POLITECNICO DI TORINO
Repository ISTITUZIONALE

Multi-agent cooperative transport across partially unknown complex obstacles with online mapping

Original
Multi-agent cooperative transport across partially unknown complex obstacles with online mapping / Blengini, C.L., Cheng, P.D.C., Indri, M.. - ELETTRONICO. - (2026). (2026 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM) Genova (Ita) 07-11 July 2026) [10.1109/aim65483.2026.11658340].

Availability:
This version is available at: 11583/3015089 since: 2026-09-02T08:28:04Z

Publisher:
IEEE

Published
DOI:10.1109/aim65483.2026.11658340

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2026 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

16 November 2026

Multi-agent cooperative transport across partially unknown complex obstacles with online mapping

Cesare Luigi Blengini, Pangcheng David Cen Cheng, Marina Indri
Dipartimento di Elettronica e Telecomunicazioni - Politecnico di Torino
Corso Duca degli Abruzzi, 24, 10129, Torino, Italy
{cesare.blengini, pangcheng.cencheng, marina.indri}@polito.it

***Abstract*—Cooperative transportation with multiple mobile robots allows carrying out tasks that exceed the capabilities of individual agents. In this paper, we propose a framework that coordinates a team of robots to transport objects via cable-tethering. The framework enables a formation control so that path planning includes (i) payload constraints, (ii) deformation to cross complex obstacles, and (iii) online updates about the environmental changes. The approach has been tested with a team of TurtleBot3 Burger robots in ROS 2, in simulated scenarios, and validated in a real-world environment.**

***Index Terms*—Multi-robot systems, cooperative transport, formation control, complex obstacles**

I. Introduction

Systems with multiple mobile robots are increasingly employed for cooperative object transportation due to their improved load capacity, fault tolerance, and cost efficiency compared to single large platforms [1]. Cooperative transport can be achieved through pushing/caging strategies or by physically coupling the robots to the payload, for instance, via cables. A transport system with cables can be modeled as a formation control problem, where maintaining appropriate relative configurations ensures load balance and payload stability. Several works demonstrate formation-based transport using aerial [2] or ground robots [3]. However, these approaches are typically validated in environments with limited obstacle complexity.

The possibility of deforming the formation can significantly increase the robots' navigation capabilities in cluttered environments, since it enables contraction to pass through narrow passages, as shown in [4]–[7]. Moreover, deforming the formation also enables obstacle-crossing maneuvers, in which robots move on opposite sides of an obstacle while maintaining payload constraints. Strategies like split-and-merge have been proposed for obstacle bypassing [8], [9]. However, these approaches assume loosely coupled robot groups that can move almost independently. In cooperative transportation, the shared payload imposes strict coordination and balancing constraints, preventing independent motion of subgroups during obstacle crossing.

Recent path-planning approaches explicitly account for formation geometry to address coordinated deformation and crossing. For instance, sampling-based methods [10]–[12] explore the high-dimensional configuration space and allow transitions between formation shapes. Optimization-based techniques [13], [14] formulate deformable transport as constrained optimal control problems, though they often suffer from high computational cost or sensitivity to initialization. Graph-based complete planners [15] discretize both workspace and formation space to guarantee optimality, but scalability depends on the grid resolution and the number of formation candidates. Note that these methods assume static environments and rely on precomputed structures that are difficult to adapt online. Other works address dynamic obstacles, for example, by modifying the Dynamic Window Approach (DWA) parameters [16], slowing down the formation [17], or learning avoidance policies via reinforcement learning [18]. However, these studies consider relatively simple environments and small obstacles. They do not address coordinated crossing of complex non-convex structures, such as fissures and crevasses, which could be encountered in space exploration scenarios.

Therefore, planning cooperative transport should consider maneuvers that (i) respect payload balancing constraints, (ii) exploit formation deformation for crossing complex non-convex obstacles, and (iii) adapt online to environmental changes, which represent a still largely unexplored issue. In this work, we propose a framework for cooperative transport with non-holonomic ground robots carrying a cable-suspended payload, which enables coordinated crossing of complex non-convex obstacles with online replanning to handle changes in the environment of the initial map.

