IEEE Robotics and Automation Letters (RAL) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

MCCA: A Decentralized Method for Collision and Deadlock Avoidance With Nonholonomic Robots

Ruochen Zheng and Siyu Li

***Abstract*—Navigation in dense and narrow environments with multiple robots is a standing challenge since deadlock is prone to occur. In this letter we present masked cooperative collision avoidance (MCCA), a fully decentralized method to avoid both collision and deadlock effectively. The concept of masked velocity is introduced, which is an implicit state of each robot and acts as an intention of avoiding deadlock. Robots are prioritized by a decentralized mechanism and masked velocities of robots with different priorities propagate among robots, promoting fluent and efficient deadlock avoiding behaviors in a local and collective manner. The solving process is reduced to a quadratic programming problem. Nonholonomic constraints are taken into account. We conduct extensive experiments in both simulation and real-world application, and the results verify the effectiveness of our method.**

***Index Terms*—Collision avoidance, multi-robot systems, nonholonomic motion planning.**

I. Introduction

MULTI-ROBOT systems have garnered significant success in industrial applications such as smart warehousing and automated transportation. As a key element, multi-agent navigation algorithms enable robots to navigate to their targets without colliding with obstacles or other robots. Enhancing the overall navigational performance of robots remains a challenging task, especially in dense environments where deadlock is prone to occur.

Among various multi-agent navigation algorithms, optimal reciprocal collision avoidance (ORCA) [1] is a popular and well-established method. By extending the ideas of reciprocal velocity obstacles (RVO) [2], it introduces half-plane constraints induced by all neighbors to form a convex polygon, inside which a velocity nearest to the preferred velocity is chosen by an efficient linear program. Each robot computes its new velocity independently based on observed positions and velocities of other robots, which is decentralized and therefore scalable to a large number of robots.

Manuscript received 18 August 2023; accepted 16 January 2024. Date of publication 25 January 2024; date of current version 8 February 2024. This letter was recommended for publication by Associate Editor G. Notomista and Editor M. A. Hsieh upon evaluation of the reviewers' comments. *(Corresponding author: Siyu Li.)*

The authors are with the Department of Modeling, Optimization and Simulation, Megvii Automation & Robotics, Beijing 100096, China (e-mail: zhengruochen02@megvii.com; lisiyu02@megvii.com).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3358623, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3358623

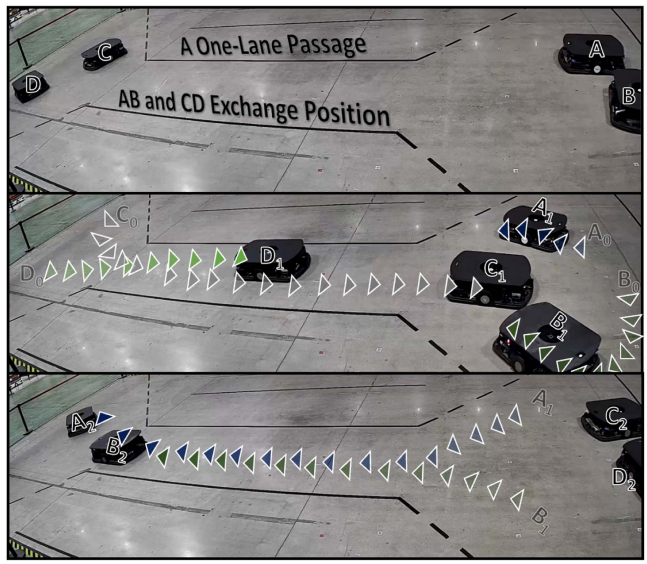


Fig. 1. Real-World Experiments.[1] Two groups of robots exchange position via a one-lane passage. The robots on one side are able to give way to robots from the other side, thus avoiding deadlock.

However, ORCA assumes that each robot is equal and shares an equivalent responsibility of collision avoidance, which promotes deadlock in dense environments. Deadlock represents an equilibrium of system dynamics that causes the robots to stall before reaching their goals. In this case, a control favoring the goal will violate safety, hence the only feasible strategy is to remain static [3].

To improve the navigation performance, some previous ORCA variants, such as [4] and [5], use ORCA to avoid collision and leverage a grid-based multi-agent path planing (MAPF) framework to coordinate robots when deadlock is detected. However, the MAPF approach is centralized and suffers from heavy computational cost. Another ORCA variant [6] allows robots to take different responsibilities of collision avoidance, as long as they sum up to one. However, this approach is ineffective for deadlock avoidance in dense environments.

To address the deadlock issue while still maintaining decentralization, we propose a masked cooperative collision avoidance (MCCA) method. It adopts the ideas of collision avoidance from ORCA and is extended to explicitly handle deadlock situations. An illustration of four robots exchanging position via a one-lane passage is shown in Fig 1. The concept of masked velocity is introduced, which is an implicit state of each individual robot. It represents an intention to either reach the goal regardless of other

robots or make way for others based on its priority determined by a decentralized approach. The problem is reduced to a quadratic program (QP), where the intention is seamlessly integrated into the velocity planning of each individual robot.

The main contributions of this letter are as follows:

- The concept of masked velocity is introduced, which is an implicit state of each robot and helps promote coordination among robots to mitigate deadlock. A special MCCA half-plane is designed to constrain the solution of masked velocity and the new velocity.
- A decentralized approach to determine the priority for each robot is presented. It allows each robot to independently decide its priority based on conflict of masked velocity with its neighbors, and a consistent strategy is enforced to prevent undesirable frequent switches of priorities.
- A new QP formulation is presented, where the objective terms, related to approaching the goal, avoiding collision and mitigating deadlock, can be prioritized, allowing robots to avoid deadlock before reaching their goals on condition that safety is maintained. It can also incorporate nonholonomic constraints such that the control inputs can be directly solved. Collision-freeness of the solving paradigm is guaranteed.

The rest of this letter is organized as follows. In Section II a brief summary of related works is provided. In Section III, the methodology of our approach is presented. An example of differential-drive robots is provided in Section IV. We report simulation and real-world experimental results in Section V and then conclude in VI.

II. Related Work

In this section, we briefly summarize the related works of multi-agent navigation algorithms.

The multi-agent navigation algorithms can be categorized into centralized methods and decentralized methods. The centralized methods, such as [7], [8], [9], utilize a central controller for the monitoring and control of all the agents in the system. Some of them, such as [10], couples the path planning with the task assignment to yield a more efficient solution. These centralized approaches assume that all robots' navigation decisions are made jointly and can generally guarantee a safe, optimal and complete solution for navigation. Deadlock can be resolved at a global level. However, they become computationally prohibitive as the number of robots increases and thus inapplicable for deployment in physical systems. Besides, such methods mostly require that the planned paths are along the grids, which makes the motion behaviors overly conservative.

