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

Collaborative Planning for Catching and Transporting Objects in Unstructured Environments

Liuao Pei[1,2], Junxiao Lin[1,2], Zhichao Han[1,2], Lun Quan[1,2], Yanjun Cao[2], Chao Xu[1,2], and Fei Gao[1,2]

***Abstract*— Multi-robot teams have attracted attention from industry and academia for their ability to perform collaborative tasks in unstructured environments, such as wilderness rescue and collaborative transportation. In this paper, we propose a trajectory planning method for a non-holonomic robotic team with collaboration in unstructured environments. For the adaptive state collaboration of a robot team to catch and transport targets to be rescued using a net, we model the process of catching the falling target with a net in a continuous and differentiable form. This enables the robot team to fully exploit the kinematic potential, thereby adaptively catching the target in an appropriate state. Furthermore, the size safety and topological safety of the net, resulting from the collaborative support of the robots, are guaranteed through geometric constraints. We integrate our algorithm on a car-like robot team and test it in simulations and real-world experiments to validate our performance. Our method is compared to state-of-the-art multi-vehicle trajectory planning methods, demonstrating significant performance in efficiency and trajectory quality.**

I. Introduction

With the advancement of autonomous navigation technology for car-like robots, some scenes depicted in science fiction movies, such as adaptive and flexible collaborative rescue and transportation in complex environments, have become a realistic possibility [1, 2]. The key to realizing these tasks lies in achieving precise and computing-efficient collaborative planning for the car-like robotic team, which is a high-dimensional, strongly-coupled system with non-holonomic motion constraints. However, achieving both high precision and high efficiency can be a dilemma, remaining challenging to perform online adaptive and precise collaborative trajectory planning in unstructured environments.

For collaborative catching and transportation tasks, in addition to considering individual motion constraints and obstacle avoidance, precisely and efficiently modeling the collaborative relationships among robotic teams is of paramount importance. When the robot team supports a net to perform catching and transporting tasks, there are multiple complex relationships among the robot team, the net, and the target to be rescued. Firstly, when a rescue target falls, the net supported by the robot team needs to catch the target in a theoretically feasible state of the robot team, defined as *adaptive catching*. The state of team formation when catching the target should be adjusted according to the environment and the distance to the target. Secondly, The trajectories need to keep the physical size of the net within the range of fatigue and damage at every moment, defined as *size safety*. At any given moment, the distance between any two robots must not exceed the corresponding distance limit associated with the two vertices. Thirdly, the net edges must not be entangled by any agent, defined as *topological safety*. From the initial time to any subsequent time, a robot should not pass through an edge of the net supported by any two other robots. The implementation of these collaborative relationships places great demands on agents in terms of both time and space, thereby creating a high-dimensional and strongly non-convex

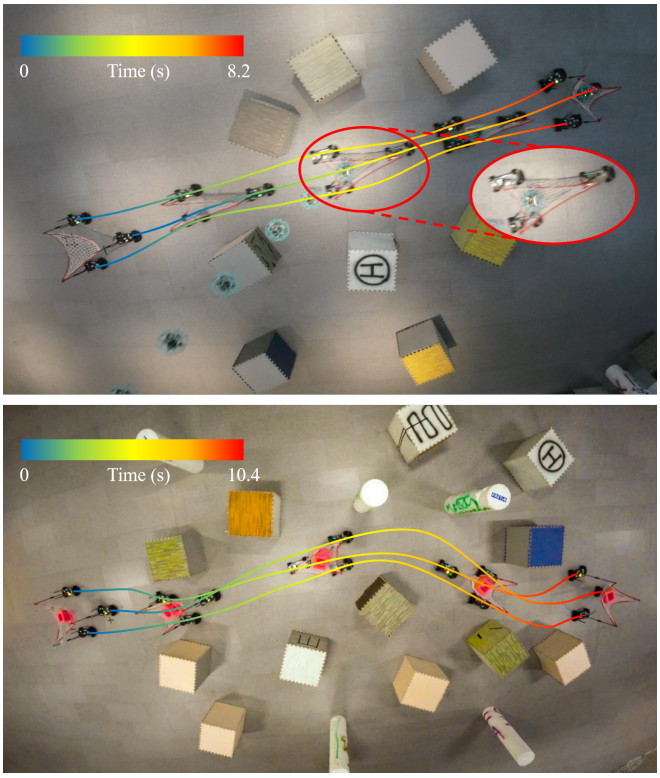

Fig. 1: Experimental results of a car-like robot team cooperatively catching and transporting objects in unstructured environments. Top: the robotic team collaboratively catches a falling drone with a supported net. Bottom: Some objects are transported through the unstructured environment by the robotic team.

Manuscript received: June, 23, 2023; Revised: August, 26, 2023; Accepted: October, 30, 2023. This paper was recommended for publication by Editor M. Ani Hsieh upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the National Natural Science Foundation of China under grant no. 62322314 and the Fundamental Research Funds for the Central Universities. (*Corresponding author: Fei Gao*)

[1]State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China.
[2]Huzhou Institute, Zhejiang University, Huzhou 313000, China.
E-mail:{plaa, fgaoaa}@zju.edu.cn
Digital Object Identifier (DOI): see top of this page.

problem. However, there currently exists no method capable of planning the trajectories of non-holonomic robot teams to complete the catching and transporting tasks online.

To bridge this research gap, we propose an efficient centralized planning framework for car-like robotic teams. Based on this framework, we model catching targets in a continuous and differentiable form for any feasible robot formation, allowing for fast gradient descent in optimization. As a result, the robot team is capable of fully exploiting its potential, deforming itself to a more appropriate state to catch the target. Furthermore, we impose restrictions on the size of the net and the topology of the team formation by accurately modeling the formation relationships. The *size safety* and *topological safety* of the rescue net supported by the robot team are guaranteed during navigation. Moreover, due to the adoption of efficient trajectory representation and decision variables transformation in optimization, all of these collaborative trajectories planning can be effectively completed online, making it possible for the robot team to perform catching and transporting tasks timely and accurately.

We apply our algorithm to a team of car-like robots and conduct catching and transporting experiments for falling targets in both simulation and real-world environments. As shown in Fig. 1, the experimental results demonstrate the ability of our algorithm to catch targets within its capability range in unstructured environments. As a foundational framework for collaborative planning, our proposed method is compared with some state-of-the-art multi-vehicle trajectory planning (MVTP) methods [3]–[7]. The results demonstrate that our method has significant advantages in terms of both computational time and success rates as agents number increases. Our contributions are summarized as follows:

1) We propose an efficient collaborative car-like robot team planning method that enables the generation of safe trajectories in unstructured environments with kinodynamic feasibility.
2) For real-time collaborative catching and transporting planning, we model multiple essential collaboration relationships and implement them based on the above framework.
3) We integrate our proposed method into a car-like robot team and validate the method in simulations and real-world experiments. We release our software for the community's reference.