The rest of the paper is structured as follows: Section II overviews the proposed framework. The transportation model is presented in Section III, while the obstacle crossing is described in Section IV. Section V introduces the formation control for the mobile robots, which navigate in the map illustrated in Section VI. Simulation and experimental results are presented in Section VII and VIII, respectively. Section IX concludes the paper with some analysis for future research.

II. The proposed architecture

The core of the proposed framework is the *Joint A** algorithm, which employs grid decomposition to compute a distance-optimal path that enables coordinated crossing of complex non-convex obstacles. Unlike existing graph-based approaches, the Joint A* does not rely on precomputed data structures, making it suitable for online replanning on an updated map. Similar to [14], payload dynamics and interaction forces are not explicitly modeled. Instead, payload balancing requirements are encoded as geometric constraints

This study was carried out within the Space It Up project funded by the Italian Space Agency, ASI, and the Ministry of University and Research, MUR, under contract n. 2024-5-E.0 - CUP n. I53D24000060005.

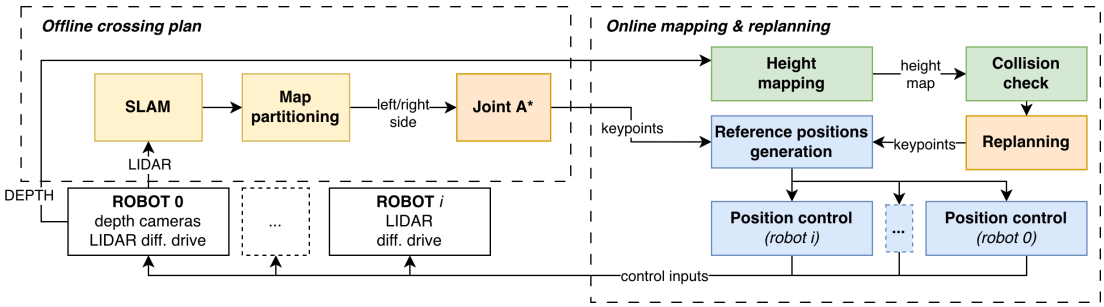

Fig. 1. The overall workflow of the proposed obstacle crossing framework.

within the planning process, allowing the method to remain computationally feasible while preserving load stability.

As shown in Figure 1, the architecture includes an offline phase and an online one. In the offline phase, the environment is mapped with a LiDAR to scan the obstacle's outline while neglecting its height; then, the Joint A* is used to generate an initial crossing path. During the online phase, the robots track the planned trajectory through a decentralized position control. In parallel, a system of depth cameras updates the obstacle's height information and detects any new obstacles. If a potential collision with other robots or payload is detected, a local segment of the path is replanned online without stopping the formation.

III. The Transportation System Model

A multi-robot transportation system (Figure 2) showcases the performance of the proposed framework. It consists of N homogeneous non-holonomic robots, each equipped with a vertical pole of length h supporting the payload through a cable of length l_r (Figure 2(a)).

1) Formation model: The robots are arranged into two symmetric groups, referred to as *side-formations*, to enable obstacle crossing while preserving payload balance. Each side-formation consists of robots equally spaced along an arc of a virtual circumference $C_{balance}$ of radius $r_{balance}$. The payload is suspended between the two side-formations, and its height depends on $r_{balance}$. Equal angular spacing within each side-formation ensures that all cables remain stretched and that the payload weight is evenly distributed. Each side-formation is approximated by its minimum enclosing circle C_{side} with radius r_{side} (see Figure 2(b)). This circular approximation is used for planning and collision checking.

2) Crossing constraints: The *complex obstacle crossing algorithm* computes trajectories for the two side-formations, each containing an arbitrary number of robots. The separation between the side-formations is defined as the Euclidean distance $d(c_1, c_2)$ between the centers c_1 and c_2 of their enclosing circles. Such an inter-formation distance $d(c_1, c_2)$ is not directly controlled; it varies, instead, as a function of the radius $r_{balance}$ of the virtual circumference $C_{balance}$. Variations of $r_{balance}$ change the geometric setup of the system and, hence, the distance between the two side-formations. The admissible variation Δd is bounded by the system's physical parameters, e.g., l_r, h, and the robots' payload capacity.