On the contrary, decentralized methods have drawn significant attention recently due to their scalability, where each agent makes the decision independently by taking into account the information of others from sensors and communications. There have been extensive studies, using techniques such as potential fields [11], graph-based searches [12], [13], velocity obstacles (VO) [14], control barrier functions [15] and reinforcement learning [16]. Among them, VO is a popular framework and most related to our approach. It defines regions in the velocity space where collisions are avoided given the robot velocities and positions. Reciprocal collision avoidance (RVO) is introduced in [2] to let each pair of robots share the responsibility for collision avoidance to eliminate oscillations. The ORCA algorithm [1] introduces a pair of half-plane constraints in the velocity space for each pair of nearby agents. It transforms the problem into a linear programming problem that can be solved efficiently and provides a safety guarantee under certain mild assumptions. Some ORCA variants, such as [17] and [18], incorporate dynamics of nonholonomic robots to produce smooth trajectories for different kinematic models.

However, these decentralized methods lack effective coordination among robots to avoid deadlock in dense environments. The VR-ORCA algorithm [6] extends the ORCA framework to allow each pair of robots to take different responsibilities of collision avoidance. It can improve the performance in open spaces, but in dense situations the new velocity can still be zero and robots remain stalled. The V-RVO algorithm [19] utilizes a position exchange strategy across two robots' common voronoi edge when deadlock is identified, which is only suitable for open spaces. The approach in [15] enforces a constant perturbation to encourage a clockwise motion for all robots. However, in dense environments admissible perturbation may not exist and deadlock may persist. The methods in [4], [5] use ORCA for collision avoidance and leverage centralized MAPF algorithms to resolve deadlock when detected. Again, they still suffer from heavy computational cost. [13] presents a decentralized MAPF algorithm on search graphs which provides optimality guarantees, but is not suitable for narrow passages.

III. Approach

In this section we present our novel MCCA method. Masked velocity, an implicit state of each individual robot, is introduced to represent an intention to avoid deadlock. A special MCCA half-plane is designed to constrain the solution of the masked velocity and real velocity. Each robot determines its priority by a decentralized approach. Robots with lower priority will try to make way for robots with higher priority based on their masked velocities. The solution is computed by a new QP formulation, where half-planes and nonholonomic constraints can be incorporated.

A. Head and Normal Robots

Assume the set of all robots $\mathcal{R}$ can be divided into two groups, $\overline{\mathcal{R}}$ denotes the set of head robots and $\underline{\mathcal{R}}$ the set of normal robots. Normal robots are assumed to yield and make way for head robots. In dense environments, such as a one-lane passage, if two robots encounter with opposite moving directions, none of them can successfully reach the goal unless one of them compromises and diverts from its own path. Therefore, it is necessary that one of them is granted higher precedence for passage.

B. Masked Velocity and MCCA Half-Plane

We define masked velocity as an implicit state of each robot. For normal robots it serves as an intention to yield and make

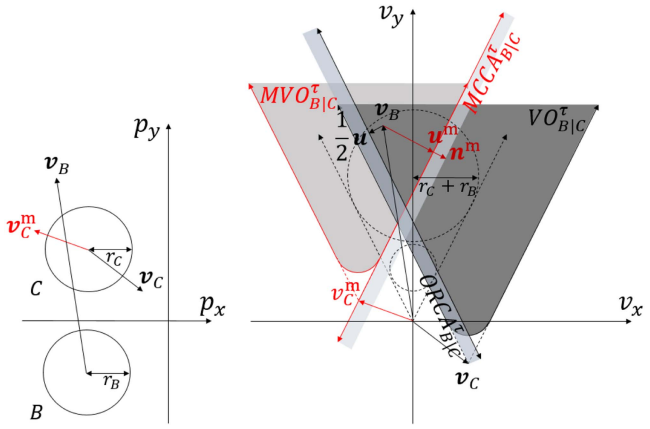


Fig. 2. Demonstration of MVO and MCCA half-plane.

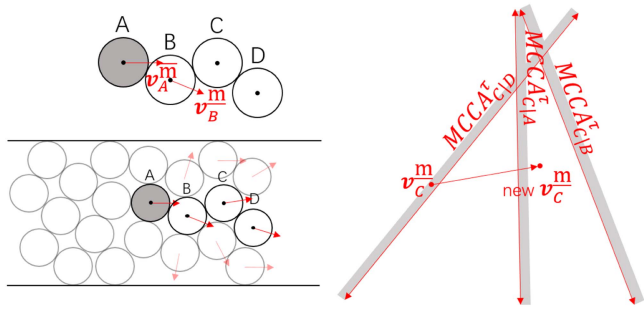


Fig. 3. Propagation of masked velocity in a deadlock situation.

way for head robots and other normal robots. While for head robots, it is an intention to reach their goals regardless of all other robots. Let $\boldsymbol{v}_A^{\overline{\mathrm{m}}}$ denote the masked velocity of a head robot $A \in \overline{\mathcal{R}}$, and $\boldsymbol{v}_B^{\underline{\mathrm{m}}}$ the masked velocity of a normal robot $B \in \underline{\mathcal{R}}$. Masked velocities of all robots, both head and normal, are initially set to zero.

$\boldsymbol{v}_A^{\overline{\mathrm{m}}}$ is estimated as the velocity closest to its preferred velocity respecting only ORCA half-planes of obstacles:

$$\boldsymbol{v}_A^{\overline{\mathrm{m}}} = \underset{\boldsymbol{v}^{\mathrm{m}} \in ORCA^{\tau}_{A|\mathcal{O}}}{\operatorname{argmin}} \left\| \boldsymbol{v}^{\mathrm{m}} - \boldsymbol{v}_A^{\mathrm{pref}} \right\|, \tag{1}$$

where $ORCA^{\tau}_{A|\mathcal{O}} = \bigcap_{o \in \mathcal{O}} ORCA^{\tau}_{A|o}$, o and $\mathcal{O}$ represent an obstacle and the set of all neighboring obstacles, respectively.

