B-UAVC: Buffered Uncertainty-Aware Voronoi Cells for Probabilistic Multi-Robot Collision Avoidance

Hai Zhu and Javier Alonso-Mora

***Abstract*— This paper presents B-UAVC, a distributed collision avoidance method for multi-robot systems that accounts for uncertainties in robot localization. In particular, Buffered Uncertainty-Aware Voronoi Cells (B-UAVC) are employed to compute regions where the robots can safely navigate. By computing a set of chance constraints, which guarantee that the robot remains within its B-UAVC, the method can be applied to non-holonomic robots. A local trajectory for each robot is then computed by introducing these chance constraints in a receding horizon model predictive controller. The method guarantees, under the assumption of normally distributed position uncertainty, that the collision probability between the robots remains below a specified threshold. We evaluate the proposed method with a team of quadrotors in simulations and in real experiments.**

I. Introduction

Multi-robot collision avoidance is a fundamental problem when deploying a team of autonomous mobile robots. Given the robots' 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. Most existing algorithms solve the problem in a deterministic manner, where the robots' states are perfectly known. Practically, however, the robots' states are obtained by an estimator and are not deterministically known. Taking this uncertainty into consideration is of utmost importance in multi-robot collision avoidance. In this paper, we present a probabilistic method for multi-robot collision avoidance under localization uncertainty. The method is distributed, does not require communication, and relies on the computation of buffered uncertainty-aware Voronoi cells (B-UAVC).

A. Related Works

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 [1], which builds on the concept of velocity obstacles (VO) [2]. 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 [3] casting the problem into a linear programming formulation which can be solved efficiently, the generalized RVO method [4] applying for heterogeneous teams of robots, and the ε-cooperative collision avoidance (εCCA) method [5] 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 [6], decoupled MPC [7], and sequential MPC [8], [9].

Some of the above deterministic collision avoidance methods have been extended to scenarios where robot motion uncertainty is considered. Based on RVO, the COCALU method [10] takes into account bounded localization uncertainty of the robots by constructing an error-bounded convex hull of the VO of each robot. [11] presents a probabilistic RVO method where robot state uncertainty is assumed to follow a Gaussian distribution. In [12] the authors present 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 [13] for planar robots, where rectangular regions were computed and inter-robot collision avoidance was transformed to avoiding overlaps of those regions. Using local linearization, [14] introduced a chance constrained nonlinear MPC (CCNMPC) method to guarantee 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 only apply to homogeneous teams of robots. Probabilistic MPC-based methods typically demand communication of the planned trajectory of each robot to guarantee collision avoidance, which is not well suited for systems with many robots. The alternative is to assume that all other robots move with constant velocity, which has been shown to produce collisions in cluttered environments [14]. In this paper, we build on the buffered Voronoi cell (BVC) method [15] for deterministic multi-robot collision avoidance, where each robot only needs to know the relative positions of neighboring robots. We extend the method into probabilistic scenarios considering robot localization uncertainty by mathematically formalizing a buffered uncertainty-aware Voronoi cell (B-UAVC).

B. Contribution

The main contribution of this paper is an on-line distributed and communication-free method for probabilistic multi-robot collision avoidance under localization uncertainty. The method relies on the computation of buffered

This work is supported in part by the Netherlands Organization for Scientific Research, Applied Sciences domain and ONR Global.

The authors are with the Department of Cognitive Robotics, Delft University of Technology, 2628 CD, Delft, The Netherlands {h.zhu; j.alonsomora}@tudelft.nl

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 and neighboring robots, and plans its motion within the B-UAVC. Probabilistic collision avoidance is guaranteed by constraining each robot's motion within its corresponding B-UAVC, via chance constraints, such that the inter-robot collision probability is below a user-specified threshold. We assume that robots can obtain the position and uncertainty information of their neighbors via communication or via onboard sensors and a filter. Communication among robots is not necessary if we assume that the estimated position uncertainty of other robots is higher than the real one.

The method can be applied to non-holonomic robots, such as a team of drones, by computing a local trajectory for each robot with a model predictive controller that keeps each robot within its B-UAVC. The method guarantees, under the assumption of normally distributed position uncertainty, that the collision probability between the robots remains below a specified threshold. We compare the proposed method in simulation with state-of-the-art approaches and evaluate it in real-world experiments with a team of quadrotors.

II. Preliminaries

A. Problem Statement

A group of n robots freely moving in a d-dimensional space $W \quad \mathbb{R}^d, d \in \{2,3\}$ is considered. For each robot $i \in I = \{1,2,\ldots,n\}$, $\mathbf{p}_i \in \mathbb{R}^d$ denotes its position, $\mathbf{v}_i = \dot{\mathbf{p}}_i$ its velocity and $\mathbf{a}_i = \dot{\mathbf{v}}_i$ its acceleration. Each robot has a safety radius r_i. 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. $\mathbf{p}_i \quad \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$, where a hat $\hat{}$ denotes the mean of a random variable. In this paper, we assume that each robot only knows the mean position and covariance of other robots in the group (or has an estimate of them).