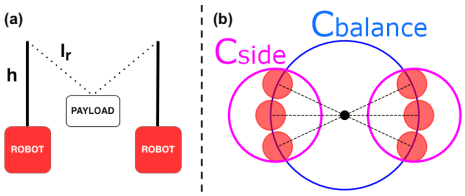

Fig. 2. (a) How the robots hold the payload. (b) The two side-formations.

IV. Obstacle Crossing Strategy

Two paths are computed to cross an obstacle, one for each side-formation, so they can pass on opposite sides of the obstacle (left and right). Small or convex obstacles can be handled using simple geometry-based strategies. In contrast, crossing large or non-convex obstacles is non-trivial; for this reason, the proposed crossing algorithm formulates the problem as a joint path planning task for two agents, each one representing a side-formation.

Since the robots operate in a planar environment, the position of each agent is represented by $p_i = (x_i, y_i)$, corresponding to the center of the side-formation's enclosing circle. Joint planning for the two agents is therefore performed in a joint state space $js = \big((x_1, y_1), (x_2, y_2)\big) \in \mathbb{R}^4$.

The two agents must satisfy a distance constraint throughout the motion, which acts as a constraint-based pruning mechanism, significantly reducing the feasible joint search space. However, joint planning over the entire environment remains computationally expensive, especially at high map resolutions. For this reason, this work restricts joint path planning to regions containing crossable obstacles. In all other cases, the formation could be treated as a single entity. Note that a global planning approach over the full environment is beyond the scope of this paper.

A. Obstacle processing

Large obstacles typically admit only one feasible crossing direction, enabling search-space reduction by assigning each side-formation to a specific side of the obstacle.

The obstacle outline is assumed to be previously obtained through SLAM performed by a single robot equipped with a LiDAR. This process produces an occupancy map, where the colors white, black, and gray denote free, occupied, and unknown areas, respectively. The free space around the

obstacle is partitioned into two regions, either manually or via the main branch of the obstacle medial axis (Figure 3).

Fig. 3. The original obstacle map and its partition into a left side (red) and right side (cyan).

B. Joint A* algorithm

Paths are computed using a modified A* algorithm that jointly plans a distance-minimizing trajectory for the two agents. Each agent can move in eight directions or remain stationary, resulting in $9^2 = 81$ possible joint actions at each expansion step. To enforce the distance constraint between agents, the inter-agent distance is evaluated at every candidate state. The states that violate the admissible distance bounds are discarded from the search space. Each agent motion is associated with a cost that depends on the direction: horizontal and vertical have a unit cost, while diagonals have a cost equal to $\sqrt{2}$. The cost of a joint action is the sum of the individual motion costs of the two agents.

The heuristic function is defined as the sum of the octile distances between each agent and its respective goal. For an agent at position $p = (x, y)$ and a goal position $g = (x_g, y_g)$, the octile distance is given by

$$h_{\text{oct}}(p, g) = \max(\Delta x, \Delta y) + (\sqrt{2} - 1)\min(\Delta x, \Delta y), \quad (1)$$

where $\Delta x = |x - x_g|$ and $\Delta y = |y - y_g|$. The overall heuristic is therefore $h(js) = h_{\text{oct}}(p_1, g_1) + h_{\text{oct}}(p_2, g_2)$.

V. Distributed formation control implementation

Each side-formation contains an equal number of non-holonomic mobile robots operating on a 2D workspace. For each robot, the planning layer provides a feasible sequence of reference positions generated from the side-formation path. The control strategy tracks the planned references while preserving coordinated formation motion, inspired by [19]. Note that only velocity commands are available for control.

A. Position-based control strategy

The kinematics of mobile robot i is modeled as a unicycle:

$$\dot{x}_i = v_i \cos\theta_i, \quad \dot{y}_i = v_i \sin\theta_i, \quad \dot{\theta}_i = \omega_i, \quad (2)$$

where (x_i, y_i, θ_i) denotes the robot pose and (v_i, ω_i) are the linear and angular control inputs.