We now define MCCA half-plane, which is specially designed for normal robots. It is generated based on the masked velocity obstacle (MVO) constructed by the current normal robot B's velocity $\boldsymbol{v}_B$ and another robot C's masked velocity $\boldsymbol{v}_C^{\mathrm{m}}$, as shown in Fig. 2. $MVO^{\tau}_{B|C}$, is constructed with $(\boldsymbol{v}_C^{\mathrm{m}}, \boldsymbol{v}_B)$. $\boldsymbol{u}^{\mathrm{m}}$ is the vector from $\boldsymbol{v}_B - \boldsymbol{v}_C^{\mathrm{m}}$ to the closest point on the boundary of $MVO^{\tau}_{B|C}$, and $\boldsymbol{n}^{\mathrm{m}}$ is the outward normal of this boundary at point $(\boldsymbol{v}_B - \boldsymbol{v}_C^{\mathrm{m}}) + \boldsymbol{u}^{\mathrm{m}}$. The MCCA half-plane is formulated as follows:

$$MCCA^{\tau}_{B|C} = \{\boldsymbol{v}^{\underline{\mathrm{m}}} | (\boldsymbol{v}^{\underline{\mathrm{m}}} - (\boldsymbol{v}_B + \boldsymbol{u}^{\mathrm{m}})) \cdot \boldsymbol{n}^{\mathrm{m}} \geq 0\}, C \in \mathcal{R}. \tag{2}$$

The masked velocity of a normal robot B is estimated as the velocity closest to its preferred velocity respecting both ORCA half-planes of obstacles and MCCA half-planes induced by other robots:

$$\boldsymbol{v}_B^{\underline{\mathrm{m}}} = \underset{\boldsymbol{v}^{\mathrm{m}} \in MCCA^{\tau}_{B}}{\operatorname{argmin}} \left\| \boldsymbol{v}^{\mathrm{m}} - \boldsymbol{v}_B^{\mathrm{pref}} \right\|, \tag{3}$$

where $MCCA^{\tau}_{B} = ORCA^{\tau}_{B|\mathcal{O}} \cap \bigcap_{X \in \mathcal{R}} MCCA^{\tau}_{B|X}$.

MCCA half-plane is used to reduce the level of conflict against C's masked velocity. It can be described that C's masked velocity is propagated to the current robot B. C can either be a head or normal robot. If C is normal, even if B has no conflict of masked velocity with any head robot, its newly solved masked velocity

[1][Online]. Available: https://www.youtube.com/watch?v=_DcZ9MHws7A

Algorithm 1: Robot Priority Determination for A_i.

Input $\mathcal{R}$: All robots A_i: Current robot
Output Updated priority for A_i
Initialization $S_{A_i} = 0$, $T_{A_i} = 0$, $A_i \in \underline{\mathcal{R}}$
if *(A_i reaches its goal)* **then**
  Set A_i as a normal robot, $S_{A_i} = 0$, $T_{A_i} = 0$;
else
  if *($T_{A_i} > 0$)* **then**
    Set A_i as a normal robot, $T_{A_i} = T_{A_i} - 1$;
  else
    if *($\exists\ A_j \in \overline{\mathcal{R}}$ s.t. $\boldsymbol{v}_{A_i}^{\overline{\mathrm{m}}} \in MVO^{\infty}_{A_i|A_j}$ and $\boldsymbol{v}_{A_i}^{\overline{\mathrm{m}}} \cdot \boldsymbol{v}_{A_j}^{\overline{\mathrm{m}}} < 0$ and $S_{A_i} < S_{A_j}$)* **then**
      Set A_i as a normal robot, $T_{A_i} = \eta$;
    else
      Set A_i as a head robot, $S_{A_i} = S_{A_i} + 1$
    end
  end
end

$\boldsymbol{v}_B^{\underline{\mathrm{m}}}$ will still avoid conflict with C, and other neighboring normal robots' masked velocities will also be effected by $\boldsymbol{v}_B^{\underline{\mathrm{m}}}$. It can be viewed that an intention to make way for certain head robots is propagated along normal robots. We will later show that the solution of control inputs is influenced by the MCCA half-plane so that the intention is indeed carried out.

An example of the propagation of masked velocity is shown in Fig. 3. A is a head robot. The masked velocity of A has propagated to B, and C is being propagated at this time instance. When D is propagated in the near future, $MCCA^{\tau}_{C|D}$ will move rightward, which will further enlarge the masked velocity of C. In such process, the deadlock avoiding intention keeps propagating from a head robot to normal robots from the near to the distant.

$MCCA^{\tau}_{B|C}$ is built with B adapting $\boldsymbol{u}^{\mathrm{m}}$ alone, as $\boldsymbol{u}^{\mathrm{m}}$ is not divided by 2 like the ORCA half-plane, suggesting that B takes full responsibility of avoiding the masked velocity of C. It is obvious if C is a head robot since a head robot does not have to consider avoiding other masked velocities. And if C is a normal robot, the masked velocity of C is losslessly propagated to B, which ensures that the propagation of masked velocity along normal robots will not shrink at a rate below 1 and diminish too soon.

C. Robot Priority Determination

We adopt a fully decentralized approach for each robot to determine its priority. Since a head robot's masked velocity is not effected by other robots, the conflict of masked velocity between two head robots cannot be resolved and may result in deadlock, therefore it should be resolved by priority determination in the first place. Algorithm 1 presents the algorithm. S_{A_i} is the number of time steps where A_i is of higher priority by far, which is an accumulative value to indicate the importance of A_i. η s a preset integer as the initial tabu steps for a newly determined normal robot. T_{A_i} denotes the current tabu steps of A_i, which will be subtracted by one at each following time step. The robot's chance to raise priority is postponed until T_{A_i} is reduced to zero. Both T_{A_i} and S_{A_i} are reset to zero once the robot reaches its goal. The introduction of the two indicators helps prevent frequent switches between high and normal priority and maintain the consistency of deadlock resolution.

Should a normal robot $A_i \in \underline{\mathcal{R}}$ become a head robot, its masked velocity solved as a head robot, denoted by $\boldsymbol{v}^{\overline{\mathrm{m}}}_{A_i}$, should not be in conflict with other head robots. A MVO cone of infinite time window $MVO^{\infty}_{A_i|A_j}$ is used to detect the conflict of masked velocity with any $A_j \in \overline{\mathcal{R}}$. Further, if $\boldsymbol{v}^{\overline{\mathrm{m}}}_{A_i}$ is in $MVO^{\infty}_{A_i|A_j}$, which means a conflict exists, if the projection of $\boldsymbol{v}^{\overline{\mathrm{m}}}_{A_i}$ onto $\boldsymbol{v}^{\overline{\mathrm{m}}}_{A_j}$ is in the same direction with $\boldsymbol{v}^{\overline{\mathrm{m}}}_{A_j}$, or $\boldsymbol{v}^{\overline{\mathrm{m}}}_{A_i} \cdot \boldsymbol{v}^{\overline{\mathrm{m}}}_{A_j} \geq 0$ equivalently, deadlock is unlikely because the robot behind can follow the one in front, thus both can remain head robots while also capable of advancing towards their goals. On condition that $\boldsymbol{v}^{\overline{\mathrm{m}}}_{A_i}$ is in $MVO^{\infty}_{A_i|A_j}$ and $\boldsymbol{v}^{\overline{\mathrm{m}}}_{A_i} \cdot \boldsymbol{v}^{\overline{\mathrm{m}}}_{A_j} < 0$, A_i is set as a head robot only when $S_{A_i} > S_{A_j}$, which means A_i is of higher priority for more steps than A_j by far and is more qualified to be a head robot.

