# Distributed Multi-Vehicle Task Assignment and Motion Planning in Dense Environments

Gang Xu, Xiao Kang, Helei Yang, *Graduate Student Member, IEEE*, Yuchen Wu, Weiwei Liu, Junjie Cao, and Yong Liu

***Abstract*— This article investigates the multi-vehicle task assignment and motion planning (MVTAMP) problem. In a dense environment, a fleet of non-holonomic vehicles is appointed to visit a series of target positions and then move to a specific ending area for real-world applications such as clearing threat targets, aid rescue, and package delivery. We presented a novel hierarchical method to simultaneously address the multiple vehicles' task assignment and motion planning problem. Unlike most related work, our method considers the MVTAMP problem applied to non-holonomic vehicles in large-scale scenarios. At the high level, we proposed a novel distributed algorithm to address task assignment, which produces a closer to the optimal task assignment scheme by reducing the intersection paths between vehicles and tasks or between tasks and tasks. At the low level, we proposed a novel distributed motion planning algorithm that addresses the vehicle deadlocks in local planning and then quickly generates a feasible new velocity for the non-holonomic vehicle in dense environments, guaranteeing that each vehicle efficiently visits its assigned target positions. Extensive simulation experiments in large-scale scenarios for non-holonomic vehicles and two real-world experiments demonstrate the effectiveness and advantages of our method in practical applications. The source code of our method can be available at https://github.com/wuuya1/LRGO.**

***Note to Practitioners*—The motivation for this article stems from the need to solve the multi-vehicle task assignment and motion planning (MVTAMP) problem for non-holonomic vehicles in dense environments. Many real-world applications exist, such as clearing threat targets, aid rescue, and package delivery. However, when vehicles need to continuously visit a series of assigned targets, motion planning for non-holonomic vehicles becomes more difficult because it is more likely to occur sharp turns between adjacent target path nodes. In this case, a better task allocation scheme can often lead to more efficient target visits and save all vehicles' total traveling distance. To bridge this, we proposed a hierarchical method for solving the MVTAMP problem in large-scale complex scenarios. The numerous large-scale simulations and two real-world experiments show the effectiveness of the proposed method. Our future work will focus on the integrated task assignment and motion planning problem for non-holonomic vehicles in highly dynamic scenarios.**

***Index Terms*— Distributed system, task assignment, motion planning for non-holonomic vehicles.**

## I. Introduction

MULTIPLE vehicle systems are becoming increasingly popular for research and applications, such as surveillance [1], [2], search and rescue [3], [4], transportation [5], [6], and delivering packages [7], [8], where a fleet of vehicles is required to visit several task positions efficiently. Due to the unparalleled endurance and practical environmental conditions, non-holonomic vehicles that can not move omni-directionally are widely utilized in these applications, such as fixed-wing unmanned aerial vehicles (UAVs) and unmanned surface vehicles (USVs), where their motion models can be simplified as the Dubins model [9]. Therefore, implementing large-scale autonomous non-holonomic vehicle systems is an inevitable trend. In particular, solving the multi-vehicle task assignment and motion planning (MVTAMP) problem is essential [10] for implementing large-scale vehicle systems. It guarantees that each non-holonomic vehicle can efficiently select tasks to visit under its limited capacity while ensuring that it does not collide with other vehicles or obstacles.

Many existing works, like [11], [12], [13], only focus on the multi-vehicle task assignment without considering the vehicles' motion planning. Recently, some works [14], [15], [16] addressed the MVTAMP problem by integrated task assignment and motion planning and considered large-scale systems. However, these proposed approaches only consider holonomic vehicles that can move in any direction. Since what is widely used in real-world application scenarios are non-holonomic vehicles, this leads to the gap in applications of multi-vehicle systems. In addition, solving the MVTAMP problem for non-holonomic vehicles is much more complex than in the holonomic one due to its more complex constraints. Meanwhile, non-holonomic vehicles often encounter deadlocks due to myopic local planning, especially in dense environments.

This article aims to resolve the above issues and ensure the multi-vehicle systems' practicality in large-scale dense environments. We first improved the task assignment scheme quality by introducing a novel review consensus method for conflict resolution while balancing the computational overhead. Meanwhile, based on the optimal reciprocal collision

Manuscript received 20 October 2023; accepted 20 November 2023. Date of publication 27 November 2023; date of current version 16 October 2024. This article was recommended for publication by Associate Editor L. Bascetta and Editor J. Yi upon evaluation of the reviewers' comments. This work was supported by the National Natural Science Foundation of China under Grant U21A20484. *(Gang Xu and Xiao Kang contributed equally to this work.) (Corresponding authors: Yong Liu; Junjie Cao.)*

Gang Xu, Helei Yang, Yuchen Wu, Weiwei Liu, Junjie Cao, and Yong Liu are with the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310027, China (e-mail: cjunjie@zju.edu.cn; yongliu@iipc.zju.edu.cn).

Xiao Kang is with the China North Artificial Intelligence and Innovation Research Institute, Beijing 100072, China.

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TASE.2023.3336076.

Digital Object Identifier 10.1109/TASE.2023.3336076

avoidance (ORCA) concept [17], we consider the non-holonomic vehicles' kinematic constraints, collision avoidance, and deadlock challenges in motion planning while maintaining real-time performances. To the best of our knowledge, this article presents the first comprehensive solution for simultaneously addressing the above issues in dynamic large-scale dense environments while verifying the proposed method in the real world. The main contributions can be summarized as follows.

- At the high level, we proposed an efficient algorithm, called the lazy-based review consensus algorithm (LRCA), to address the multi-vehicle task assignment problem, where each vehicle can independently compute its allocation solution by solving an overall objective function and resolute conflict of allocation solution by a novel review consensus strategy. The proposed algorithm can achieve a closer to the optimal assignment solution and balance the computation cost in large-scale scenarios.
- At the low level, we proposed a novel motion planning algorithm to solve the multi-vehicle motion planning problem, where our proposed guidance point strategy (GOS) addresses the vehicles' deadlock issues, and an improved Dubins curve overcomes the non-holonomic vehicles' constraints. In particular, the proposed method can quickly solve a feasible new velocity for each non-holonomic vehicle in dense environments.
- Intergrated the above two points, we presented a hierarchical distributed method to solve the task assignment and motion planning problem for multiple non-holonomic vehicles in dense environments. Meanwhile, enormous simulations and two real-world experiments demonstrated the performance advantages and practicability of the proposed method, respectively.

The rest of the article is organized as follows. Section II summarizes the related works. In Section III, the problems investigated in this article are formulated. Section IV presents the details of the proposed methods and the associated analysis. Section V demonstrates the performances of our method by conducting simulation and real-world experiments, and Section VI concludes this article.

## II. Related Works

This section summarizes the related works of multi-vehicle task assignment and motion planning.

The multi-vehicle task assignment is a canonical NP-hard optimization problem, where solving for the optimal solution is essentially infeasible, except for small instances. Many approaches have been proposed to solve this problem in centralized settings. Among them, the Hungarian algorithm [18] is widely used as a classical method for task assignment, in which the method can provide the optimal scheme. Besides, the heuristic-based approaches [19], [20], [21] and auction-based approaches [22], [23], [24] have also been in the spotlight as its has proficiently solved task assignment problems in many scenarios. However, these methods are unsuitable for large-scale scenarios as they have crucial drawbacks, especially the computational complexity. Unlike centralized approaches, although distributed approaches do not provide optimal solutions for multi-vehicle task assignment problems in most cases, they have attracted more attention because most of them can provide computational complexity guarantees. Among them, auction-based methods [25], [26], [27], [28] are popular for quickly providing feasible solutions, in which each vehicle updates the assignment of all tasks iteratively by receiving bids from its neighbors until all the tasks are assigned. Recently, the authors proposed a distributed algorithm [29] by introducing a consensus ADMM (alternating direction method of multipliers) [30], resulting in the optimal task assignment. The work [31] addresses the task-level path planning problem of forest wildfire monitoring in large-scale dynamic scenarios. However, these methods only address the task assignment problem in the MVTAMP problem rather than the remaining motion planning simultaneously.

Notably, some research works have studied the MVTAMP problem in recent years. In [32], the authors consider this problem by a Markov decision process (MDP) in an underwater workspace but fail to consider any obstacles. A reinforcement learning-based method [33] was proposed to solve the problem but failed to guarantee collision avoidance in many scenarios. In [34], the authors integrated the task assignment and path planning for multi-robot pickup and delivery. It should be noted that these works only considered the holonomic vehicles. Several methods take into account the non-holonomic vehicles, such as [35], [36], [37], [38], [39], [40], and [41]. In [35], [36], and [37], these works address the MVTAMP problem for the vehicles of the Dubins model [9]. The work [38] considers unmanned surface vehicles' kinematic and dynamic constraints in solving the assignment and planning problem for rescuing targets in a complex ocean environment. In [39], a coordinated method for target assignment and UAV path planning was developed, where collision avoidance is achieved by reducing the velocity of the air vehicle. The work [40] modifies the consensus-based auction algorithm [25] to complete the dynamic reallocating tasks for robots while making the robots avoid unexpected obstacles. In [41], a method that combined the A* [42] and task allocation algorithms was proposed for search and rescue missions. However, these methods are unsuitable for large-scale scenarios due to the computational overhead and deadlocks. Recently, the work [14] presented a novel method for large-scale multi-robot systems, in which the method integrated task assignment and motion planning into a comprehensive optimization problem that aims to minimize the total traveling cost and potential path conflicts simultaneously. The work [15] proposed a distributed algorithm based on a primal decomposition approach to solve the MVTAMP problem for addressing multi-vehicle pickup and delivery. The work [43] proposes a novel suboptimal complete algorithm, TSWAP, that adopts an arbitrary initial target assignment and then repeats one step of path planning by exchanging targets. However, these works assume that the vehicle is holonomic. As a result, deploying large-scale non-holonomic vehicle systems into real-world applications remains incredibly challenging.

Unlike the research works above, the method proposed in this article aims to simultaneously address task assignment and motion planning for multiple non-holonomic vehicle systems, further promoting large-scale non-holonomic vehicles' application in the real world.

## III. Problem Formulation

In this section, we first describe the non-holonomic vehicle's kinematic model. Then, we summarize the formulations of the task assignment and motion planning.

### A. Vehicle Kinematic Model

Considering the applications, such as fixed-wing UAVs and USVs, in the real world, we simplified the non-holonomic vehicle's kinematic constraints using the Dubins model, which assumes that vehicles have a minimum turning radius and can move only forward. Let the vehicle's vector position and velocity be $\mathbf{p} = [x, y]^T$ and $\mathbf{v} = \dot{\mathbf{p}} = [\dot{x}, \dot{y}]^T$, respectively, where $(x, y)$ is its Cartesian coordinates in a global frame. Let the vehicle's orientation angle be $\theta$, that is the angle between the X-axis's orientation and the vehicle's heading. Then, let the vehicle's linear and angular speeds be $v$ and $\omega$, respectively. Finally, the non-holonomic vehicles' kinematic models are formulated as

$$\begin{aligned}\dot{x} &= v\cos\theta\\ \dot{y} &= v\sin\theta\\ \dot{\theta} &= \omega.\end{aligned} \tag{1}$$

According to the Dubins model, the kinematic constraints of the vehicle can be deduced as

$$|\omega| \le \frac{v}{r_{min}}, \tag{2}$$

where $r_{min}$ is the minimum turning radius of the vehicle.