Rather than generating reference velocities, the control architecture adopts a position-based approach. This choice avoids the difficulties associated with producing curvature-continuous velocity references for non-holonomic robots, especially when the formation undergoes geometric changes, e.g., variations in radius or curvature. Each robot is then assigned a desired position $\mathbf{p}_i^{des}$, computed from the current path segment and a local progression variable.

A body-frame position controller is used to generate admissible velocity commands. The position error is first transformed into robot-local coordinates:

$$\begin{aligned} e_x &= \cos\theta_i\, \Delta x_i + \sin\theta_i\, \Delta y_i, \\ e_y &= -\sin\theta_i\, \Delta x_i + \cos\theta_i\, \Delta y_i, \end{aligned} \quad (3)$$

where $\Delta x_i = x_i^{des} - x_i$ and $\Delta y_i = y_i^{des} - y_i$. The control law is defined as:

$$v_i = k_v\, e_x, \quad \omega_i = k_\omega\, \text{atan2}(e_y, e_x), \quad (4)$$

with $k_v, k_\omega > 0$. This formulation allows smooth motion with mild backward corrections when the goal lies behind the robot.

B. Formation compactness via path coordination

The path is divided into M segments connecting successive formation keypoints. Within each segment s, a local progression variable $\alpha_s \in [0, 1]$ parametrizes the interpolation between the start and goal settings. For a given formation geometry $\mathcal{F}_s$, the desired position of robot i is

$$\mathbf{p}_i^{des}(\alpha_s) = f_i(\alpha_s, \mathcal{F}_s). \quad (5)$$

When α_s reaches 1, the system advances to segment $s + 1$ with $\alpha_{s+1} = 0$.

To maintain formation compactness, the evolution of α_s is defined by a slowdown function that depends on the maximum tracking error:

$$\Delta\alpha_s = \frac{\gamma(e_{max})}{T_s}\Delta t, \quad (6)$$

where T_s is the nominal traversal time for a segment s, Δt is the control step and the slowdown factor γ is defined as:

$$\gamma(e_{max}) = \frac{1}{1 + \left(\frac{e_{max}}{E_{ref}}\right)^p}. \quad (7)$$

Here, $e_{max} = \max_i \|\mathbf{p}_i^{des} - \mathbf{p}_i\|$, E_{ref} is a reference error threshold, and p controls the slowdown's intensity. This creates a self-stabilizing feedback loop: as tracking errors increase, the reference progression slows down, allowing robots to recover; as errors decrease, the formation resumes nominal speed.

C. Decentralized coordination via max-consensus

A max-consensus protocol is employed to compute e_{max} without centralized coordination. Robots communicate only with neighbors defined by a connected communication graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{N}_i$ denotes the neighbor set of robot i.

At each control step, every robot initializes its local estimate with its current tracking error, $me_i^{(0)} = e_i$. The estimate is then iteratively updated by exchanging information with neighbors:

$$me_i^{(k+1)} = \max\left(me_i^{(k)}, \max_{j \in \mathcal{N}_i} me_j^{(k)}\right), \quad (8)$$

where k denotes the iteration index within a single control step. Under standard assumptions on graph connectivity, the protocol converges to the true maximum error in a number of iterations equal to the graph diameter. By reinitializing the estimates at each control step, the protocol correctly tracks a time-varying maximum. Each robot then computes its own

slowdown factor $\gamma(me_i)$ and updates its local path progression α_s independently. Although consensus ensures that all robots converge to the same estimate of e_{max}, the path variables α_s and segment indices are maintained locally by each robot, resulting in a fully decentralized architecture that relies only on neighbor-to-neighbor communication.

VI. Local map enhancement and replanning

The global planning map may not accurately consider execution-time conditions due to sensor noise, incomplete prior knowledge, or environmental changes. Also, global planners assume obstacles as planar, ignoring height information that is critical for safe payload transportation and can lead to collisions. To overcome these limitations, we introduce a local map update and a replan strategy that integrates online perception with the global plan. During execution, one robot acts as a coordinator, monitoring the scenario through depth measurements from RGB-D cameras mounted on its pole to provide a 360° view of the environment. The locally updated map is then used to detect potential collisions and trigger replanning when needed.