D. Quadratic Programming Formulation

Next, we present a new solving paradigm formulated as a quadratic programming problem. Compared to the linear programming in ORCA, we would like a formulation to have the following features: (1) The distance between the new velocity and the preferred velocity can be explicitly minimized. (2) If a solution strictly respecting all half-plane constraints does not exist, the violation can be minimized. (3) Constraints can be weighted differently. (4) Control inputs and kinematic constraints can be incorporated directly.

A corresponding formulation is as follows:

$$\min \quad \alpha_1 \left\| \boldsymbol{v} - \boldsymbol{v}^{\mathrm{pref}} \right\|_2^2 + \alpha_2 \sum_{i \in \mathcal{O}} \delta_i^2 + \alpha_3 \sum_{j \in \mathcal{R}} \delta_j^2 + \alpha_4 \sum_{k \in \mathcal{M}} \delta_k^2 \tag{4a}$$

$$\text{s.t.} \quad \det\left(\boldsymbol{B}_{\boldsymbol{\zeta}}\right) \leq \delta_\zeta, \zeta \in \mathcal{O} \cup \mathcal{R} \cup \mathcal{M}, \delta_\zeta \geq 0 \tag{4a}$$

$$\dot{\boldsymbol{q}} = \boldsymbol{S}(\boldsymbol{q}) \cdot \boldsymbol{r} \tag{4b}$$

The first objective term is the square of the distance between the new velocity and the preferred velocity. Each of the rest is a sum of squares of an auxiliary variable δ_ζ that represents the degree of violation of the corresponding half-plane constraint by the new velocity. For example, $\delta_\zeta, \zeta \in \mathcal{O}$ represents the violation against an ORCA half-plane induced by an obstacle. Objective terms are prioritized by preset weights α_1 to α_4. Constraint (4a) expresses the violation mathematically. As shown in Fig. 4, a half-plane is defined by its position (p_x^ζ, p_y^ζ) and direction (d_x^ζ, d_y^ζ). The new velocity (v_x, v_y) lies within the half-plane iff the determinant of $\boldsymbol{B}_\zeta = \begin{bmatrix} v_x - p_x^\zeta & d_x^\zeta \\ v_y - p_y^\zeta & d_y^\zeta \end{bmatrix}$ is non-positive. ORCA half-plane constraints induced by obstacles $\mathcal{O}$ and other robots $\mathcal{R}$, along with MCCA half-planes $\mathcal{M}$, are all considered. The violation against them is punished in the objective with different weights. Note that $\mathcal{M}$ is empty for a head robot. Constraint (4b) incorporates nonholonomic constraints, which describes the relationship between the velocity and the control inputs. Consider a robot with motion states in the generalized coordinates $\boldsymbol{q} = [q_1, \ldots, q_n]^T$. $\dot{\boldsymbol{q}} = [\dot{q_1}, \ldots, \dot{q_n}]^T$ is the velocity vector. $\boldsymbol{r} = [r_1, \ldots, r_m]^T$ represents the m independent control motors and $\boldsymbol{S}(\boldsymbol{q}) = [\boldsymbol{s}_1(\boldsymbol{q}), \ldots, \boldsymbol{s}_m(\boldsymbol{q})]$ comprises m reachable motion vectors. The linear form is suitable for most wheeled robots such as differential-drive, tricycle, Ackermann-drive robots, and a robot with a trailer [20]. The proposed formulation is a convex QP problem that can be efficiently solved by a QP solver.

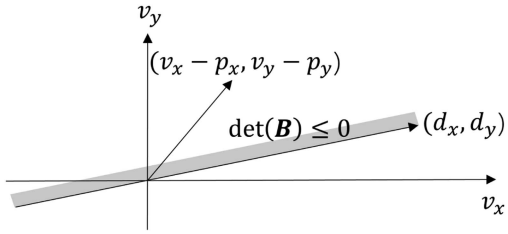


Fig. 4. Visualization of a half-plane and its permitted velocities.

Masked velocities of both head and normal robots are solved by a holonomic version of (4), where only the half-planes in their definitions (1) and (3) are considered and constraint (4b) is omitted. Masked velocity is not actual and does not need to be constrained by kinematics. In this situation, the solved masked velocity is usually larger, which strengthens the intention to avoid deadlock.

For each robot, it is required that $\alpha_2, \alpha_3 \gg \alpha_4 \gg \alpha_1$ so that the most prioritized target is to remain clear of obstacles and other robots, then it is necessary to avoid the masked velocities of other robots to carry out the intention to avoid deadlock, and finally it could catch up with its preferred velocity. We provide a proof which shows α_1 to α_4 can be properly set so that collision-freeness is guaranteed if the bounding circle radius of each robot increases only marginally.

Theorem 1: If there exists a velocity satisfying all ORCA half-planes, then the new velocity solved in (4) is collision-free if the bounding circle radius R is increased by at least $2\tau v^{\max} \sqrt{\frac{2\alpha_1 + \alpha_4 n}{\min(\alpha_2, \alpha_3)}}$, where τ is the time horizon, $v^{\max}$ is the maximum linear velocity and n is the number of other robots.

Proof: Given that there exists a velocity satisfying all ORCA half-planes, a solution to (4), $x' = (v_x', v_y', \delta_{i_1}', \cdots, \delta_{i_o}', \delta_{i_{o+1}}', \cdots, \delta_{i_{o+n}}', \delta_{k_1}', \cdots, \delta_{k_n}')$ exists, where $i_1, \cdots, i_{o+n} \in \mathcal{O} \cup \mathcal{R}, k_1, \cdots, k_n \in \mathcal{M}$, such that $\delta_{i_1}', \cdots, \delta_{i_{o+n}}' = 0$. Let $f(x)$ denote the objective function in (4) and $x^* =$