II. Related Work

Currently, methods for solving the MVTP problem are mainly divided into coupled methods and decoupled methods. In order to reduce the enormous scale of the problem brought by coupled methods, the decoupled method decomposes the MVTP problem into several sub-problems and solves them gradually. Li et al. [8] proposed a prioritized trajectory optimization method for non-holonomic mobile robots. The method initially obtains collision-free paths using enhanced conflict-based search (ECBS) [9] and establishes safe corridors based on these paths. They improved the overall solution efficiency through grouping strategies. A penalty cost of deviating from the initial trajectory was introduced for the success rate of the optimization, which significantly limited the solution space. Moreover, sequential optimization can lead to deadlock issues in dense environments, resulting in failure to solve the problem. Luis et al. [7] proposed a distributed model predictive control (DMPC) based multi-agent motion planning framework and used on-demand collision avoidance for partitioning the free space, which allows for real-time trajectory planning with the less conservative movement and faster transition times. However, DMPC focused on planning in the range of specific horizon, which means the complete trajectories of the agents are often not optimal.

To account for the future states of multiple agents in the solution, the centralized methods solve for the trajectories of all agents in a single problem. Chen et al. [6] solved the MVTP problem by tightening collision constraints incrementally, thus forming a sequence of more relaxed and feasible intermediate optimization problems. They considered agents as circles for avoidance is conservative, and it may not guarantee that each quadratic programming (QP) problem is solvable when the kinematics are complex. Li et al. [10] proposed an adaptive-scaling constrained optimization (ASCO), which divides the intractably scaled constraints in the coupled optimal control problem (OCP) and conquers them in an iterative framework. In each iteration, partial collision avoidance constraints are implemented within an adaptive range. Ouyang et al. [5] proposed a two-stage method, which identifies a feasible homotopy class and finds a local optimum based on the identified homotopy class. However, they considered the car-like robot as two circles for obstacle avoidance, which is conservative, and the problem uses dense states as optimization variables, which makes it challenging to guarantee real-time performance. Wen et al. [11] proposed a spatiotemporal hybrid state A* for non-holonomic kinematic models. They introduced a hierarchical search-based solver called Car-like Conflict-Based Search (CL-CBS) and applied a body conflict tree to address collisions considering the shapes of the agents.

III. Collaborative Trajectories Optimizaiton

In this section, we introduce the motion model and differential flatness property of the car-like robots [12] and formulate collaborative planning as an optimization problem. Then we present the constraint handling approach, the basic kinematic constraints, and environmental obstacle avoidance constraints for the agents.

A. Differential Flatness of the car-like robots

We use the simplified kinematic bicycle model in the Cartesian coordinate frame to describe the kinematics of car-like robots. Assuming that the car is front-wheel driven and steered with perfect rolling and no slipping, the model can

be described as

$$\begin{cases} \dot{p}_x = v\cos\theta, \dot{p}_y = v\sin\theta, \\ \dot{\theta} = \dfrac{1}{L} v\tan\phi, \\ \dot{v} = a_t, \dot{a}_n = v^2\kappa, \\ \kappa = \tan\phi / L. \end{cases} \tag{1}$$

The state vector is $\boldsymbol{x} = (p_x, p_y, \theta, v, a_t, a_n, \phi, \kappa)^T$, where $\boldsymbol{p} = (p_x, p_y)^T$ denotes the position at the center of the rear wheels, θ is the yaw angle of the robot's body, v is the longitudinal velocity with respect to the robot's body frame, a_t is the longitudinal acceleration, a_n is latitude acceleration, ϕ is the steering angle of the front wheels, κ is the curvature and L is the wheelbase length of the car-like robot. We choose two differentially flat outputs $\boldsymbol{\sigma} := (\sigma_x, \sigma_y)^T$ as the optimization variables with a physical meaning that $\boldsymbol{\sigma} = \boldsymbol{p}$ is the position centered on the rear wheel of the car and the dynamics are given by:

$$\begin{cases} v = \eta\sqrt{\dot{\sigma}_x^2 + \dot{\sigma}_y^2}, \\ \theta = \arctan 2(\eta\dot{\sigma}_x, \eta\dot{\sigma}_y), \\ a_t = \eta(\dot{\sigma}_x\ddot{\sigma}_x + \dot{\sigma}_y\ddot{\sigma}_y)/\sqrt{\dot{\sigma}_x^2 + \dot{\sigma}_y^2}, \\ a_n = \eta(\dot{\sigma}_x\ddot{\sigma}_y - \dot{\sigma}_y\ddot{\sigma}_x)/\sqrt{\dot{\sigma}_x^2 + \dot{\sigma}_y^2}, \\ \phi = \arctan\left(\eta(\dot{\sigma}_x\ddot{\sigma}_y - \dot{\sigma}_y\ddot{\sigma}_x)L/(\dot{\sigma}_x^2 + \dot{\sigma}_y)^{\frac{3}{2}}\right), \\ \kappa = \eta(\dot{\sigma}_x\ddot{\sigma}_y - \dot{\sigma}_y\ddot{\sigma}_x)/(\dot{\sigma}_x^2 + \dot{\sigma}_y^2)^{\frac{3}{2}}, \end{cases} \tag{2}$$

where the additional variable η is to characterize the motion direction of the robot with $\eta = -1$ and $\eta = 1$ representing backward and forward movements respectively [13].

B. Optimization Problem Formulation

For the k-th agent in the swarm, the i-th segment of the trajectory is formulated as a 2-dimensional polynominal, which is divided into $M_{i,k}$ pieces with degree $2s-1$, a coefficient matrix $\boldsymbol{c}_{i,k} = [\boldsymbol{c}_{i,1,k}^T, \boldsymbol{c}_{i,2,k}^T, ..., \boldsymbol{c}_{i,M_{i,k},k}^T]^T \in \mathbb{R}^{2sM_{i,k}\times 2}$ and a uniform time interval $T_{i,k} \in \mathbb{R}^+$. Then, this piece of the trajectory can be expressed as:

$$\begin{aligned} \boldsymbol{\sigma}_{i,j,k}(t) = \mathbf{c}_{i,j,k}^T\boldsymbol{\beta}(t), \quad \boldsymbol{\beta}(t) := [1, t, ..., t^{2s-1}]^T, \\ \forall t \in [0, T_{i,k}], \forall j \in \{1, 2, ..., M_{i,k}\}, \\ \forall i \in \{1, 2, ..., n_k\}, \\ \forall k \in \{1, 2, ..., K\}, \end{aligned} \tag{3}$$

where K is the number of cooperative agents, n_k is the number of trajectory of the k-th agent. The i-th segment of the trajectory $\boldsymbol{\sigma}_{i,k} : [0, T_{i,k}]$ is obteined:

$$\begin{aligned} \boldsymbol{\sigma}_{i,k}(t) = \boldsymbol{\sigma}_{i,j,k}(t - (j-1) * T_{i,k}), \\ \forall j \in \{1, 2, ..., M_{i,k}\}, t \in [(j-1) * T_{i,k}, j * T_{i,k}]. \end{aligned} \tag{4}$$