Any two robots i and j in the group are mutually collision-free if $\|\mathbf{p}_i \quad \mathbf{p}_j\| \quad r_i + r_j$. Note that the robot position is a random variable described by a Gaussian distribution, which has 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 $\mathbf{p}_i \quad \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$ is probabilistic collision-free with a robot j at position $\mathbf{p}_j \quad \mathcal{N}(\hat{\mathbf{p}}_j, \Sigma_j)$ if*

$$\Pr(\|\mathbf{p}_i \quad \mathbf{p}_j\| \quad r_i + r_j) \quad 1 \quad \delta, \tag{1}$$

where δ is the collision probability threshold, $\Pr(\,)$ *indicates the probability of an event.*

The objective of probabilistic collision avoidance is to compute a local motion for each robot in the group, that respects its kinematic and dynamic constraints, makes progress towards its goal location and is probabilistic collision-free with other robots in the team for a short time horizon.

In our proposed method, to achieve probabilistic collision avoidance, we rely on two concepts, which are described in the following: a) Voronoi cells for partitioning the workspace among the robots and b) linear separators of a pair of Gaussian distributions. In the upcoming Sec. III we will show how the linear separator can be employed to compute the B-UAVC.

B. Voronoi Cell

For a set of deterministic points $(\mathbf{p}_1, \ldots, \mathbf{p}_n) \in \mathbb{R}^d$, the normal Voronoi cell (VC) of each point $i \in I$ is defined as [16]

$$V_i = \{\mathbf{p} \in \mathbb{R}^d : \|\mathbf{p} \quad \mathbf{p}_i\| \quad \|\mathbf{p} \quad \mathbf{p}_j\|, \forall j \neq i\}, \tag{2}$$

where $\|\cdot\|$ denotes the Euclidean distance. Equation (2) can be written as

$$V_i = \{\mathbf{p} \in \mathbb{R}^d : \mathbf{p}_{ij}^T \mathbf{p} \quad \mathbf{p}_{ij}^T \frac{\mathbf{p}_i + \mathbf{p}_j}{2}, \forall j \neq i\}, \tag{3}$$

where $\mathbf{p}_{ij} = \mathbf{p}_j \quad \mathbf{p}_i$. It can be observed that V_i is the intersection of a set of planes 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 plane between each pair of points.

C. Best Linear Separator

In contrast to only separating two deterministic points in VC, in [17], a best linear separator for two Gaussian distributions with different covariance is developed. Given $\mathbf{p}_i \quad \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$ and $\mathbf{p}_j \quad \mathcal{N}(\hat{\mathbf{p}}_j, \Sigma_j)$, 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: $\mathbf{a}_{ij}^T \mathbf{p} \quad b_{ij}$ 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, which is briefly described in the following.

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

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

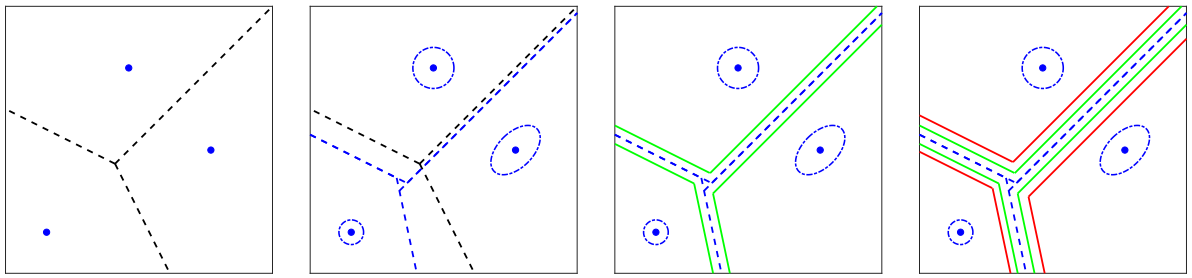
where $\Phi(\,)$ denotes the cumulative distribution function (CDF) of the standard normal distribution. Similarly, the misclassification probability when $\mathbf{p}$ is from the second distribution is

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

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

$$(\mathbf{a}_{ij}, b_{ij}) = \arg\min_{\mathbf{a}_{ij}, b_{ij}} \max(\Pr_i, \Pr_j), \tag{4}$$

which can be solved using a fast minimax procedure. Here we give the method to compute $\mathbf{a}_{ij}$ and b_{ij} without proof.



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

Fig. 1: Example of buffered uncertainty-aware Voronoi cell (B-UAVC). Blue dots are robots; blue dash-dot line indicates the 3-σ confidence ellipsoid of the position uncertainty. (a) Deterministic Voronoi cell (VC, the boundary in black dashed line). (b) Uncertainty-aware Voronoi cell (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).