Then, let $T_s$ be the sample time of the computer system equipped with the vehicle. We discretize the Eqn. (1) as

$$\begin{aligned}x_{k+1} &= x_k + T_s v_k \cos\theta_k\\ y_{k+1} &= y_k + T_s v_k \sin\theta_k\\ \theta_{k+1} &= \theta_k + T_s\omega_k,\end{aligned} \tag{3}$$

where $\omega_k$ is subject to $|\omega_k| \le v_k/r_{min}$, and $[x_{k+1}, y_{k+1}, \theta_{k+1}]^T$ and $[v_k, \omega_k]^T$ are the state $\mathbf{q}_{k+1}$ at the next time and the control input $\mathbf{u}_k$ at the current time of the vehicle, respectively. Note that we regard the vehicle as a circle with a radius $r$ in this article.

### B. Task Assignment Formulation

Let $V = \{1, 2, \ldots, i, \ldots m\}$ be the indices' set of a team of vehicles that need to orderly complete the assigned targets' visit, where $m$ is the number of vehicles. Each vehicle can be assigned maximum $L_i$ targets where $i \in V$. Meanwhile, let $T = \{1, 2, \ldots, j, \ldots n\}$ be the indices' set of $n$ randomly placed targets in 2-D space. For generalization, we define $v_j$ as the target's important factor that indicates the priority of the target $j$ visited by a vehicle. In this article, each target has the same priority, i.e., $v_j = 1, \forall j \in T$. We also denote $\lambda_l$ as the distance discount factor that makes an effort to decrease the traveled distance needed to visit the target by vehicles, and $\lambda_l = 0.95$ in this article. Referred to [25], the total reward value $f_i(T_i)$ of vehicle $i$ visiting the targets in $T_i$ can be formulated as

$$f_i(T_i) = \sum_{j=1}^{|T_i|} v_j \lambda_l^{d(Path_i^j)}, \tag{4}$$

where $|\cdot|$ is the cardinality of the list, $T_i \subseteq T$ is the set of targets that are assigned to the vehicle $i$ in sequence, $Path_i$ is the initial path of vehicle $i$ generated by $T_i$, $Path_i^j$ is the sub-path of $Path_i$ formed from the position of vehicle $i$ to the position of the target $j$, and $d(Path_i^j)$ is the length of the sub-path $Path_i^j$.

To achieve the assignment scheme of vehicle $i$, we add the unassigned target $k$ to the set $T_i$. Thus, the marginal reward $\omega_i(k)$ of adding target $k$ to $T_i$ can be defined as Eqn. (5).

$$\omega_i(k) = f_i(T_i \cup \{k\}) - f_i(T_i) \quad \forall k \in T, k \notin T_i. \tag{5}$$

Consequently, the marginal reward $\omega_i(k)$ can be obtained by combining Eqn. (4) and Eqn. (5):

$$\omega_i(k) = \sum_{j=1}^{|T_i \cup \{k\}|} v_j \lambda_l^{d(Path_i^j)} - \sum_{j=1}^{|T_i|} v_j \lambda_l^{d(Path_i^j)}. \tag{6}$$

Note that $\omega_i(k)$ guarantees that the objective function is a submodular function, which has an essential property of diminishing marginal reward, i.e., the marginal reward $\omega_i(k)$ for a given task $k$ will not increase as the selected tasks (refer to [25]), ensuring convergence of our method.

Finally, the overall marginal reward function of the task assignment can be described as

$$\begin{aligned}&\max \sum_{i=1}^{|V|}\left(\sum_{j=1}^{|T_i|} v_j \lambda_l^{d(Path_i^j)}\right)\\ &\text{s.t. } |T_i| \le L_i \quad \forall i \in V\\ &\qquad \sum_{i=1}^{|V|}\left(x_{ij}\right) = 1 \quad \forall j \in T,\end{aligned} \tag{7}$$

where $x_{ij} = 1$ if target $j$ is assigned to vehicle $i$ and 0 otherwise. The constraints indicate that the number of targets assigned to each vehicle does not exceed its maximum $L_i$, each target can be assigned to only one vehicle, and one vehicle can select multiple targets.

### C. Motion Planning Formulation

Considering the real-time performance of multi-vehicle systems, we expanded the ORCA concept [17] for multi-vehicle motion planning in dense environments. Here, we first introduce the concept of ORCA and clarify its limitations.

We illustrate the ORCA by considering the collision avoidance behavior of two vehicles $A$ and $B$ whose radii are $r_A$ and $r_B$, respectively, where $A \in V$ and $B \in V$. As shown in Fig. 1(a), $A$ and $B$ move to the states $\mathbf{q}_A = [p_x^A, p_y^A, \theta_A]^T$ and $\mathbf{q}_B = [p_x^B, p_y^B, \theta_B]^T$ with the velocities $\mathbf{v}_A$ and $\mathbf{v}_B$, respectively. Note that the positions of $A$ and $B$ are $\mathbf{p}_A = [p_x^A, p_y^A]^T$ and $\mathbf{p}_B = [p_x^B, p_y^B]^T$, respectively. In ORCA, the velocity obstacle $VO_{A|B}^{\tau}$ is described as the set of relative velocities $\mathbf{v}_{AB} = \mathbf{v}_A - \mathbf{v}_B$ that will lead to a collision between $A$ and $B$ before time $\tau$. Let $D(\mathbf{p}, r)$ be an open disc whose

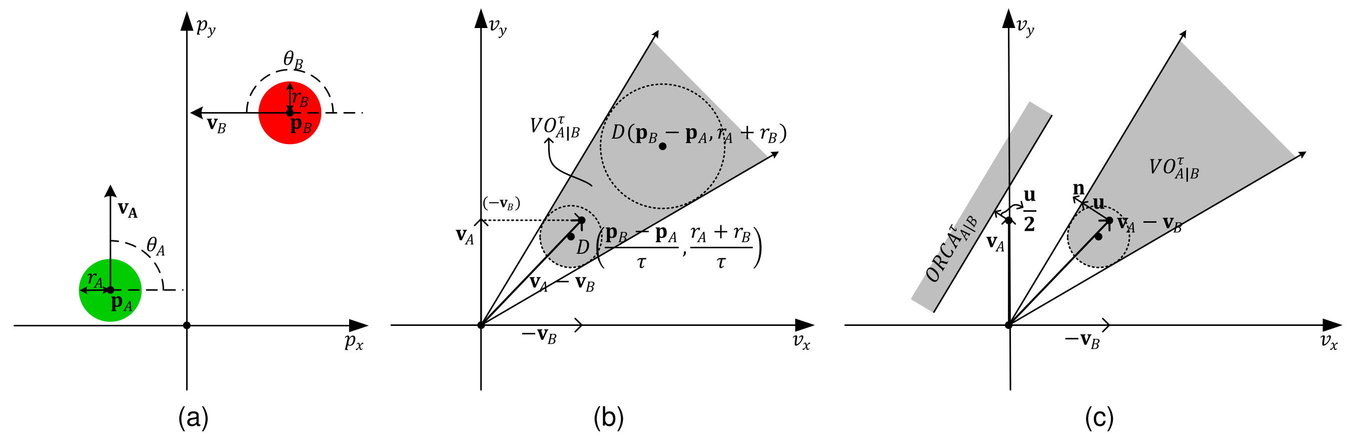


Fig. 1. (a) A configuration of two vehicles $A$ and $B$. (b) The geometric illustration of velocity obstacle with time window $\tau$ of $A$ with respect to $B$. Here, $VO^{\tau}_{A|B}$ is calculated by a cone with its apex at the origin (in velocity space) and its legs tangent to disc $D(\mathbf{p}_B - \mathbf{p}_A, r_A + r_B)$, in which an arc of disc $D((\mathbf{p}_B - \mathbf{p}_A)/\tau, (r_A + r_B)/\tau)$ truncates the cone, and the amount of truncation depends on the value of $\tau$. (c) The geometric illustration of the ORCA concept of $A$ with respect to $B$.

radius is $r$ and center is $\mathbf{p}$. Thus, we can formulate the $VO^{\tau}_{A|B}$ as

$$VO^{\tau}_{A|B} = \{\mathbf{v}_{AB} \mid \exists t \in [0, \tau] :: t\mathbf{v}_{AB} \in D(\mathbf{p}_B - \mathbf{p}_A, r_A + r_B)\}, \tag{8}$$

where the geometric illustration of velocity obstacle $VO^{\tau}_{A|B}$ is presented in Fig. 1(b) with the gray area. It indicates that $A$ and $B$ will collide before time $\tau$ if $\mathbf{v}_{AB} \in VO^{\tau}_{A|B}$. Otherwise, $A$ and $B$ will be collision avoidance with at least time $\tau$.

Subsequently, Let $ORCA^{\tau}_{A|B}$ be the set of permitted velocities of $A$. As shown in Fig. 1(c), the $ORCA^{\tau}_{A|B}$ can be formulated as

$$ORCA^{\tau}_{A|B} = \left\{\mathbf{v} \mid \left(\mathbf{v} - \left(\mathbf{v}_A + \frac{1}{2}\mathbf{u}\right)\right) \cdot \mathbf{n} \geq 0\right\}, \tag{9}$$

where $ORCA^{\tau}_{A|B}$ is a half-plane with starting point $\mathbf{v}_A + \frac{1}{2}\mathbf{u}$ and direction $\mathbf{n}$, $\mathbf{u}$ is the vector from $\mathbf{v}_{AB}$ to the closest point on the boundary of the $VO^{\tau}_{A|B}$, and $\mathbf{n}$ is the normal vector pointing outward of the boundary of $VO^{\tau}_{A|B}$.

Considering all neighbors $B$ of $A$, the feasible velocity set $ORCA^{\tau}_{A}$ for $A$ is the intersection of the disc $D(\mathbf{0}, v^{max}_A)$ and all half-planes $ORCA^{\tau}_{A|B}$, where $v^{max}_A$ is the maximum speed of $A$. Therefore, the $ORCA^{\tau}_{A}$ can be described as

$$ORCA^{\tau}_{A} = D\left(\mathbf{0}, v^{\max}_A\right) \cap \bigcap_{B \neq A} ORCA^{\tau}_{A|B}. \tag{10}$$

Then, vehicle $A$ can achieve the optimal new velocity $\mathbf{v}^{\text{new}}_A$ at next time by

$$\mathbf{v}^{\text{new}}_A = \begin{cases} \underset{\mathbf{v} \in ORCA^{\tau}_A}{\arg\min} \left\| \mathbf{v} - \mathbf{v}^{\text{pref}}_A \right\|, & \text{if } ORCA^{\tau}_A \neq \emptyset \\ \underset{\mathbf{v} \in D\left(\mathbf{0}, v^{\max}_A\right)}{\arg\min} \ \underset{B \neq A}{\max}\, d_{A|B}(\mathbf{v}), & \text{otherwise} \end{cases}, \tag{11}$$

where $\mathbf{v}^{\text{pref}}_A$ is the preferred velocity of $A$, and $d_{A|B}(\mathbf{v})$ indicates the signed distance of $\mathbf{v}$ to the edge of the any half-plane $ORCA^{\tau}_{A|B}$.

