Collision Avoidance Algorithm Based on Buffered Vonoroi Cell

Weixing Guo
University of California, Berkeley
CA, US
guowx0704@berkeley.edu

Dongqi Zuo
University of California, Berkeley
CA, US
zuodi@berkeley.edu

***Abstract*—Our research is focused on the collision avoidance problem in the context of autonomous vehicle navigation. We employed Buffered Voronoi Cells for collision detection, addressing the collision challenge, and utilized a geometric algorithm based on greedy algorithm concepts for path planning. Additionally, we implemented an MPC method for optimization. Ultimately, we achieved smooth collision avoidance results, demonstrating good performance in intersection experiments. This confirms that we have developed a multi-agent collision avoidance algorithm with high success rates, low computational costs, and fast convergence.**

***Index Terms*—Buffered Voronoi cell, collision avoidance, MPC, multi-agent, autonomous vehicle navigation**

I. Motivation

Due to the unsatisfactory progress of our project during the final presentation, we decided to start from the very beginning and work on a new project from the ground up.

Our motivation for this project is to address collision avoidance for autonomous vehicles in scenarios where paths intersect in narrow areas, such as a crossing. For nowadays, collision avoidance is crucial for the success of autonomous vehicles (AVs) today. As AV technology advances, it is imperative to ensure that these vehicles can effectively detect and avoid obstacles and other vehicles, to reduce accidents. Statistics show that around 94% of traffic accidents are caused by human errors, highlighting the need for introducing autonomous systems that can prevent such mistakes.

Collision avoidance technologies in AVs are designed to address common crash types, including rear-end collisions, which account for nearly 30% of all crashes. These systems rely on sensors like radar and lidar, which can detect objects in real time, giving AVs a significant safety edge over human drivers who may not react in time. Ensuring that these systems are capable of quickly and accurately avoiding collisions is essential for reducing fatalities and improving road safety.[1]

In class, we learned about vehicle collision avoidance methods using Buffered Voronoi Cells(BVC). Therefore, we wanted to take this opportunity to apply this algorithm. By using BVC as collision avoidance algorithm, we can drive the robots to destinations without collisions, rather than to provide sensor coverage.

Compared to traditional methods such as Velocity Obstacles (VO), Artificial Potential Fields (APF) and Distributed Optimization[2], BVC offers a more scalable and less computationally intensive solution for multiple robots in simple environments. Also, The buffered Voronoi cell method ensures that agents maintain a safe distance from obstacles and other agents by calculating exclusive safe zones, which guarantee the collision-free requirement.

In this project, we successfully implemented the BVC method in specific intersection scenarios and resolved the deadlock and constraint ignorance issues encountered during the implementation process. We improve a path planing algorithm which Dingjiang[3] proposed using QP and geometric approaches in his paper. And we use MCP method in locally optimization progress. Finally, We utilized the $Shapely$ package in Python to implement BVC for collision avoidance functionality, and ultimately achieving visualization.

II. Problem Formulation

We consider a group of N robots operating in a 2D plane. Let the position of each robot $i \in \{1, 2, \ldots, N\}$ be $p_i \in \mathbb{R}^2$. The robots navigate in discrete time steps $t = 0, 1, \ldots$, and evolve under the single-integrator dynamics:

$$p_{i,t+1} = p_{i,t} + u_{i,t}\Delta t, \tag{1}$$

where $u_{i,t} \in \mathbb{R}^2$ is the control input (velocity command) for robot i at time t, and $\Delta t > 0$ is the time step.

A. Environment and Constraints

The robots operate in a corridor-like intersection area represented by a convex polygon $P \subseteq \mathbb{R}^2$. Thus, for

each time step t and each robot i, we require:

$$p_{i,t} \in P. \tag{2}$$

Each robot i starts from an initial position $p_i^{\text{init}} \in P$ and aims to reach its designated final goal position $p_i^{\text{goal}} \in P$. To facilitate turning or reorientation at the intersection, we introduce a two-phase navigation strategy. We define a center region:

$$\mathcal{C} = \{p \in \mathbb{R}^2 \mid \|p - p^{\text{center}}\| \leq r_{\text{center}}\}, \tag{3}$$

where $p^{\text{center}} \in P$ and $r_{\text{center}} > 0$ define the center area of the intersection.

The robot i must first guide itself to enter $\mathcal{C}$ (Phase 1). Once $p_{i,t}$ satisfies $\|p_{i,t} - p^{\text{center}}\| \leq r_{\text{center}}$, it switches to Phase 2, where it attempts to reach p_i^{goal}. Finally, the robot must achieve $\|p_{i,t} - p_i^{\text{goal}}\| \leq \epsilon$ for some tolerance $\epsilon > 0$ to consider the task complete.

