2025 IEEE International Conference on Robotics and Automation (ICRA)
May 19-23, 2025. Atlanta, USA

Multi-Nonholonomic Robot Object Transportation with Obstacle Crossing Using a Deformable Sheet

Weijian Zhang, Charlie Street, Masoumeh Mansouri

***Abstract*— In this paper, we address multi-robot formation planning where nonholonomic robots collaboratively transport objects using a *deformable sheet* in unstructured, cluttered environments. The formation can expand or contract to adjust the height of the object on the sheet. However, interactions between the robots and sheet introduce complex constraints for formation planning. Complexity increases further when the only feasible solution requires *crossing* an obstacle, i.e. where robots navigate in different *homotopy classes* around an obstacle such that the object hovers above it. Most existing nonholonomic formation planners do not admit obstacle crossing, limiting performance. In this paper, we present a two-stage iterative trajectory optimization framework which explicitly considers obstacle crossing. First, we capture the set of all feasible homotopy classes for each robot using a topological probabilistic roadmap. We then iteratively apply numerical optimization techniques to find a safe and feasible solution for the formation. We demonstrate the efficacy of our framework in simulation and on real robot hardware.**

I. Introduction

Recently, significant research has considered collaborative object transportation using formations of mobile robots manipulating objects by pushing, grasping, or caging [1]. Outside of robotics, *deformable sheets* have been used to carry objects such as hospital patients or fragile objects [2]. Deformable sheets can be applied to multi-robot object transportation, where robots hold up the corners of the sheet and the object lies in the middle. This improves formation flexibility, as robots can adjust the sheet height by moving further apart to avoid obstacles [3]. However, it is challenging to synthesize feasible and efficient formation trajectories using these sheets, as we must consider how the robots and sheet interact, and how to maintain the formation while avoiding collisions with obstacles. Complexity increases further when the only feasible solution requires robots to navigate through multiple distinct *homotopy classes*. For example, robots in the same formation may be positioned on opposite sides of an obstacle while collectively transporting an object on a sheet above the obstacle (see Figure 1). For object transportation, such obstacle crossing behaviors should be handled while maintaining the formation. In this paper, we use nonholonomic robots to perform these challenging behaviours. We

Weijian Zhang, Charlie Street and Masoumeh Mansouri are with the School of Computer Science at the University of Birmingham, wxz163@student.bham.ac.uk and {c.l.street, m.mansouri}@bham.ac.uk

Charlie Street and Masoumeh Mansouri are UK participants in Horizon Europe Project CONVINCE, and supported by UKRI grant number 10042096.

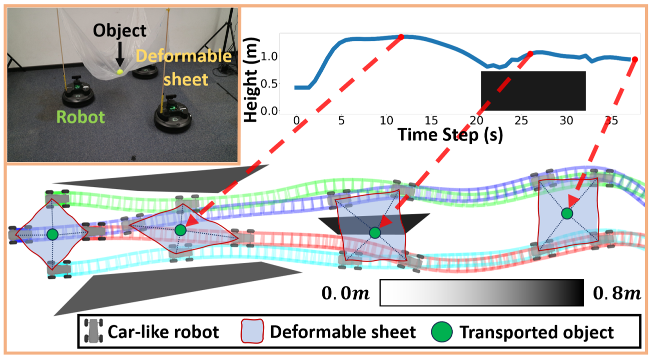

Fig. 1. Four car-like robots collaboratively transporting an object with a deformable sheet in an unstructured environment. The irregular obstacles randomly distributed in the environment vary in height. The image in the top left demonstrates our transportation setup on three Turtlebots.

consider nonholonomic robots for their enhanced robustness during transportation and widespread applicability in industrial settings.

In this paper, we address collaborative object transportation for nonholonomic robot formations in cluttered environments using a deformable sheet (see Fig. 1). For this, we introduce a two-stage iterative trajectory optimization framework. First, we identify a set of paths for each robot that are homotopically distinct using a topological probabilistic roadmap (see Sec. IV). Each homotopically distinct path makes different routing decisions around each of the obstacles in the environment. These paths are then used as an initial guess for the collaborative trajectory optimization problem, which is formulated as a joint optimal control problem (OCP). To simplify trajectory optimization, we capture object height constraints implicitly in the constraints of the formation. With this, we then iteratively synthesize a safe and feasible locally optimal trajectory for a given homotopy class (see Sec. V).

Though previous work has considered multi-robot object transportation using a deformable sheet [3]–[10], ours is the first to allow nonholonomic robots to cross obstacles in dense, unstructured environments. The primary contributions of this paper are i) a heuristic path exploration method which efficiently evaluates a set of homotopically distinct solution spaces for the formation; and ii) a two-stage iterative motion planning framework for finding locally time-optimal collision-free formation trajectories using a deformable sheet. We demonstrate the efficacy of our proposed framework in simulation and on hardware. All corresponding software is released open source online[1].

[1]https://github.com/HyPAIR/CPDOT

2025 IEEE International Conference on Robotics and Automation (ICRA) | 979-8-3315-4139-2/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICRA55743.2025.11128313

II. Related Work

Deformable Object Modelling. Deformable sheets have high degrees of freedom, making it difficult to predict the motion of objects lying in the sheet. Deformable sheets are often modelled for robotic manipulation using physics- and geometry-based models [11]. Physics-based models use physics simulators to calculate object motion relative to the sheet. However, these models are often highly complex and challenging to construct accurately, making them unsuitable for real-time applications. Geometry-based models approximate the sheet through a set of geometric constraints to provide efficient estimates of object deformation [12].