Although each vehicle can quickly achieve the feasible velocity, the ORCA is unsuitable for non-holonomic vehicles due to the absence of kinematic constraints. In addition, the ORCA may also encounter deadlock issues, especially in dense environments. This article overcame these limitations.

## IV. Proposed Method

This section first presents a novel distributed lazy-based review consensus algorithm (LRCA) to address the multi-vehicle task assignment. Then, we propose a novel path generation method named guidance point strategy (GOS) for addressing the deadlock issues. Finally, we propose a hierarchical method, LRGO, to simultaneously address the task assignment and motion planning for non-holonomic vehicles, which integrated the above-proposed methods with the ORCA and the improved Dubins curve.

### A. Lazy-Based Review Consensus Algorithm

In multi-vehicle task assignment methods, the auction-based method must resolve the conflict during bids for tasks, in which maximum consensus strategy is an efficient and fast method [28]. However, assigning the best task to its highest bidder in each iteration may not guarantee that the solution is closer to the global optimal. In this case, assigning this task to the vehicle with the same best task and a higher global reward can always lead to the assignment solution closer to the global optimal. Based on the inspiration, we propose a novel review consensus strategy for conflict resolution over all tasks.

We first introduce some notations used next. Let $N_i$ be the unassigned task set of vehicle $i$ and $T_i$ be the ordered task set that has already been allocated to the vehicle $i$. Let $T$ be the set of $T_i$; note that each vehicle can update $T$ in each iteration without communication. In each iteration, we denote $j^*_i$ as the current best task with the highest marginal reward $\omega^*_i$ for vehicle $i$. In addition, we denote $\Omega_i$ as the set of marginal rewards of tasks in $N_i$. Let $I_i$ be the set of the ID of all vehicles. Let $J^*_i$ and $W^*_i$ be the set of $j^*_i$ and $\omega^*_i$ of all vehicles, respectively. Here, we sort the set $W^*_i$ in descending order, sorting the rest of the sets $I_i$ and $J^*_i$ according to the sorted $W^*_i$. In the auction phase, the LRCA is the same as the LSTA [28], except that the former considers all tasks, and the last one considers only the tasks in a sample set selected with probability $p$ from all tasks. The reader is referred to [28] for a detailed auction process. After completing the auction process, the vehicle $i$ shares its ID, the best task $j_i^*$, the marginal reward $\omega_i^*$ of task $j_i^*$, and the position $\mathbf{p}_i$ with all its neighbors while receiving corresponding information from its neighbors to enter the consensus phase of conflict resolution.

In the consensus phase, the vehicle negotiates with all its neighbors to determine who will acquire the current global best task, where the global best task and vehicle assigned to this task are denoted as $j_{a^*}^*$ and $a^*$, respectively. Then, $j_{a^*}^*$ is added to the set $T_i$ if $a^* = i$. Meanwhile, the task $j_{a^*}^*$ and its corresponding marginal reward are removed from the set $N_i$ and $\Omega_i$, respectively. We describe the details of implementing the review consensus strategy in Algorithm 1. In the review consensus strategy, instead of allocating the best task $j_{a^*}^*$ directly to the vehicle $a^*$ with the highest marginal reward, the other vehicle with the same best task is also considered. For this, we first find this candidate vehicle $b^*$ with the following three conditions: vehicle $b^*$ has the same best task as vehicle $a^*$, vehicle $b^*$ has been assigned at least one task, and the difference $\varepsilon$ between the highest marginal reward of vehicle $a^*$ and the highest marginal reward of vehicle $b^*$ must be small enough, empirically with $\varepsilon$ being 0.02 in this article. The above process is presented in lines 1-6 of Algorithm 1. After finding candidate vehicle $b^*$, the review consensus strategy first determines whether the line segment formed by the position of task $j_{a^*}^*$ and the position of the latest assigned task of vehicle $a^*$ intersects a line segment formed by the positions of two adjacent tasks assigned to the vehicle $b^*$, where $\mathbf{p}(*)$ in lines 10-11 is the position of the task $*$. If the above two line segments intersect, the review consensus strategy assigns the current best task $j_{a^*}^*$ to the vehicle that generates the maximum global reward (see lines 12-19), as shown in Fig. 2. Otherwise, the review consensus strategy assigns the best task $j_{a^*}^*$ to vehicle $a^*$ for conflict resolution.

**Algorithm 1** Review Consensus Strategy for Vehicle $i$

```
Input: Sorted $I_i$, $J_i^*$, and $W_i^*$
Output: $j_{a^*}^*$, $a^*$
1:  $j_{a^*}^*$, $a^* \leftarrow$ the first element from $J_i^*$ and $I_i$, respectively
2:  for $j_b^*$ in $J_i^*$, $b$ in $I_i$, $\omega_b^*$ in $W_i^*$, $b! = a^*$ do
3:      if $j_b^* == j_{a^*}^*$ and $|T_b| > 0$ and $\left|\omega_b^* - \omega_{a^*}^*\right| < \varepsilon$ then
4:          $b^* \leftarrow b$; break
5:      end if
6:  end for
7:  if $b^*$ is NULL then
8:      return $j_{a^*}^*$, $a^*$
9:  else
10:     $Segment1 \leftarrow \{\mathbf{p}(j_{a^*}^*), \mathbf{p}(T_{a^*}[-1])\}$
11:     $Segment2 \leftarrow \{\mathbf{p}(j_{b^*}^{n-1}), \mathbf{p}(j_{b^*}^{n})\}, \forall j_{b^*}^{n-1}, j_{b^*}^{n} \in T_{b^*}$
12:     if $Segment1$ intersects $Segment2$ then
13:         $Reward1 \leftarrow f_{a^*}(T_{a^*} \cup \{j_{a^*}^*\}) + f_{b^*}(T_{b^*})$
14:         $Reward2 \leftarrow f_{a^*}(T_{a^*} \cup T_{b^*}[n:]) + f_{b^*}(T_{b^*}[:n] \cup \{j_{a^*}^*\})$
15:         if $Reward1 < Reward2$ then
16:             $T_{a^*} \leftarrow T_{a^*} \cup T_{b^*}[n:]$, $T_{b^*} \leftarrow T_{b^*}[:n]$
17:             return $j_{a^*}^*$, $b^*$
18:         end if
19:     end if
20: end if
21: return $j_{a^*}^*$, $a^*$
```

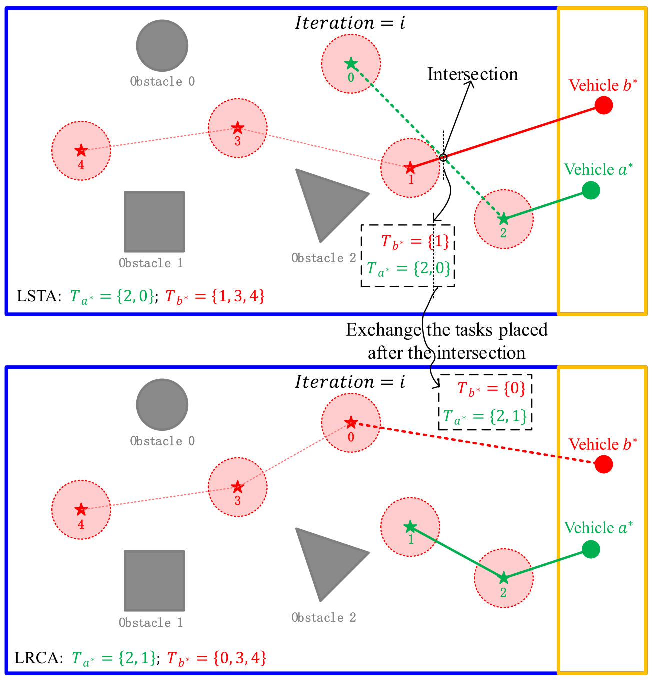


Fig. 2. Illustration for the review consensus strategy. The allocation scheme within the dashed box indicates the result at the $i^{th}$ iteration. Notably, the top and bottom figures also depict the final allocation results obtained using the LSTA and the LRCA, respectively.

## B. Analysis for LRCA

LRCA exhibits identical theoretical performance to LSTA [28] in terms of approximation ratio and computational complexity. We summarize the performance of LRCA as follows. The LRCA achieves an expected approximation guarantee of 50% for monotone submodular objective functions while achieving an expected total computational complexity of $O(mn)$ and individual complexity of $O(n^2)$ for each vehicle, where $n = |T|$ is the number of tasks and $m = |V| \times |T|$ is the number of task-vehicle pairs.

Before analyzing why LRCA and LSTA are entirely consistent in terms of theoretical performance, it is necessary to explain how the review consensus strategy works. Assume that the global best task at the $i^{th}$ iteration using the LRCA is $j_{a^*}^* = 0$, which corresponds to the two vehicles with the highest and second highest marginal rewards, $a^*$ and $b^*$, respectively, like Fig. 2. At this moment, $T_{a^*} = \{2\}$ is the task set assigned to vehicle $a^*$, while $T_{b^*} = \{1\}$ is the task set assigned to vehicle $b^*$. In the presence of satisfying the condition stated in line 3 of Algorithm 1, the newly added segment $\{\mathbf{p}(0), \mathbf{p}(2)\}$ if assigning the current best task $j_{a^*}^* = 0$ to $a^*$ (i.e., $T_{a^*} = T_{a^*} \cup \{0\} = \{2, 0\}$) intersects with the segment formed by the position of $b^*$ and the position of task 1 (note that $T_{b^*} = \{1\}$), as stated in line 12 of Algorithm 1. Then, the LRCA exchanges the tasks placed after the intersection, i.e., from $T_{a^*} = \{2, 0\}$ and $T_{b^*} = \{1\}$ to $T_{a^*} = \{2, 1\}$ and $T_{b^*} = \{0\}$, respectively, and checks if the total marginal reward of the allocation scheme after the exchange is greater than the one before the exchange, i.e., satisfying the inequality in line 15 of Algorithm 1, as shown in Fig. 2. If the inequality is satisfied, the LRCA will adjust the allocation results from the $(i-1)^{th}$ iteration to obtain a better assignment scheme through line 16 of Algorithm 1. Since the proposed method reviews the allocation result from the $(i-1)^{th}$ iteration, it is called the review consensus strategy.

Using the inequality of line 15 of Algorithm 1 as the determination condition ensures that the results of the previous iteration are only adjusted if a higher global gain can be obtained. Therefore, in the worst case, the LRCA provides the same approximation guarantee as the LSTA. However, in most cases, the LRCA is superior to the LSTA. Additionally, since the LRCA only adds conflict resolution from the last iteration in the consensus phase, it does not change the overall computational complexity. Therefore, the complexity of the LRCA is the same as the LSTA. Thus, the proof of the LRCA's theoretical performance can refer to the LSTA [28].

## C. Guidance Point Strategy for Path Generation

This part describes the proposed guidance point strategy (GOS) that addresses the deadlocks of vehicles in the ORCA by quickly generating a guided path. In particular, the GOS does not need the grid map while possessing the ability of the D* algorithm [44] to handle unknown obstacles.