B. Collision-Free Configuration and Buffered Voronoi Cells (BVC)

We assume each robot has a physical size represented by a safety radius $r_s > 0$. A configuration is called *collision-free* if for all $i \neq j$,

$$\|p_i - p_j\| \geq 2r_s. \tag{4}$$

To ensure collision avoidance, we exploit a Voronoi-based approach. Suppose we have N robots with positions $\{p_1, \ldots, p_N\}$. The general Voronoi cell of robot i with respect to others is defined by:

$$V_i = \{p \in \mathbb{R}^2 \mid \|p - p_i\| \leq \|p - p_j\|,\ \forall j \neq i\}. \tag{5}$$

To account for the robot size, we define the *Buffered Voronoi Cell (BVC)*. For a collision-free configuration, the BVC of robot i is:

$$V_i^{\text{B}} = P \cap \bigcap_{j \neq i} \left\{ p \in \mathbb{R}^2 \mid \left(p - \frac{p_i + p_j}{2}\right)^T (p_j - p_i) + r_s\|p_j - p_i\| \leq 0 \right\}. \tag{6}$$

By staying within its BVC, i.e. $p_{i,t} \in V_i^{\text{B}}$, robot i ensures no collisions with others, as shown in related lemmas (see, e.g., Lemma 1 in reference works).

C. Control Strategy and Local Optimization

For Path Planning algorithm, We use Analytical Geometric Algorithm[2]. Due to the difficulty to implement a QP solver for path planning, we use this algorithm which ignore the cost of intermediate state and control input. By this method, we not only ensure the efficiency but also conform the collision avoidance.

In this algorithm, Our goal is to find the closest point towards destination, which is similar to greedy algorithm's concept. So, we assume this point is selected among vertices g_1 and g_2, point g_p which is on the edge of BVC and itself g. Therefore, we define $V = (\epsilon, e)$ as a convex polygon in R^2, where ϵ is the set of edges and e is the set of vertices, the closest point $g^* \in V$ and also the minimum distance d_{min}. So we do series of judgments according to parameter λ_i for edge ϵ_i shown in (7),

$$\lambda_i = -\frac{(g_1 - g)^T(g_2 - g)}{||g_1 - g_2||^2} \tag{7}$$

If λ_i is less than 1 which means g's projection is between the vertices, then g^* would be g_p which is shown in (8). Else if λ_i is larger than 1 then g^* would be g_1 or g_2 according to the distance. As a result, we find out the closest point by going through all the boundaries. Finally, the agent will move to this point at that time step.

$$g_p = (1 - \lambda_i)g_1 + \lambda_i g_2 \tag{8}$$

For local optimization, We define a finite-horizon cost function over a horizon T:

$$J = \sum_{t=0}^{T-1} \left[(p_t - p_{\text{target}})^T Q (p_t - p_{\text{target}}) + u_t^T R u_t\right] + (p_T - p_{\text{target}})^T Q_f (p_T - p_{\text{target}}), \tag{9}$$

subject to the dynamics (1) and feasibility constraints $p_t \in V_i^{\text{B}} \cap P$. Here, Q, Q_f, R are positive definite (or semi-definite) weighting matrices, and p_{target} is the current waypoint (either p^{center} or p_i^{goal}).

This algorithm iteratively linearizes the dynamics and quadratizes the cost, performing a backward pass to compute feedback gains K_t and feedforward increments k_t, and a forward pass with a line search on step size α to ensure improvement. The result is a locally optimal sequence $\{u_t\}$ that helps the robot advance while maintaining safety and staying within V_i^{B} and P. The regularization by R and careful weighting in Q, Q_f encourages smooth, low-jerk motions.

D. Termination and Outputs

The process is repeated until all robots reach their final goals. Initially, robot i moves towards $\mathcal{C}$ and, upon

entering this region ($\|p_{i,t} - p^{\text{center}}\| \leq r_{\text{center}}$), it switches its objective to p_i^{goal}. The algorithm terminates when:

$$\|p_{i,t} - p_i^{\text{goal}}\| \leq \epsilon \quad \forall i. \tag{10}$$

As output, we obtain a set of collision-free, two-phase trajectories $\{p_{i,t}\}$ that respect the corridor constraints, maintain non-collision through BVC constraints, achieve intermediate center-region passage, and finally converge to each robot's individual goal position.