One can refer to [17], [18] for more details and proof. First we write $\mathbf{a}_{ij}$ and b_{ij} as

$$\mathbf{a}_{ij} = [t\Sigma_i + (1 \quad t)\Sigma_j] \ ^1(\hat{\mathbf{p}}_j \quad \hat{\mathbf{p}}_i), \tag{5}$$

$$b_{ij} = \mathbf{a}_{ij}^T\hat{\mathbf{p}}_i + t\mathbf{a}_{ij}^T\Sigma_i\mathbf{a}_{ij} = \mathbf{a}_{ij}^T\hat{\mathbf{p}}_j \quad (1 \quad t)\mathbf{a}_{ij}^T\Sigma_j\mathbf{a}_{ij}, \tag{6}$$

where $0 < t < 1$. Then substituting Eq. (5) into the following equality

$$\mathbf{a}_{ij}^T[t^2\Sigma_i \quad (1 \quad t)^T\Sigma_j]\mathbf{a}_{ij} = 0, \tag{7}$$

we can obtain a nonlinear equation which can be solved to obtain the value of t. Finally, we can compute $\mathbf{a}_{ij}$ and b_{ij} using Eqs. (5) and (6).

***Remark* 1.** *The best linear separator coincides with the separating plane of Eq. (3) when $\Sigma_i = \Sigma_j = \sigma^2\mathbf{I}$. In this case, $t = 0.5$, $\mathbf{a}_{ij} = \frac{2}{\sigma^2}(\hat{\mathbf{p}}_j \quad \hat{\mathbf{p}}_i)$ and $b_{ij} = (\hat{\mathbf{p}}_j \quad \hat{\mathbf{p}}_i)^T(\hat{\mathbf{p}}_i + \hat{\mathbf{p}}_j)/\sigma^2$.*

***Remark* 2.** *8i 6= j 2 I , $\mathbf{a}_{ji} = \quad \mathbf{a}_{ij}, b_{ji} = \quad b_{ij}$. This can be obtained according to the definition of best linear separator.*

III. Buffered UAVC

In this section, we formally introduce our definition of buffered uncertainty-aware Voronoi cell, give the construction method and present its important properties.

A. Definition of UAVC

To consider the robot position uncertainty, we introduce the uncertainty-aware Voronoi cell using the concept of best linear separator described in section II-C.

***Definition* 2** (Uncertainty-Aware Voronoi Cell)**.** *For a set of Gaussian random points with mean $(\hat{\mathbf{p}}_1, \ldots, \hat{\mathbf{p}}_n)$ 2 R^d and covariance $(\Sigma_1, \ldots, \Sigma_n)$ 2 $\mathsf{R}^d \ ^d$, the uncertainty-aware Voronoi cell (UAVC) of each point i 2 I is defined as*

$$V_i^u = \{\mathbf{p} \ 2 \ \mathsf{R}^d : \mathbf{a}_{ij}^T\mathbf{p} \quad b_{ij}, 8j \ 6= i\}, \tag{8}$$

where $\mathbf{a}_{ij}, b_{ij}$ formulate a best linear separator between i and j as described in section II-C.

Figure 1b illustrates the UAVC of three robots which have different uncertainty covariances. It can be observed that the robot with larger uncertainty covariance is assigned a larger partition of the space.

***Remark* 3.** *$V_i^u = V_i$ when $\Sigma_i = \Sigma_j = \sigma^2\mathbf{I}$. This can be derived from Remark 1.*

***Remark* 4.** *In contrast to deterministic Voronoi cells, the UAVCs generally do not constitute a full tessellation of the workspace, i.e. $\bigcup_1^n V_i^u \quad W$, as shown in Fig. 1b.*

B. Collision Avoidance Buffer

Now we introduce two buffers to the UAVC, to account for the robot physical safety radius r and the collision probability threshold δ.

1) Robot safety radius r: We create the robot safety radius buffer by shifting the boundary of the UAVC towards the robot and define

$$V_i^{u,r} = \{\mathbf{p} \ 2 \ \mathsf{R}^d : \mathbf{a}_{ij}^T\mathbf{p} \quad b_{ij} \quad \beta_i^r, 8j \ 6= i\}, \tag{9}$$

where the safety radius buffer $\beta_i^r = r_i\|\mathbf{a}_{ij}\|$. Figure 1c shows the buffered UAVC of each robot after taking into account their safety radius.