Formation-Level Obstacle Avoidance. In [7], [9], formation motion planning is simplified using the geometric constraints of the sheet, reducing the complexity of the interaction model. In [10], a centralized planning and control algorithm is presented which efficiently guides a formation of car-like robots while transporting a net carrying an object in unstructured environments. However, [7], [9], [10] treat the formation as a solid two-dimensional object which must remain strictly separated from obstacles to avoid collisions. This collision avoidance strategy can easily fail in environments with dense obstacles [8].

Formations can flexibly avoid obstacles by passing through them, i.e. allowing robots to travel either side of an obstacle assuming the object is high enough. In [4]–[6], a formation of three omnidirectional mobile robots grip and support a deformable sheet to manipulate and transport objects along a given trajectory. However, as the number of robots increases, the transported object may have different equilibrium states on the sheet, i.e. different parts of the sheet can be taut or slack. This makes it challenging to identify the location of the object. A virtual variable cable model (VVCM) is presented in [3] which simplifies object-sheet interactions into a set of dynamically changing cables. With this, the equilibrium of the object on the sheet is represented using the tension state of the virtual cables in a quasi-static condition. This allows the object's position under different formation configurations to be estimated quickly. The authors then present a local planner which allows the formation to navigate through obstacles, however it relies heavily on the robots being holonomic. In [8], a formation-level path planning algorithm is introduced to navigate through dense environments, however its success rate is highly dependent on how finely the environment is discretized. Further, all virtual cables must remain taut during transportation which reduces the flexibility of the formation. In [13], [14], cable-driven parallel robots (CDPRs) use physical cables to transport objects. A reinforcement learning approach for CDPR dynamic obstacle avoidance is presented in [13], however learning is complex and does not transfer to different environments. In [14], an optimization-based trajectory planner for CDPRs is presented. This approach requires the coordinates and timesteps of the object's passage through obstacles prior to planning, reducing its applicability and flexibility.

In contrast to [3]–[8], our framework is applicable to non-holonomic robots. Also, unlike [9], [10], we allow formations to navigate through obstacles, improving the transportation success rate in dense and unstructured environments.

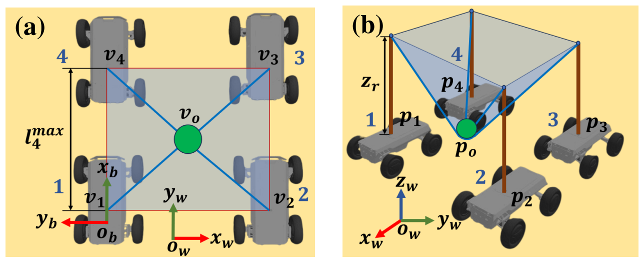

Fig. 2. A formation of four car-like robots supports an object (green) with a deformable sheet (blue). (a) The $2D$ coordinate system of the undeformed sheet (top view). (b) The $3D$ coordinate system of the robot formation based on the virtual variable cables model [3].

III. Problem Statement

In this paper, we consider a formation of N car-like nonholonomic robots that transport an object using a deformable sheet. Let $\mathcal{W} \subset SE(3)$ be the $3D$ workspace, and $\mathcal{O} \subset SE(3)$ be the set of static convex polyhedral obstacles which are known *a priori*. During object transportation, the robots move in $2D$ space, whereas the transported object moves in $3D$. To simplify the object transportation problem, we apply the assumptions in [3], i.e. the object is a point mass, the object's motion is quasi-static, and the deformable sheet is inelastic, such that the object moves freely under gravity towards the position of minimum potential energy on the sheet. For robot $i \in \{1, ..., N\}$, we define $\mathbf{p}_i(t) = (x_i(t), y_i(t))^T \subset SE(2)$ as the coordinate of the midpoint of the rear axle and $\theta_i(t) \in [-\pi, \pi)$ as the robot's orientation in Cartesian space. $\mathbf{p}_o = [x_o, y_o, z_o]^T \subset SE(3)$ denotes the coordinate of the object. A $2D$ view of a four-robot formation is shown in Fig. 2(a), where the support points of the sheet $\mathbf{v}_i$ form a convex polygon with N vertices. We capture the formation topology by labelling the robots sequentially as $\{1, 2, 3, 4\}$ in an anticlockwise direction. The topological order must remain unchanged during transportation or the sheet will fold or twist. We define $N(i)$ as the robot number adjacent to robot i, i.e. $N(i) = i + 1$ if $i < N$, and $N(i) = 1$ otherwise. A $3D$ view of the same formation is shown in Fig. 2(b), where each robot's support point $\mathbf{r}_i$ is fixed at a constant height z_r, i.e., $\mathbf{r}_i = [\mathbf{p}_i, z_r]^T$. In this paper, we use a bicycle model [15] to describe the kinematics of the robots and assume the robots steer with perfect rolling and no slipping. We denote a trajectory for robot i as $\mathcal{T}_i(t) = [\mathbf{z}_i(t), \mathbf{u}_i(t)]^T$, $t \in [0, t_f]$, where $\mathbf{z}_i(t)$ and $\mathbf{u}_i(t)$ represent the state and control parameters, and t_f represents the time robot i should reach its target while fully tracking $\mathcal{T}_i$.