Let the target position that vehicle $i$ is about to visit be $\mathbf{p}_i^g$, the next visit position be $\mathbf{p}_i^n$ (also called the guidance point), and the current position be $\mathbf{p}_i$ of vehicle $i$. Note that before updating the path $Path_i$ of vehicle $i$, $\mathbf{p}_i^n$ equals $\mathbf{p}_i^g$. Let $N_i^{\max}$ be the maximum number of neighbors perceivable by vehicle $i$. Then, let the sub-path formed by $\mathbf{p}_i$ and $\mathbf{p}_i^n$ be $L_{cn}$, i.e., $L_{cn} = \{\mathbf{p}_i, \mathbf{p}_i^n\}$. The core idea of the GOS is as follows. Suppose $L_{cn}$ intersects with the obstacles around vehicle $i$. In this case, the GOS first solves an optimal guidance point in the feasible area outside the nearest intersecting obstacle while adding it to $Path_i$ and updating the sub-path $L_{cn}$ for vehicle $i$. Repeat the above steps until $L_{cn}$ does not intersect with the obstacles around the vehicle. The critical point of the GOS is how to solve an optimal guidance point $\mathbf{p}_i^n$ for collision avoidance when $L_{cn}$ intersects with the obstacles around vehicle $i$. To the completeness of the GOS, we assume that the obstacles in dense environments satisfy *Condition 1*.

*Condition 1:* All obstacles are either convex obstacles or obstacles with at most one concave vertex, and at most $N$ ($N < N_i^{\max}$) circular obstacles are connected, and all polygonal obstacles are separated obstacles. Here, we refer to obstacles close to each other so that the vehicle $i$ cannot traverse the clearance between obstacles as connected obstacles. In contrast, a separated obstacle means that vehicle $i$ can pass through the clearance between the obstacles.

Note that this assumption is reasonable, as the above obstacles can describe obstacles of various shapes, such as U-shaped or other shaped obstacles, in large-scale environments, especially outdoor ones. Next, we introduce how the GOS works and present its implementation in Algorithm 2.

We first consider that there is a circular static obstacle $ob$ with radius $r_{ob}$ and position $\mathbf{p}_{ob}$ in the workspace, closest to vehicle $i$ among all intersected obstacles with sub-path $L_{cn}$ of vehicle $i$. In this case, the formulation of solving the guidance point $\mathbf{p}_i^n$ can be derived by

**Algorithm 2** Guidance Point Strategy (GOS) for Vehicle $i$

```
Input: $\mathbf{p}_i$, $\mathbf{p}_i^n$, $Neighbors_i$, $Path_i$, vehicle $i$
Output: the collision-free sub-path $L_{cn}$
1: while $Segment(\mathbf{p}_i, \mathbf{p}_i^n)$ intersects $Neighbor_i$ do
2:   for $obj$ in $Neighbors_i$ do
3:     Inflate $obj$ using $MinkowskiSum(obj$, vehicle i)
4:     if $obj$ is circular obstacle then
5:       if $obj$ intersects $Segment(\mathbf{p}_i, \mathbf{p}_i^n)$ then
6:         Compute the obstacles connected to $obj$
7:         $\mathbf{p}_i^* \leftarrow$ Compute the guide point by Eqn. (12)
8:         $Path_i \leftarrow Path_i \cup \{\mathbf{p}_i^n\}$, $\mathbf{p}_i^n \leftarrow \mathbf{p}_i^*$
9:         $L_{cn} \leftarrow \{\mathbf{p}_i, \mathbf{p}_i^n\}$
10:      end if
11:    else
12:      if $obj$ is polygon obstacle then
13:        if $obj$ intersects $Segment(\mathbf{p}_i, \mathbf{p}_i^n)$ then
14:          $\mathbf{p}_i^* \leftarrow$ Compute the guide point by Eqn. (14)
15:          $Path_i \leftarrow Path_i \cup \{\mathbf{p}_i^n\}$, $\mathbf{p}_i^n \leftarrow \mathbf{p}_i^*$
16:          $L_{cn} \leftarrow \{\mathbf{p}_i, \mathbf{p}_i^n\}$
17:        end if
18:      end if
19:    end if
20:  end for
21: end while
22: return $L_{cn}$
```

$$\mathbf{p}_i^n = \begin{cases} \mathbf{p}_{ob} + \tau\mathbf{u}, & \text{if } ob = ob^* \\ \mathbf{p}_{ob^*} + \tau\mathbf{u}, & \text{otherwise}, \end{cases} \tag{12}$$

where $\tau$ is the extension factor, $\mathbf{p}_{ob^*}$ is the position of the connected obstacle $ob^*$ if it exists, and $\mathbf{u}$ indicates the minimum cost of updating $L_{cn}$ so that it does not intersect with the obstacle. Specifically, the $ob^*$ connects only one obstacle and is closest to the segment $L_{cn}$, and $\mathbf{u}$ can be expressed as

$$\mathbf{u} = \begin{cases} (r_i + r_{ob})(\mathbf{n} \mid \underset{\mathbf{n} \perp L_{cn}}{\arg\min}\; d(\mathbf{p}_{ob} + \mathbf{n}, L_{cn})), & \text{if } ob = ob^* \\ (r_i + r_{ob^*})(\mathbf{n} \mid \underset{\mathbf{n} \perp L_{cn}}{\arg\max} \sum_{k=1}^{N} \|\mathbf{p}_{ob^k} - \mathbf{p}_c\|), & \text{otherwise}, \end{cases} \tag{13}$$

where $\mathbf{n}$ is the unit normal vector of $L_{cn}$, $d(\mathbf{p}_{ob} + \mathbf{n}, L_{cn})$ is the distance from point $\mathbf{p}_{ob} + \mathbf{n}$ to the line passing through $L_{cn}$, $\mathbf{p}_c = \mathbf{p}_{ob^*} + (r_i + r_{ob^*})\mathbf{n}$ indicates the potential guidance point, and $\|\mathbf{p}_{ob^k} - \mathbf{p}_c\|$ denotes the distance from $\mathbf{p}_{ob^k}$ to $\mathbf{p}_c$. Now, we need to consider how to calculate $\tau$. Specifically, if the obstacle $ob$ belongs to the connected one, $\tau \geq 1$. Otherwise $\tau = 1$. Note that $\tau \geq 1$ ensures that the potential guidance points are not in the area of other obstacles. Therefore, we can extend $\tau$ incrementally by the radius $r_i$ of vehicle $i$ until $\mathbf{p}_i^n$ is placed in the feasible region outside these connected obstacles. Since the number of connected obstacles is smaller than the number of neighbors perceivable by vehicle $i$, we can always obtain the value of $\tau$ within constant iterations.

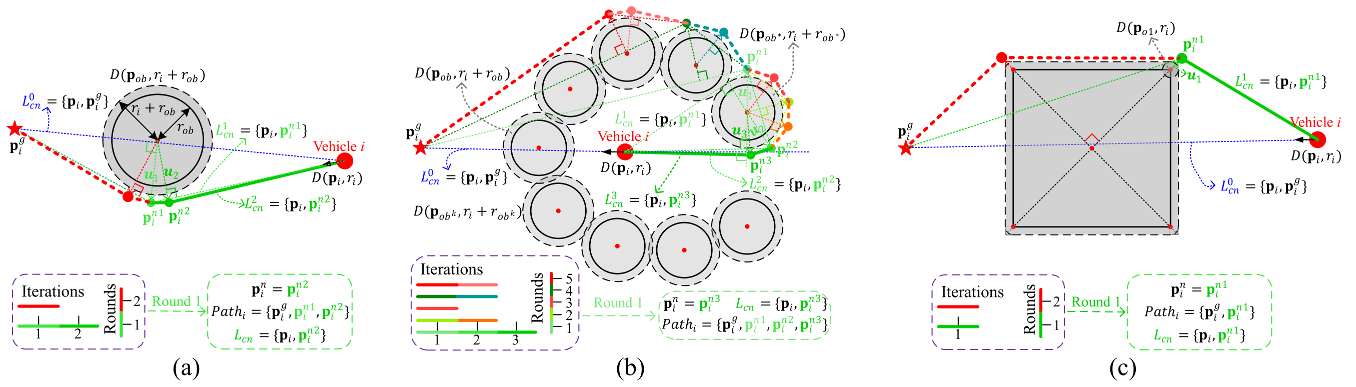

$D(\mathbf{p}_{ob}, r_i + r_{ob})$
$L_{cn}^0 = \{\mathbf{p}_i, \mathbf{p}_i^g\}$
$L_{cn}^1 = \{\mathbf{p}_i, \mathbf{p}_i^{n1}\}$
$L_{cn}^2 = \{\mathbf{p}_i, \mathbf{p}_i^{n2}\}$
Vehicle $i$
$D(\mathbf{p}_i, r_i)$
Iterations
Rounds
Round 1
$\mathbf{p}_i^n = \mathbf{p}_i^{n2}$
$Path_i = \{\mathbf{p}_i^g, \mathbf{p}_i^{n1}, \mathbf{p}_i^{n2}\}$
$L_{cn} = \{\mathbf{p}_i, \mathbf{p}_i^{n2}\}$
(a)
$D(\mathbf{p}_{ob^*}, r_i + r_{ob^*})$
$D(\mathbf{p}_{ob^k}, r_i + r_{ob^k})$
$L_{cn}^3 = \{\mathbf{p}_i, \mathbf{p}_i^{n3}\}$
$\mathbf{p}_i^n = \mathbf{p}_i^{n3}$
$L_{cn} = \{\mathbf{p}_i, \mathbf{p}_i^{n3}\}$
$Path_i = \{\mathbf{p}_i^g, \mathbf{p}_i^{n1}, \mathbf{p}_i^{n2}, \mathbf{p}_i^{n3}\}$
(b)
$D(\mathbf{p}_{o1}, r_i)$
$\mathbf{p}_i^n = \mathbf{p}_i^{n1}$
$Path_i = \{\mathbf{p}_i^g, \mathbf{p}_i^{n1}\}$
$L_{cn} = \{\mathbf{p}_i, \mathbf{p}_i^{n1}\}$
(c)

Fig. 3. The GOS's geometric illustrations, where the numerical superscripts and subscripts indicate the number of iterations in the first round. (a) In the separated circular obstacle situation. (b) In multiple connected circular obstacles. (c) In the polygon obstacle situation.

If the obstacle $ob$ belongs to the separated circular one, the effective $\mathbf{p}_i^n$ can always be solved Eqn. (13), updating the $L_{cn}$ of vehicle $i$ to the collision-free subpath, as shown in Fig. 3(a). In addition, when the obstacle $ob$ belongs to the connected one, if vehicle $i$ wants to avoid all these connected obstacles, $\mathbf{p}_i^n$ must be outside these connected obstacles in geometry. It can be expressed mathematically as $\sum_{k=1}^{N} \|\mathbf{p}_{ob^k} - \mathbf{p}_i^n\| > \sum_{k=1}^{N} \|\mathbf{p}_{ob^k} - \mathbf{p}_i^{\text{candidate}}\|$, where $\mathbf{p}_i^{\text{candidate}}$ is another candidate guidance point surrounded by these connected obstacles but in the feasible region, as shown in Fig. 3(b). This shows that Eqn. (13) can still find an effective $\mathbf{p}_i^n$, such that the $L_{cn}$ updated by the $\mathbf{p}_i^n$ is a collision-free sub-path. We summarize the above implementation of solving $\mathbf{p}_i^n$ in circular obstacle situations in lines 4-10 of Algorithm 2. Notably, line 6 constructs connected obstacles' spatial topological relationship, putting all the connected obstacles in a set and calculating how many connected obstacles each.