2) Collision probability threshold δ: We create the collision probability buffer by shifting the boundary of the UAVC towards the robot and define

$$V_i^{u,\delta} = \{\mathbf{p} \ 2 \ \mathsf{R}^d : \mathbf{a}_{ij}^T\mathbf{p} \quad b_{ij} \quad \beta_i^\delta, 8j \ 6= i\}, \tag{10}$$

in which the collision probability buffer is

$$\beta_i^\delta = \sqrt{2\mathbf{a}_{ij}^T\Sigma_i\mathbf{a}_{ij}} \ \text{erf} \ ^1(2\sqrt{1 \quad \delta} \quad 1), \tag{11}$$

where erf( ) is the Gauss error function [19] defined as $\text{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{\ t^2}dt$ and erf ^{1}( ) is its inverse. In this paper, we assume the threshold $0 < \delta < \frac{3}{4}$, which is reasonable in practice. Hence, erf $^1(2\sqrt{1 \quad \delta} \quad 1) > 0, \beta_i^\delta > 0$. This buffer can be obtained by following the proof of Theorem 1.

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

$$V_i^{u,b} = \{\mathbf{p} \ 2 \ \mathsf{R}^d : \mathbf{a}_{ij}^T\mathbf{p} \quad b_{ij} \quad \beta_i^r \quad \beta_i^\delta, 8j \ 6= i\}. \tag{12}$$

Note that we have $V_i^{u,r} \quad V_i^{u,b}$ and $V_i^{u,\delta} \quad V_i^{u,b}$. Figure 1d shows the final B-UAVC of each robot in the team.

C. Properties of B-UAVC

***Theorem* 1** (Probabilistic Collision-Free)**.** *$\forall \mathbf{p}_i \quad \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$ and $\mathbf{p}_j \quad \mathcal{N}(\hat{\mathbf{p}}_j, \Sigma_j)$, where $\hat{\mathbf{p}}_i \in V_i^{u,b}, \hat{\mathbf{p}}_j \in V_j^{u,b}, i \neq j \in \mathcal{I}$, we have $\Pr(\|\mathbf{p}_i \quad \mathbf{p}_j\| \quad r_i + r_j) \quad 1 \quad \delta$, i.e. the probability of collision for both robots i and j is below the threshold δ.*

Proof. We first introduce the following lemma:

***Lemma* 1** (Linear Chance Constraint [20])**.** *A multivariate random variable $\mathbf{x} \quad \mathcal{N}(\hat{\mathbf{x}}, \Sigma)$ satisfies*

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

Then we prove that

$$\mathbf{p}_i \in V_i^{u,r}, \mathbf{p}_j \in V_j^{u,r} \Rightarrow \|\mathbf{p}_i \quad \mathbf{p}_j\| \quad r_i + r_j. \tag{13}$$

According to Eq. (9), if $\mathbf{p}_i \in V_i^{u,s}, \mathbf{p}_j \in V_j^{u,s}$ we have

$$\mathbf{a}_{ij}\mathbf{p}_i \quad b_{ij} \quad r_i\|\mathbf{a}_{ij}\|, \tag{14}$$

$$\mathbf{a}_{ji}\mathbf{p}_j \quad b_{ji} \quad r_j\|\mathbf{a}_{ji}\|. \tag{15}$$

Note that $\mathbf{a}_{ij} = \quad \mathbf{a}_{ji}$, $b_{ij} = \quad b_{ji}$ (Remark 2). Hence, the summation of the above two equations yields

$$\mathbf{a}_{ij}(\mathbf{p}_j \quad \mathbf{p}_i) \quad (r_i + r_j)\|\mathbf{a}_{ij}\|. \tag{16}$$

Therefore,

$$\|\mathbf{p}_i \quad \mathbf{p}_j\| \quad \frac{(r_i + r_j)\|\mathbf{a}_{ij}\|}{\|\mathbf{a}_{ij}\|} = (r_i + r_j).$$

Next, we compute the probability of $\mathbf{p}_i \in V_i^{u,r}$ when $\mathbf{p}_i \quad \mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_i)$ where $\hat{\mathbf{p}}_i \in V_i^{u,b}$ (similarly for jth robot). According to Eq. (12), we have

$$\mathbf{a}_{ij}^T\hat{\mathbf{p}}_i \quad b_{ij} \quad r_i\|\mathbf{a}_{ij}\| \quad \sqrt{2\mathbf{a}_{ij}^T\Sigma_i\mathbf{a}_{ij}} \quad \text{erf}^{-1}(2\sqrt{1 \quad \delta} \quad 1). \tag{17}$$

Hence, applying Lemma 1 and substituting Eq. (17) yields

$$\begin{aligned}\Pr(\mathbf{p}_i \in V_i^{u,r}) &= \Pr(\mathbf{a}_{ij}\mathbf{p}_i \quad b_{ij} \quad r_i\|\mathbf{a}_{ij}\|) \\ &= \frac{1}{2} + \frac{1}{2}\text{erf}\left(\frac{b_{ij} \quad r_i\|\mathbf{a}_{ij}\| \quad \mathbf{a}_{ij}^T\hat{\mathbf{p}}_i}{\sqrt{2\mathbf{a}_{ij}^T\mathbf{a}_{ij}}}\right) \\ &\quad \frac{1}{2} + \frac{1}{2}\text{erf}\left(\text{erf}^{-1}(2\sqrt{1 \quad \delta} \quad 1)\right) \\ &= \frac{1}{2} + \frac{1}{2}(2\sqrt{1 \quad \delta} \quad 1) \\ &= \sqrt{1 \quad \delta}.\end{aligned} \tag{18}$$

Similarly for the robot j, $\Pr(\mathbf{p}_j \in V_j^{u,r}) \quad \sqrt{1 \quad \delta}$.