Under the above assumptions, to solve the object transportation problem we must obtain trajectories $\mathcal{T}_i$ for each robot which transport the object from its initial configuration $\mathbf{z}_i^s$ to its target configuration $\mathbf{z}_i^g$ while avoiding robot and object collisions. We formulate this as a joint OCP (Problem (1)) as below:

$$\min_{\mathbf{z}_i,\,\mathbf{u}_i,\,t_f} \sum_{i=1}^{N} \int_0^{t_f} \mathbf{u}_i(t)^T \mathbf{W} \mathbf{u}_i(t)\, dt + \omega_t t_f \tag{1a}$$

$$\text{s.t. kinematic constraints,} \tag{1b}$$

$$\underline{\mathbf{z}} \leq \mathbf{z}_i(t) \leq \overline{\mathbf{z}},\ \underline{\mathbf{u}} \leq \mathbf{u}_i(t) \leq \overline{\mathbf{u}}, \tag{1c}$$

$$\mathbf{z}_i(0) = \mathbf{z}_i^s,\ \mathbf{u}_i(0) = \mathbf{u}_i^s, \tag{1d}$$

$$\mathbf{z}_i(t_f) = \mathbf{z}_i^g,\ \mathbf{u}_i(t_f) = \mathbf{u}_i^g, \tag{1e}$$

$$t_f > 0, \tag{1f}$$

$$\mathcal{G}(\mathbf{z}_i(t),\, \mathbf{u}_i(t),\, t) \preceq 0, \tag{1g}$$

$$\forall t \in [0,\, t_f], \quad \forall i \in \{1, 2, \ldots, N\},$$

where $\mathbf{W} \in \mathbb{R}^{2\times 2}$ is a diagonal matrix which penalizes control effort, and $\omega_t t_f$ is a time regularization term which minimizes the trajectory execution time. The OCP constraints capture the robots' kinematic constraints (1b), solution feasibility constraints (1c), initial boundary constraint (1d), terminal boundary constraint (1e), and positivity constraints on the terminal time (1f). (1g) comprises the robots' safety constraints, which we describe in Sec. V.

To solve (1), we propose Alg. 1. The remainder of the paper describes each step of Alg. 1 in detail. In summary, we begin by identifying a set of homotopically distinct paths for each robot which make different decisions to avoid obstacles. We build combinations of these paths, one for each robot, and then filter and rank them to form a rich set of initial guesses (line 1-3) to guide the solution of the joint OCP in (1). We then iteratively test each combination of paths in order and solve an independent OCP until a feasible solution is found (lines 4-23). To simplify the OCP while ensuring object collision avoidance, we convert the object height constraint into a set of constraints over the formation. These constraints are iteratively adjusted until the object is collision-free (lines 13-20). We proceed by discussing the initial guess generation in Sec. IV, and the OCP safety constraints in Sec. V.

IV. Initial Guess Generation

In this section, we present our approach for generating initial guesses for formation trajectory optimization.

A. Generating Homotopically Distinct Paths

The solution quality for the OCP in (1) is reliant on having a high-quality initial guess. The initial guess should approximate the homotopy class of the optimal joint trajectory. Therefore, the first step in generating our initial guess is to account for all homotopically distinct paths through the environment. To achieve this, we build a topological probabilistic roadmap (PRM) [16] (see line 1 in Alg. 1) using the centres of the initial and target formation configurations. An example PRM is shown in Fig. 3. Multiple homotopically distinct shortest paths from the initial to target formation configurations are then extracted from the PRM using path-searching techniques (see the blue lines in Fig. 3). We then rewire these paths for each of the robots using a reverse search to connect the robot's initial and target configurations to the path (the dashed red lines in Fig. 3). The search is halted if a

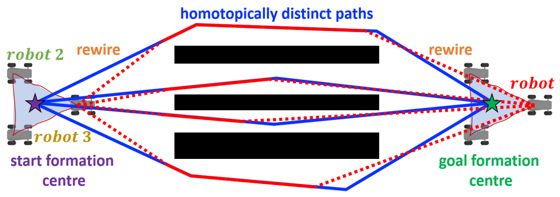

Fig. 3. The outcome of running [16] in a simple environment with three obstacles. The set of homotopically distinct paths are shown in blue and the rewired shortest paths for robot 1 are shown in red.

collision occurs or the robot's initial and target configurations meet. The final robot path combines a segment of the homotopically distinct path (the solid red lines in Fig. 3) and the routes to and from it. By selecting a path for each robot we build a guiding path for the entire formation (line 2).