III. Challenges

The main Challenges we anticipated are whether the algorithm can scale to an arbitrary number of agents. Moreover, the computation of BVC maybe increase heavily as the number of agents grows. Also, the number of agents growth may raises the possibility of encountering non-convex situations that could lead to computation stalls. And it finally turns out that our algorithm can cope with those challenges.

The issues we actually faced during implementation was the deadlock problem and "Reward Hacking" likewise problem. The first issue is deadlock problem, where agents become locally stable when they both move to the BVC boundary. We primarily adopted a right-selection principle based on orientation, but the smoothness still needs improvement. The second one is that our agent ignored the barrier constraints and go straight to the destination. We dealt with the challenge by forcing agents go to a reward zone located near the intersection shown in (3).

IV. Finding

A. Numerical Results

We conducted numerical simulations with $N = 3$ robots navigating through a corridor-like intersection region. Initially, each robot is positioned in different arms of the intersection, and its final goal lies in another arm. To reach the final goal, each robot must first move into a designated center region $\mathcal{C}$ before proceeding to its own final goal.

Figure 2 illustrates the initial positions of the 3 robots (colored dots) and their corresponding Buffered Voronoi Cells (BVCs), shown as colored polygons. The BVC constraints ensure that each robot remains collision-free while advancing through the corridor. In the initial phase, all robots head towards the center region $\mathcal{C}$. As time progresses (Figure 3), the robots successfully converge towards the center, each following a feasible path within its BVC. After entering the center region, each robot changes its target to its assigned final goal.

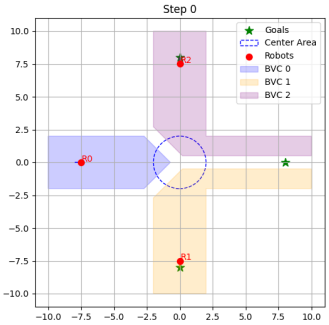

Fig. 1: Initial configuration of 3 robots and their BVCs in the corridor intersection.

By the end of the simulation (Figure 4), all robots have passed through the center and reached their respective final goals. The resulting trajectories are collision-free and reasonably smooth, demonstrating the effectiveness of the two-phase objective combined with the BVC-based collision avoidance strategy.

In our simulation, the system time step is Δt=0.2s second, and the planning horizon is $\mathcal{T}$ = 20. The safety radius is r_s=0.2 meter and the minimum distance among the robots during the simulation is dmin = 0.4 meter.

Our approach is also scalable to a larger number of robots. To demonstrate this, we conducted an additional simulation with $N = 100$ robots. Similar to the smaller-scale experiments, the robots were initially placed evenly on a circle of radius 20 m, with slight random perturbations to their positions to break symmetry. All robots aimed to navigate across the circle to the diametrically opposite points, requiring them to pass through the designated center region.

As in the smaller cases, we computed the BVC for each robot at every time step, ensuring that each robot maintained collision-free motion. (Figure 5) shows the BVCs for all 100 robots, each plotted in a distinctive color corresponding to that robot. The snapshots at intermediate steps, such as $t = 160$ and $t = 480$, illustrate the dynamic progression: the robots gradually converged toward the center region and then proceeded to their respective opposite destinations. To avoid visual clutter, only the trajectories of a subset of robots are shown in the later figures.

Even with 100 robots, the algorithm successfully arranged collision-free paths for all agents within the given horizon. This demonstrates the potential scalability of our method, as it can handle a significantly larger population of robots while preserving the feasibility and

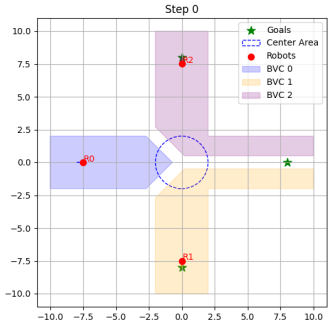

Fig. 2: Initial configuration of 3 robots and their BVCs in the corridor intersection.

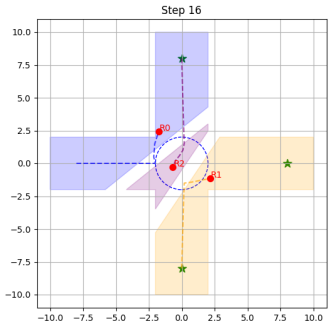

Fig. 3: Robots converging towards the center region. Each robot stays within its BVC.

