Object Gathering with a Tethered Robot Duo

Yao Su\textsuperscript{1} Yuhong Jiang\textsuperscript{1} Yixin Zhu\textsuperscript{1,2} Hangxin Liu\textsuperscript{1}

Abstract—We devise a cooperative planning framework to generate optimal trajectories for a robot duo tethered by a flexible net to gather scattered objects spread in a large area. Specifically, the proposed planning framework first produces a set of dense waypoints for each robot, serving as the initialization for optimization. Next, we formulate an iterative optimization scheme to generate smooth and collision-free trajectories while ensuring cooperation within the robot duo to gather objects efficiently and avoid obstacles properly. We validate the generated trajectories in simulation and implement them in physical robots using Model Reference Adaptive Control (MRAC) to handle unknown dynamics of carried payloads. In a series of studies, we find that: (i) a U-shape cost function for maintaining separation distance is effective in planning cooperative robot duo, and (ii) the task efficiency is not always proportional to the tethered net’s length. Given an environment configuration, our framework can gauge the optimal net length. To our best knowledge, ours is the first that provides such estimation for tethered robot duo.

Index Terms—Cooperative path planning, optimization, tethered WMRs, adaptive control.

I. INTRODUCTION

We consider the task of gathering and carrying objects scattered on the floor by deploying two Wheeled Mobile Robots (WMRs) tethered by a flexible net or rope as a duo. Compared to picking and placing individual objects one by one, this setting is more intriguing and compelling for robots to collect small items spread across a large area autonomously. Fig. 1 showcases two similar tasks in the physical world that share a similar spirit, wherein humans adopt a net to collect scattered objects with high efficiency.

The tethered robot duo must resolve two challenges. First, how to effectively generate cooperative trajectories spanned between two individuals? Despite a robot duo could accomplish more complex and dexterous tasks than a single robot\textsuperscript{1}, the physical connection among it also introduces additional constraints in planning. Compared with prior work concerning rigid connections (e.g.,\textsuperscript{2,3}), planning for the robot duo with a non-rigid connection (e.g., a rope or a net) is more challenging\textsuperscript{4,5}; one has to consider the task goals, obstacle avoidance, and net shape maintenance through individual’s behaviors. Second, the tethered robot duo is subjected to increasing payloads during operation. In addition to the friction force introduced by the net, the dragging force increases significantly and deteriorates the trajectory tracking of each robot as the task progresses and more objects are gathered. Of note, this challenge is mostly overlooked in literature\textsuperscript{4-7}.

In this paper, we devise an optimization-based cooperation framework for a pair of tethered WMRs (i.e., a duo) to gather and transport objects with a net; see Fig. 1\textsuperscript{b} for the setup. The core of our framework is a trajectory optimization scheme that produces smooth trajectories for the robot duo, which jointly accounts for task goals, obstacle avoidance, and net shape maintenance. A set of waypoints extended from a centerline connecting all the target objects along the robots’ initial and final configurations serve as the initialization for the optimization scheme. An MRAC controller is further implemented to track the produced trajectories under unknown increasing payloads stably. Our framework is validated in both simulation and experiment.

A. Related Work

Although cooperative planning has incubated many successful applications, such as search and rescue\textsuperscript{8}, exploration\textsuperscript{9}, object manipulation\textsuperscript{10,11}, payload transportation\textsuperscript{12,13}, the planning problem of tethered robot duo to date is less explored. Prior work\textsuperscript{4,14} mostly formulates it as a separation problem: The goal is to find trajectories for the robot duo to separate all target objects from obstacles. In such a scheme, prior methods assume an infinite separation distance between the robots, impractical and unrealistic in many cases. Although Teshnizi et al.\textsuperscript{5} relax this assumption and introduce a modified A* planning algorithm given specific tether lengths, it fails to properly account for the obstacle avoidance (e.g., in final configuration) and net shape maintenance. We fully address these challenges in a new framework.
Gathering objects using the robot duo necessitates maintaining the shape of the flexible tether connecting the robots, though not necessarily precisely. Several setups have showcased the ability to handle flexible connections in diverse scenarios among robots. For instance, a dual-arm robot manipulates rope-like objects in clutter [15], three quadcopters connected by a net catch and throw a ball [11], multiple robots manipulate deformable bed sheet [16], a visual servoing scheme maintains the tether’s shape between two-wheel robots [17][18], and a pair of tethered quadcopters move an object using the tether [5][7]. Although the above literature strongly suggests the significance of maintaining tether/net shapes in cooperative planning, directly and precisely controlling its shape is oftentimes unnecessary, resulting in increased complexity in both modeling and control. To tackle this problem, our proposed framework adopts a U-shape cost function to maintain a proper distance between the two tethered robots, such that the net shape is indirectly controlled.