B. Heuristics Filtering and Sorting

An initial guess for the OCP in (1) can be computed by selecting one of the paths in Sec. IV-A for each robot. This generates a set of combinations of paths $comb_j$. It becomes intractable to solve (1) for each $comb_j$, as the number of homotopy classes increases exponentially with the number of obstacles. Therefore, we introduce a number of heuristics to filter and sort these combinations (see line 3 of Alg. 1). Given a path for each robot, we begin by uniformly discretizing the paths into K waypoints. We then record any obstacles which intersect the line segments between corresponding waypoints on any two paths. If the height of the obstacle is higher than the maximum height of the object in the deformable sheet, that combination of paths is discarded. We also record the Euclidean distance between corresponding waypoints. If the inter-distance exceeds a threshold ϵ_d, the formation is unlikely to be maintained, and thus the combination is discarded. Finally, to prevent the sheet from folding or twisting, we require the robots assigned to the same homotopy class to be adjacent [17], i.e. their labels are consecutive. After completing this initial filtering, we introduce a heuristic cost function to sort the remaining combinations:

$$J = \lambda_1 \cdot J_l + \lambda_2 \cdot J_t + \lambda_3 \cdot J_h + \lambda_4 \cdot J_s. \tag{2}$$

Here, J_l and J_t represent the average path length and the average number of waypoints for all robots, respectively. J_h is 1 if all selected paths are in the same homotopy class and 0 otherwise. J_s counts the number of topological violations using the method proposed in [10], and then estimates the feasibility of homotopy assignments. λ_1, λ_2, λ_3 and λ_4 are weight parameters. The first two terms encourage shorter paths and fewer detours. The third term prioritizes crossing obstacles over avoiding them to avoid robot congestion and exploit the full solution space. The fourth term tries to maintain the consistency of the formation's topology.

C. Joint Trajectory Generation

Given a combination of paths $comb_j$, we now generate a corresponding joint trajectory which can act as an initial guess for the OCP in (1) (line 5). For this, we construct a safe corridor for each robot that ensures the initial guess remains in the same homotopy class as the paths in $comb_j$.

Algorithm 1: The Proposed Formation Planner.

```
1  path ← TopoPRM($map$, $\mathbf{p}^s_{ctr}$, $\mathbf{p}^g_{ctr}$);
2  comb ← IdentifyComb(path, $\{\mathbf{p}^s_i\}^N_{i=1}$, $\{\mathbf{p}^g_i\}^N_{i=1}$);
3  comb* ← Filtering&Sorting(comb);
4  foreach $comb_j \in$ comb* do
5      [$\mathcal{T}^{guess}$, $\mathcal{SC}_j$] ← GenerateInitialGuess($comb_j$);
6      foreach $t \in [0, t_f]$ do
7          // Set min inter-robot distance
8          $\mathcal{C}^H(t) \leftarrow d_0$;
9      GenerateOCP($\mathcal{T}^{guess}$, $\mathcal{SC}_j$, $\mathcal{C}^H$);
10     [$\mathcal{T}$, $is\_failed$] ← Solve (1);
11     if $is\_failed$ then
12         continue the outer loop in line 4;
13     while !SatisfiedHeightCons($\mathcal{T}$) do
14         foreach $t \in [0, t_f]$ do
15             if HeightViolation($\mathcal{T}(t)$) then
16                 if MaximumFormation($\mathcal{C}^H$) then
17                     continue the outer loop in line 4;
18                 foreach $i \in [1, ..., N]$ do
19                     $\mathcal{C}^H_i(t) \leftarrow \max\{l^{max}_i, \mathcal{C}^H_i(t) + \delta\}$;
20         [$\mathcal{T}$, $is\_failed$] ← Solve (1);
21         if $is\_failed$ then
22             continue the outer loop in line 4;
23     return $\mathcal{T}$;
```

Safe corridors are computed by using the waypoints in each robot's path as a seed to compute the corresponding obstacle-free convex polygon using a convex decomposition technique called IRIS [18]. Formally, the safe corridor for robot i using the path in $comb_j$ can be represented as $\mathcal{SC}^i_j = \{q \in \mathbb{R}^2 : \mathbf{A}^i_j q \leq \mathbf{b}^i_j\}$, where $\mathbf{A}^i_j \in \mathbb{R}^{n^i_j \times 2}$ and $\mathbf{b}^i_j \in \mathbb{R}^{n^i_j}$ describe the hyperplane of a convex polygon [18]. n^i_j is the number of hyperplanes, equivalent to the number of edges in the convex region. Using the safe corridors, we then adapt hybrid A^* [19] to find a homotopically consistent, collision-free, and dynamically feasible path for each robot. When expanding a child node in hybrid A^*, it must be constrained to the safe corridor to ensure it is collision-free, and remain homotopically consistent with the selected initial guess, i.e.:

$$\mathbf{p}_i(t) \subset \bigcup_{j=1}^{K} \mathcal{SC}^i_j,\ t \in [0,\ t_f]. \tag{3}$$

These paths are then augmented with a minimum-time velocity profile to form trajectories [20]. Finally, we proportionally relax the velocities of all robots [21] to ensure that they reach their respective target positions simultaneously. The resulting trajectories are recorded in $\mathcal{T}^{guess}$.

V. Iterative Trajectory Optimization

In this section, we describe the safety constraints in OCP (1) and how they are incorporated during optimization.