$(v_x^*, v_y^*, \delta_{i_1}^*, \cdots, \delta_{i_o}^*, \delta_{i_{o+1}}^*, \cdots, \delta_{i_{o+n}}^*, \delta_{k_1}^*, \cdots, \delta_{k_n}^*)$ denote an optimal solution. $f(x^*) \leq f(x') \Rightarrow \alpha_1(v_x^* - v_x^{\text{pref}})^2 + \alpha_1 (v_y^* - v_y^{\text{pref}})^2 + \alpha_2 \sum_{i\in\mathcal{O}}(\delta_i^*)^2 + \alpha_3 \sum_{j\in\mathcal{R}}(\delta_j^*)^2 + \alpha_4 \sum_{k\in\mathcal{M}} (\delta_k^*)^2 \leq \alpha_1(v_x' - v_x^{\text{pref}})^2 + \alpha_1(v_y' - v_y^{\text{pref}})^2 + \alpha_2 \sum_{i\in\mathcal{O}}(\delta_i')^2 + \alpha_3 \sum_{j\in\mathcal{R}}(\delta_j')^2 + \alpha_4 \sum_{k\in\mathcal{M}}(\delta_k')^2$. $\delta_{i_1}', \cdots, \delta_{i_{o+n}}' = 0 \Rightarrow \sum_{i\in\mathcal{O}\cup\mathcal{R}}(\delta_i')^2 = 0$. Then we have $\min(\alpha_2, \alpha_3) \sum_{i\in\mathcal{O}\cup\mathcal{R}}(\delta_i^*)^2 \leq \alpha_1((v_x' - v_x^{\text{pref}})^2 - (v_x^* - v_x^{\text{pref}})^2 + (v_y' - v_y^{\text{pref}})^2 - (v_y^* - v_y^{\text{pref}})^2) + \alpha_4 \sum_{k\in\mathcal{M}}((\delta_k')^2 - (\delta_k^*)^2)$. Further, $|v_x| \leq v^{\max} \Rightarrow |v_x' - v_x^{\text{pref}}|, |v_x^* - v_x^{\text{pref}}| \leq 2v^{\max} \Rightarrow (v_x' - v_x^{\text{pref}})^2 - (v_x^* - v_x^{\text{pref}})^2 \leq 4(v^{\max})^2$. The same applies to v_y. The half-plane constraints are effective iff $\det(\boldsymbol{B}) > 0$ and can be reduced to equality in this situation, therefore $\delta_k = \det(\boldsymbol{B})$ when $\delta_k > 0$. The distance between a velocity and a half-plane violated can be derived as $d_y^{\zeta}(v_x - p_x) - d_x^{\zeta}(v_y - p_y)$, which is the value of $\det(\boldsymbol{B})$ and equals δ_k as mentioned above. From the definition the distance is less than the norm of the relative velocity, therefore $\delta_k \leq 2v^{\max}$ and $(\delta_k')^2 - (\delta_k^*)^2 \leq 4(v^{\max})^2$. Then $\min(\alpha_2, \alpha_3) \sum_{i\in\mathcal{O}\cup\mathcal{R}}(\delta_i^*)^2 \leq 8\alpha_1(v^{\max})^2 + 4\alpha_4 n(v^{\max})^2$. $(\delta_i^*)^2 \geq 0 \Rightarrow \min(\alpha_2, \alpha_3)(\delta_i^*)^2 \leq 8\alpha_1(v^{\max})^2 + 4\alpha_4 n(v^{\max})^2, i \in \mathcal{O}\cup\mathcal{R} \Rightarrow \delta_i^* \leq 2v^{\max}\sqrt{\frac{2\alpha_1+\alpha_4 n}{\min(\alpha_2,\alpha_3)}}, i \in \mathcal{O}\cup\mathcal{R}$. Hence the violation of any of the ORCA half-planes is bounded by a distance $d = 2v^{\max}\sqrt{\frac{2\alpha_1+\alpha_4 n}{\min(\alpha_2,\alpha_3)}}$. The boundary of the VO constructed with the enlarged radius should also keep a distance d from the boundary of the VO constructed with the real radius, such that collision is avoided. To achieve this, in terms of collision avoidance against another robot, the radius of both the discs centered at $(\mathrm{p}_B - \mathrm{p}_A)$ and $(\mathrm{p}_B - \mathrm{p}_A)/\tau$ should increase at least $\frac{d}{2}$, thus R should increase at least $\frac{\tau d}{2}$. For an obstacle, the increase should be at least τd. We can conclude that if R is $2\tau v^{\max}\sqrt{\frac{2\alpha_1+\alpha_4 n}{\min(\alpha_2,\alpha_3)}}$ larger than the real radius of each robot, then all ORCA half-planes are satisfied and collision-freeness is guaranteed.

E. Masked Cooperative Collision Avoidance

The complete MCCA method is described in Algorithm 2. Note that for each robot, among the states of other robots, positions and velocities can either be observed or shared through communication, while masked velocities and priorities can only be obtained through the latter. Its own priority, masked velocity and control inputs are solved at each time step. A visualization of the progression of the states is shown in Fig. 5.

F. Communication

Unlike ORCA, which only requires the observable positions and velocities of neighboring robots, our method requires their current masked velocities and priorities in addition. These are implicit states that should be shared through communication. Given the very limited data to be transmitted, our method enjoys a low communication overhead. The information can either be gathered and broadcast through a centralized node or be exchanged through inter-agent communications in a decentralized manner, where each robot gathers information from robots

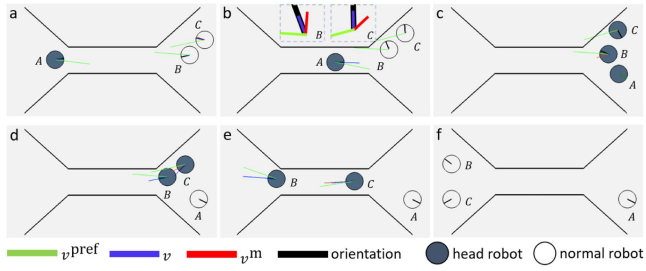


Fig. 5. (a) A upgrades to head, B and C downgrade to normal by Algorithm 1. (b) B starts to avoid A's masked velocity, and its solved masked velocity changes direction accordingly. C starts to avoid B's masked velocity, such that both B and C make way for A. (c) B and C upgrade to head by Algorithm 1. (d) A downgrades to normal when reaching its goal. (e) B and C pass through the passage. (f) B and C downgrade to normal when reaching their goals.

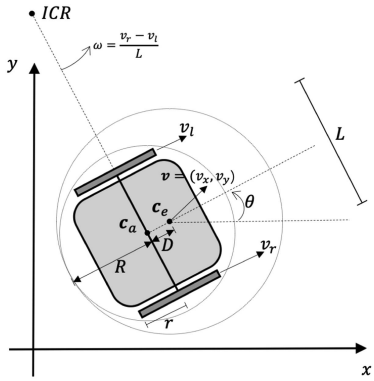


Fig. 6. For differential-drive robots, the left and right wheel velocities are controlled with separate motors. r is the wheel radius. L is the distance between the wheel centers. The velocity of the wheel axis center $\boldsymbol{c}_a$ is always perpendicular to the robot axis thus not fully controllable. The effective center $\boldsymbol{c}_e$ converts the bounding circle into a disc of radius $R + D$ and its velocity $\boldsymbol{v} = (v_x, v_y)$. Instantaneous Center of Rotation (ICR) is used to compute the angular velocity ω [20]. The robot orientation is in the direction of $\boldsymbol{c}_e - \boldsymbol{c}_a$.

Algorithm 2: MCCA.

