Bareiss D, van den Berg J (2015) Generalized reciprocal collision avoidance. *The International Journal of Robotics Research*, 34(12):1501–1514

Van den Berg J, Lin M, Manocha D (2008) Reciprocal velocity obstacles for real-time multi-agent navigation.

In: 2008 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp 1928–1935

Blackmore L, Ono M, Williams BC (2011) Chance-constrained optimal path planning with obstacles. *IEEE Transactions on Robotics*, 27(6):1080–1094

Breitenmoser A, Martinoli A (2016) On Combining Multi-robot Coverage and Reciprocal Collision Avoidance. *In: Springer Tracts in Advanced Robotics*, vol 112, Springer Japan, Tokyo, pp 49–64

Chen Y, Cutler M, How JP (2015) Decoupled multiagent path planning via incremental sequential convex programming. *In: 2015 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, pp 5954–5961

Claes D, Hennes D, Tuyls K, Meeussen W (2012) Collision avoidance under bounded localization uncertainty. *In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, pp 1192–1198

Dawson C, Jasour A, Hofmann A, Williams B (2020) Provably Safe Trajectory Optimization in the Presence of Uncertain Convex Obstacles. *In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, pp 6237–6244

Deits R, Tedrake R (2015a) Computing large convex regions of obstacle-free space through semidefinite programming. *In: Springer Tracts in Advanced Robotics*, vol 107, pp 109–124

Deits R, Tedrake R (2015b) Efficient mixed-integer planning for uavs in cluttered environments. *In: 2015 IEEE international conference on robotics and automation (ICRA)*, IEEE, pp 42–49

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

Gopalakrishnan B, Singh AK, Kaushik M, Krishna KM, Manocha D (2017) Prvo: Probabilistic reciprocal velocity obstacle for multi robot navigation under uncertainty. *In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, pp 1089–1096

Hardy J, Campbell M (2013) Contingency planning over probabilistic obstacle predictions for autonomous road vehicles. *IEEE Transactions on Robotics*, 29(4):913–929

Hönig W, Preiss JA, Kumar TK, Sukhatme GS, Ayanian N (2018) Trajectory planning for quadrotor swarms. *IEEE Transactions on Robotics*, 34(4):856–869

Kamel M, Alonso-Mora J, Siegwart R, Nieto J (2017) Robust collision avoidance for multiple micro aerial vehicles using nonlinear model predictive control. *In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, pp 236–243

Kozlov MK, Tarasov SP, Khachiyan LG (1980) The polynomial solvability of convex quadratic programming. *USSR Computational Mathematics and Mathematical Physics*, 20(5):223–228

Liu S, Watterson M, Mohta K, Sun K, Bhattacharya S, Taylor CJ, Kumar V (2017) Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments. *IEEE Robotics and Automation Letters*, 2(3):1688–1695

Luis CE, Vukosavljev M, Schoellig AP (2020) Online trajectory generation with distributed model predictive control for multi-robot motion planning. *IEEE Robotics and Automation Letters*, 5(2):604–611

Luo W, Sun W, Kapoor A (2020) Multi-robot collision avoidance under uncertainty with probabilistic safety barrier certificates. *In: 2020 Advances in Neural Information Processing Systems (NeurIPS)*, vol 33

Lyons D, Calliess J, Hanebeck UD (2012) Chance constrained model predictive control for multi-agent systems with coupling constraints. *In: 2012 American Control Conference (ACC)*, IEEE, pp 1223–1230

Morgan D, Subramanian GP, Chung SJ, Hadaegh FY (2016) Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming. *International Journal of Robotics Research*, 35(10):1261–1285

Nägeli T, Meier L, Domahidi A, Alonso-Mora J, Hilliges O (2017) Real-time planning for automated multi-view drone cinematography. *ACM Transactions on Graphics*, 36(4):1–10

Okabe A, Boots B, Sugihara K, Chiu SN (2009) *Spatial tessellations: Concepts and applications of Voronoi diagrams*. John Wiley & Sons

Pierson A, Schwarting W, Karaman S, Rus D (2020) Weighted buffered voronoi cells for distributed semi-cooperative behavior. *In: 2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, pp 5611–5617

Schmerling E, Pavone M (2017) Evaluating trajectory collision probability through adaptive importance sampling for safe motion planning. *In: Robotics: Science and Systems*, vol 13

Serra-Gómez A, Brito B, Zhu H, Chung JJ, Alonso-Mora J (2020) With whom to communicate: Learning efficient communication for multi-robot collision avoidance. *In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, pp 11770–11776

Shim D, Kim H, Sastry S (2003) Decentralized nonlinear model predictive control of multiple flying robots. *In: 2003 IEEE Conference on Decision and Control (CDC)*, IEEE, pp 3621–3626

Tordesillas J, Lopez BT, How JP (2019) Faster: Fast and safe trajectory planner for flights in unknown environments. *In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, pp 1934–1940

Van Den Berg J, Guy SJ, Lin M, Manocha D (2011) Reciprocal n-body collision avoidance. *In: Springer Tracts in Advanced Robotics*, vol 70, pp 3–19

Wang M, Schwager M (2019) Distributed collision avoidance of multiple robots with probabilistic buffered voronoi cells. *In: 2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, IEEE, pp 169–175

Zanelli A, Domahidi A, Jerez J, Morari M (2020) FORCES NLP: an efficient implementation of interior-point methods for multistage nonlinear nonconvex programs. *International Journal of Control*, (1):13–29

Zhou D, Wang Z, Bandyopadhyay S, Schwager M (2017) Fast, on-line collision avoidance for dynamic vehicles using buffered voronoi cells. *IEEE Robotics and Automation Letters*, 2(2):1047–1054

Zhou L, Tzoumas V, Pappas GJ, Tokekar P (2018) Resilient active target tracking with multiple robots. *IEEE Robotics and Automation Letters*, 4(1):129–136

Zhu H, Alonso-Mora J (2019a) B-uavc: Buffered uncertainty-aware voronoi cells for probabilistic multi-robot collision avoidance. *In: 2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, IEEE, pp 162–168

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

Zhu H, Juhl J, Ferranti L, Alonso-Mora J (2019) Distributed multi-robot formation splitting and merging in dynamic environments. *In: 2019 International Conference on Robotics and Automation (ICRA)*, IEEE, pp 9080–9086