A. Safety Constraints

Environmental Obstacle Avoidance Constraints. In our framework, we consider the shape of the robots when defining obstacle collision avoidance constraints. Recall that a robot's centroid and orientation are defined as in Fig. 2(a). At time t, the footprint of robot i can be described by the vertices $\{\mathbf{V}_{i,l}(t) \in \mathbb{R}^2 : \mathbf{V}_{i,l}(t) = \mathbf{p}_i(t) + \mathbf{R}(\theta_i(t)) \cdot \mathbf{l}_{i,l},\ l \in \{1, ..., 4\}\}$, where $\mathbf{R}(\theta_i(t))$ is the rotation matrix representing the orientation of the body frame relative to the world frame, and $\mathbf{l}_{i,l}$ is the relative coordinate of the l-th vertex within the body frame. Given the initial trajectories computed in Sec. IV, we run IRIS [18] once more to generate a safe corridor for each robot. To avoid collisions with obstacles and maintain topological equivalence with the initial guess trajectory, robot i's footprint must remain inside the safe corridor at each timestep:

$$\begin{aligned} &\mathbf{A}_i(t) \cdot \mathbf{V}_{i,l}(t) \leq \mathbf{b}_i(t), \\ &\forall i \in \{1, ..., N\},\ l \in \{1, ..., 4\},\ t \in [0,\ t_f]. \end{aligned} \tag{4}$$

Algorithm 2: Function **SatisfiedHeightCons**($\mathcal{T}$).

```
1  foreach $t \in [0, t_f]$ do
2      $\{\mathbf{r}_i\}^N_{i=1}$ ← DeriveRobotPosition($\mathcal{T}(t)$);
3      $\mathbf{P_o}$ ← FK($\{\mathbf{r}_i\}^N_{i=1}$, $\{\mathbf{v}_i\}^N_{i=1}$, $z_r$);
4      $\mathcal{O}_{cross}$ ← DetectObstacle($\mathbf{x_o}$, $\mathbf{y_o}$);
5      $\tilde{h} \leftarrow \min(\mathbf{z_o})$;
6      foreach $obstacle \in \mathcal{O}_{cross}$ do
7          if DeriveHeight($obstacle$) $\geq \tilde{h} + \epsilon_h$ then
8              return false;
9  return true;
```

Inter-Robot Collision Avoidance Constraints. To avoid collisions, robot footprints should never intersect. In our framework, we apply the collision avoidance constraints as represented in [15], which check that each vertex of a robot's footprint lies outside the footprints of the other robots.

Size Safety Constraint and Topological Safety Constraint. During transportation, the robot formation must cooperate to support a deformable sheet. The sheet is of finite length and the distance between any two robots must not exceed this length. Therefore, we constrain the distance between each robot and its neighbours as follows:

$$\|\mathbf{p}_i(t) - \mathbf{p}_{N(i)}(t)\|_2 - l^{max}_i \leq 0,\ \forall t \in [0,\ t_f], \tag{5}$$

where l^{max}_i is the maximum allowable distance between robot i and $N(i)$ (see Fig. 2(a)). As discussed in Sec. III, the topological ordering of the robots should remain fixed during locomotion to prevent the sheet becoming twisted. For this, we apply the topological safety constraints in [10], which ensure that the signed distance from the robot to the lines representing the non-neighboring net edges are non-negative. Formally, given matrix $\mathbf{B} = [0\ \ -1,\ 1\ \ 0]$:

$$\begin{aligned} &\frac{\mathbf{B}(\mathbf{p}_{N(j)} - \mathbf{p}_j)^T}{\|\mathbf{p}_{N(j)} - \mathbf{p}_j\|_2}(\mathbf{p}_i - \mathbf{p}_j) \geq 0, \\ &\forall j \in \{1, 2, \ldots, N\} : j \neq i, N(j) \neq i, \\ &\qquad \forall i \in \{1, 2, \ldots, N\}. \end{aligned} \tag{6}$$

Object Height Constraint. The object's trajectory is highly coupled to robot movement. In the $3D$ coordinate system in Fig. 2(b), the connection between each robot and the object can be taut or slack [22]. With this, there are multiple equilibrium states for the object with different numbers of taut cables. We compute the different equilibrium states of the object by applying the forward kinematics function in [22]:

$$f_{FK}(\{\mathbf{p}_i\}_{i=1}^{N},\, \{\mathbf{v}_i\}_{i=1}^{N},\, z_r) \rightarrow \mathbf{P}_o \subset SE(3). \quad (7)$$

Given the footprints for each robot and the fixed height of the robot contact points with the sheet, we obtain the set of equilibrium states $\mathbf{P}_o$ under all possible combinations of taut cables. This results in a set of corresponding object height values $\mathbf{z_o}$, and x and y coordinates $\mathbf{x_o}$ and $\mathbf{y_o}$, respectively. Let $\mathcal{O}_{cross}$ be the set of obstacles which intersect with the object in the $2D$ plane. To avoid collisions with the object, all obstacles must be lower than $\tilde{h}+\epsilon_h$, where $\tilde{h} = \min(\mathbf{z_o})$, and ϵ_h is a safety margin. This process is summarized in Alg. 2.

Terminal Constraints. It is challenging for car-like robots to precisely reach a target position. Therefore, to improve smoothness at the end of the formation trajectory, we relax the terminal constraints in (1e) to $\|\mathbf{p}_o(t_f) - \mathbf{p}_o^{target}\| \leq \epsilon_t$, where $\mathbf{p}_o^{target}$ is the target position of the object and ϵ_t is a distance threshold. The terminal state of the object is approximated as the average of the robot terminal positions, i.e. $\mathbf{p}_o(t_f) = \sum_{i=1}^{N} \mathbf{p}_i(t_f)$. This assumes the object remains near the geometric center of the sheet, which holds under quasi-static object motion and an inelastic sheet.