Then, we consider that a polygon obstacle intersects with the sub-path $L_{cn}$ and is closest to vehicle $i$ among all intersected obstacles, as shown in Fig. 3(c). In this case, the GOS first inflates all the convex vertices of the obstacle to the vertices of the Minkowski sum of the vehicle $i$ and the obstacle. Notably, the Minkowski sum can be achieved in constant time complexity under *Condition 1*. Let the $V_{\text{convex}}$ be the set of inflated convex vertices, and the $\mathbf{p}_{vt}$ be a vertex from $V_{\text{convex}}$. In addition, we define the $V_{opt}$ ($V_{opt} \subset V_{\text{convex}}$) as the vertices set, where any vertex $\mathbf{p}_{vt}$ from $V_{opt}$ satisfies that both line segments $L1 = \{\mathbf{p}_i, \mathbf{p}_{vt}\}$ and $L2 = \left\{\mathbf{p}_{vt}, \mathbf{p}_i^g\right\}$ do not intersect with the obstacle. At the same time, we denote the $V_{\text{subopt}}$ ($V_{\text{subopt}} \subset V_{\text{convex}}$) as another set, where any vertex $\mathbf{p}_{vt}$ from $V_{\text{subopt}}$ satisfies that the line segment $L = \{\mathbf{p}_i, \mathbf{p}_{vt}\}$ do not intersect with the obstacle. Thus, we can solve the guidance point $\mathbf{p}_i^n$ with Eqn. (14), where $d(\mathbf{p}_{vt}, L_{cn})$ is the distance from point $\mathbf{p}_{vt}$ to the line passing through $L_{cn}$.

$$\mathbf{p}_i^n = \begin{cases} \arg\min\limits_{\mathbf{p}_{vt} \in V_{opt}} d(\mathbf{p}_{vt}, L_{cn}), & \text{if } V_{opt} \neq \emptyset \\ \arg\min\limits_{\mathbf{p}_{vt} \in V_{\text{subopt}}} d(\mathbf{p}_{vt}, L_{cn}), & \text{otherwise} . \end{cases} \tag{14}$$

According to the conditions of $V_{opt}$, we can obtain the $\mathbf{p}_i^n$ only through one iteration if $V_{opt}$ is not empty because the $L_{cn}$ updated with $\mathbf{p}_i^n$ has avoided the obstacle. Otherwise, we achieve $\mathbf{p}_i^n$ from the $V_{\text{subopt}}$. It should be noted that $V_{\text{subopt}}$ is a non-empty set because there must be a vertex in $V_{\text{convex}}$ that satisfies the condition of $V_{\text{subopt}}$. This shows that Eqn. (14) can always find an effective $\mathbf{p}_i^n$. Lines 12-18 presented in Algorithm 2 summarize the implementation of solving $\mathbf{p}_i^n$ in polygon obstacle situations.

## D. Analysis for GOS

We first analyze the completeness of the GOS. Under *Condition 1*, stated in Section IV-C, if a collision-free path exists between the $\mathbf{p}_i$ and $\mathbf{p}_i^g$ for vehicle $i$, the GOS will definitely find a collision-free path.

To demonstrate the completeness of the GOS, we assume the existence of collision-free paths between $\mathbf{p}_i$ and $\mathbf{p}_i^g$ of vehicle $i$. Therefore, it is evident that both $\mathbf{p}_i$ and $\mathbf{p}_i^g$ must fall within the feasible region, as shown in Fig. 3. In the description of how the GOS works in Section IV-C, we know that the guidance point $\mathbf{p}_i^n$ found in each iteration is always valid, which means there is always a collision-free sub-path $L_{cn}$ between $\mathbf{p}_i$ and $\mathbf{p}_i^n$. According to the Algorithm 2, the GOS will check whether the $L_{cn}$ intersects with obstacles in each iteration. If an intersection exists, repeat using the GOS method for updating $L_{cn}$ until a non-intersecting $L_{cn}$ is obtained. Thus, vehicle $i$ travels along $L_{cn}$ and updates $L_{cn}$ when it arrives $\mathbf{p}_i^n$. The vehicle $i$ can always reach the goal position $\mathbf{p}_i^g$ without collision by moving along each $L_{cn}$. Therefore, the path composed of a series of $L_{cn}$ must be a collision-free path from the $\mathbf{p}_i$ to the $\mathbf{p}_i^g$. Therefore, if it exists, the GOS can always find the collision-free path between $\mathbf{p}_i$ and $\mathbf{p}_i^g$. In addition, since GOS prioritizes avoiding the obstacle closest to the vehicle, it guarantees that the collision-free path obtained by avoiding the closest obstacle is locally optimal.

Then, we analyze the time complexity of the GOS. According to the details described in Algorithm 2, we can derive the time complexity of solving the guidance point $\mathbf{p}_i^n$ is $O(r)$, where $r$ is the number of obstacles perceived by the vehicle. Since the GOS can always compute the collision-free sub-path $L_{cn}$ in constant times, the overall time complexity of GOS remains $O(r)$. GOS is a lightweight path generation algorithm that can replace the global D* planner in most common outdoor environments to generate feasible collision-free sub-paths more quickly, ensuring real-time system performance while possessing the ability of the D* to handle unknown obstacles.

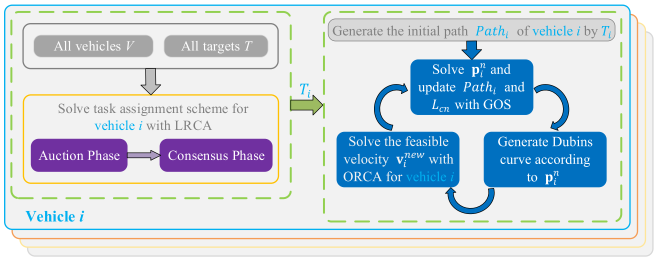

All vehicles $V$
All targets $T$
Solve task assignment scheme for vehicle $i$ with LRCA
Auction Phase
Consensus Phase
$T_i$
Generate the initial path $Path_i$ of vehicle $i$ by $T_i$
Solve $\mathbf{p}_i^n$ and update $Path_i$ and $L_{cn}$ with GOS
Solve the feasible velocity $\mathbf{v}_i^{new}$ with ORCA for vehicle $i$
Generate Dubins curve according to $\mathbf{p}_i^n$
Vehicle $i$

Fig. 4. The structure of the proposed hierarchical multi-vehicle task assignment and motion planning method LRGO.

Therefore, we compare the advantages of GOS by analyzing the time complexity of the D*. According to the reference [44], the time complexity for solving the initial global path can be optimized to $O(e + vlogv)$, where $e$ is the number of edges connecting adjacent grid cells and $v$ is the number of grid cells. For the D* to be effective, the size of grid cells must be smaller than that of the obstacle or the vehicle. Therefore, the $v$ is more significant than $r$, resulting in $O(e+vlogv) \gg O(r)$. Although the D* planner can locally adjust the generated path to avoid the newly emerged obstacles with a time complexity comparable to the GOS, it needs to recalculate the path when the goal position is changed. Therefore, the time complexity remains $O(e + vlogv)$, making it challenging to guarantee real-time performance throughout visiting multiple targets. On the contrary, the GOS can quickly provide a feasible collision avoidance sub-path for the vehicle, ensuring the real-time performance of the systems.

### E. The Proposed Comprehensive Method

Based on the proposed LRCA and GOS methods, we proposed our hierarchical method named LRGO that integrates them with the ORCA method and the improved Dubins curve to simultaneously address the task assignment and motion planning for non-holonomic vehicles. We present the structure of the LRGO in Fig. 4. First, each vehicle $i$ solves the task assignment scheme $T_i$ using the LRCA, resulting in its initial path $Path_i$, like in Fig. 4. Note that all vehicles communicate with each other, share their initial position, and know all the target positions. Then, according to its current and next visit positions, each vehicle updates its path with the GOS to avoid static obstacles. At the same time, we introduce the Dubins curve [9] for meeting the kinematic constraints of non-holonomic vehicles. Now, we describe the process of introducing the Dubins curve as follows. Firstly, we obtain the current pose and the next terminal pose of vehicle $i$ by $\mathbf{q}_i = (\mathbf{p}_i, \theta(\mathbf{v}_i))$, and $\mathbf{q}_i^n = \left(\mathbf{p}_i^n, \theta(\mathbf{p}_i^n - \mathbf{p}_i)\right)$, respectively, where $\mathbf{v}_i$ is its current velocity, and $\theta(\cdot)$ is the angle of a vector in the coordinate system. Then, the Dubins path connecting the two poses can be achieved if the starting circular arc does not intersect with the static obstacles in this environment. Otherwise, we choose another sub-optimal turning arc to generate the Dubins path for collision avoidance with static obstacles if it exists. We use the ORCA algorithm to achieve dynamic collision avoidance after achieving the current path that satisfies the non-holonomic vehicle kinematic constraints. Repeat the above steps until the vehicle has visited all its targets and reached the specific ending area.

## V. Experimental Results

This section demonstrates the advantages of our method in large-scale scenarios by conducting simulations and real-world experiments on an ASUS desktop equipped with the Intel(R) Core(TM) i7-8700 CPU with 16 GB memory. Here, we first conducted a simulation to verify the effectiveness of our method. Then, we combined the state-of-the-art (SOTA) task assignment methods, including the CBBA [25], the DSTA [27], and the LSTA [28], with our motion planning method and compared them with the proposed method, demonstrating the advantages of our method. All methods in this article are implemented by using Python. Note that the CBBA algorithm is from the open-source[1] repository. Since there are no open-source DSTA and LSTA algorithms, we implement them by referring to the papers [27], [28]. Finally, we also performed two real-world experiments to verify the validity and application potential in physical environments.

We consider a potential application scenario where a vehicle fleet needs to visit a series of threat targets to clear them and move to a specific ending area. Each threat target is cleared when its vehicle traverses its position in an allowable distance deviation. Note that each vehicle cannot collide with any obstacle and cannot traverse into other threat target areas assigned to the other vehicle. In contrast, each vehicle will no longer regard the cleared threat target's area as forbidden. We set the parameters of the proposed algorithm as follows for simulations. Let the max sensing range be $D_{\max} = 1000\ m$, let the maximum number of neighbors for each vehicle be $N_{\max} = 20$, and let the time horizon in the ORCA be $\tau = 10$. In addition, six metrics are given for the performance evaluation.

- *Total Rewards (TR)*: the overall objective function values for the rewards of task assignment.
- *Total Travel Distance (TTD)*: the actual total traveling distance of all vehicles.
- *Maximum Angular Speed (MAS)*: the maximum heading angle rate among all vehicles during moving.
- *Task Accomplishing Rate (TAR)*: the ratio of the number of cleared threat targets by vehicles to the total number of threat targets under a specific time limit.
- *Task Assignment Computation Cost (TAC)*: the computation time cost for solving a task assignment scheme.
- *Average Computation Cost (ACC)*: the average computation time cost for solving a new velocity in each step.