The total duration of the i-th segment of k-th agent's trajectory is $\tau_k = M_{i,k} * T_{i,k}$.Based on this, we can obtain the trajectory of the k-th agent:

$$\begin{aligned} \boldsymbol{\sigma}_k(t) = \boldsymbol{\sigma}_i(t - \tau_{i,k}), \\ \forall i \in \{1, 2, ..., n_k\}, t \in [\tau_{i,k}, \tau_{i+1,k}], \end{aligned} \tag{5}$$

where $\tau_k = \sum_{i=1}^{n_k} T_{i,k}$ is the duration of the whole trajectory, $\tau_{i,k} = \sum_{\tilde{i}=1}^{i} T_{i,k}$ is the timestamp of the starting point of the i-th segment and $\tau_1 = 0$. Moreover, we define a coefficient set $\mathbf{c}_k = [\mathbf{c}_{1,k}, \mathbf{c}_{2,k}, ..., \mathbf{c}_{n,k}]^T \in \mathbb{R}^{(\sum_{i=1}^{n_k} M_{i,k}s)\times 2}$ and time vector $\mathbf{T}_k = [T_{1,k}, T_{2,k}, ..., T_{n,k}]^T \in \mathbb{R}^n$ for k-th agent. Then we define the coefficient set $\mathbf{c} = [\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_K] \in \mathbb{R}^{2M_i s\times 2nK}$ and time vector $\mathbf{T} = [\mathbf{T}_1, \mathbf{T}_2, ..., \mathbf{T}_K]^T \in \mathbb{R}^{n_k}$ for the subsequent derivation.

The desired collaboration trajectories are optimized on the control effort, time regularization, and collaborative cost of all the agents to ensure smoothness, aggressiveness, and collaboration. Moreover, to guarantee that the trajectories are kinodynamic feasible and competent for the cooperation task, corresponding constraints are necessary. The optimization problem formulation is:

$$\min_{\mathbf{c},\mathbf{T}} \sum_{k=1}^{K}\left(\int_0^{\tau_k} \boldsymbol{\mu}_k(t)^T\mathbf{W}\boldsymbol{\mu}_k(t)dt + w_T\tau_k\right) + \Psi_{obj}(\mathbf{c},\mathbf{T}) \tag{6a}$$

$$\text{s.t.}\boldsymbol{\mu}_k(t) = \boldsymbol{\sigma}_k^{[s]}(t), \forall t \in [0, \tau_k], \tag{6b}$$

$$\boldsymbol{\sigma}_{i,k}^{[s-1]}(0) = \bar{\boldsymbol{\sigma}}_{0,i,k}, \tag{6c}$$

$$\boldsymbol{\sigma}_{i,j,k}^{[\tilde{d}]}(\delta T_i) = \boldsymbol{\sigma}_{i,j+1,k}^{[\tilde{d}]}(0), \tag{6d}$$

$$T_{i,k} > 0, \tag{6e}$$

$$\begin{aligned} &\mathcal{G}_d(\boldsymbol{\sigma}(t), ..., \boldsymbol{\sigma}^{(s)}(t), t) \preceq 0, \forall d \in \mathcal{D}, \forall t \in [0, \tau_k], \\ &\forall i \in \{1, 2, ..., n_k\}, \forall j \in \{1, 2, ..., M_{i,k} - 1\}, \\ &\forall k \in \{1, 2, ..., K\}, \end{aligned} \tag{6f}$$

where $\mathbf{W} \in \mathbb{R}^{2\times 2}$ is a diagonal matrix to penalize control efforts, $w_T\tau_k$ is the time regularization term, and $\Psi_{obj}(\mathbf{c},\mathbf{T})$ is the collaborative cost when considering collaborative catching and transporting tasks, will be introduced in IV-C. Based on the proved optimality condition in the previous work [14], the entire piece-wise polynomial trajectory can be parameterized solely based on waypoints and the duration for each piece, thereby satisfying the equality constraints Eq.(6c-6d). For trajectory durations that are strictly constrained to be positive in Eq.(6e), we map them to unconstrained virtual times $\varsigma \in \mathbb{R}$ with a smooth bijection. Eq.(6f) is composed of environmental obstacle avoidance, inter-agent avoidance, kinodynamic constraints and collaborative constraints, where $\mathcal{D} = \{v, a_t, a_n, \kappa_l, \kappa_r, \mathfrak{C}, \Theta, \mathbb{L}_{net}, \mathbb{T}_{net}\}$ will be introduced in the following sections. We adopt the penalty term S_Σ to relax the feasibility constraints Eq. (6f). Consequently, the original optimization problem is reformulated into an unconstrained formulation, which can be efficiently solved using the L-BFGS algorithm [15]. Further application details are introduced in Section 2 of the supplementary material [16].

C. Single Agent Constraints

1) Kinodynamic Constraints: During the motion of the robot, the linear velocity, linear acceleration, latitude acceleration, and steering angle of the robot are limited by the physical properties of the robot motor, servo, tire, etc. These

kinodynamic constraints can be expressed as follows:

$$\begin{cases} \mathcal{G}_v(\dot{\boldsymbol{\sigma}}) = \dot{\boldsymbol{\sigma}}^T\dot{\boldsymbol{\sigma}} - v_{max}^2, \\ \mathcal{G}_{a_t}(\dot{\boldsymbol{\sigma}}, \ddot{\boldsymbol{\sigma}}) = \dfrac{(\ddot{\boldsymbol{\sigma}}^T\dot{\boldsymbol{\sigma}})^2}{\dot{\boldsymbol{\sigma}}^T\dot{\boldsymbol{\sigma}}} - a_{tmax}^2, \\ \mathcal{G}_{a_n}(\dot{\boldsymbol{\sigma}}, \ddot{\boldsymbol{\sigma}}) = \dfrac{(\ddot{\boldsymbol{\sigma}}^T\mathbf{B}\dot{\boldsymbol{\sigma}})^2}{\dot{\boldsymbol{\sigma}}^T\dot{\boldsymbol{\sigma}}} - a_{nmax}^2, \\ \mathcal{G}_{\kappa_l}(\dot{\boldsymbol{\sigma}}, \ddot{\boldsymbol{\sigma}}) = \dfrac{\ddot{\boldsymbol{\sigma}}^T\mathbf{B}\dot{\boldsymbol{\sigma}}}{||\dot{\boldsymbol{\sigma}}||_2^3} - \kappa_{max}, \\ \mathcal{G}_{\kappa_r}(\dot{\boldsymbol{\sigma}}, \ddot{\boldsymbol{\sigma}}) = -\dfrac{\ddot{\boldsymbol{\sigma}}^T\mathbf{B}\dot{\boldsymbol{\sigma}}}{||\dot{\boldsymbol{\sigma}}||_2^3} - \kappa_{max}, \end{cases} \tag{7}$$