Finally, by applying Eq. (13), we have

$$\begin{aligned}\Pr(\|\mathbf{p}_i \quad \mathbf{p}_j\| \quad r_i + r_j) \quad &\Pr(\mathbf{p}_i \in V_i^{u,r}) \quad \Pr(\mathbf{p}_j \in V_j^{u,r}) \\ &\sqrt{1 \quad \delta}\ \sqrt{1 \quad \delta} \\ = 1 \quad \delta.\end{aligned} \tag{19}$$

□

IV. Collision Avoidance Using B-UAVC

In this section, we present our distributed collision avoidance method using the B-UAVC. We start with describing a one-step controller for single integrator robots, followed by a receding horizon planning formulation for high-order dynamical systems and a method discussion.

A. Single-Integrator Dynamics: One-Step Controller

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 [15], a fast one-step controller can be designed to make the robot move towards its goal location $\mathbf{g}_i$, see Algorithm 1 . At each time step, each robot first computes its B-UVAC $V_i^{u,b}$, then finds the closest point $\mathbf{g}_i \in V_i^{u,b}$ to $\mathbf{g}_i$ and generates a control velocity $\mathbf{u}_i = v_{i,\max}(\mathbf{g}_i \quad \mathbf{g}_i)/\|\mathbf{g}_i \quad \mathbf{g}_i\|$, where $v_{i,\max}$ is the robot maximal speed. The closest point $\mathbf{g}_i$ can be found by checking each edge and vertex of $V_i^{u,b}$ [15].

Algorithm 1 One-Step Controller

1: **for** Each robot $i \in \mathcal{I}$ **do**
2: Compute the B-UAVC $V_i^{u,b}$
3: Find the closest point $\mathbf{g}_i \in V_i^{u,b}$ to $\mathbf{g}_i$
4: Compute $\mathbf{u}_i = v_{i,\max}(\mathbf{g}_i \quad \mathbf{g}_i)/\|\mathbf{g}_i \quad \mathbf{g}_i\|$
5: **end for**

B. High-Order Dynamics: Receding Horizon Planning

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

***Problem* 1** (Receding Horizon Trajectory Planning)**.**

$$\begin{aligned}\min_{\hat{\mathbf{x}}_i^{1:N}, \mathbf{u}_i^{0:N-1}} \quad &\sum_{k=0}^{N \quad 1}((\hat{\mathbf{p}}_i^k \quad \hat{\mathbf{p}}_{i,r}^k)^T Q(\hat{\mathbf{p}}_i^k \quad \hat{\mathbf{p}}_{i,r}^k) + \mathbf{u}_i^k R\mathbf{u}_i^k) \\ &+ (\hat{\mathbf{p}}_i^N \quad \hat{\mathbf{p}}_{i,r}^N)^T Q_N(\hat{\mathbf{p}}_i^N \quad \hat{\mathbf{p}}_{i,r}^N) \\ \text{s.t.} \quad &\mathbf{x}_i^0 = \hat{\mathbf{x}}_i, && \text{(20a)} \\ &\hat{\mathbf{x}}_i^k = \mathbf{f}_i(\hat{\mathbf{x}}_i^{k \quad 1}, \mathbf{u}_i^{k \quad 1}), && \text{(20b)} \\ &\hat{\mathbf{p}}_i^k \in V_i^{u,b}, && \text{(20c)} \\ &\mathbf{u}_i^{k \quad 1} \in U_i, && \text{(20d)} \\ &\forall i \in \mathcal{I}, \forall k \in \{1, \ldots, N\}. && \text{(20e)}\end{aligned}$$

In Problem 1, $\mathbf{p}_{i,r}$ is the reference trajectory of robot i, which generally comes from some high-level path planner, e.g. [21]; U_i is the admissable control space; Q, R, Q_N are positive semi-definite symmetric matrices. The constraint (20c) restrains the planned trajectory to be within the robot's B-UAVC $V_i^{u,b}$. According to the definition of $V_i^{u,b}$ in Eq. (12), the constraint can be formulated as a set of linear inequality constraints:

$$\mathbf{a}_{ij}^T\hat{\mathbf{p}}_i^k \quad b_{ij} \quad \beta_i^r \quad \beta_i^\delta, \forall j \in \mathcal{I}, j \neq i. \tag{21}$$