B. Iterative Process

The initial guesses computed in Sec. 2 may be imperfect as they lack information about the kinematic constraints of the robots. Further, the complex height safety constraints in Alg. 2 make (1) challenging to solve directly. To mitigate these issues, we propose a two-layer iterative framework, as illustrated in Alg. 1. In the inner loop (lines 13 to 20 of Alg. 1), we iteratively add formation constraints at any timestep where the height constraint is violated:

$$\|\mathbf{p}_i(t) - \mathbf{p}_{N(i)}(t)\|_2 - \mathcal{C}_i^H(t) \geq 0,\, \forall t \in [0,\, t_f], \quad (8)$$

where $\mathcal{C}_i^H(t)$ is the minimum allowable distance between adjacent robots at time t. Whenever an object height violation is detected, $\mathcal{C}_i^H(t)$ is iteratively increased by δ, starting from an initial default distance d_0 (line 19). This raises the sheet and consequently the object. If the maximum distance between robots is reached (line 16) or any other constraints are violated (lines 11 and 21), our planner fails and starts again from the next initial guess. This process is repeated until a feasible solution is found. Note that the loop in lines 4-23 can be executed in parallel to improve solution time as each combination of paths is independent.

VI. Experimental Results

In this section, we demonstrate the efficacy of our approach in simulation and on hardware. All simulations are run on Ubuntu 20.04 with an Intel Core i9 processor @ 3.2GHz and 16GB of RAM. All software is implemented in C++ using the robot operating system (ROS) [23]. To solve the OCP in (1), we use the explicit Runge-Kutta method [24] to construct the nonlinear programming (NLP) problem, ADOL-C [25] for automatic differentiation, and the primal-dual interior-point solver IPOPT [26] to solve the NLP. Experimental runs can be viewed in the supplementary video[2]. All parameter settings are described in the documentation for our open-source software release.

A. Simulation Experiment

1) Ablation Study: We conducted an ablation study to demonstrate the benefits of the initial guesses in Sec. IV. We compared our framework in Alg. 1 to a modified version, where hybrid A^* computes the initial guess in (1) without identifying multiple homotopy classes. All experiments take place in a maze environment with cuboid obstacles of heights ranging from 0.4-0.8m, as shown in Fig. 4. We generated 100 problem instances with randomised initial and goal configurations which are consistent across both methods. We then evaluate the success rate (SR), object height ($height$), task completion time (t_f), and control efforts (c_e) for both methods, as shown in Table I. We also present the average/maximum formation similarity error $\bar{e_f}$ and e_f^{max} used in [27], which describes how much the formation adjusts its shape. The version of our framework without an initial guess can only identify a single homotopy class, which requires significant deformation of the formation to maneuver through the congested passage in Fig. 4(a). This results in greater variation in the object height during transportation, and in 43 instances, the poor initial guess resulted in an infeasible solution. In contrast, our method benefits from a more comprehensive exploration of the solution space, achieving a 100% success rate while also improving trajectory quality and transportation stability (see Fig. 4(b)).

2) Comprehensive Analysis: We do not compare against state-of-the-art methods as the algorithms proposed in [3]–[8] only apply to omnidirectional robots, and the approaches in [9], [10] lack the capability to navigate through obstacles. To evaluate the robustness of our framework, we consider object transportation in a more complex $60m \times 60m$ environment. We considered formations of three and four robots, where each robot's footprint is a $1m \times 0.8m$ rectangle. The initial and target positions of the formation center are set to $(-35,\, -35)m$ and $(35,\, 35)m$, respectively. We consider problems with 20, 40, 60, 80, and 100 cuboid obstacles. For each number of obstacles, we randomly generate 100 problem instances, where obstacles have dimensions $1.5 \times 0.75m$ or $2 \times 1m$, and the obstacle height is uniformly sampled in the range 0.4-0.8m. For each instance, we record a failure if a solution is not found within 60 seconds. We record the success rate (SR), the planning time (the combined time for initial guess generation and trajectory optimization), and the average iterations for the outer and inner loops of Alg. 1. We present the results in Table II.

[2]https://youtu.be/DvEniLKDaH0

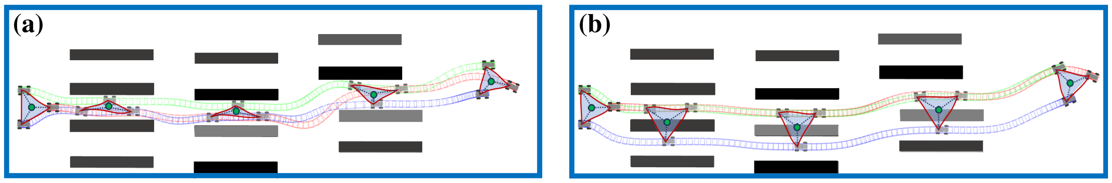

Fig. 4. The generated trajectories for the ablation study. (a) Without heuristic initial guess generation. (b) With heuristic initial guess generation.

TABLE I
FRAMEWORK PERFORMANCE WITH AND WITHOUT HEURISTIC INITIAL GUESS GENERATION.