### A. Simulation Experiments

In this part, extensive simulations are performed to verify the effectiveness and advantages of our method. To validate the scalability of our algorithm, we set three groups of parameters for heterogeneous non-holonomic vehicles that have different sizes and maximum velocities as follows. The first group parameters of vehicles are: the radius is $r = 5\ m$, the maximum number of clearing targets is $L = 8$, the preferred speed is $v^{\text{pref}} = 6\ m/s$, the maximum linear speed is $v^{\max} = 8\ m/s$, the maximum angular speed is $\omega^{\max} = \frac{\pi}{6}\ rad/s$, and the minimum turning radius is $\rho_{min} = \frac{v^{\max}}{\omega^{\max}}$. Similarly, the other two groups'

[1]https://github.com/keep9oing/consensus-based-bundle-algorithm

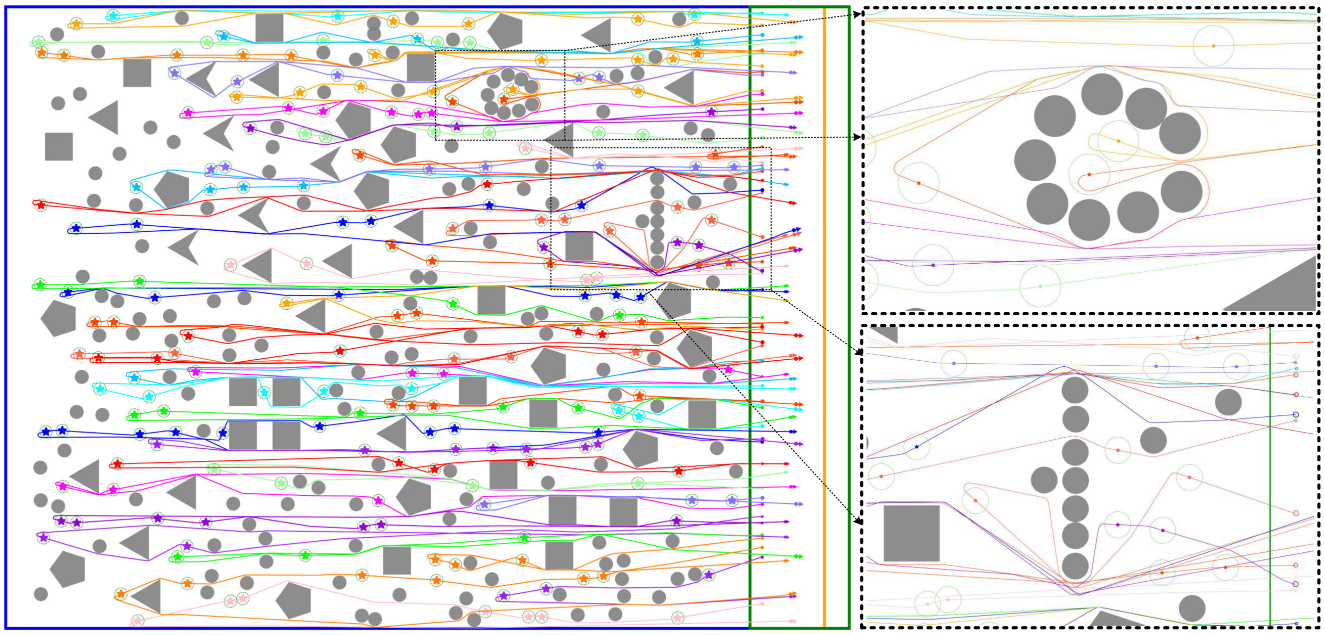


Fig. 5. All vehicles' trajectories cleared targets are presented. Note that the orange square, the blue square, and the green square indicate the starting area, the task area, and the ending area, respectively.

parameters are set as $r = 8\ m$, $L = 10$, $v^{\text{pref}} = 6\ m/s$, $v^{\max} = 9\ m/s$, $\omega^{\max} = \frac{\pi}{6}$ rad/s, and $\rho_{min} = \frac{v^{\max}}{\omega^{\max}}$, as well as $r = 10\ m$, $L = 12$, $v^{\text{pref}} = 6\ m/s$, $v^{\max} = 10\ m/s$, $\omega^{\max} = \frac{\pi}{6}\ rad/s$, and $\rho_{min} = \frac{v^{\max}}{\omega^{\max}}$, respectively. Here, each vehicle's configurations are determined randomly from the three groups. Each threat target is cleared when its vehicle traverses its position within a distance deviation of 3 $m$. Meanwhile, once the vehicle collides with other objects in the environment, it no longer moves.

In the first simulation, 50 heterogeneous non-holonomic vehicles are placed randomly in a starting area with 0.6 $km \times 5.0\ km$. They are required to clear 203 threat targets with a radius of 50 $m$ located randomly in the task area with 5.8 $km \times 5.0\ km$, which is a dense environment with rich static obstacles, including circular obstacles with a radius of 50 $m$ and polygon obstacles with a larger area. Each vehicle must move to the pointed ending area with 0.8 $km \times 5.0\ km$ once it completes all its missions. Note that the ending area of vehicles completely overlaps with the starting area in this simulation. The simulation results are shown in Fig. 5 and Table I. All vehicles' trajectories are shown in detail in Fig. 5. By inspecting Fig. 5, we can observe that the deployed vehicles successfully cleared all threat targets. Each vehicle overcomes the sharp turn caused by the switching target and plans its U-turn action after clearing its assigned threat targets. Additionally, Table I presents the metrics results achieved by our method and the combined LSTA method, a benchmark in the same scenario for comparison. In Table I, we can find that our method obtains higher quality than the combined LSTA method, except for the computational time in task assignment. Especially, our method reduces the total traveling distance by over 14 $km$, significantly saving energy consumption. Although our method is not as fast as the combined LSTA method in solving task assignments, it still

TABLE I
COMPARED THE PROPOSED METHOD WITH THE BENCHMARK METHODS IN SIX METRICS. *TTD*: *km*, *MAS*: *rad/S*, *TAR*: %, *TAC*: *s*, *ACC*: *ms*

| Methods | *TR* | *TTD* | *MAS* | *TAR* | *TAC* | *ACC* |
|---|---|---|---|---|---|---|
| LRGO (Ours) | **174.893** | **433.95** | $\pi/6$ | 100.0 | 0.87 | 2.78 |
| Combined LSTA | 174.889 | 448.13 | $\pi/6$ | 100.0 | **0.46** | 2.78 |

meets the real-time performance of multi-vehicle systems. In addition, the actual angular speed during the traveling of all vehicles is less than their maximum angular speed, and the planning method proposed in this paper can calculate a new velocity within 3 $ms$. To sum up, these results presented in Fig. 5 and Table I verified the effectiveness and practicability of our method in a dense environment.

To achieve the quantitative performance evaluation of our algorithm, we compared our method with the combined CBBA, the combined DSTA, and the combined LSTA methods in large-scale scenarios. Note that we do not consider the combination of the ORCA and our LRCA because using the ORCA alone in a dense environment always makes vehicles unable to avoid deadlocks. In addition, we do not compare our method with the combined D* [44] due to its time complexity analyzed in Section IV-D, which can not meet the real-time performance of the multi-vehicle system. The configurations of the comparison experiments are the same as the first simulation, except that the starting and ending areas are at the task areas' two ends. The positions of vehicles, threat targets, and obstacles are randomly generated, where the threat target and the obstacle can invade each other for 5 $m$. Then, we conduct 16 group simulations, and each group simulation is repeated ten times independently, in which the number $M$ of vehicles is set from 10 to 160. Meanwhile, the number $N$

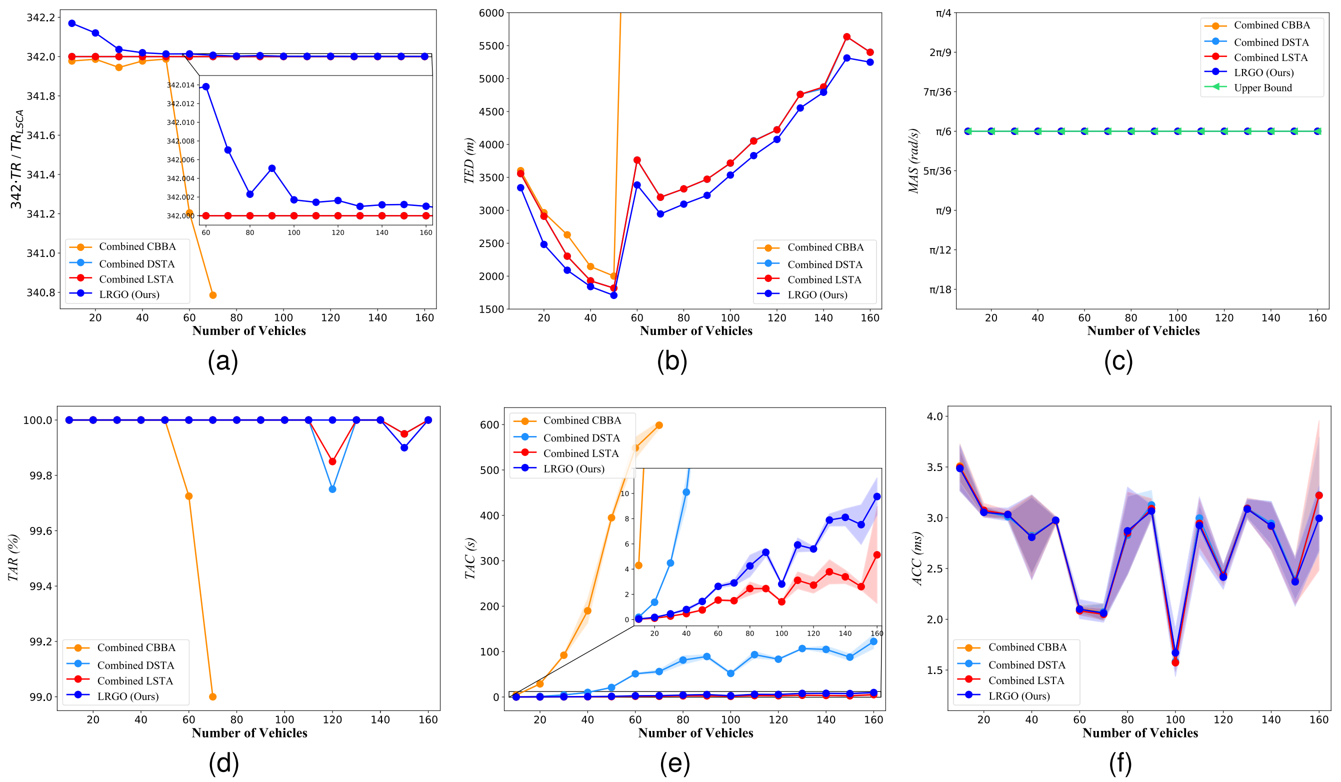


Fig. 6. (a) The normalized total rewards value for targets assignment. (b) The extra traveling distance for all vehicles to clear targets. (c) The actual angular speed during the traveling of all vehicles and the maximum angular speed. (d) The success rate of clearing threat targets. (e) The computation cost of solving task assignment solution. (f) The computational time for calculating a velocity solution.

of threat targets is set by $N = max(5 \times M, 400)$, and the number of obstacles is 200, including circular and polygon obstacles. Note that the maximum number of threat targets is set due to the limited task area. In addition, if the computation time for solving a task assignment scheme is more than 600 $s$, we think solving the task assignment is a failure. For a clear comparison, we adjust the metric $TR$ to a normalized metric $342 \cdot TR/TR_{\mathrm{LSTA}}$ with the mean rewards of $N = 400$ and the metric $TTD$ to $TED$ to indicate the difference between the actual total traveling distance and the straight line distance to the ending area of all vehicles.