As an essential branch of robust control, Model Reference Adaptive Control (MRAC) can maintain the platform’s stability with unmodeled dynamics. It has been implemented on various robot platforms with inaccurate physical parameters, unknown payloads, or external disturbance [19]. For example, a tilt-rotor quadcopter platform using MRAC pulls an unmodeled cart [20], and WMRs adopts MRAC maintain the platform stability with inaccurate physical parameters [21]. Similarly, we implement MRAC on the tethered robot duo for the object gathering task, since the physical parameters of the objects are unknown, and the drag force increases with more objects collected by the net.

### II. Cooperative Path Planning

This section formally describes the proposed cooperative path planning framework for the tethered robot duo, assuming the environment and object locations are known.

#### A. Problem Setup

Fig. 2 illustrates our problem setup. Given an environment with a set of objects (red squares) and obstacles (black blocks), a robot duo is tasked to navigate from the initial configuration \( P_s = (P^1_s, P^2_s) \) to reach the end configuration \( P_e = (P^1_e, P^2_e) \) while collecting the objects on-the-way and avoiding obstacles. The environment is described as an occupancy grid \( \text{Map} \), wherein \( \text{Map}(i,j) = 0, 1 \), and 2 denote empty grids, grids occupied by obstacles, and grids occupied by objects, respectively. A safety margin \( \gamma \) defined based on the robot’s dimension is further added to obstacles for collision avoidance and to the radius of each target object’s minimum bounding circle for safety. We also merge the circles that are closely overlapped. As a result, we have \( M \) circles in total \( O = \{O_1, \ldots, O_M\} \), where \( O_i = [x^i_0, y^i_0, r^i_0] \) includes the center position and radius of each circle. Of note, avoiding obstacles not only requires the robot duo to avoid directly colliding with the obstacle but further demands the tethered net not to enclose any obstacle at any moment.

We devise a two-stage cooperative path planning framework to generate an optimal trajectory, shown also in Fig. 2. First, our framework generates a set of waypoints (blue dots) from the centerline path (dashed green line) that connects the robots’ start and end configurations along with the circles that enclose target objects. We call these waypoints baseline trajectory; they serve as a good initialization for the subsequent step. Next, our framework performs trajectory optimization to produce feasible trajectories (solid red line).

**Assumption:** Our framework has two assumptions: (i) The circles do not overlap with obstacles; namely, the objects cannot be too close to the obstacle such that the robot duo fails to navigate without violating the safety margin. (ii) The size of the circles is smaller than the length of the tethered net; the net constrains the robot duo’s motions.

#### B. Baseline Trajectory

Let the middle point of the tethered robot duo at the start configuration be \( P^c_s = \frac{1}{2}(P^1_s + P^2_s) \) and at the end configuration be \( P^c_e = \frac{1}{2}(P^1_e + P^2_e) \). First, we adopt a conventional path planning algorithm (hybrid \( A^* \) [22] or \( RRT^* \) [23]) to construct the centerline by connecting \( P^c_s \), each circle’s center point \( O_i \), and \( P^c_e \); see dashed green line in Fig. 2. Formally, the centerline is denoted as \( C = \{P^c_s, \ldots, P^c_e\} \in \mathbb{R}^{N \times 3} \).
Case 2
Case 3

Fig. 3: The selection of baseline points. Given the centerline point \( P_0 \) (green arrow), four candidates of baseline points are proposed; Eqs. 2 and 3 yield \( P_1, P_2 \) (blue dots), and Eq. 1 yields \( P_3, P_4 \) (red dots). Shaded points indicate the selected baseline points \( P_1, P_2 \).