Solving the above trajectory optimization problem may drive the robot towards to the boundary of its B-UAVC. For second or higher dynamics, the robot has limited acceleration. When the robot is moving to the boundary of its B-UAVC with a large velocity, it might not be able to continue to stay within the B-UAVC. Hence, to improve safety, as illustrated in Fig. 2 we introduce an additional safety stopping buffer, which is defined as

$$ds_i^k = \begin{cases} \frac{k\mathbf{a}_{ij}^T \mathbf{v}_i^k k^2}{2a_{\max}}, & \text{if } \mathbf{a}_{ij}^T \mathbf{v}_i^k > 0; \\ 0, & \text{otherwise}, \end{cases} \tag{22}$$

where $a_{\max}$ is the maximal acceleration of the robot. The constraints in Eq. (21) now become

$$\mathbf{a}_{ij}^T \hat{\mathbf{p}}_i^k \quad b_{ij} \quad \beta_i^r \quad \beta_i^\delta \quad ds_i^k, 8j \; 2 \; I, j \; 6= \; i. \tag{23}$$

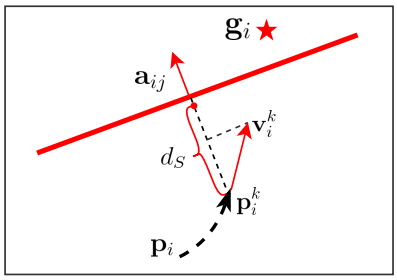


Fig. 2: Additional buffer is added to allow robots with second-order dynamics to have enough space to decelerate.

C. Discussion

1) Collision avoidance guarantee: We assume that the robot's position uncertainty within the short planning horizon is less or equal to its real-time localization uncertainty. Under the assumption, by constraining each robot's motion within its corresponding B-UAVC, our formulation guarantees, by construction, that the collision probability of the robot with each other robot at every stage of the planned trajectory is below δ. (Section III-C Theorem 1).

2) Deadlock resolution: Since the proposed collision avoidance method is local and distributed, deadlocks may happen. For the one-step controller, each robot must be at $\mathbf{g}_i$ when the system is in a deadlock configuration. In this case, each robot chooses one of the nearby edges within its B-UAVC to move along [15]. 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 temporally change the goal location $\mathbf{g}_i$ of each robot by clockwise rotating it along the z axis with 90 , i.e.

$$\mathbf{g}_{i,temp} = R_Z(\quad 90 \quad)(\mathbf{g}_i \quad \hat{\mathbf{p}}_i) + \hat{\mathbf{p}}_i, \tag{24}$$

where R_Z denotes the rotation matrix for rotations around z-axis. Once the robot recovers from stuck, i.e. once it moves with a positive speed, 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 to the goals since livelock (robots continuously repeat a sequence of behaviors that bring them from one deadlock situation to another one) may still occur.

V. Results

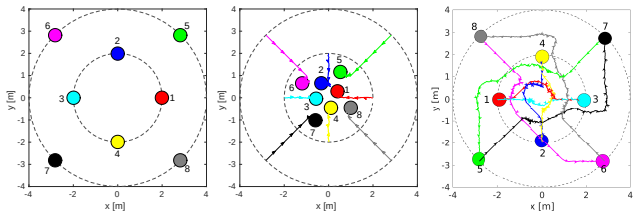
In this section, we evaluate the proposed method in both simulations and experiments. A video demonstrating the results accompanies this paper. We use a commodity Intel i7 CPU@2.6GHz computer for computations, which are performed in MATLAB. The Forces Pro solver [22] is employed to generate fast optimized MPC code for the receding horizon planning problem.

A. Simulation Results

1) Simulation with single-integrator robots: We first evaluate our method in simulation with single-integrator robots moving in a planar space. The one-step controller described in Section IV-A is employed. As shown in Fig. 3, eight robots are initially placed around two circles with their goal locations at the opposite side of the same circle. The robot radius is set as $r = 0.3$ m. Robots 1-4 have a localization uncertainty $\Sigma^h = \text{diag}(0.06 \text{ m}, \; 0.06 \text{ m})^2$ while robots 5-8 have $\Sigma^l = \text{diag}(0.04 \text{ m}, \; 0.04 \text{ m})^2$. The inter-robot collision probability threshold is set as $\delta = 0.03$.