Method	SR	$\bar{e_f}$	e_f^{max}	$height\,(m)$	$t_f\,(s)$	c_e
w\o Sec. IV	57%	0.1647 ± 0.0628	0.2992 ± 0.1271	0.5168 ± 0.4315	84.6118 ± 13.0238	0.5468 ± 0.3148
w Sec. IV	**100%**	$\mathbf{0.0644 \pm 0.0135}$	$\mathbf{0.1981 \pm 0.0645}$	$\mathbf{0.5913 \pm 0.1385}$	$\mathbf{78.1998 \pm 14.2406}$	$\mathbf{0.2411 \pm 0.1374}$

TABLE II
FRAMEWORK PERFORMANCE AS THE NUMBER OF ROBOTS AND OBSTACLES INCREASE.

Obstacles	SR	Planning Time (s)		Avg. Iterations	Obstacles	SR	Planning Time (s)		Avg. Iterations
(3 robots)		IG	TO	O (I)	(4 robots)		IG	TO	O (I)
20	99%	4.25 ± 2.41	4.39 ± 5.54	1.04 (1.14)	20	97%	4.19 ± 2.58	7.81 ± 10.57	1.08 (1.08)
40	94%	5.72 ± 2.98	7.43 ± 6.94	1.49 (1.51)	40	86%	4.55 ± 2.29	8.15 ± 8.76	1.12 (1.05)
60	89%	7.27 ± 2.09	12.27 ± 12.58	1.49 (1.51)	60	82%	5.56 ± 1.56	15.10 ± 16.06	1.33 (1.11)
80	79%	8.26 ± 2.73	12.59 ± 11.15	1.41 (1.65)	80	66%	6.86 ± 1.97	16.16 ± 11.68	1.33 (1.09)
100	71%	9.74 ± 2.25	16.46 ± 13.65	1.55 (1.93)	100	57%	9.76 ± 2.71	18.22 ± 17.79	1.37 (1.37)

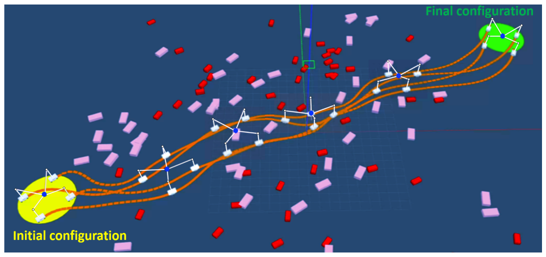

Fig. 5. Example trajectories (orange) for a formation of 4 car-like robots. The transported object is a blue sphere, and the pink and red areas indicate obstacles. Yellow and green areas show the initial and target formation configurations, respectively. White virtual cables are shown instead of a sheet as they are simpler to visualize in Gazebo.

As the number of obstacles and robots increase, the success rate decreases. This is because the increased environment complexity reduces the feasible solution space, increasing the difficulty of planning. Similarly, Alg. 1 requires more outer iterations to explore different topologies, and more inner iterations to impose the object height constraints. However, the total number of iterations remains low due to the quality of initial guesses generated in Sec. IV. In all successful instances, our planner generates safe and feasible collision-free trajectories. Fig. 5 shows example trajectories in Gazebo for four car-like robots in a scenario with 100 obstacles. The object's position is estimated by selecting the lowest height equilibrium state obtained through forward kinematics [22]. Our framework adjusts the formation to adapt to the obstacles while ensuring safety.

B. Real-World Experiment

We also evaluated our framework in a real indoor scenario with cuboid obstacles, as shown in Fig. 6. Due to robot availability we used three Turtlebots (see Fig. 1), which are

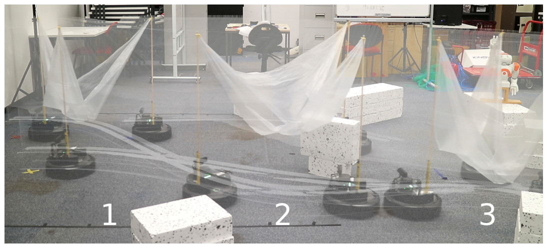

Fig. 6. Three snapshots of three Turtlebots transporting an object with trajectories overlaid, where the formation expands to cross an obstacle.

not car-like robots. We demonstrated the performance of our framework on car-like robots in Sec. VI-A. Robot states are obtained through ROS, and a closed loop controller adapted from [28] tracks robot trajectories at 100Hz. We consider three problem instances with different obstacle heights but the same start and end points. Fig. 6 illustrates robot trajectories for one problem instance, with snapshots taken at three points along the trajectory. Here, the formation begins in its default shape, expands to cross an obstacle, and then contracts back to the smaller shape once the obstacle is cleared.

VII. CONCLUSION

In this paper, we proposed a motion planning framework for collaborative object transportation using a team of nonholonomic robots and a deformable sheet. Our iterative two-stage framework synthesizes high quality trajectories while ensuring completeness. We improve the efficiency of our solver by decoupling object height constraints from the optimization problem. The efficacy of our framework has been validated in simulation and in the real world. In future work, we will improve our framework to account for dynamic obstacles such as humans.