Fig. 6 shows the results, where we present the average values of these six metrics and the standard deviations of $TAC$ and $ACC$. From Fig. 6(a), it can be observed that our method shows the best performance. Note that the approaching the normalized $TR$ metrics are due to diminishing marginal gains. However, the benefits of a better allocation scheme are manifested in terms of saving overall travel distance and improving the completion rate of tasks, as demonstrated in Fig. 6(b) and (d), respectively, which showcases the superiority of our LRGO. However, our method has a lower rate of clearing targets when the vehicle's number is 150 in Fig. 6(d). It should be because the vehicle collides with a static obstacle to avoid other vehicles when they traverse the same narrow channel formed by multiple obstacles. A similar situation also occurs in the combined LSTA and DSTA methods when the number of vehicles is 120. In Fig. 6(c), we can observe that the actual angular speed during the traveling of all vehicles is less than its maximum, meeting the non-holonomic vehicles' kinematic constraints. From Fig. 6(e), it can be observed that the CBBA can not solve an assignment scheme within 600 $s$ when the vehicles' number is more than 70. The LSTA achieves the best performance in terms of computation cost for solving task assignment solutions. This is because our method considers more reasonable conflict resolution, resulting in more computation time than the LSTA. Despite this, our algorithm can solve closer to the optimal allocation scheme quickly. Furthermore, in Fig. 6(f), it can be found that although in dense large-scale scenarios, our method can still calculate the feasible velocity for the vehicle within only 4 $ms$, meeting the real-time requirements of almost all multi-vehicle systems. To sum up, the comparisons with these SOTA methods show the advantages of the method proposed in this paper.

## B. Real-World Experiments

To verify the practicality of our method, we conducted two real-world experiments on a team of unmanned ground vehicles (UGVs) to show its potential. We conducted the experiments on Ubuntu 18.04 with ROS Melodic. The configurations are as follows. The workspace is rectangular with a square 6.8 $m \times 4.5$ $m$ where all the projecting walls are regarded as obstacles. Each vehicle is regarded as a disc whose radius is 0.18 $m$ according to its size. The vehicle's minimum turning radius is 0.35 $m$, obtained by the kinematic model in which the actual angular speed does not exceed $\frac{\pi}{4}$ $rad/s$. All vehicles are connected via the WiFi network provided by a router. In addition, the radius of each threat target and static circular obstacle is set as 0.18 $m$. We believe that a vehicle

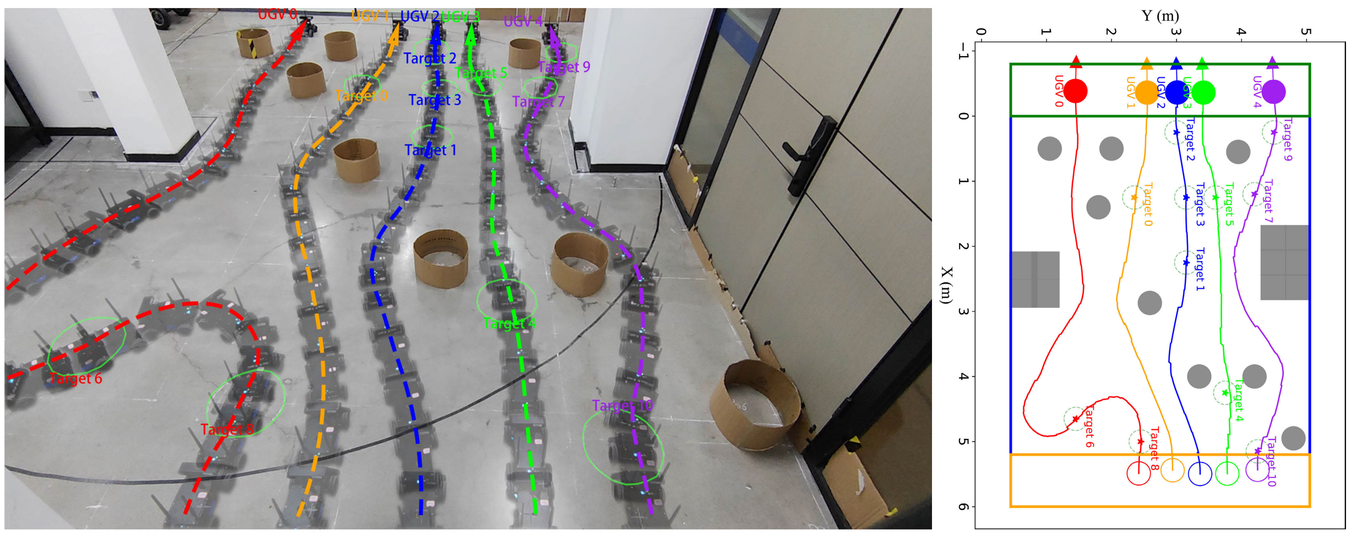


Fig. 7. All vehicles' trajectories of the first experiment in the real world are presented. *TR*: 10.998, *TTD*: 33.3 *m*, *MAS*: 0.59 *rad/s*, *TAR*: 100%, *TAC*: 3.13 *ms*, *ACC*: 2.58 *ms*.

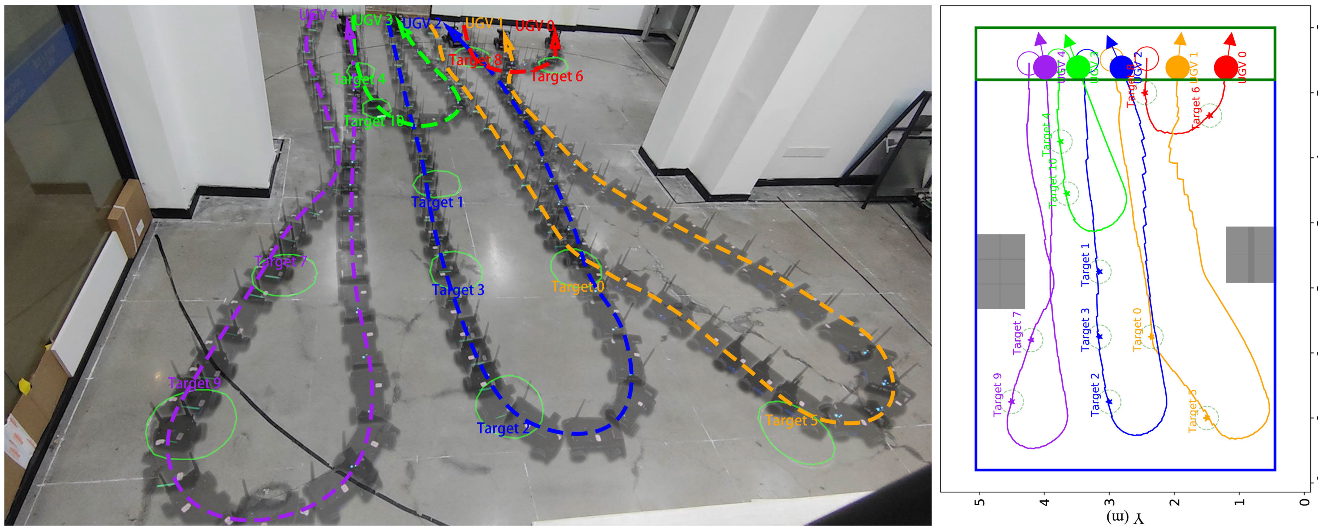


Fig. 8. All vehicles' trajectories of the second experiment in the real world are presented. *TR*: 10.998, *TTD*: 47.1 *m*, *MAS*: 0.671 *rad/s*, *TAR*: 100%, *TAC*: 2.99 *ms*, *ACC*: 2.34 *ms*.

successfully clears its target when the distance of positions between the vehicle and its target is less than 10 *cm*. Finally, we build the map of the workspace with the Cartographer algorithm [45], and each vehicle locates its current position using the AMCL algorithm, meanwhile broadcasting its position to its neighbors. Note that the vehicles' error within 10 *cm* for locating is allowed in the two experiments.

In the first experiment, five vehicles and eleven threat targets are deployed on the workspace, where the starting and ending areas are at the two ends of the task area, respectively. Meanwhile, we place eight static circular obstacles on the ideal path of vehicles to verify the proposed method's collision avoidance performance in the real world. Fig. 7 presents the details of the experimental results. In Fig. 7, the indexing color of each target is the same as that of the vehicle assigned to the target. At the same time, we mark all cleared targets by using green circles. From Fig. 7, we can observe that all threat targets are successfully cleared, where when the two adjacent path nodes and the vehicles' heading occur at acute angles, the vehicle cleans the next target through smooth turns, such as UGV 0. In addition, by inspection of the six above metrics, it can be observed that all vehicles' angular speed is less than their maximum. At the same time, it should be noted that our method generates an optimal task assignment scheme due to the experiment being a small-scale instance. These results shown in Fig. 7 demonstrate the effectiveness and robustness of the proposed method.

We conducted the second experiment by deploying five vehicles and eleven threat targets on the workspace, where the ending area of vehicles completely overlaps with the starting area. In this scenario, each vehicle will return to the starting area once it clears all assigned targets. Note that all vehicles do not consider the cleared target an obstacle. The experimental results are shown in Fig. 8. Similar to the first experiment, Fig. 8 presents all vehicles' trajectories, where the index color of each target is the same as its vehicles. Each cleared target is marked by using a green circle. In Fig. 8, we can find that all vehicles successfully cleared all targets and returned to the starting area under the vehicle's constraints. In addition, by inspection of the metric of actual maximum angular speed, it can be observed that all vehicles' angular speed is less than their maximum, demonstrating that the minimum turning radius is satisfied. Meanwhile, the other metrics are the optimal ones. The experiment results also show our method's validity and robustness again.

## VI. Conclusion

In this article, we investigated the multi-vehicle task assignment and motion planning (MVTAMP) problem for a fleet of non-holonomic vehicles, where these vehicles are required to visit a series of targets and move to a specific ending area. Considering the kinematic constraints of the non-holonomic vehicles, we proposed a novel hierarchical method for simultaneously solving the multiple vehicle task assignment and motion planning in a large-scale dense environment. We first conducted a simulation in large-scale scenarios to demonstrate the effectiveness of our method. The results of the simulation show that the proposed method achieves the best performance compared with its benchmark method. In addition, extensive simulation results compared with the SOTA methods show that the proposed method achieves the best performance in almost all cases, especially the quality of task assignment and the total travel distance of all vehicles. Finally, the results of two real-world experiments demonstrated the potential of the proposed method in real-world applications. In the future, we will mainly overcome the collision situation in that multiple vehicles pass through the same narrow channel to guarantee that the fleet vehicles can complete all tasks even in a complex dynamic environment.

## References

[1] J. Scherer and B. Rinner, "Multi-robot persistent surveillance with connectivity constraints," *IEEE Access*, vol. 8, pp. 15093–15109, 2020.