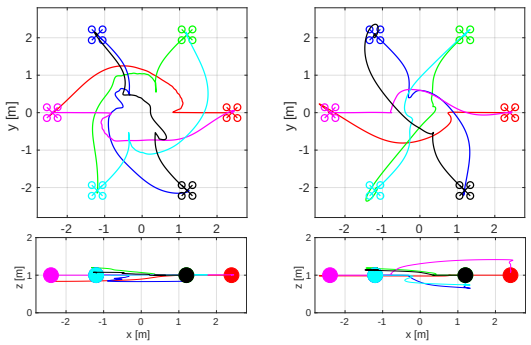
We run the simulation 10 times and compare our B-UAVC method with the deterministic BVC method [15]. Simulation results show that when taking into account robot localization uncertainty, collision happened in all runs with BVC. Instead, B-UAVC successfully navigates the robots without collision in every run. An average minimum distance among robots of those runs is 0.68 m and the minimum minimum distance is 0.65 m. Fig. 3 shows an instance of the simulations.

2) Comparison with other methods: We then 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 [14], which requires communication of future planned trajectories. For both methods, we adopt the same quadrotor dynamics model described in [14] for planning. The quadrotor radius is $r = 0.3$ m and the collision probability threshold is set as $\delta = 0.03$. The time step is



(a) (b) (c)

Fig. 3: An instance of simulation with eight single-integrator robots under localization uncertainty. Robot 1-4 have a localization uncertainty $\Sigma^h = \text{diag}(0.06 \text{ m}, \; 0.06 \text{ m})^2$ while robots 5-8 have $\Sigma^l = \text{diag}(0.04 \text{ m}, \; 0.04 \text{ m})^2$. (a) Initial positions. (b) At time step 300. (c) At time step 1224.



(a) (b)

Fig. 4: 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 [14].

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

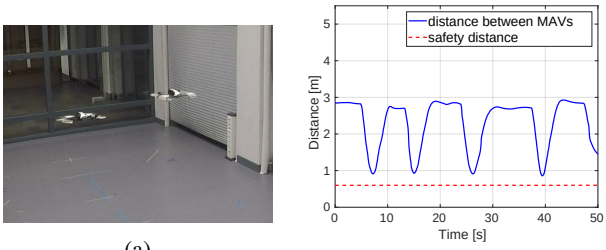
As shown in Fig. 4, we simulate with six quadrotors exchanging their initial positions in 3D space. Each quadrotor is under localization uncertainty $\Sigma = \text{diag}(0.04\text{ m},\ 0.04\text{ m},\ 0.04\text{ m})^2$. For each method, we run the simulation 10 times and calculate the minimum distance 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.

B. Experimental Results

Experimental setup: Our experimental platform is the Parrot Bebop 2 quadrotor [23]. An external motion capture system (OptiTrack) is used to measure the pose of each quadrotor, which is regarded as the real pose. We then add Gaussian noise to the data to simulate the localization uncertainties. The added measurements noise is zero mean with covariance $\Sigma = \text{diag}(0.04\text{ m},\ 0.04\text{ m},\ 0.04\text{ m})^2$. Taking the noisy measurements as inputs, an Unscented Kalman Filter (UKF) is employed to estimate the state of quadrotors, which results in an average estimated position uncertainty covariance $\Sigma^0 = \text{diag}(0.03\text{ m},\ 0.03\text{ m},\ 0.03\text{ m})^2$. A MPC controller is implemented with the same setup as Section V-A.2. The collision probability threshold is set as $\delta = 0.03$. Control commands are sent to the quadrotors using Robot Operating System (ROS).

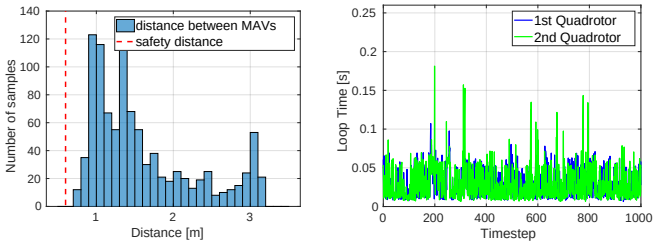
First experiment: Two quadrotors, initially located at (1.4, 0, 1.2) m and (–1.4, 0, 1.2) m, are required to swap their positions. We performed the swapping action multiple times. Fig. 5a shows a snapshot from our experiment. Fig. 5b shows the distance between the two quadrotors over time during the experiment. It can be seen that the distance was always larger than the required safety value 0.6 m.

Second experiment: One of the quadrotors follows a goal which moves with a period of 18 s along a lemniscate (a



(a) (b)

Fig. 5: Experimental results with two quadrotors swapping their positions. (a) A snapshot during the experiment. (b) Distance between the two quadrotors over time.



(a) (b)

Fig. 6: Experimental results with two quadrotors following crossing paths. (a) Histogram of distance between the two quadrotors. (b) Computation time of one control loop of each quadrotor during the experiment (mean 28 ms).