Input $\mathcal{R}$: All robots $\mathcal{O}$: All obstacles
Output Control inputs $\boldsymbol{r}_{A_i}$ for each $A_i \in \mathcal{R}$
Initialization;
Loop Process
for *each* $A_i \in \mathcal{R}$ **do** *Control Cycle*
  Sense position/velocity of A_i;
  Sense obstacle line segments;
  Receive from each $A_{j\neq i} \in \mathcal{R}$ the states;
  Execute Algorithm 1 for A_i;
  Construct $ORCA^{\tau}_{A_i|\mathcal{O}}$;
  for $A_{j\neq i} \in \mathcal{R}$ **do**
    Construct $ORCA^{\tau}_{A_i|A_j}$;
    Construct $MCCA^{\tau}_{A_i|A_j}$ if $A_i \in \underline{\mathcal{R}}$;
  end
  Get $ORCA^{\tau}_{A_i}$ from all $ORCA^{\tau}_{A_i|A_j}$ and $ORCA^{\tau}_{A_i|\mathcal{O}}$;
  Get $MCCA^{\tau}_{A_i}$ from all $MCCA^{\tau}_{A_i|A_j}$ and $ORCA^{\tau}_{A_i|\mathcal{O}}$ if $A_i \in \underline{\mathcal{R}}$;
  Get preferred velocity $\boldsymbol{v}^{\text{pref}}_{A_i}$;
  Solve masked velocity $\boldsymbol{v}^{\text{m}}_{A_i}$ with holonomic QP;
  Solve control inputs $\boldsymbol{r}_{A_i}$ with nonholonomic QP;
end

TABLE I
VARIABLES AND PARAMETERS

Variable	Description
v_{l}	Left wheel linear speed
v_{r}	Right wheel linear speed
$\boldsymbol{v} = (v_x, v_y)$	velocity of $\boldsymbol{c}_e$
Parameter	**Description**
$v_{\mathrm{l}}^{\mathrm{c}}$	Current left wheel linear speed
$v_{\mathrm{r}}^{\mathrm{c}}$	Current right wheel linear speed
θ	Orientation
$v^{\max}$	Maximum linear velocity
$a^{\max}$	Maximum linear acceleration
Δt	Time step

within its communication range and sends to its neighboring robots. Both approaches guarantee real-time communication.

IV. An Example of Differential-Drive Robots

We present a detailed implementation for differential-drive robots. Fig. 6 shows the kinematic model. The wheel axis center is not fully controllable and we adopt the effective center approach in [21]. It is only necessary to ensure $D > 0$ for the sole purpose of kinematic adaptation, therefore we assume $D/R \approx 0.1$ or less, which is much smaller than the recommended value $D = R$.

Table I summarizes the variables and parameters of kinematic adaptation. $\boldsymbol{v} = (v_x, v_y)$ is the effective velocity of the effective center, which is at a distance of $D > 0$ from the wheel axis center. v_{l} and v_{r} are the control inputs. It follows that $v_x = \left(\frac{\cos\theta}{2} + \frac{D\sin\theta}{L}\right) v_l + \left(\frac{\cos\theta}{2} - \frac{D\sin\theta}{L}\right) v_r$ and $v_y = \left(\frac{\sin\theta}{2} - \frac{D\cos\theta}{L}\right) v_l + \left(\frac{\sin\theta}{2} + \frac{D\cos\theta}{L}\right) v_r$. Both the wheels should not exceed the maximum speed, therefore $|v_l|, |v_r| \leq v^{\max}$. $a^{\max}\Delta t$ is the largest possible change of wheel speed between two consecutive time steps, hence $|v_l - v_l^{\mathrm{c}}|, |v_r - v_r^{\mathrm{c}}| \leq a^{\max}\Delta t$. The above are linear constraints of (4b) that should be incorporated in the QP formulation.

V. Experimentation and Application

In this section, both simulation and real-world application results are presented. Compared with the state-of-the-art, the proposed method performs significantly better in difficult scenarios where deadlock is prone to occur. We recommend watching our video for a more detailed view of the results.

A. Implementation Details

Both simulation and real-world application are based on a commercially available SLAM-based mobile robot MegBot-S800.[2] All robots are differential-drive with 950 mm × 725 mm × 285 mm in three dimensions. Each robot is equipped with two powered wheels of radius $r = 0.1$ m and two passive caster wheels to balance the robot. Each wheel is powered by a servomotor, which given the control inputs will power the wheel with up to the maximum linear acceleration. In the simulation the maximum linear velocity is 2 m/s and the maximum linear acceleration is 2 m/s^2. The distance between $\boldsymbol{c}_a$ and $\boldsymbol{c}_e$ is $D = 0.015$ m. The bounding circle radius R is 0.485 m. The effective radius is $R + D = 0.5$ m, which is only 3 percent larger than R. The time step is $\Delta t = 0.25$ s. The time horizon τ is 12 s. The weight parameters in (4) are $\alpha_1 = 10^{-2}$, $\alpha_2 = 10^4$, $\alpha_3 = 10^4$, $\alpha_4 = 1$. The initial value of tabu steps η is 30.

For both simulation and real-world application, we apply a simple method to set $\boldsymbol{v}^{\mathrm{pref}}$ of a robot to point to the goal from its current location with magnitude no more than $v^{\max}$. For the purpose of this simulation, sensing and localization are not implemented, and for each robot accurate positions, velocities, masked velocities, priorities of all robots and obstacle line segments can be acquired. At each time step, based on above information each robot computes the control inputs using MCCA in a decentralized manner and applies them to the actuators. Webots [22], a mobile robot simulation software, is used for our simulation. PROX-QP [23] is used to solve the proposed QP problem.

B. Simulation Design and Results

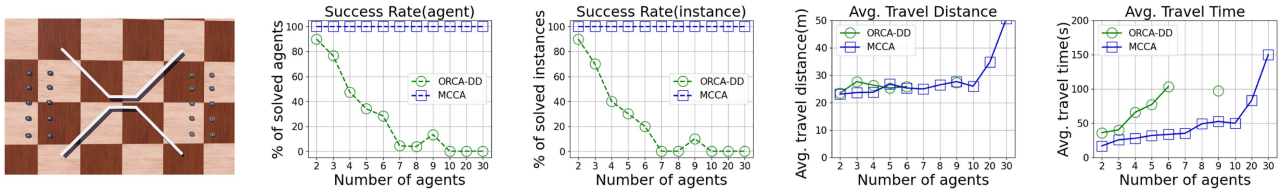
Simulation involves various configurations of narrow passages and dense obstacles, which are frequently encountered, hard to solve when numerous robots participate and considered as the main bottleneck of deadlock-free navigation in real-world environments. We conduct intensive testing on such cases to showcase the benefit of our approach over state-of-the-art. Besides, we also conduct experiments in open spaces, where no obstacle is involved.