where v_{max} is the magnitude of maximal logitude velocity, a_{tmax}/a_{nmax} is the maximal longitude/latitude acceleration, $\mathbf{B} := \begin{bmatrix} 0 & -1; & 1 & 0 \end{bmatrix}$ is an auxiliary 2×2 matrix with the property of $\mathbf{B}^T = -\mathbf{B}$, κ_{max} is the corresponding maximum curvature. Due to the monotonicity of tangent function, we restrict the front steer angle by limiting the curvature $\kappa = \tan\phi/L$ in $[-\tan\phi_{max}/L, \tan\phi_{max}/L] := [-\kappa_{max}, \kappa_{max}]$, where ϕ_{max} is the preset maximum steer angle.

2) Environmental obstacle avoidance: Considering the shape of the robot and in order to make full use of the safe solution space, we consider the geometric shapes for obstacle avoidance rather than as a point. After obtaining the grid map and the initial paths from hybrid A* [17], we sample the path to generate the safety corridor [18] and constrain the robot's whole body on the trajectory to be inside the safety corridor. We define the center of mass and orientation of the robot as the origin and x-direction of the body frame, which is illustrated in Fig.2. The rotation matrix $\mathbf{R}$, representing the orientation of the body frame relative to the world frame, can be expressed as:

$$\mathbf{R} = \frac{\eta}{\|\dot{\boldsymbol{\sigma}}\|_2}\left[\dot{\boldsymbol{\sigma}}, \mathbf{B}\dot{\boldsymbol{\sigma}}\right]. \tag{8}$$

We define the vertice set $\mathcal{E}$ of the k-th robot as:

$$\mathcal{E}_k = \{\boldsymbol{v}_e \in \mathbb{R}^2 : \boldsymbol{v}_e = \boldsymbol{\sigma} + \mathbf{R}\boldsymbol{l}_e, e = 1, 2, ..., n_e\}, \tag{9}$$

where n_e is the number of vertexs and $\boldsymbol{l}_e$ is the coordinate of the e-th vertex in the body frame. The H-representation [19] of each convex polygon $\mathfrak{C}$ in the driving corridor is obtained:

$$\begin{aligned} &\mathfrak{C} = \{\boldsymbol{q} \in \mathbb{R}^2 : \boldsymbol{A}\boldsymbol{q} \leq \boldsymbol{b}\}, \\ &\boldsymbol{A} = [\boldsymbol{A}_1, ..., \boldsymbol{A}_z, ..., \boldsymbol{A}_{n_z}]^T \in \mathbb{R}^{n_z \times 2}, \\ &\boldsymbol{b} = [b_1, ..., b_z, ..., b_{n_z}]^T \in \mathbb{R}^{n_z}, \end{aligned} \tag{10}$$

where n_z is the number of hyperplanes, $\boldsymbol{A}_z \in \mathbb{R}^2$ and $b_z \in \mathbb{R}$ are the descriptors of the hyperplane, which can be determined by a point on the hyperplane and the normal vector. Therefore the constraint of the robot's whole body in the safety corridor on the trajectory can be expressed as:

$$\mathcal{G}_{\mathfrak{C}}(\boldsymbol{\sigma}, \dot{\boldsymbol{\sigma}}) = \boldsymbol{A}_z^T(\boldsymbol{\sigma} + \mathbf{R}\boldsymbol{l}_e) - b_z, \forall z \in \{1, 2, ..., n_z\} \tag{11}$$

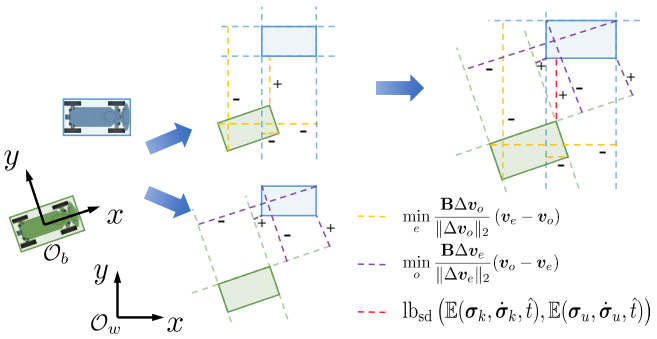

Fig. 2: The illustration of the lower bound of signed distance. First, we calculate the minimum distances from each vertex of one polygon to the edges of another polygon, resulting in purple and yellow dashed lines. Then, we determine the maximum value among these distances, which corresponds to the red dashed line. The positive or negative signs of the distances are indicated alongside them.

IV. Collaboratively Catching and Transporting Objects

A. Inter-agents Obstacle Avoidance

To ensure the safety of agents, it is required that the convex multi-deformation distance under each moment between agents is greater than the safety distance d_m. Therefore, the constraint of collision avoidance between agents is defined as $\mathcal{G}_\Theta(\boldsymbol{\sigma}, \dot{\boldsymbol{\sigma}}) = [\mathcal{G}_{\Theta_1}, ..., \mathcal{G}_{\Theta_u}, ..., \mathcal{G}_{\Theta_{N_u}}]^T \in \mathbb{R}^{N_u}$ where $N_u = \frac{k(k-1)}{2}$ is the number of swarm agents in two-by-two combinations. The constraint of the k-th agent and u-th agent at a constraint point is defined as:

$$\mathcal{G}_{\Theta_{k,u}}(\boldsymbol{\sigma}_k, \dot{\boldsymbol{\sigma}}_k, \boldsymbol{\sigma}_u, \dot{\boldsymbol{\sigma}}_u) = d_m - U\left(\mathcal{E}_k, \mathcal{E}_u\right), \tag{12}$$

where d_m is the minimum safe distance and $U(\mathbb{E}_k, \mathbb{E}_u)$ means the convex polygons distance between the k-th and u-th agent. In collaboratively trajectory optimization of agents, the computation of convex polygon distances between agents needs to be computed in a continuously differentiable manner. In this work, we use the maximum-minimum smoothing Minsky difference-based algorithm [20] to compute the lower approximation of the signed distances of convex polygons and obtain their derivatives, which can be expressed as:

$$\begin{aligned} \mathrm{lb_{sd}}\left(\mathcal{E}_k, \mathcal{E}_u\right) = \max\Bigg\{ &\max_e \min_o \frac{\mathbf{B}\Delta\boldsymbol{v}_e}{\|\Delta\boldsymbol{v}_e\|_2}(\boldsymbol{v}_o - \boldsymbol{v}_e), \\ &\max_o \min_e \frac{\mathbf{B}\Delta\boldsymbol{v}_o}{\|\Delta\boldsymbol{v}_o\|_2}(\boldsymbol{v}_e - \boldsymbol{v}_o)\Bigg\}, \\ &\boldsymbol{v}_e \in \mathcal{E}_k, \boldsymbol{v}_o \in \mathcal{E}_u, \end{aligned} \tag{13}$$

where $\Delta\boldsymbol{v}_e = \boldsymbol{v}_{e+1} - \boldsymbol{v}_e$ and $\Delta\boldsymbol{v}_u = \boldsymbol{v}_{u+1} - \boldsymbol{v}_u$. The physical meaning of the lower bound of the signed distance between two vehicles is shown in Fig. 2. To overcome the gradient discontinuity problem of eq.(13), we use the log-sum-exp function to smooth the maximum and minimum operations [21].

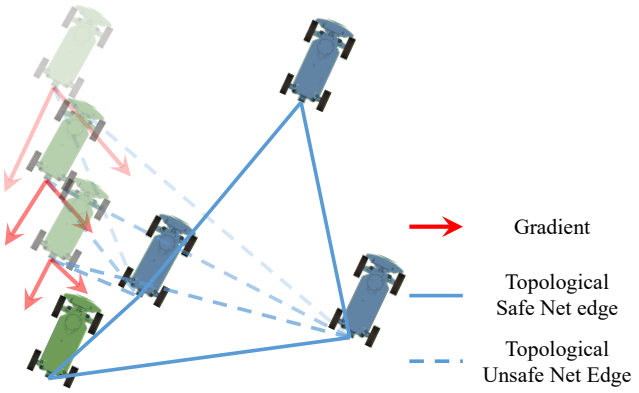

Fig. 3: The process of achieving topological safety during the optimization. The lightest shade of green represents the initial state of the green robot. By utilizing the aforementioned formulas, gradients in two directions are obtained and make the net achieve topological safety.

B. Collaborative Transportation

In order to achieve the capability of cooperative transportation, usually the robot team cooperates to support a net, which requires maintaining the *size safety* and *topology safety* of the net. For *size safety*, we constrain the distance between each agent and the neighboring agents to be less than the corresponding net edge length, which can be expressed as:

$$\begin{aligned} \mathcal{G}_{\mathbb{L}_{net}}(\boldsymbol{\sigma}_k, \boldsymbol{\sigma}_{N(k)}) = \|\boldsymbol{\sigma}_k - \boldsymbol{\sigma}_{N(k)}\|_2 - L_k^{net}, \\ \forall k \in \{1, 2, ..., K\}, \end{aligned} \tag{14}$$

where $N(k)$ denotes the next agent number adjacent to the k-th agent, which can be expressed as:

$$N(k) = \begin{cases} k+1, & k < K, \\ 1, & k = K, \end{cases} \tag{15}$$

L_k^{net} denotes the length of net edge between the k-th and $N(k)$-th agent. Moreover, in order to maintain the topology of the net during the collaborative support net, it is necessary to constrain the signed distance of the agent to the line where the non-neighboring net edges are located to be positive, which can be expressed as:

$$\begin{aligned} \mathcal{G}_{\mathbb{T}_{net}}(\boldsymbol{\sigma}_k, \boldsymbol{\sigma}_p, \boldsymbol{\sigma}_{N(p)}) \ = \frac{\mathbf{B}(\boldsymbol{\sigma}_{N(p)} - \boldsymbol{\sigma}_p)^T}{\|\boldsymbol{\sigma}_{N(p)} - \boldsymbol{\sigma}_p\|_2}(\boldsymbol{\sigma}_k - \boldsymbol{\sigma}_p), \\ \forall p \in \{p \in \{1, 2, ..., K\} : p \neq k, N(p) \neq k\}, \\ \forall k \in \{1, 2, ..., K\}. \end{aligned} \tag{16}$$

The specific process is illustrated in Fig. 3. Through gradient descent, the collaborative robot team achieves topological safety. It is worth noting that the blue robot also adjusts its state simultaneously to ensure topological safety, although it is not depicted in the image for clarity.

By imposing two constraints on the positions of agents simultaneously along the trajectory, we ensure both *size safety* and *topological safety* of the net throughout the support period.

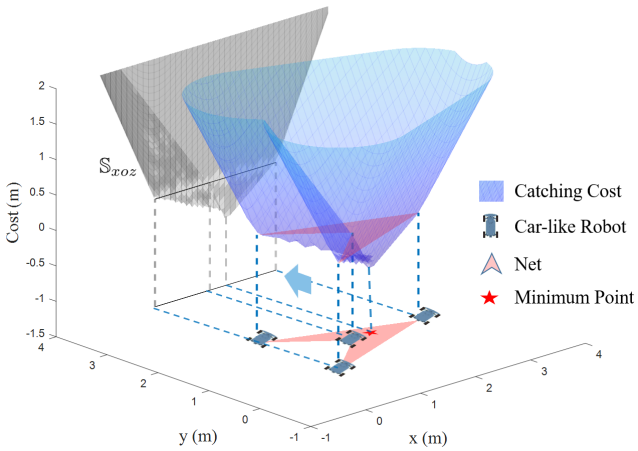

Fig. 4: The collaborative catching cost Ψ_{obj} formed when the formation of the robot team is a non-convex polygon. During the optimization process, by changing the positions of each agent, the minimum point of the cost is made as close as possible to the landing point of the target.

C. Adaptive Catching

By coordinating the robot team to catch the object and solving it in the optimization, we can expand the successful feasible spatial-temporal solution space as much as possible in order to achieve catching at high-speed status in complex environments by agents.