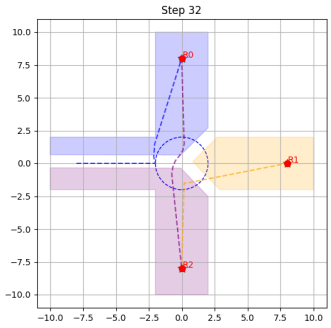

Fig. 4: All robots have passed through the center and reached their final goals.

smoothness of the resulting trajectories.

B. Computational Complexity

The proposed approach involves multiple components: BVC computation at each time step and a repeated MPC-like loop. We analyze the complexity focusing on these key operations.

At each time step, for each robot $i \in \{1, \ldots, N\}$, we compute its Buffered Voronoi Cell (BVC). This involves intersecting a given bounding polygon P with up to $(N-1)$ half-plane constraints derived from the positions of the other robots.

Assuming that polygonal intersection operations scale linearly with the number of constraints, each BVC computation takes approximately $O(N)$ time per robot. Thus, for all N robots each time step, the complexity is:

$$O(N \cdot N) = O(N^2).$$

The outer loop runs for up to max_steps iterations (or until convergence). Each iteration requires updating all robots' BVCs, which is $O(N^2)$, plus executing simple geometric checks for movement feasibility. These checks are relatively inexpensive compared to BVC computations.

Therefore, without any additional optimization overhead, over max_steps steps, the total complexity is on the order of:

$$O(\text{max steps} \cdot N^2).$$

- The $O(N^2)$ scaling arises naturally due to the pairwise nature of constraints when forming BVCs.

- If N is large, this $O(N^2)$ term may dominate the overall runtime.

- Depending on the environment shape and the complexity of polygon intersection routines, practical runtimes may vary.

In summary, the complexity mainly comes from repeated BVC computations, yielding $O(\text{max\_steps} \cdot N^2)$ scaling for large N. The exact performance depends on implementation details and problem parameters, but this order-of-magnitude analysis provides guidance on expected runtime behavior.

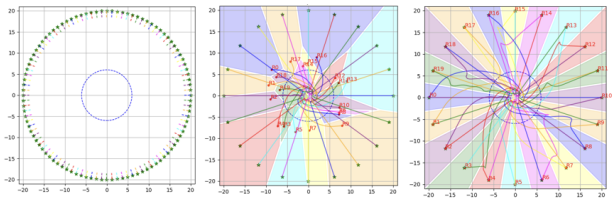

Fig. 5: Simulation with 100 robots with single integrator dynamics move to their goal configuration using the geometric algorithm. (a) Initial Configuration (b) At step 60. (c) At step 110.

V. Conclusion

To sum up, in this research we achieve following topics

- We developed an algorithm that successfully drive agents converging to the destination while protecting them away from collision.
- Our algorithm lowered the complexity while ensuring the success rate.

Our next step is to fix following issues

- Improve the moving progress and make trajectory smoother by optimize the deadlock algorithm and geometric algorithm.
- Utilize our algorithm in more complex environment such as more crowded situation, more flexible setting up and more complex surroundings.

References

[1] Mehak Raibail et al. "Decentralized Multi-Robot Collision Avoidance: A Systematic Review from 2015 to 2021". In: Mar. 2022, p. 610. DOI: 10.3390/sym14030610.

[2] Mingyu Wang and Mac Schwager. "Distributed Collision Avoidance of Multiple Robots with Probabilistic Buffered Voronoi Cells". In: Aug. 2019, pp. 169–175. DOI: 10.1109/MRS.2019.8901101.

[3] Dingjiang Zhou et al. "Fast, On-line Collision Avoidance for Dynamic Vehicles Using Buffered Voronoi Cells". In: Apr. 2017, pp. 1047–1054. DOI: 10.1109/LRA.2017.2656241.

AUTHOR CONTRIBUTIONS

Weixing Guo: Weixing Guo led the conceptual design and theoretical formulation of the buffered Voronoi cell (BVC) approach. He established the mathematical framework for the two-phase navigation strategy, incorporating the center region objective. He also performed the primary complexity analysis and structured the problem in a manner that facilitated a clear comparison with ORCA. During the writing process, Weixing Guo ensured the academic rigor of the paper, refined the LaTeX presentation, and integrated the formal proofs and key theoretical insights.

Dongqi Zuo: Dongqi Zuo focused on the implementation details and numerical simulations. He developed the code to compute BVCs, performed the experiments in the corridor-like intersection scenario, and generated the figures and GIF animations. He worked on optimizing the polygon intersection routines, tested different configurations to evaluate scalability, and contributed to simplifying the complexity discussion in practical terms. Dongqi Zuo also assisted in editing the manuscript, ensuring that the results and methodologies were communicated effectively and concisely.