The cost of objects as obstacles treats circles \( O_j \) as obstacles, designed similarly to potential field methods [26]:

\[
J_{\text{obs}, 2}(i, j) = \begin{cases} 
\frac{1}{2} K_2 \left( \frac{1}{D(i, j)} - \frac{1}{r_j^o} \right)^2, & D(i, j) \leq r_j^o, \\
0, & D(i, j) > r_j^o,
\end{cases}
\]

where \( D(i, j) \) denotes the Euclidean distance between the trajectory point \( P_i \) and the object circle center \( O_j \), and \( K_2 \) is a positive parameter to penalize the distance.

As such, the combined obstacle cost of \( P_i \) is defined as

\[
J_{\text{obs}}(i) = J_{\text{obs}, 1}(i) + \sum_{j=1}^{M} J_{\text{obs}, 2}(i, j).
\]

Distance Cost: The distance cost is calculated as the weighted sum of the Euclidean distance between two consecutive points and the orientation difference:

\[
J_{\text{dist}}(i) = t_1 \sqrt{\Delta x_i^2 + \Delta y_i^2} + t_2 |\Delta \phi_i|,
\]

where
\[
\Delta x_i = P^1(i, 1) - P^1(i + 1, 1),
\]
\[
\Delta y_i = P^1(i, 2) - P^1(i + 1, 2),
\]
\[
\Delta \phi_i = P^1(i, 3) - P^1(i + 1, 3),
\]

with \( t_1 \) and \( t_2 \) as two non-negative parameters.

Expansion Cost: The cost function for expansion is designed as a U-shape cost function [27]:

\[
J_e(i) = \begin{cases} 
k_{d1} \tan^2(\gamma_1 d + \gamma_2), & d_{\text{min}} \leq d \leq d_{\text{rest}}, \\
k_{d2}(d-d_{\text{rest}})^2 - \frac{(d-d_{\text{max}})^2}{(d-d_{\text{max}})^2}, & d_{\text{rest}} < d \leq d_{\text{max}}, \\
K_3, & d < d_{\text{min}} \text{ or } d > d_{\text{max}}
\end{cases}
\]

where \( \gamma_1 = \frac{\pi}{2(d_{\text{max}}-d_{\text{min}})} \) and \( \gamma_2 = -\gamma_1 d_{\text{rest}} \). \( d \) is the Euclidean distance between trajectory points \( P^1 \) and \( P^2 \), representing the expansion of the tethered net. \( k_{d1}, k_{d2}, \) and \( K_3 \) are gain parameters. \( d_{\text{max}}, d_{\text{min}}, \) and \( d_{\text{rest}} \) denote the maximum, minimum, and the ideal expansion of the tethered net, set based on various hardware setup and task scenarios.

The reason for having this expansion cost is to maintain the U-shape of the tethered net. Conversely, the net shape would make it infeasible to collect objects when the two WMRs are too close or too far. In this paper, we set \( d_{\text{min}} = 0.1 d_{\text{max}} \) and \( d_{\text{rest}} = \frac{d_{\text{max}}}{2} \) d_{\text{max}}—a half-circle shape. Of note, if \( d_{\text{max}} \) is not specified, we can treat it as a random variable to estimate the optimal net length in certain task scenarios.

Smoothness Cost: The smoothness cost is defined as the sum of squared accelerations along the trajectory:

\[
J_s = \sum_{j=1}^{3} b_j P^1(:, j)^T Q P^1(:, j),
\]

where \( Q = A^T A \) and \( A \) is the finite difference matrix [28]. \( b_j \) are weighting gains.

Taken together, the total cost along the whole trajectory is defined by the sum of the above cost functions:

\[
J_{\text{total}} = a_1 J_s + \sum_{i=1}^{N} a_2 J_{\text{dist}}(i) + a_3 J_{\text{obs}}(i) + a_4 J_e(i)
\]

where \( a_1, a_2, a_3, \) and \( a_4 \) are weighting gains for costs.
Algorithm 1: Trajectory Optimization Algorithm