For E objects that need to be caught, sorted by the landing time, let $(\boldsymbol{p}_e, t_e)$ represent the predicted landing position and landing time, respectively. In order to ensure that the robot team is capable of completing the rescue mission in any formation, we utilize the following formula to describe the relationship between the landing point and the formation. Firstly, shoot a ray from the landing point in any direction, and check the number of intersection points between the ray and the polygon formed by the current formation (in accordance with IV.B). If it is odd, define the landing point as being inside the polygon with $\Upsilon = -1$. On the other hand, if it is even, define the landing point as being outside the polygon with $\Upsilon = 1$. Then, the signed distance between the point and the polygon formed by the formation can be expressed as:

$$\begin{aligned} \mathbb{D}(\boldsymbol{\sigma}, \boldsymbol{p}_e, t_e) = \Upsilon \min_k \Bigg\{ \Bigg\| - \boldsymbol{p}_e + \boldsymbol{\sigma}_k + \\ \mathcal{L}\Big(\frac{(\boldsymbol{\sigma}_{N(k)} - \boldsymbol{\sigma}_k)^T(\boldsymbol{p}_e - \boldsymbol{\sigma}_k)}{\|\boldsymbol{\sigma}_{N(k)} - \boldsymbol{\sigma}_k\|_2^2}\Big)(\boldsymbol{\sigma}_{N(k)} - \boldsymbol{\sigma}_k)\Bigg\|_2 \Bigg\}, \\ k \in \{1, ..., K\}, e \in \{1, ..., E\}, \end{aligned} \tag{17}$$

where smoothing function $\mathcal{L}(x)$ is defined as:

$$\mathcal{L}(x) = \begin{cases} 0, & x \leq 0, \\ -\frac{1}{\epsilon^2}x^3 + \frac{2}{\epsilon}x^2, & 0 < x \leq \epsilon, \\ x, & \epsilon < x \leq 1-\epsilon, \\ -\frac{1}{\epsilon^2}x^3 + \frac{3-2\epsilon}{\epsilon^2}x^2 & \\ +\frac{4\epsilon-3}{\epsilon^2}x + \frac{(\epsilon-1)^2}{\epsilon^2}, & 1-\epsilon < x \leq 1, \\ 1, & x > 1, \end{cases} \tag{18}$$

where $\epsilon \in \mathbb{R}^+$ is an excessive micro value . Based on this, we can calculate the signed distance between the target landing point and the formation of the robot team. We propose a collaborative cost function Ψ for the collaborative catching of the target by the robots, expressed as:

$$\Psi_{obj} = \begin{cases} \omega_{ce}\mathbb{D}, & \mathbb{D} < -\epsilon, \\ \frac{\omega_{ce}-\omega_{ca}}{4\epsilon^3}\mathbb{D}^4 - \frac{3(\omega_{ce}-\omega_{ca})}{4\epsilon}\mathbb{D}^2 & \\ +\frac{\omega_{ce}+\omega_{ca}}{2}\mathbb{D}, & -\epsilon < \mathbb{D} < \epsilon, \\ \omega_{ca}\mathbb{D}, & \mathbb{D} > \epsilon, \end{cases} \tag{19}$$

where ω_{ce} and ω_{ca} are the weights of taking the target to the center of the net and guaranteeing the catch of the target, respectively. As shown in Fig. 4, a non-convex polygonal formation can form a continuously differentiable cost based on Eq.(19), allowing the robot team to adaptively catch the target and actively increase the formation when catching. In addition, in order to make the trajectory end more adaptable to the environment and smoother, we relaxed the end state of the trajectory.

$$\mathbf{M}_{i,k}\left(T_{i,k}\right) c_{i,k} = \mathbf{b}_{i,k}, 1 \le i \le N_k, 1 \le k \le K, \tag{20}$$

where $\mathbf{M}_{i,k}$ is an invertible matrix defined in [14]. In this work, $\boldsymbol{b}_{\boldsymbol{n_k}}$ is defined as follows:

$$\begin{aligned} \mathbf{b}_{n_k} = (&p_{n_k-1}, v_{n_k-1}, \mathbf{0}_{2\times(s-2)}, \boldsymbol{q}_{n_k,1}, \mathbf{0}_{2\times\tilde{d}}, ..., \\ &\boldsymbol{q}_{n_k,M_{n_k,k}-1}, \mathbf{0}_{2\times\tilde{d}}, \boldsymbol{\sigma}_k^f)^{\mathrm{T}}, \quad \forall k \in \{1, ..., K\}, \end{aligned} \tag{21}$$

where $\boldsymbol{\sigma}_k^f$ is the final position of the k-th robot. We include the final positions of trajectories as decision variables. Then, the gradients of the objective function with respect to final positions are derived as follows:

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{\sigma}_k^f} = \left(\mathbf{M}_i^{-\mathrm{T}} \frac{\partial \mathcal{J}}{c_i}\right)^{\mathrm{T}} e_{2sM_{n_k,k}}, \tag{22}$$

In this way, by optimizing the final positions of the trajectories, the robot team can freely choose a smoother parking position.

Fig. 5: Computation time under different agents numbers.

V. Evaluation And Experiments

A. Benchmark Comparisons

To verify the effectiveness and computational efficiency of our proposed method, we compared it with several state-of-the-art MVTP methods. All the methods were tested on an Intel i7-12700 CPU with 32GB memory. We conducted tests on 100 cases with 8 random-sized and positioned obstacles. To ensure fairness in the comparative experiments, we provided the same initial guess for methods using the Hybird A* algorithm [17]. More details are introduced in Section 7 of the supplementary material [16]. We tested different numbers of agents and recorded success rates, computation time, the average time to execute agent trajectories, the average time to execute the longest trajectory, average total trajectory length, and average trajectory acceleration cost. These results are shown in Table I.

From the data, it can be seen that FOTP shows a high success rate, but due to the dense decision variables of all states and inputs in trajectories, the problem dimension is extremely high, leading to a huge computational burden. In contrast, our proposed method optimizes the waypoints and durations of each piece of the polynomial trajectory, resulting in a significant advantage in computation time compared to FOTP. On the other hand, our proposed method only needs to improve the initial path without considering collision avoidance to complete collision avoidance among agents in a relatively short time, significantly reducing the dependence on initial paths. The MNHP method uses a distributed approach with grouping and priority strategies. However, there are several limitations associated with it. Firstly, it considers robots as circular shapes for obstacle avoidance, which is conservative and leads to reduced solution space. This limitation hampers its effectiveness in environments where the robot's shape needs to be taken into consideration, especially in narrow spaces where the robot's shape becomes crucial. Secondly, the MNHP method penalizes the extent to which trajectories deviate from the initial values provided by ECBS [9] in the cost function. This heavy dependency on initial values significantly impacts its performance. Thirdly, MNHP does not optimize trajectory time and requires all agents to reach their specified final states simultaneously. This constraint further diminishes the possibility of successful spatiotemporal collision avoidance. These three limitations collectively restrict the effectiveness of MNHP. DMPC uses a priority-based strategy and employs on-demand obstacle avoidance strategy. However, it only considers local information in the near future for each planning iteration, which leads to overly greedy behavior and poor trajectory quality. It often takes detours in areas with dense obstacles and agents, and may even get stuck in infeasible situations due to its greediness. From the data, it can be seen that CL-CBS has a high success rate, but it only provides a path with a time sequence that can serve as a feasible solution, without any effect on smoothness. Moreover, because of its semi-decentralized approach to seeking solutions, it has to face the problem of rapidly