1 -shaped curve) with semi-axes (1.3, 0.6) m and a constant height 1.4 m, while the other quadrotor swaps its position between two specified goals (–1.8, 0, 1.2) m and (1.8, 0, 1.5) m every 4 s. The paths of the two quadrotors are crossing. Hence, they need to avoid collisions with each other when following the path. In Fig. 6a we cumulate the distance between the two quadrotors during the experiment. A minimum safe distance of 0.6 m was maintained over the entire run. In Fig. 6b we show the computation time of one control loop of each quadrotor. The mean computation time over the experiment is 28 ms, indicating the controller can be executed efficiently online.

VI. Conclusion

In this paper we presented a distributed multi-robot collision avoidance method that accounts for the robots' localization uncertainties. By assuming the uncertainties are according to Gaussian distributions, we compute a buffered uncertainty-aware Voronoi cell (B-UAVC) for each robot. The probability of collision between robots is guaranteed to be below a specified threshold by constraining each robot's motion to be within its corresponding B-UAVC. In simulation with six quadrotors, we showed that our method can achieve the same level of safety compared with the CCNMPC method, which is centralized and requires robots to communicate future trajectories. We also validated our method in experiments with two quadrotors following crossing paths. Future work shall take into account static and dynamic obstacles in the environments.

REFERENCES

[1] J. van den Berg, Ming Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *2008 IEEE International Conference on Robotics and Automation*, vol. 48, no. 1. IEEE, 2008, pp. 1928–1935.

[2] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998.

[3] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Springer Tracts in Advanced Robotics*, 2011, vol. 70, pp. 3–19.

[4] D. Bareiss and J. van den Berg, "Generalized reciprocal collision avoidance," *The International Journal of Robotics Research*, vol. 34, no. 12, pp. 1501–1514, 2015.

[5] J. Alonso-Mora, P. Beardsley, and R. Siegwart, "Cooperative collision avoidance for nonholonomic robots," *IEEE Transactions on Robotics*, vol. 34, no. 2, pp. 404–420, 2018.

[6] D. Shim, H. Kim, and S. Sastry, "Decentralized nonlinear model predictive control of multiple flying robots," in *IEEE Conference on Decision and Control*, vol. 4, 2003, pp. 3621–3626.

[7] Y. Chen, M. Cutler, and J. P. How, "Decoupled multiagent path planning via incremental sequential convex programming," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 5954–5961.

[8] D. Morgan, G. P. Subramanian, S. J. Chung, and F. Y. Hadaegh, "Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming," *International Journal of Robotics Research*, vol. 35, no. 10, pp. 1261–1285, 2016.

[9] T. Nägeli, L. Meier, A. Domahidi, J. Alonso-Mora, and O. Hilliges, "Real-time planning for automated multi-view drone cinematography," *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1–10, 2017.

[10] D. Claes, D. Hennes, K. Tuyls, and W. Meeussen, "Collision avoidance under bounded localization uncertainty," *IEEE International Conference on Intelligent Robots and Systems*, pp. 1192–1198, 2012.

[11] B. Gopalakrishnan, A. K. Singh, M. Kaushik, K. M. Krishna, and D. Manocha, "PRVO: Probabilistic reciprocal velocity obstacle for multi robot navigation under uncertainty," *IEEE International Conference on Intelligent Robots and Systems*, vol. 2017-Septe, pp. 1089–1096, 2017.

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

[13] D. Lyons, J. Calliess, and U. D. Hanebeck, "Chance constrained model predictive control for multi-agent systems with coupling constraints," in *2012 American Control Conference (ACC)*. IEEE, 2012, pp. 1223–1230.

[14] H. Zhu and J. Alonso-Mora, "Chance-constrained collision avoidance for mavs in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 776–783, 2019.

[15] D. Zhou, Z. Wang, S. Bandyopadhyay, and M. Schwager, "Fast, online collision avoidance for dynamic vehicles using buffered voronoi cells," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1047–1054, 2017.

[16] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial tessellations: concepts and applications of Voronoi diagrams*. John Wiley & Sons, 2009, vol. 501.

[17] T. W. Anderson and R. R. Bahadur, "Classification into two multivariate normal distributions with different covariance matrices," *The Annals of Mathematical Statistics*, vol. 33, no. 2, pp. 420–431, 1962.

[18] E. Nowakowska, J. Koronacki, and S. Lipovetsky, "Tractable measure of component overlap for gaussian mixture models," *arXiv preprint arXiv:1407.7172*, 2014.

[19] L. C. Andrews, *Special functions of mathematics for engineers*. SPIE, 1997.

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

[21] W. Honig, J. A. Preiss, T. K. Kumar, G. S. Sukhatme, and N. Ayanian, "Trajectory planning for quadrotor swarms," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 856–869, 2018.

[22] A. Domahidi and J. Jerez, "Forces professional. embotech gmbh (http://embotech. com/forces-pro)," 2014.

[23] "Parrot bebop 2," https://www.parrot.com/us/drones/parrot-bebop-2.