A. Local map enhancement

The height information is encoded in an RGB map, where colors represent different elevation levels. The initial occupancy map is converted to RGB, and during robot motion, the map regions observed by the cameras are updated, as shown in Figure 4. Each pixel of the visible map is classified using the z value obtained from the point cloud generated from depth measurements:

- $z = 0$ is depicted in white (or cyan), it is the ground plane (free space).
- $z > 0$ is represented by colors ranging from green (lower heights) to red (higher elevations).
- $z < 0$ is kept in black.

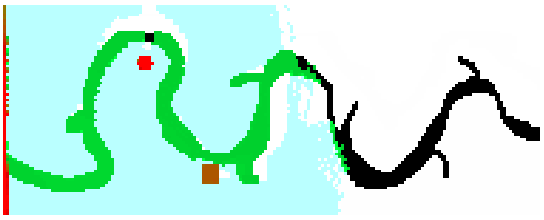

Fig. 4. The height map being built during the update process.

1) Point cloud generation: With the camera intrinsics, depth measurements are converted into a point cloud expressed in the camera frame $\{C\}$. The frame $\{C\}$ is centered at the camera optical center, with axes (X_c, Y_c, Z_c), where Z_c aligns with the optical axis and (X_c, Y_c) lie on the image plane. The points are then transformed into the map frame $\{W\}$, whose axes (X_w, Y_w, Z_w) are defined such that the X_w–Y_w plane coincides with the ground and Z_w points upward. The transformation is given by $\mathbf{p}_w = \mathbf{R}_{wc}\,\mathbf{p}_c + \mathbf{t}_{wc}$, where $\mathbf{R}_{wc}$ and $\mathbf{t}_{wc}$ are the rotation matrix and the translation vector, respectively, between the camera and map frames. This transformation depends on the camera mounting position and the robot's current pose.

2) Camera sensor fusion: Multiple RGB-D cameras are used to approximate a 360° field of view, with partially overlapping views. During each update cycle, cameras are processed sequentially and overwrite the corresponding map pixels, ensuring that only the most recent updates are kept.

3) Masking regions: To prevent other robots from being incorporated into the map as obstacles, points within a masking region are excluded from the update. This region is defined by the current footprint of the two side-formations and is inflated by a small safety margin to account for tracking errors and localization uncertainty.

B. Path replanning

The global planner generates two sequences of keypoints of equal length, one for each side-formation, indexed by $m = 0, \ldots, M-1$. Each keypoint represents the planned position of a side-formation along the path. After each local map update, the coordinator robot performs collision checks along the planned trajectories. The side-formation is approximated by a circular footprint of radius r_{side} centered at each keypoint position. A collision occurs if any map cell intersecting this footprint is not labeled as free space. For each pair of keypoints at index m, the corresponding payload and cable configurations are uniquely determined from the two side-formation positions and are checked against the height information stored in the map.

The collision checks start $n+1$ steps ahead of the current index m_{cur} and proceed for at most $n_{\max}$ steps or until the end of the keypoint sequence. This ensures that only future portions of the path are considered, assuming the next $n+1$ steps are collision-free due to the long sensing range of the depth cameras. Let m_{first} denote the first detected collision index. When a collision is found, a contiguous segment of the trajectory is selected for replanning. The start index is defined as $m_{\text{start}} = m_{\text{first}} - n$, with $m_{\text{start}} > m_{\text{cur}}$ by construction. The end index m_{end} is the smallest index greater than m_{first} such that no collisions are detected for n consecutive steps.

Note that path replanning is not triggered immediately due to transient or partially observed obstacles. Instead, each candidate segment $(m_{\text{start}}, m_{\text{end}})$ is stored in a fixed-size queue of length M_q. Replanning behavior is only executed when all queue elements are identical, indicating a persistent collision across multiple map updates.

1) Joint A with additional state validation:* The path is replanned using the Joint A* algorithm described in Section IV-B. During state expansion, an additional validation step is introduced to account for payload and cable constraints. Those candidate joint states that violate these constraints are discarded. Although this increases the computational cost of the search, replanning is restricted to short path segments, making the additional overhead negligible in practice.