TABLE I: Comparison Of Trajectory Generation With State-of-the-art MVTP Methods

Method	Success Rate	Time(s)			Average Travel Distance(m)	Acceration Cost(m^2s^{-3})	Global Planning	Centra-lization
		Computation	Mean Travel	Longest Travel				
Proposed, K = 8	**100%**	0.280	**5.956**	6.918	84.461	**244.925**	**Yes**	**Yes**
Proposed, K = 14	**100%**	**0.707**	**6.067**	**6.841**	**145.002**	**205.204**		
Proposed, K = 20	**100%**	**1.186**	**6.328**	**7.151**	**207.469**	**196.532**		
Proposed, K = 26	**100%**	**1.900**	**6.387**	**8.216**	**271.725**	218.752		
FOTP [5], K = 8	**100%**	4.897	6.169	**6.169**	**81.462**	286.648	Yes	Yes
FOTP, K = 14	**100%**	37.801	6.957	6.957	149.151	245.775		
FOTP, K = 20	**100%**	147.214	7.835	7.835	222.223	204.270		
FOTP, K = 26	**100%**	376.645	8.888	8.888	297.745	**154.748**		
MNHP [8], K = 8	85%	**0.077**	5.009	5.009	84.779	957.611	Yes	Hybrid
MNHP, K = 14	31%	-	4.850	4.850	146.166	1462.799		
MNHP, K = 20	9%	-	5.280	5.280	210.357	1705.211		
MNHP, K = 26	4%	-	5.100	5.100	280.261	2701.124		
DMPC [7], K = 8	42%	2.877	8.743	12.711	176.121	1213.8	No	No
DMPC, K = 14	19%	7.348	10.609	17.376	376.560	1597.272		
DMPC, K = 20	8%	13.158	11.748	20.100	612.625	2043.097		
DMPC, K = 26	0%	-	-	-	-	-		
SCP [22], K = 8	**100%**	164.020	5.000	5.000	79.480	272.010	Yes	Yes
SCP, K = 14	92%	1337.320	4.792	4.792	140.259	325.094		
SCP, K = 20	79%	4065.292	4.884	4.884	202.888	322.018		
SCP, K = 26	15%	9497.747	4.925	4.925	253.961	366.470		
CL-CBS [3], K = 8	**100%**	0.524	-	-	96.335	-	Yes	Hybrid
CL-CBS, K = 14	**100%**	13.813	-	-	163.819	-		
CL-CBS, K = 20	**100%**	32.344	-	-	236.304	-		
CL-CBS, K = 26	**100%**	46.991	-	-	306.870	-		

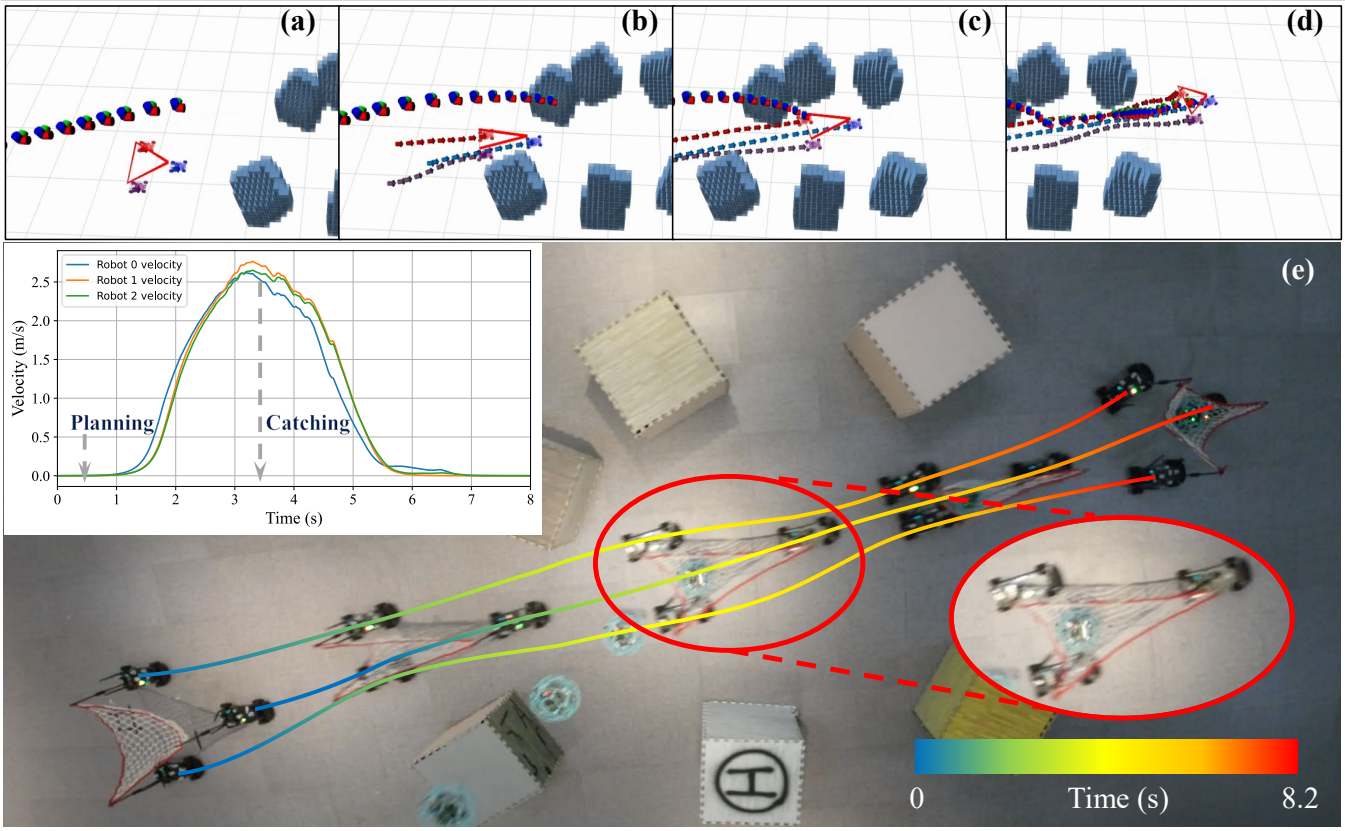

Fig. 6: Experimental validation of our method for a crashed aircraft rescuing. The collaborative rescue trajectory of the robot team is generated in 0.053 s and was able to rescue the downed aircraft successfully. (a) The initial state of the robot team and the aircraft began to experience a malfunction and consequently crashed. (b) The robot team transforms and maneuvers to avoid obstacles while maintaining the safety of agents and rescue net security. (c) The primary robot team proactively increases the net rescue size before and after rescuing the target to enhance the success rate and provide cushioning. (d) The robot team safely comes to a gentle stop in the secure area. (e) The global trajectory planning results.

increasing computation time in the continuously shrinking solution space and the lower priority agents having the worse trajectory quality. We conducted tests on two methods with the highest success rates in the table under different cases of planning for the number of agents, as shown in Fig. 5.