[2] Y. Cui, W. Dong, D. Hu, and H. Liu, "The application of improved harmony search algorithm to multi-UAV task assignment," *Electronics*, vol. 11, no. 8, p. 1171, Apr. 2022.

[3] J. Turner, Q. Meng, G. Schaefer, A. Whitbrook, and A. Soltoggio, "Distributed task rescheduling with time constraints for the optimization of total task allocations in a multirobot system," *IEEE Trans. Cybern.*, vol. 48, no. 9, pp. 2583–2597, Sep. 2018.

[4] J. Wu, C. Song, J. Ma, J. Wu, and G. Han, "Reinforcement learning and particle swarm optimization supporting real-time rescue assignments for multiple autonomous underwater vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6807–6820, Jul. 2022.

[5] R. Herguedas, G. López-Nicolás, and C. Sagüés, "Multirobot transport of deformable objects with collision avoidance," *IEEE Syst. J.*, vol. 17, no. 2, pp. 3224–3234, Jun. 2023.

[6] L. Zhang, Y. Sun, A. Barth, and O. Ma, "Decentralized control of multi-robot system in cooperative object transportation using deep reinforcement learning," *IEEE Access*, vol. 8, pp. 184109–184119, 2020.

[7] Z. Liu, H. Wang, H. Wei, M. Liu, and Y.-H. Liu, "Prediction, planning, and coordination of thousand-warehousing-robot networks with motion and communication uncertainties," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 4, pp. 1705–1717, Oct. 2021.

[8] N. Mathew, S. L. Smith, and S. L. Waslander, "Planning paths for package delivery in heterogeneous multirobot teams," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 4, pp. 1298–1308, Oct. 2015.

[9] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *Amer. J. Math.*, vol. 79, no. 3, p. 497, Jul. 1957.

[10] L. Antonyshyn, J. Silveira, S. Givigi, and J. Marshall, "Multiple mobile robot task and motion planning: A survey," *ACM Comput. Surv.*, vol. 55, no. 10, pp. 1–35, Oct. 2023.

[11] S. Chopra, G. Notarstefano, M. Rice, and M. Egerstedt, "A distributed version of the Hungarian method for multirobot assignment," *IEEE Trans. Robot.*, vol. 33, no. 4, pp. 932–947, Aug. 2017.

[12] Y. Emam, S. Mayya, G. Notomista, A. Bohannon, and M. Egerstedt, "Adaptive task allocation for heterogeneous multi-robot teams with evolving and unknown robot capabilities," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 7719–7725.

[13] X. Bai, A. Fielbaum, M. Kronmuller, L. Knoedler, and J. Alonso-Mora, "Group-based distributed auction algorithms for multi-robot task assignment," *IEEE Trans. Autom. Sci. Eng.*, vol. 20, no. 2, pp. 1292–1303, Apr. 2023.

[14] Z. Liu, H. Wei, H. Wang, H. Li, and H. Wang, "Integrated task allocation and path coordination for large-scale robot networks with uncertainties," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 2750–2761, Oct. 2022.

[15] A. Camisa, A. Testa, and G. Notarstefano, "Multi-robot pickup and delivery via distributed resource allocation," *IEEE Trans. Robot.*, vol. 39, no. 2, pp. 1106–1118, Apr. 2023.

[16] Y. Gottlieb, J. G. Manathara, and T. Shima, "Multi-target motion planning amidst obstacles for autonomous aerial and ground vehicles," *J. Intell. Robotic Syst.*, vol. 90, nos. 3–4, pp. 515–536, Jun. 2018.

[17] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics Research*. Berlin, Germany: Springer, 2011, pp. 3–19.

[18] H. W. Kuhn, "The Hungarian method for the assignment problem," *Nav. Res. logistics Quart.*, vol. 2, nos. 1–2, pp. 83–97, 1955.

[19] C. Wei, Z. Ji, and B. Cai, "Particle swarm optimization for cooperative multi-robot task allocation: A multi-objective approach," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2530–2537, Apr. 2020.

[20] J. Schwarzrock, I. Zacarias, A. L. C. Bazzan, R. Q. D. A. Fernandes, L. H. Moreira, and E. P. de Freitas, "Solving task allocation problem in multi unmanned aerial vehicles systems using swarm intelligence," *Eng. Appl. Artif. Intell.*, vol. 72, pp. 10–20, Jun. 2018.

[21] X. Bai, W. Yan, S. S. Ge, and M. Cao, "An integrated multi-population genetic algorithm for multi-vehicle task assignment in a drift field," *Inf. Sci.*, vol. 453, pp. 227–238, Jul. 2018.

[22] R. Zlot and A. Stentz, "Market-based multirobot coordination for complex tasks," *Int. J. Robot. Res.*, vol. 25, no. 1, pp. 73–101, Jan. 2006.

[23] E. Nunes and M. Gini, "Multi-robot auctions for allocation of tasks with temporal constraints," in *Proc. AAAI Conf. Artif. Intell.*, 2015, vol. 29, no. 1, pp. 1–7.

[24] M. Otte, M. J. Kuhlman, and D. Sofge, "Auctions for multi-robot task allocation in communication limited environments," *Auto. Robots*, vol. 44, nos. 3–4, pp. 547–584, Mar. 2020.

[25] H.-L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Trans. Robot.*, vol. 25, no. 4, pp. 912–926, Aug. 2009.

[26] L. Luo, N. Chakraborty, and K. Sycara, "Distributed algorithm design for multi-robot task assignment with deadlines for tasks," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 3007–3013.

[27] H.-S. Shin, T. Li, H.-I. Lee, and A. Tsourdos, "Sample greedy based task allocation for multiple robot systems," *Swarm Intell.*, vol. 16, no. 3, pp. 233–260, Sep. 2022.

[28] T. Li, H.-S. Shin, and A. Tsourdos, "Efficient decentralized task allocation for UAV swarms in multi-target surveillance missions," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2019, pp. 61–68.

[29] O. Shorinwa, R. N. Haksar, P. Washington, and M. Schwager, "Distributed multirobot task assignment via consensus ADMM," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 1781–1800, Jun. 2023.

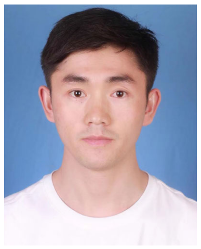
[30] G. Mateos, J. A. Bazerque, and G. B. Giannakis, "Distributed sparse linear regression," *IEEE Trans. Signal Process.*, vol. 58, no. 10, pp. 5262–5276, Oct. 2010.

[31] J. Li, Y. Xiong, and J. She, "UAV path planning for target coverage task in dynamic environment," *IEEE Internet Things J.*, vol. 10, no. 20, pp. 17734–17745, Oct. 2023.

[32] F. Nawaz and M. Ornik, "Multi-agent, multi-target path planning in Markov decision processes," *IEEE Trans. Autom. Control*, early access, Jun. 16, 2023, doi: 10.1109/TAC.2023.3286807.

[33] H. Qie, D. Shi, T. Shen, X. Xu, Y. Li, and L. Wang, "Joint optimization of multi-UAV target assignment and path planning based on multi-agent reinforcement learning," *IEEE Access*, vol. 7, pp. 146264–146272, 2019.

[34] 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. Autom. Lett.*, vol. 6, no. 3, pp. 5816–5823, Jul. 2021.
[35] W. Cai, M. Zhang, and Y. Zheng, "Task assignment and path planning for multiple autonomous underwater vehicles using 3D Dubins curves," *Sensors*, vol. 17, no. 7, p. 1607, Jul. 2017.
[36] Z. Jia, J. Yu, X. Ai, X. Xu, and D. Yang, "Cooperative multiple task assignment problem with stochastic velocities and time windows for heterogeneous unmanned aerial vehicles using a genetic algorithm," *Aerosp. Sci. Technol.*, vol. 76, pp. 112–125, May 2018.
[37] Y. Zhao et al., "Cooperative multiple task assignment problem with target precedence constraints using a waitable path coordination and modified genetic algorithm," *IEEE Access*, vol. 9, pp. 39392–39410, 2021.
[38] X. Jin and M. J. Er, "Cooperative path planning with priority target assignment and collision avoidance guidance for rescue unmanned surface vehicles in a complex ocean environment," *Adv. Eng. Informat.*, vol. 52, Apr. 2022, Art. no. 101517.
[39] L. Babel, "Coordinated target assignment and UAV path planning with timing constraints," *J. Intell. Robotic Syst.*, vol. 94, nos. 3–4, pp. 857–869, Jun. 2019.
[40] W.-Y. Yu, X.-Q. Huang, H.-Y. Luo, V.-W. Soo, and Y.-L. Lee, "Auction-based consensus of autonomous vehicles for multi-target dynamic task allocation and path planning in an unknown obstacle environment," *Appl. Sci.*, vol. 11, no. 11, p. 5057, May 2021.
[41] Y. Du, "Multi-UAV search and rescue with enhanced A* algorithm path planning in 3D environment," *Int. J. Aerosp. Eng.*, vol. 2023, pp. 1–18, Feb. 2023.
[42] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, Jul. 1968.
[43] K. Okumura and X. Défago, "Solving simultaneous target assignment and path planning efficiently with time-independent execution," *Artif. Intell.*, vol. 321, Aug. 2023, Art. no. 103946.
[44] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 1994, pp. 3310–3317.
[45] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LiDAR SLAM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 1271–1278.



**Gang Xu** received the B.S. degree in electrical engineering and automation from Hangzhou Dianzi University, Hangzhou, China, in 2019, and the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, in 2020. His current research interests include multi-robot making-decision and planning.

**Xiao Kang** received the Ph.D. degree in mechatronic engineering from the Beijing Institute of Technology in 2013. She was a Senior Scientist with the China North Vehicle Research Institute in 2018. She is mainly engaged in unmanned system research. She has published more than 20 articles in intelligent perception, information fusion, and navigation and control of robotics.

**Helei Yang** (Graduate Student Member, IEEE) received the B.S. degree in automation from the Zhejiang University of Technology, Hangzhou, China, in 2020. He is currently pursuing the master's degree in electronics and information with the Polytechnic Institute, Zhejiang University, Hangzhou. His current research interests include deep reinforcement learning and robotics.

**Yuchen Wu** received the B.S. degree in aircraft air worthiness technology from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2022. He is currently pursuing the master's degree with the Polytechnic Institute, Zhejiang University. His research interests include trajectory planning and intelligent decision.

**Weiwei Liu** was born in Hefei, China. He received the Ph.D. degree in electronics and information from Zhejiang University, Hangzhou, China, in 2023. His main research interests include artificial intelligence, decision-making, and control.

**Junjie Cao** received the B.S. degree in mechanical engineering and automation from Nanjing Tech University, Nanjing, China, in 2014, and the M.S. degree in mechanical engineering (mechatronics) and the Ph.D. degree in control science and engineering from Zhejiang University, Zhejiang, China, in 2017 and 2021, respectively. His current research interests include machine learning, sequential decision making, and robotics.

**Yong Liu** received the B.S. degree in computer science and engineering and the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 2001 and 2007, respectively. He is currently a Professor with the Department of Control Science and Engineering, Institute of Cyber Systems and Control, Zhejiang University. He has published more than 90 research papers in machine learning, computer vision, information fusion, and robotics. His research interests include machine learning, robotics, vision, information processing, and granular computing.