Data: $N_{\text{total}}, P, O, Map, b_{1-2}, K_{1-3}, k_{12}, k_{r}$

Result: $P_{\text{opt}}, d_{\text{max}}$

\[
\begin{align*}
\text{step} & \leftarrow 1, d_{\text{max}} \leftarrow N\Delta L_{\text{max}}; & \text{// Initialization} \\
\text{while} \ step \leq N_{\text{total}} \text{ do} \\
\quad \text{if} \ mod(step, 2) = 1 \text{ then} & \text{// Optimize left half} \\
\qquad P_{\text{opt}} & \leftarrow P(1, 1 : 3), P_{e} \leftarrow P(N, 1 : 3); \\
\qquad P_{1 \text{ini}} & \leftarrow P(2, N-1, 1 : 3); \\
\qquad P_{2 \text{ini}} & \leftarrow P(2, N-1, 4 : 6); \\
\qquad P_{\text{opt}} & \leftarrow \text{opt}(P_{1 \text{ini}}, P_{2 \text{ini}}, P_{e}, Map, O, d_{\text{max}}); \\
\qquad \text{// Optimization process} \\
\quad \text{else} & \text{// Optimize right half} \\
\qquad P_{\text{opt}} & \leftarrow P(1, 4 : 6), P_{e} \leftarrow P(N, 4 : 6); \\
\qquad P_{1 \text{ini}} & \leftarrow P(2, N-1, 4 : 6); \\
\qquad P_{2 \text{ini}} & \leftarrow P(2, N-1, 1 : 3); \\
\qquad P_{\text{opt}} & \leftarrow \text{opt}(P_{1 \text{ini}}, P_{2 \text{ini}}, P_{e}, Map, O, d_{\text{max}}); \\
\quad \text{end} \\
\quad \text{step} & \leftarrow \text{step} + 1; & \text{// Update steps} \\
\text{end} \\
\text{end} \\
\text{P}_{\text{opt}} & \leftarrow P; & \text{// Output optimized trajectory}
\end{align*}
\]

2) Constraints:

Maximum Velocity Constraint: We define it as:

\[
\begin{align*}
0 \leq \sqrt{\Delta x_{r}^{2} + \Delta y_{r}^{2}} & \leq \Delta L_{\text{max}}, \\
-\Delta \phi_{\text{max}} & \leq \Delta \phi_{r} \leq \Delta \phi_{\text{max}},
\end{align*}
\]

where $\Delta L_{\text{max}}$ is the maximum travel distance at each step, and $\Delta \phi_{\text{max}}$ is the maximum turn angle at each step.

Object and Obstacle Constraint: We select consecutive points on trajectory $P^{1}$ and $P^{2}$ (i.e., $P_{1 \text{ini}}^{1}, P_{2 \text{ini}}^{1}, P_{1 \text{ini}}^{2}, P_{2 \text{ini}}^{2}$) to build a quadrilateral, resulting in $N-1$ quadrilaterals. To ensure collecting all the objects, the centers of all circles $O$ must be covered by these quadrilaterals. Moreover, to prevent the path from enclosing any obstacles inside, these quadrilaterals must not overlap with any obstacles.

III. ROBUST TRAJECTORY TRACKING CONTROL

We adopt a decentralized framework to design the trajectory tracking controller for individual WMRs. Since the drag force from the tethered net and gathered objects are unknown in our proposed setup, the conventional model-based controller is not suitable, and a more advanced MRAC is implemented to improve the robustness of the trajectory tracking control.

A. Dynamics of Individual WMRs

The configuration of WMRs is defined by $\mathbf{q} = [x, y, \phi]$, shown also in Fig. 2. Li et al. [29] describe its dynamics:

\[
M(q) \ddot{q} + C(q, \dot{q}) \dot{q} = B(q) \tau - A(q)^{T} \lambda,
\]

where

\[
M(q) = \begin{bmatrix} m & 0 & m d \phi \cos \phi \\ 0 & m & -m d \phi \sin \phi \\ m d \phi \sin \phi & -m d \phi \cos \phi & J \end{bmatrix},
\]

\[
C(q, \dot{q}) = \begin{bmatrix} 0 & 0 & m d \phi \cos \phi \\ 0 & 0 & -m d \phi \sin \phi \\ 0 & 0 & 0 \end{bmatrix},
\]

\[
B(q) = \frac{1}{r} \begin{bmatrix} \cos \phi & \cos \phi \\ \sin \phi & \sin \phi \\ R & -R \end{bmatrix},
\]

\[
A(q) = \begin{bmatrix} -\sin \phi & \cos \phi & -d \end{bmatrix},
\]

\[
\lambda = -m(\dot{x} \cos \phi + \dot{y} \sin \phi) \phi.
\]