B. Real-World Experiments

The experimental field is a $6m \times 12m$ field with random obstacles and the robots collaboratively support a net. The processor of the central planning platform is an 8th Intel i5-8700 with 16 GB RAM, and the trajectories are sent to the swarm agents for tracking online. We fix the vertices of a net to the rear of each of the three car-like robots and carry out a collaborative transportation experiment on the field. After being given the target state, the central computing platform calculates the trajectories of the robotic team, which is then followed by the controllers of each agent [23]. Further application details are introduced in Section 3 of the supplementary material [16]. The catching process is shown in Fig. 6, where a failed drone acts as the rescue target. During the flight, the drone suddenly malfunctioned and fell, and the system began to plan collaborative rescue trajectories. Our method generates collaborative trajectories for three car-like robots in just 0.053s. Throughout the process, in addition to satisfying the kinematics and safety of each agent's movement, both the *size safety* and *topological safety* of the net are also guaranteed. The generated trajectories highly exploit the robots' kinematic performance to complete the rescue mission. As shown in Fig 6(e), to successfully achieve long-distance rescue, the robot team first accelerates at almost maximum acceleration and achieves the rescue at almost maximum speed. Then, due to the relaxation of the terminal state at the rear end, they smoothly stop in the safe area.

VI. Conclusion

In this work, we introduce a collaborative robotic team planning system with capabilities of collaborative target catching and transporting. By modeling cooperative tasks, we complete the planning of a car-like robot team to collaboratively catch and transport objects. Comparison with current multiple MVTP methods shows the solution quality and scalability of our method. Simulations and real-world experiments demonstrate the robustness and flexibility of our method. In the future, one research direction is to extend our method to transportation in three-dimensional environments, where the team formation changes to enable the transported object to avoid obstacles considering height.

References

[1] J. Alonso-Mora, R. Knepper, R. Siegwart, and D. Rus, "Local Motion Planning for Collaborative Multi-Robot Manipulation of Deformable Objects," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 5495–5502.

[2] H. Zhang, H. Song, W. Liu, X. Sheng, Z. Xiong, and X. Zhu, "Hierarchical motion planning framework for cooperative transportation of multiple mobile manipulators," *arXiv preprint arXiv:2208.08054*, 2022.

[3] L. Wen, Y. Liu, and H. Li, "CL-MAPF: Multi-agent path finding for car-like robots with kinematic and spatiotemporal constraints," *Robotics and Autonomous Systems*, vol. 150, p. 103997, 2022.

[4] J. Li, M. Ran, and L. Xie, "Efficient Trajectory Planning for Multiple Non-Holonomic Mobile Robots via Prioritized Trajectory Optimization," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 405–412, Apr. 2021.

[5] Y. Ouyang, B. Li, Y. Zhang, T. Acarman, Y. Guo, and T. Zhang, "Fast and Optimal Trajectory Planning for Multiple Vehicles in a Nonconvex and Cluttered Environment: Benchmarks, Methodology, and Experiments," in *International Conference on Robotics and Automation (ICRA)*, May 2022, pp. 10 746–10 752.

[6] Y. Chen, M. Cutler, and J. P. How, "Decoupled Multiagent Path Planning via Incremental Sequential Convex Programming," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 5954–5961.

[7] C. E. Luis, M. Vukosavljev, and A. P. Schoellig, "Online Trajectory Generation With Distributed Model Predictive Control for Multi-Robot Motion Planning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 604–611, Apr. 2020.

[8] J. Li, M. Ran, and L. Xie, "Efficient trajectory planning for multiple non-holonomic mobile robots via prioritized trajectory optimization," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 405–412, 2021.

[9] M. Barer, G. Sharon, R. Stern, and A. Felner, "Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 5, no. 1, 2014, pp. 19–27.

[10] B. Li, Y. Ouyang, Y. Zhang, T. Acarman, Q. Kong, and Z. Shao, "Optimal Cooperative Maneuver Planning for Multiple Nonholonomic Robots in a Tiny Environment via Adaptive-Scaling Constrained Optimization," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1511–1518, Apr. 2021.

[11] L. Wen, Z. Zhang, Z. Chen, X. Zhao, and Y. Liu, "CL-MAPF: Multi-Agent Path Finding for Car-Like Robots with Kinematic and Spatiotemporal Constraints," *Robotics and Autonomous Systems*, vol. 150, p. 103997, Apr. 2022.

[12] R. M. Murray, M. Rathinam, and W. Sluis, "Differential Flatness of Mechanical Control Systems: A Catalog of Prototype Systems," in *Proceedings of the 1995 ASME International Congress and Exposition*, 1995.

[13] Z. Han, Y. Wu, T. Li, L. Zhang, L. Pei, L. Xu, C. Li, C. Ma, C. Xu, S. Shen, and F. Gao, "An efficient spatial-temporal trajectory planner for autonomous vehicles in unstructured environments," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–18, 2023.

[14] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically Constrained Trajectory Optimization for Multicopters," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 3259–3278, Oct. 2022.

[15] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical programming*, vol. 45, no. 1-3, pp. 503–528, 1989.

[16] L. Pei, *Supplementary Materials for Collaborative Planning for Catching and Transporting Objects in Unstructured Environments*, Sept. 2023. [Online]. Available: https://doi.org/10.5281/zenodo.8376633

[17] D. A. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *The International Journal of Robotics Research*, vol. 29, pp. 485 – 501, 2010.

[18] X. Zhong, Y. Wu, D. Wang, Q. Wang, C. Xu, and F. Gao, "Generating large convex polytopes directly on point clouds," *arXiv preprint arXiv:2010.08744*, 2020.

[19] D. Avis, K. Fukuda, and S. Picozzi, "On canonical representations of convex polyhedra," in *Mathematical Software*. World Scientific, 2002, pp. 350–360.

[20] S. Cameron and R. K. Culley, "Determining the minimum translational distance between two convex polyhedra," *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, pp. 591–596, 1986.

[21] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[22] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1917–1922, 2012.

[23] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1094–1099, 2015.