2) Consistency of the control parameters: When replanning occurs, only path segments ahead of the current formation position are modified. Since path progression is locally

parametrized within each segment, the current segment index and the local progression α_s remain valid.

In limited-connectivity topologies, updated keypoints may propagate with some delay, temporarily causing inconsistent trajectory information across robots. However, because replanning only affects future segments, synchronization can be successfully completed before the formation reaches the modified portion.

VII. Simulation tests

A simulation was conducted with six TurtleBot3 Burger robots in Gazebo 11 to evaluate: (i) traversal of large non-convex obstacles while maintaining formation constraints, (ii) integration of obstacle height into the map representation, (iii) online replanning, and (iv) robustness to temporary robot failures. The simulation was run on a laptop running ROS2 Humble with an Intel Core i7-14650HX, 16GB of RAM, and an NVIDIA Laptop GeForce RTX 5070.

1) Scenario: Two three-robot side-formations with ring-topology communication were employed. The central-to-lateral robot distance was set to $30\,\mathrm{cm}$, yielding an approximate circular footprint of $r_{\text{side}} = 50\,\mathrm{cm}$.

The mechanical configuration was defined as follows:

- Pole length: $h = 135\,\mathrm{cm}$ (total robot height $160\,\mathrm{cm}$).
- Rope length: $l_r = 190\,\mathrm{cm}$.
- Resulting payload height: $13\,\mathrm{cm}$ at $d_{\min} = 240\,\mathrm{cm}$ and $100\,\mathrm{cm}$ at $d_{\max} = 360\,\mathrm{cm}$.

The perception was provided by five depth cameras (resolution 120×160 pixels), mounted at a height of $140\,\mathrm{cm}$ with $45°$ inclination.

The environment was represented by a 200×100 grayscale map ($1\,\mathrm{px} = 10\,\mathrm{cm}$), with an $8\,\mathrm{px}$ dilation for global planning and a circular collision footprint of radius $5\,\mathrm{px}$ for replanning; start and goal positions were manually selected.

2) Results and Discussion: A video of the complete experiment is available at [20]. The initial planned path is shown in Figure 5. The global plan was computed in $30\,\mathrm{s}$, with path lengths of $29.91\,\mathrm{m}$ (left side-formation) and $30.38\,\mathrm{m}$ (right).

With control parameters $E_{ref} = 10\,\mathrm{cm}$ and $p = 2.0$, the controllers achieved a mean lateral error of $1.2\,\mathrm{cm}$ (max $7.5\,\mathrm{cm}$). The maximum tracking error propagated through consensus was $13.2\,\mathrm{cm}$. while the maximum deviation from the desired central-to-lateral robot distance was $2.5\,\mathrm{cm}$.

To evaluate online adaptability, three previously unknown obstacles were added to the map: one $50 \times 50 \times 50\,\mathrm{cm}$ box and two cylinders (radius $20\,\mathrm{cm}$, heights $70\,\mathrm{cm}$ and $90\,\mathrm{cm}$). During execution, five replanning events occurred, updating between 15 and 24 segments per event with computation times between $0.28\,\mathrm{s}$ and $1.15\,\mathrm{s}$. As shown in Figure 6, the trajectory was locally adjusted to avoid the obstacles without stopping the formation at moderate speeds.

Robustness was assessed by halting one robot for $10\,\mathrm{s}$. The remaining robots progressively slowed down until the formation stopped, inducing a deformation of $C_{balance}$ proportional to the control parameters E_{ref} and p (Figure 7).

Fig. 5. The initial path computed for the simulation test. The $\star$ and $\bullet$ represent the goal and start position of the side-formation center, respectively.

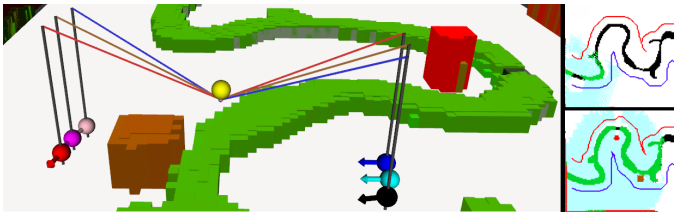

Fig. 6. Local trajectory replanning to avoid a new obstacle (brown).