where $m$ is the WMR’s mass, $J$ its rotational inertia, $2R$ the distance between two driving wheels, $r$ the radius of each wheel, $d$ the distance from the coordinate origin $P_{0}$ to the CoM $P_{e}$, $M(q)$ the inertia matrix, $C(q, \dot{q})$ the Coriolis and Centrifugal force matrix, $B(q)$ the input transformation matrix, $\tau$ the torque vector for two wheels, $A(q)$ matrix associated with the non-holonomic constraints (i.e., $A(q) \dot{\phi} = 0$), and $\lambda$ the vector of constraint forces. Selecting $S(q)$ as a basis of $A(q)$ nullspace (i.e., $S(q)^{T}A(q) \dot{q} = 0$),

\[
S(q) = \begin{bmatrix} \cos \phi & -d \sin \phi \\ \sin \phi & d \cos \phi \\ 0 & 1 \end{bmatrix},
\]

we can rewrite the WMR’s kinematic equation with the velocity vector,

\[
\dot{q} = S(q)v,
\]

where $v = [\nu, \omega]^{T}$, and $\nu$ and $\omega$ are the WMRs’ linear and angular velocity. Taking the derivative of Eq. (19), we have

\[
\dot{q} = \dot{S}(q)v + S(q)\dot{v}.
\]

Substituting Eqs. (19) and (20) into Eq. (12) and multiplying $S^{T}$ at each side, we have

\[
\dot{M} \dot{v} + \dot{C} \dot{v} = \tau,
\]

where $\dot{M} = S(q)^{T}M(q)S(q)$, $\dot{\tau} = \dot{B} \tau$, $\dot{B} = S(q)^{T}B(q)$, and $C = S(q)M(q)S(q) + S(q)^{T}C(q, \dot{q})S(q)$.

B. Model-Based Control of Individual WMRs

We design a conventional model-based controller based on Eq. (21) as a baseline,

\[
\tau = \ddot{\phi}(\dot{M} \ddot{v} + \ddot{C} \dot{v}),
\]

where $\nu$ is the desired velocity vector, designed to track the reference trajectory [29]:

\[
\nu = \begin{bmatrix} \nu^{d} \\ \omega^{d} \end{bmatrix} = \begin{bmatrix} \nu^{r} \cos \phi + k_{1}(\epsilon_{x} + d(1 - \cos \phi)) \\ \omega^{r} + k_{2}\nu^{r}(\epsilon_{y} - d \sin \epsilon \phi + k_{3}\nu^{r} \sin \epsilon \phi) \end{bmatrix},
\]

where $\nu^{r}$ and $\omega^{r}$ are reference linear and angular velocity, $k_{1}, k_{2}$, and $k_{3}$ are positive parameters, and $\epsilon_{x}, \epsilon_{y}$, and $\epsilon_{\phi}$ are errors between the reference trajectory and the real trajectory in $x, y$, and $\phi$, respectively [29].

\[
\epsilon_{x} = x' - x, \epsilon_{y} = y' - y, \epsilon_{\phi} = \phi' - \phi.
\]

Although this controller has proven to work well on WMRs given accurate physical parameters, it fails in our setting because the tethered net will collect multiple objects with unknown physic parameters (mass, inertia, etc.). As such, we further design an adaptive control law; see the next section.
C. Adaptive Control with Unknown Payload

Since MRAC laws guarantee the asymptotic convergence of trajectory tracking error in the presence of parametric and matched uncertainties [19], we implement MRAC on WMRs to improve its robustness under unknown payload.

Specifically, we rewrite the matrices in Eq. (21) with unknown dynamics,

\[
\dot{\bar{r}} = u + (\bar{M}_k - I)\bar{v}(t) + \bar{C}_v \bar{v}(t),
\]

and the reference model is given as

\[
\dot{X}_m(t) = A_m X_m(t) + B_m u_m(t),
\]

where \(X_m = v^d\), \(A_m = -K\), \(K\) is a Hurwitz matrix, and \(B_m = I\). The reference input \(u_m\) can be calculated by

\[
u_m = B_m^{-1}(\dot{X}_m - A_m X_m) = \dot{v}^d + Kv^d.
\]

The adaptive law is designed as

\[
\begin{align*}
\dot{u} &= \bar{u}_a + \bar{u}_k, \\
\dot{\bar{u}}_k &= -KX + u_m, \\
\dot{\bar{u}}_a &= -\Delta X X + \Delta u_m,
\end{align*}
\]

where \(\bar{u}_k\) and \(\bar{u}_a\) are the linear and adaptive feedback component, respectively. \(\Delta X\) and \(\Delta u_m\) can be calculated with adaptive gains and state errors; please refer to Canigur et al. [21] for more details. Of note, the motor torque saturation and the adaptive gains of MRAC jointly determine the maximum payload while the stability of tracking controller is still maintained. Larger adaptive gains can increase the response speed but may lead to oscillations and higher overshoot [30].

