2026 IEEE/SICE International Symposium on System Integration (SII)
January 11-14, 2026. Cancún, México

Energy-Efficient Objects Collection with Tethered Mobile Robots Twins with Dynamic Grouping of Scattered Objects

Haruka Nakamura, Hiroki Nagai and Genya Ishigami

***Abstract*— This paper addresses an energy-efficient object collection problem using two mobile robots twins, equipped with a flexible tool such as a net or tether. This problem is subject to a routing problem for a collection sequence that minimizes total energy consumption of both robots for collecting scattered, multiple objects. To this end, we propose a dynamic programming-based route planning method, where the robot state is defined by energy consumption and its position. As the flexible tool allows the robots to collect a number of objects all at once, we implement a grouping circle of the tool, the radius of which depends on the number of the collected objects. The method incorporates a load-dependent velocity model as well as the load-dependent radius of object grouping circles to realistically reflect robot dynamics during the cooperative collection and transportation of the objects. Simulations were conducted under various scenarios with different object distributions, start positions of the robots, and the maximum velocities of the robots. The simulation results demonstrate that the proposed method reduces both energy consumption and travel time compared to baseline methods.**

I. INTRODUCTION

In this paper, we consider a task to collect/gather multiple scattered objects by mobile robots twins connected by a flexible tool such as a net or tether one another. Fig. 1 is the concept of this study; two robots are tethered with a flexible tool used for towing an object indirectly.

Collecting scattered objects is often time-consuming for one single robot; the robot needs to detect the object's precise location and capture them with its specific mechanism. Moreover, the robot with small payload capability can only transport a few number of objects, requiring repeated tasks to complete the object collection. Cooperating two robots having a flexible tool such as a net or a tether for those tasks will make the task more time-efficient, enabling the robots to roughly detect the object locations and to collect them with a simpler mechanism.

This robotic system as *Cooperative objects transport*, is useful and has been studied in the past in various situations; the collection and transportation of lightweight objects [1] and the skimming of oil or cleanup of surface debris on the ocean surface [2], [3]. These studies, however, did not consider any mechanical effect in the object dragging and surface friction as the mass of the towed objects is just $0 \sim 7\%$ of that of the robot or vessel. When the mass of the object is large comparing to the robots, or when the surface friction is large, the robots' behaviors dynamically change. One of the critical effects on the robots is slip against the ground. When the mass of the objects is more than 15% of that of the robot, the twins turn outwards with respect to the object on the high-slip ground even though each robot is controlled to align to straight motion, and eventually pull each other outwards, making themselves immobilized [4]. This implies that the direction of the flexible tool from the robot to the object is critical to avoid or decrease the mechanical effect on the robot.

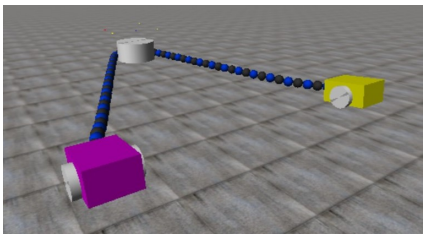


Fig. 1: Object transportation by tethered mobile robots twins

Considering the tether tension vectors, only the aligned component of the traction force is useful; perpendicular components reduce towing efficiency [5]. Therefore, the robot and object should be in-line such that the both vectors of the robot and object are in the same direction, minimizing the tension [6].

The above shows that shorter the distance between the twins, higher the towing efficiency. However, if the robot twins are too close one another, only a few objects can be collected in one time, resulting the twins are required to move longer distance to collect all the objects. Therefore, the distance between the robot twins effects to the total efficiency and to the collecting order of the objects with considering dynamic grouping of multiple objects with regard to the distance between the robots that varies with carrying load due to the objects dynamically.

The problem mentioned above holds another well-known problem as Traveling Salesman Problem (TSP), where the robots twins collect all the objects in one task. The objective function is commonly minimizing the travel distance, which is replaceable for energy consumption in terms of decreasing the load to the robots. Therefore, our scenario can be classified as energy-dependent TSP with dynamic grouping based on the carrying load. Energy-dependent TSP has been studied in various sce-

*This work was not supported by any organization

Haruka Nakamura, Hiroki Nagai, and Genya Ishigami are with the Space Robotics Group, Department of Mechanical Engineering, Keio University, Yokohama 223-8522, Japan. harunaka7, jh7sv9weog-5@keio.jp, ishigami@mech.keio.ac.jp

narios. Cha et al. [7] model the arc cost as a function of both the load and the visitation sequence, considering the impact of hazardous materials. Xiao et al. [8] propose a mathematical formulation, which captures the increasing energy consumption as the transported load increases. Similarly, fuel consumption is modeled in [9], [10], where the fuel cost per unit distance depends linearly on vehicle load. However, none of these studies consider dynamic grouping of the objects during task execution.

Some studies have incorporated clustering into the TSP framework, while the cluster sizes are static and predefined [11], [12]. Other studies consider dynamic or adaptive clustering within TSP formulations [13], [14], yet they do not take into account the load-dependent effects of the transported objects.