VIII. Real-world experiments

Two experiments were conducted to validate the control scheme with four TurtleBot3 Burgers and to assess the effect of different distance constraints when traversing non-convex obstacles. The robots were not equipped with depth sensors; therefore, obstacle height was assumed fixed at $30cm$. Each robot uses a LiDAR sensor for AMCL-based localization.

1) Scenario: Two side-formations of two robots each with ring-topology communication were deployed. Each side-formation was approximated by a circular footprint with radius $r_{\text{side}} = 40\,\mathrm{cm}$. Each robot had a pole of length $h = 100\,\mathrm{cm}$ (total robot height $125\,\mathrm{cm}$).

Two rope settings were evaluated:

- *Setup 1:* $l_r = 95\,\mathrm{cm}$, yielding a payload height between $44\,\mathrm{cm}$ ($d_{\min} = 100\,\mathrm{cm}$) and $67\,\mathrm{cm}$ ($d_{\max} = 150\,\mathrm{cm}$).
- *Setup 2:* $l_r = 105\,\mathrm{cm}$, yielding a payload height between $33\,\mathrm{cm}$ ($d_{\min} = 100\,\mathrm{cm}$) and $93\,\mathrm{cm}$ ($d_{\max} = 200\,\mathrm{cm}$).

Both configurations were tested on the same 90×80 occupancy map with resolution $1\,\mathrm{px} = 5\,\mathrm{cm}$. The map was dilated using a circular kernel of radius $8\,\mathrm{px}$ to account for formation size. Start and goal positions were manually defined.

2) Results and discussion: A video of the complete experiment is available at [20]. The initial planned paths are shown in Figure 8. The global path for Setup 1 was computed in $2.4\,\mathrm{s}$, resulting in a total path length of $4.87\,\mathrm{m}$ for the left side-formation and $4.86\,\mathrm{m}$ for the right side-formation. For Setup 2, the global path was computed in $11.8\,\mathrm{s}$, yielding a total path

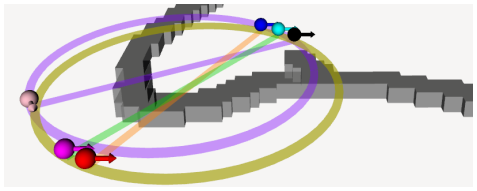

Fig. 7. Deformation of the formation due to a temporary failure of an agent.

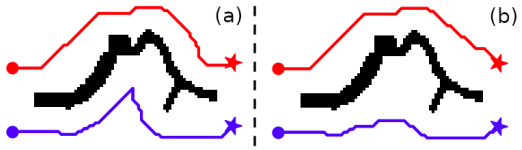

Fig. 8. The initial path computed for setup 1 (a) and setup 2 (b). The $\star$ and $\bullet$ represent the goal and start position of the side-formation center, respectively.

Fig. 9. Experimental setup with 4 robots in a real-world environment, with tighter constraint (left) and broader constraint (right).

length of $4.68\,\mathrm{m}$ for the left side-formation and $4.18\,\mathrm{m}$ for the right side-formation.

Real-world experiments confirm the effectiveness of the proposed approach. Increasing the admissible distance range between side-formations yields shorter and smoother trajectories (Figures 8 and 9). However, this comes at the cost of higher computational time due to the enlarged feasible search space.

Empirical tests show that computational time scales polynomially with the problem parameters: approximately linearly with map dimensions, quadratically with the range of admissible inter-formation distances, and as s^4 with grid resolution s. Notably, since the robots are abstracted into two side-formations during planning, the total number of robots does not directly impact planning complexity.

IX. Conclusions and Future Work

This paper presented a framework for multi-robot formations to jointly transport a payload with obstacle crossing. The proposed approach reduces the crossing problem to a two-agent path planning formulation by abstracting the robots into two side-formations, hence favoring scalability even for large teams. The framework was validated through both simulation and real-world experiments; the results demonstrate that the system can successfully traverse non-convex obstacles, adapt to unknown obstacles through online map updates, and perform local replanning when necessary. The obstacle height is added with depth measurements, enabling collision checks for the robots, cables, and payload.