IV. SIMULATION

This section evaluates our cooperative planning framework in a simulation environment based on Gazebo, a 3D dynamic environment simulator. Our results demonstrate that (i) the proposed framework produces optimal trajectories for the tethered robots, and (ii) our U-shape cost function design effectively maintains the tethered net during the tasks.

A. Setup

For the tethered robot platform, we utilize two TurtleBot3 Waffle Pi robots and tether them with a flexible “net” by connecting several 0.1-meter-long thin cuboid links with passive revolute joints; the length of the net can be easily modified by inserting or removing cuboid links. With such a design, the tethered net would deform in accord to gathered objects’ weights, and drag forces could be introduced to robots in an unspecified direction, crucial in evaluating our MRAC implementation. To enable torque command to the robots in simulation, we use JointEffortController in ros_control to command desired torques to TurtleBot3’s wheels instead of using the built-in controller with velocity control only.

Fig. 4 illustrates the simulated environment with several keyframes of a task performance. The tethered robot duo starts from the bottom left to gather red movable cubes while avoiding blue obstacles at various locations. Table I tabulates some essential physical and software properties of the simulation setup. We set the parameters involved in the trajectory planning as following: \(l = 0.3\) m, \(N_{\text{total}} = 8\), \(t_1 = 300\), \(t_2 = 10\), \(K_1 = K_2 = K_3 = 1e7\), \(k_{d1} = 20\), \(k_{d2} = 15\), \(b_1 = b_2 = 110\), \(b_3 = 1\), \(a_1 = a_2 = a_3 = a_4 = 1\), \(\Delta L_{\text{max}} = 0.05\) m, and \(\Delta \phi_{\text{max}} = 0.1\) rad. We utilize the nonlinear optimization solver fmincon in Matlab to solve the optimal trajectory with the interior point method.

B. Trajectory Tracking

Fig. 5 shows the simulation results with various lengths of the tethered net \(d_{\text{max}}\). Given a configuration during the

<table>
<thead>
<tr>
<th>TABLE I: Physical and Software Properties used in Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Group</strong></td>
</tr>
<tr>
<td>Robot</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Net</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Others</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
optimization, our framework can further estimate the optimal length of the tethered net in terms of defined cost function $J_{\text{total}}$ by treating $d_{\text{max}}$ as a variable in Algorithm 1. If $d_{\text{max}}$ is already specified, Algorithm 1 can be easily modified by treating $d_{\text{max}}$ as a constant instead of a variable. Among five cases with various maximum net lengths shown in Fig. 5, our framework estimates $d_{\text{max}} = 2.32$ m is the optimal net length.

In these simulations, we implement both MRAC controller and the conventional model-based controller. Specifically, the MRAC controller can maintain the platform stability along the whole trajectory while successfully gathering and transporting all the objects to the endpoint. By contrast, the conventional model-based controller is unstable when more objects are collected, resulting in collisions between the two robots or between the robot and objects/obstacles.

For each of the above cases, the distance cost $J_{\text{dist}}$ and total cost $J_{\text{total}}$ are first obtained with recorded robot trajectory. Next, these costs are compared with planned costs; the results are shown in Fig. 5. In particular, if we only care about $J_{\text{dist}}$, both $d_{\text{max}} = 2.5$ m and $d_{\text{max}} = 3.0$ m are better. However, it is difficult to maintain the shape of the tethered net under these two settings. As such, our optimization framework deems they are not the optimal solution. In comparison, although $d_{\text{max}} = 2.32$ m results in a slightly higher $J_{\text{dist}}$, this length is considered as the optimal one due to the lowest $J_{\text{total}}$.

This simulation result demonstrates the tracking performance of the proposed MRAC controller to handle increasing payloads. It also suggests that a longer tether between robots does not necessarily correspond to better efficiency. This finding may impact prior arts wherein the length is not considered or is fixed.