We note that centralized methods will not be competitive against our method nor other decentralized methods like ORCA-based methods in real-world applications, as they are not able to handle more than a few robots. Besides, such methods usually require discretization of time and space, which is considered to lack adaptability, difficult to implement and overly conservative in motion behaviors. As for ORCA-based methods considering nonholonomic robots, they mainly differ in approaches of incorporating kinematics of nonholonomic motion models, and generally share the same mechanism in regard to resolving deadlock in dense situations. Hence in this section we mainly compare our method with ORCA-DD [17], which applies ORCA formulation to differential-drive robots.

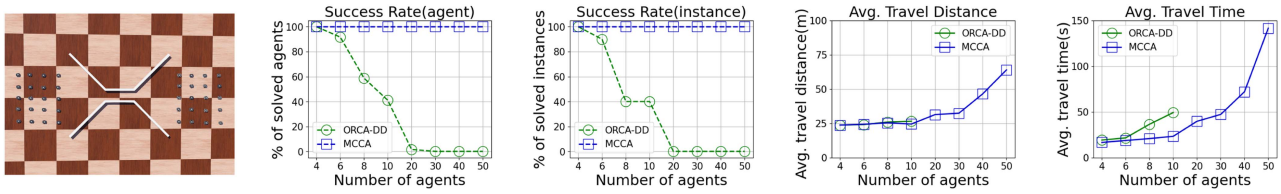
Six different scenarios are designed to compare the performance of MCCA and ORCA-DD. For each scenario and each group of N (ranging from 2 to 50) robots, we run 10 instances varying in the origin and goal positions and the following four metrics are computed and depicted in Fig. 7:

- Robot success rate: The percentage of robots out of all the robots in 10 instances that are successful. A robot fails if it remains extremely low speed (0.001m/s) for 5 minutes before reaching its goal or fails to reach its goal in 10 minutes.
- Instance success rate: The percentage of instances where all robots are successful.
- Average travel distance per robot in all successful instances.

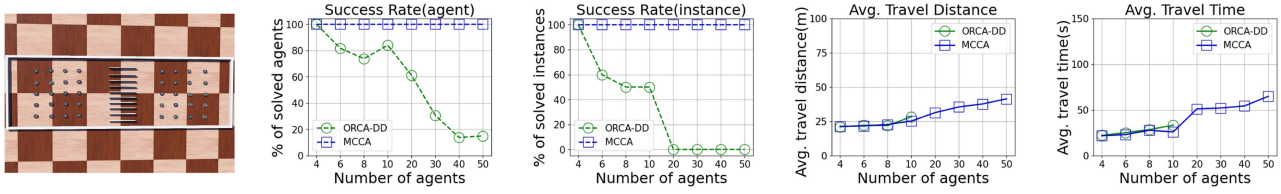
[2][Online]. Available: https://en-robotics.megvii.com/AMR-AGV.html



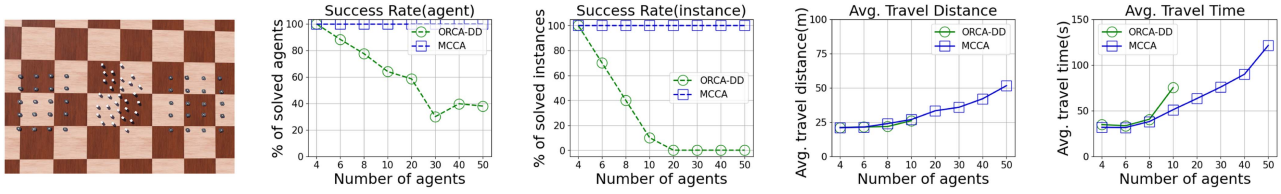
(a) Challenge Scenario 1: N robots exchange position via a one-lane passage



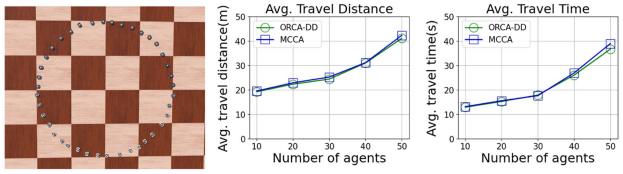
(b) Challenge Scenario 2: N robots exchange position via a two-lane passage



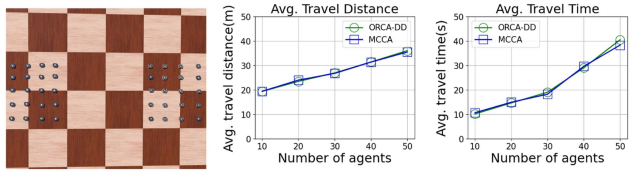
(c) Challenge Scenario 3: N robots exchange position via several one-lane passages



(d) Challenge Scenario 4: N robots exchange position in an environment of densely placed obstacles.



(e) Open Space Scenario 1: N robots in a circle exchange position



(f) Open Space Scenario 2: N robots from two sides exchange position

Fig. 7. Summary of simulation results.

- Average travel time per robot in all successful instances.

For the four scenarios featuring difficult obstacle configurations, the results clearly favor MCCA with no deadlock observed, which remains successful for all instances, while ORCA-DD deteriorates and fails quickly as the number of robots increases.

For the open space scenarios, all instances are successfully solved by both evaluated methods. In such cases, ORCA-based methods have proven to show significantly better performance conventionally, and the results indicate that our method performs competitively with ORCA-DD in terms of average travel distance and time.

We also compare the computational cost between the two methods. Similar to ORCA-DD, our method requires that each robot utilizes its onboard computer to run an instance of MCCA to obtain control inputs. For both approaches, the computational time is highly dependent on the maximum number of considered robots $\mathcal{N}$, which is determined by the number of all robots and the observation or communication range. The maximum response time of each individual robot of 10 runs in each scenario is illustrated in Table II. Although MCCA has a higher computational cost than ORCA-DD, it can still provide real-time responses.