References

[1] E. Tuci, M. H. Alkilabi, and O. Akanyeti, "Cooperative object transport in multi-robot systems: A review of the state-of-the-art," *Frontiers in Robotics and AI*, vol. 5, p. 59, 2018.

[2] B. Roy, A. Basmajian, and H. H. Asada, "Repositioning of a rigid body with a flexible sheet and its application to an automated rehabilitation bed," *IEEE Transactions on automation science and engineering*, vol. 2, no. 3, pp. 300–307, 2005.

[3] J. Hu, W. Liu, H. Zhang, J. Yi, and Z. Xiong, "Multi-robot object transport motion planning with a deformable sheet," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9350–9357, 2022.

[4] K. Hunte and J. Yi, "Collaborative object manipulation through indirect control of a deformable sheet by a mobile robotic team," in *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*, pp. 1463–1468, IEEE, 2019.

[5] K. Hunte and J. Yi, "Collaborative manipulation of spherical-shape objects with a deformable sheet held by a mobile robotic team," *IFAC-PapersOnLine*, vol. 54, no. 20, pp. 437–442, 2021.

[6] K. Hunte and J. Yi, "Pose control of a spherical object held by deformable sheet with multiple robots," *IFAC-PapersOnLine*, vol. 55, no. 37, pp. 414–419, 2022.

[7] J. Alonso-Mora, R. Knepper, R. Siegwart, and D. Rus, "Local motion planning for collaborative multi-robot manipulation of deformable objects," in *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 5495–5502, IEEE, 2015.

[8] W. Liu, J. Hu, H. Zhang, M. Y. Wang, and Z. Xiong, "A novel graph-based motion planner of multi-mobile robot systems with formation and obstacle constraints," *IEEE Transactions on Robotics*, 2023.

[9] S. Chand, P. Verma, R. Tallamraju, and K. Karlapalem, "Transportation of deformable payload through static and dynamic obstacles using loosely coupled nonholonomic robots," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pp. 228–230, IEEE, 2019.

[10] L. Pei, J. Lin, Z. Han, L. Quan, Y. Cao, C. Xu, and F. Gao, "Collaborative planning for catching and transporting objects in unstructured environments," *IEEE Robotics and Automation Letters*, 2023.

[11] F. Nadon, A. J. Valencia, and P. Payeur, "Multi-modal sensing and robotic manipulation of non-rigid objects: A survey," *Robotics*, vol. 7, no. 4, p. 74, 2018.

[12] M. Ruan, D. McConachie, and D. Berenson, "Accounting for directional rigidity and constraints in control for manipulation of deformable objects without physical simulation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 512–519, IEEE, 2018.

[13] Y. Liu, Z. Cao, H. Xiong, J. Du, H. Cao, and L. Zhang, "Dynamic obstacle avoidance for cable-driven parallel robots with mobile bases via sim-to-real reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 8, no. 3, pp. 1683–1690, 2023.

[14] T. Rasheed, P. Long, A. S. Roos, and S. Caro, "Optimization based trajectory planning of mobile cable-driven parallel robots," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6788–6793, IEEE, 2019.

[15] W. Zhang, C. Street, and M. Mansouri, "A decoupled solution to heterogeneous multi-formation planning and coordination for object transportation," *Robotics and Autonomous Systems*, p. 104773, 2024.

[16] B. Zhou, J. Pan, F. Gao, and S. Shen, "Raptor: Robust and perception-aware trajectory replanning for quadrotor fast flight," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1992–2009, 2021.

[17] R. Herguedas, M. Aranda, G. López-Nicolás, C. Sagüés, and Y. Mezouar, "Double-integrator multirobot control with uncoupled dynamics for transport of deformable objects," *IEEE Robotics and Automation Letters*, 2023.

[18] R. Deits and R. Tedrake, "Computing large convex regions of obstacle-free space through semidefinite programming," in *Proceedings of Algorithmic Foundations of Robotics XI: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics*, pp. 109–124, Springer, 2015.

[19] D. 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, no. 5, pp. 485–501, 2010.

[20] T. Lipp and S. Boyd, "Minimum-time speed optimisation over a fixed path," *International Journal of Control*, vol. 87, no. 6, pp. 1297–1311, 2014.

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

[22] J. Hu, W. Liu, J. Yi, and Z. Xiong, "Forward kinematics of object transporting by a multi-robot system with a deformable sheet," *IEEE Robotics and Automation Letters*, 2024.

[23] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "Ros: An open-source robot operating system," in *Proceedings of the ICRA Workshop on Open Source Software*, vol. 3, p. 5, 2009.

[24] P. J. van der Houwen and B. P. Sommeijer, "Explicit runge–kutta (–nyström) methods with reduced phase errors for computing oscillating solutions," *SIAM Journal on Numerical Analysis*, vol. 24, no. 3, pp. 595–617, 1987.

[25] A. Walther and A. Griewank, "Getting started with adol-c.," *Combinatorial scientific computing*, vol. 1, no. 06.02, 2009.

[26] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, pp. 25–57, 2006.

[27] Q. Liu, B. Tian, X. Zhang, J. Lu, and Z. Li, "Sampling-based hierarchical trajectory planning for formation flight," *arXiv preprint arXiv:2407.17392*, 2024.

[28] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," in *Proceedings., IEEE International Conference on Robotics and Automation*, pp. 384–389, IEEE, 1990.