C. Maintaining Net Shape

We design a U-shape cost function in Eq. (9) to penalize the tethered robot duo for being too close/far, so that the curvature of the net is indirectly controlled to embrace new objects and carry gathered objects. To demonstrate the efficacy of the U-shape design, we compare with two baseline cost functions:

- **No separation constraint included.** This strategy does not take separation distance into consideration; referred to as Baseline 1 in Fig. 6.
- **Desired separation distance $d_{\text{rest}}$ as hard constraint.** This strategy maintains the desired separation distance along the path; referred to as Baseline 2 in Fig. 6.

Figs. 6a, 6c and 6d quantitatively compare the distance cost of each cost design with various net lengths in three scenarios, and Figs. 6a, 6e and 6c show the corresponding reference trajectories produced by our framework using the 3 types of expansion cost designs with a specific net length in each scenario. Although Baseline 1 yields the shortest travel distance without including any constraint on separation distance, it may require unrealistic net length. Baseline 2 keeps a specified separation distance at all time, which may find no solution for a small $d_{\text{max}}$ or cause unnecessarily long travel distance for a large $d_{\text{max}}$. In comparison, the proposed U-shape cost function produces feasible solutions for various $d_{\text{max}}$ while properly maintaining efficient travel distances.
V. Experiment

Section IV has evaluated the proposed cooperative planning framework and the MRAC controller design in simulation. This section further validates the proposed framework in the physical environment.

A. Setup

The setup of our proposed tethered robot duo is shown in Fig. 1 wherein a net is connected to two TurtleBot3 Waffle Pi robots at each end. Each TurtleBot is equipped with a Raspberry Pi 3B+ running Robot Operating System (ROS), an OpenCR 1.0 board with an IMU module and a microprocessor for low-level motor control, an LDS-01 Lidar for localization, and two Dynamixel XM430 motors with a maximum torque of 3 N\text{-}m for wheel actuation. The path planning component runs on a desktop (AMD Ryzen9 5950X CPU, 64.00 GB RAM) and takes an average of 180.3 s to produce the optimal trajectory with $N = 200$ in Algorithm 1.

To enable torque command, we modify the built-in low-level controller on OpenCR board designed to take velocity command inputs, such that it receives torque command inputs. The Gmapping Simultaneous Localization and Mapping (SLAM) algorithm is utilized to localize the WMRs in experiment and outputs $q$ and $v$ as feedback. Fig. 7 shows the experimental environment, a replication of the simulated environment presented in Fig. 6a; eight objects are placed on the floor with various weights ranging from $15 \text{ g}$ to $50 \text{ g}$.

B. Results

The net length used in our experiment is $1.45 \text{ m}$, the optimal length estimated by our cooperative path planner according to this specific environment configuration. Fig. 8 plots the reference and actual trajectories with error ribbon aggregated from five trials. Together with the simulation results, our experiments have demonstrated (i) the implemented MRAC controller robustly tracks the planned trajectories, and (ii) the tethered robot duo, using the proposed cooperative planning framework, successfully gathers all the objects along the path and carries them to the end position.
VI. CONCLUSION

This paper investigated an interesting task of gathering and transporting scattered objects with a tethered robot duo. We addressed it by proposing a two-stage cooperative planning framework based on trajectory optimization and implementing it with MRAC. In planning, we designed a U-shape cost function and incorporated other constraints to produce trajectories, capable of indirectly maintaining the flexible net’s shape w.r.t. the distance between two robots. In the implementation, we demonstrated that MRAC could robustly handle the increased payload with unknown dynamics as more objects were carried. As an extra feature, our planning framework can also estimate the most efficient length given an environment configuration, which led to a crucial extension to existing work in tethered robots that assumed an infinite or fixed tether length.

In the implementation, we demonstrated that MRAC could robustly handle the increased payload with unknown dynamics as more objects were carried. As an extra feature, our planning framework can also estimate the most efficient length given an environment configuration, which led to a crucial extension to existing work in tethered robots that assumed an infinite or fixed tether length.

The authors would like to thank Dr. Bo Dai, Mr. Ziyuan Jiao, Mr. Aoyang Qin, and Ms. Zhen Chen at BIGAI for their constructive discussions and help on experiments and figures.

REFERENCES