A distributed position-based control strategy ensures trajectory tracking and allows the formation to react to temporary robot slowdowns or failures through continuous inter-robot communication. The current setup relies on geometric constraints for the payload's balance and does not consider its dynamics. While this assumption is sufficient under quasi-static conditions, including dynamic modeling and more advanced control strategies would improve robustness, particularly during aggressive maneuvers or transient disturbances. Extending the framework to operate over larger environment maps with multiple obstacles remains an important direction for future research. Such an extension would require careful consideration of computational scalability and planning efficiency.

References

[1] E. Tuci, M. H. M. Alkilabi, and O. Akanyeti, "Cooperative Object Transport in Multi-Robot Systems: A Review of the State-of-the-Art," *Frontiers in Robotics and AI*, 2018.

[2] D. K. D. Villa, A. S. Brandao, R. Carelli, and M. Sarcinelli-Filho, "Cooperative load transportation with two quadrotors using adaptive control," *IEEE Access*, vol. 9, pp. 129 148–129 160, 2021.

[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] J. Alonso-Mora, S. Baker, and D. Rus, "Multi-robot formation control and object transport in dynamic environments via constrained optimization," *The International Journal of Robotics Research*, vol. 36, no. 9, pp. 1000–1021, 2017.

[5] S.-M. Lee and J.-U. Lee, "Multi-robot formation planning in maze-like environments consisting of narrow passages using graph search," *IEEE Access*, vol. 12, pp. 167 694–167 704, 2024.

[6] J. Li, Y. Xu, X. Shi, and L. Li, "Deform: Adaptive formation reconfiguration of multi-robot systems in confined environments," *IEEE Robotics and Automation Letters*, 2025.

[7] Y. Sirineni, R. Tallamraju, A. Rawat, and K. Karlapalem, "Decentralized collision avoidance and motion planning for multi-robot deformable payload transport systems," in *IEEE SSRR*, 2020.

[8] H. Zhu, J. Juhl, L. Ferranti, and J. Alonso-Mora, "Distributed Multi-Robot Formation Splitting and Merging in Dynamic Environments," in *International Conference on Robotics and Automation (ICRA)*, 2019.

[9] F. Sahebsara and M. de Queiroz, "Distance-based formation maneuvering of mobile robots with static obstacles," in *2022 IEEE Conference on Control Technology and Applications (CCTA)*, 2022, pp. 470–475.

[10] H. Jiang, W. Xie, B. Zhou, W. Zeng, H. Xiong, and Y. Lou, "Simultaneous path and motion planning approaches for cooperative cable-driven transportation with mobile robots: A comparative study," *IEEE Transactions on Automation Science and Engineering*, 2025.

[11] S. Borate, R. Rana, P. Venkatesh, and M. Vadali, "FF-RRT*: A Sampling-Based Planner for Multirobot Global Formation Path Planning," *Journal of Mechanisms and Robotics*, 2024.

[12] H. Sha, Y. Cui, W. Lu, D. Zhang, C. Wang, J. Wu, R. Xiong, and Y. Wang, "Efficient Global Trajectory Planning for Multi-robot System with Affinely Deformable Formation," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2024.

[13] 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 IROS*.

[14] W. Zhang, C. Street, and M. Mansouri, "Multi-Nonholonomic Robot Object Transportation with Obstacle Crossing Using a Deformable Sheet," in *IEEE ICRA*, 2025.

[15] 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*, 2024.

[16] Y. Cao and N. Mohamad Nor, "An improved dynamic window approach algorithm for dynamic obstacle avoidance in mobile robot formation," *Decision Analytics Journal*, vol. 11, p. 100471, 2024.

[17] S. Chand, P. Verma, R. Tallamraju, and K. Karlapalem, "Transportation of Deformable Payload through Static and Dynamic Obstacles using Loosely Coupled Nonholonomic Robots," in *International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, 2019.

[18] 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.

[19] I.-A. F. Ihle, M. Arcak, and T. I. Fossen, "Passivity-based designs for synchronized path-following," *Automatica*, vol. 43, no. 9, pp. 1508–1518, 2007.

[20] "Simulation/experiments demo video," https://youtu.be/r62twBp9j0U, [Online; accessed February 2026].