TABLE II
MAXIMUM RESPONSE TIME OF 10 RUNS IN 6 SCENARIOS

$\mathcal{N}$	Time(ORCA-DD)[ms]	Time(MCCA)[ms]
2	0.024	0.317
10	0.058	1.103
20	0.089	2.450
40	0.199	5.016

C. Real-World Experiments

We proceed to verify the performance of MCCA with typical narrow scenarios that are frequently encountered in real-world industrial applications such as smart warehousing and in-plant

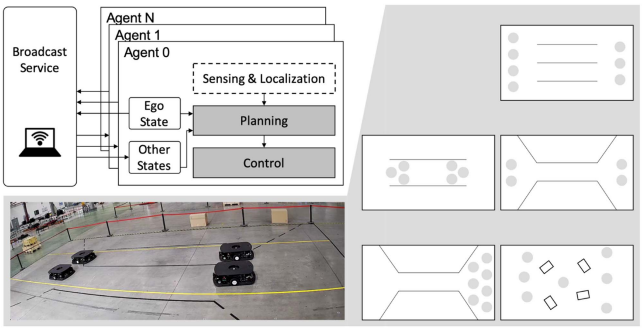


Fig. 8. Configuration for real-world application.

logistics. The configuration is shown in Fig. 8. A broadcast service is set up on a laptop. For each robot, it sends its current states(including the position, velocity, masked velocity and priority) to the service at each time step. The service broadcasts the received newest states of all robots to each individual robot at a constant interval of 0.1 s. At each time step, each robot solves the control inputs using MCCA and applies them to its actuators. Duct Tapes are used to create various obstacle configurations in a rectangular area of 120m^2. Obstacles are stored in a pre-build map and real-time localization is obtained by on-board perception module. All parameter settings are the same with our simulation, except the bounding circle radius is moderately increased to 0.6m due to error in localization.

The performance is in consistent with the simulation, which proves the effectiveness of our method in real-world environments. Deadlock is avoided entirely through local velocity planning, therefore in a complete task scenario by applying a simple global planner to guide a feasible path only considering obstacles, each robot can accomplish its moving task when deadlock with other robots is avoided automatically and no extra global coordination is needed.

VI. Conclusion and Future Work

In this letter, a new MCCA method is presented. Masked velocity, MCCA half-planes and robot priority determination are introduced to encourage a fully decentralized mechanism for deadlock avoidance, accompanied by an easy-to-solve multi-objective QP formulation. Collision-freeness of the solving paradigm is guaranteed. We demonstrate that our method can efficiently and reliably avoid deadlock and enhance overall performance in dense environments where numerous robots participate. One of our next steps is to apply MCCA method to mobile robots with different kinematics, which can be achieved by applying corresponding kinematic constraints in the QP formulation. Another direction is to compensate for uncertainties, which we believe can be realized by adding customized constraints in the QP formulation.

References

[1] J. Van den, S. J. Berg, M. Guy Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Proc. Robot. Research: 14th Int. Symp.*, 2011, pp. 3–19.

[2] J. Van den, M. Berg Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2008, pp. 1928–1935.

[3] J. Grover, C. Liu, and K. Sycara, "The before, during, and after of multi-robot deadlock," *Int. J. Robot. Res.*, vol. 42, no. 6, pp. 317–336, 2023.

[4] S. Dergachev, K. Yakovlev, and R. Prakapovich, "A combination of theta*, orca and push and rotate for multi-agent navigation," in *Proc. Interactive Collaborative A Robot.*, G. Ronzhin Rigoll and R. Meshcheryakov, Eds. Cham: Springer International Publishing, 2020, pp. 55–66.

[5] S. Dergachev and K. Yakovlev, "Distributed multi-agent navigation based on reciprocal collision avoidance and locally confined multi-agent path finding," in *Proc. IEEE 17th Int. Conf. Automat. Sci. Eng.*, 2021, pp. 1489–1494.

[6] K. Guo, D. Wang, T. Fan, and J. Pan, "Vr-orca: Variable responsibility optimal reciprocal collision avoidance," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 4520–4527, Jul. 2021.

[7] M. Čáp, J. Gregoire, and E. Frazzoli, "Provably safe and deadlock-free execution of multi-robot plans under delaying disturbances," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 5113–5118.

[8] J. Van den, J. Berg, M. C. Snoeyink Lin, and D. Manocha, "Centralized path planning for multiple robots: Optimal decoupling into sequential plans," in *Proc. Robotics: Sci. Syst.*, J. Trinkle, Y. Matsuoka, and J. A. Castellanos, Eds. The MIT Press, 2009.

[9] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, "Conflict-based search for optimal multi-agent pathfinding," *Artif. Intell.*, vol. 219, pp. 40–66, 2015.

[10] Z. Chen, J. Alonso-Mora, X. Bai, D. D. Harabor, and P. J. Stuckey, "Integrated task assignment and path planning for capacitated multi-agent pickup and delivery," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 5816–5823, Jul. 2021.

[11] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Trans. Syst., Man, Cybern.*, vol. 19, no. 5, pp. 1179–1187, Sep./Oct. 1989.

[12] X. Bai, W. Yan, M. Cao, and D. Xue, "Distributed multi-vehicle task assignment in a time-invariant drift field with obstacles," *IET Control Theory Appl.*, vol. 13, pp. 2886–2893, 2019.

[13] H. Wang and M. Rubenstein, "Walk, stop, count, and swap: Decentralized multi-agent path finding with theoretical guarantees," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 1119–1126, Apr. 2020.

[14] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. J. Robot. Res.*, vol. 17, no. 7, pp. 760–772, 1998.

[15] L. Wang, "Multi-robot coordination and safe learning using barrier certificates," Ph.D. dissertation, Georgia Institute of Technology, 2018.

[16] T. Fan, P. Long, W. Liu, and J. Pan, "Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios," *Int. J. Robot. Res.*, vol. 39, no. 7, pp. 856–892, 2020.

[17] J. Snape, J. Van Den, S. J. Berg Guy, and D. Manocha, "Smooth and collision-free navigation for multiple robots under differential-drive constraints," in *Proc. IEEE/ RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 4584–4589.

[18] J. Alonso-Mora, A. Breitenmoser, M. Rufli, P. Beardsley, and R. Siegwart, *Optimal Reciprocal Collision Avoidance for Multiple Non-Holonomic Robots*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 203–216.

[19] S. H. Arul and D. Manocha, "V-rvo: Decentralized multi-agent collision avoidance using voronoi diagrams and reciprocal velocity obstacles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 8097–8104.

[20] G. Klančar, A. Zdešar, S. Blažič, and I. Škrjanc, "Chapter 2 - motion modeling for mobile robots," in *Proc. Wheeled Mobile Robot.*, G. Klančar, A. Zdešar, S. Blažič, and I. Škrjanc, Eds. Butterworth-Heinemann, 2017, pp. 13–59.

[21] B. Kluge, D. Bank, E. Prassler, and M. Strobel, *Coordinating the Motion a Human and a Robot in a Crowded, Natural Environment*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 207–219.

[22] O. Michel, "Cyberbotics Ltd. Webots: Professional mobile robot simulation," *Int. J. Adv. Robotic Syst.*, vol. 1, no. 1, pp. 40–43, 2004.

[23] A. Bambade, S. El-Kazdadi, A. Taylor, and J. Carpentier, "PROX-QP: Yet another quadratic programming solver for robotics and beyond," in *Proc. Robotics: Sci. Syst.*, 2022. [Online]. Available: https://inria.hal.science/hal-03683733/file/Yet_another_QP_solver_for_robotics_and_beyond.pdf/