Therefore, we aim to elaborate an energy-minimizing routing method with consideration of the dynamic grouping based on the load-dependent effects of the carried objects. The main contributions of this work are as follows:

- Formulation of an energy-efficient collection route using dynamic programming.
- Introduction of a dynamic object grouping method that determines the area for collecting multiple objects at once with a flexible tool.
- Modeling of energy consumption and maximum output velocity as functions of the current transported load, enabling the planner to account for the mechanical effects of heavy loads on the robots.

II. Assumptions and Problem Statement

In this study, we consider an environment where multiple objects uniform in size and mass, are scattered within a flat, homogeneous surface. Two mobile robots, connected with a flexible tool such as a net or tether on their rear end one another, collect and transport multiple objects all together by enclosing them within a flexible tool and dragging them along the ground. Both robots are provided with information regarding the location and masses of all objects.

The robots task is to determine a collection sequence that minimizes total energy consumption as the two robots, starting from a designated initial position, sequentially collect all objects. A group of objects that can be collected together is represented by a circle of radius R, and the process of forming such a group is referred to as *grouping*.

The route planner performs grouping at each step based on the current transported load. The robots finish their task when all the objects are collected, and stay at the last object group. For route planning, the planner views the two robots at an abstract level as a single point mass, and similarly, each object is assumed as a point mass. Fig. 2 shows the relationship between the transportation system and the planning target. Details of the robots' actual movement—such as whether the flexible tool is deployed throughout navigation or only during object collection—are beyond the scope of this study.

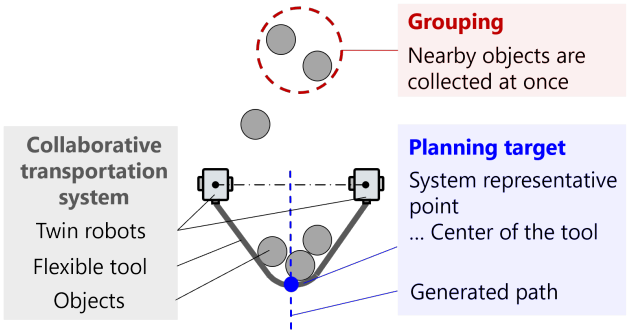
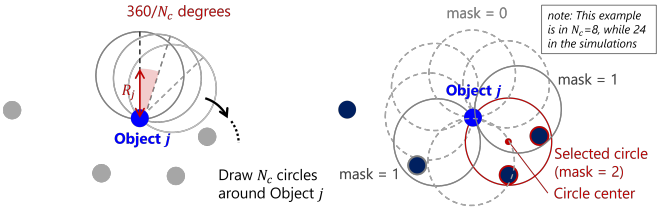


Fig. 2: Planning target in the system



(a) Grouping Step 1 (b) Grouping step 2–4

Fig. 3: Grouping procedure

III. Energy Efficient Routing with Dynamic Grouping

A. Grouping with Sweeping Strategy

To enhance collection efficiency, we introduce a *grouping* that enables the collection of multiple nearby objects all at once using a flexible tool. This is particularly beneficial in scenarios where objects are spatially clustered and collecting/transporting them one by one is inefficient.

Given a target object j, the proposed planner considers a set of circular regions centered at the location of object j whose radii are less than or equal to the current maximum allowable radius $R_{\text{max,current}}$. Let M_{current} denote the carried mass before reaching object j, then $R_{\text{max,current}}$ is given by:

$$R_{\text{max,current}} = R_{\text{max},0} - K_R \cdot M_{\text{current}} \tag{1}$$

where K_R is the radius reduction coefficients as the robot twins tend to get closer each other as the carried load increases. These circles are generated by rotating a sweep angle in $360/N_c$-degree increments with the total of N_c circles around object j, simulating a radial search for possible groupings (Fig. 3a).

Each circular region is evaluated by computing a *mask value*, which represents the number of additional objects that fall within the region (Fig. 3b). The region with the highest mask value is selected as the grouping candidate. The grouping flow is as below:

Algorithm 1 Grouping: `findSweepGroup()`

function FINDSWEEPGROUP($j, S_{\text{current}}, M_{\text{current}}$)
  $R \leftarrow R_0 - K_M \cdot M_{\text{current}}$
  for $i = 0$ to $N_c - 1$ **do** ▷ Angle: 0° to 345°
    Compute group center c at direction i
    Initialize $mask \leftarrow 0, count \leftarrow 0$
    for $k = 0$ to $N - 1$ **do**
      if $k \notin S_{\text{current}}$ and $D_{k,c} \leq R$ **then**
        Add k to $mask$, increment $count$
      end if
    end for
    if $count > best\_count$ **then**
      Update best group: $mask$, $center$
    end if
  end for
  return $mask, c$
end function

Step 1 Placing circles of radius $R < R_{\text{max, current}}$ every $360/N_c$ degrees around object j (We set $N_c = 24$).

Step 2 Counting the number of objects contained in each circle.

Step 3 Selecting the circle with the highest count (mask value).

Step 4 Computing the center of the selected circle to define the group.

In cases where multiple regions contain the same number of the objects, the region with the smallest sweep angle is chosen to ensure determinism and geometric consistency. However, it does not affect the optimality of the route as the planner calculated all the cases so that the energy-inefficient center is removed in the update of the state.

The maximum grouping radius $R_{\text{max, current}}$ is dynamically adapted based on the current transported load M_{current}, reflecting constraints on maneuverability or energy efficiency. Note that our method is a greedy approach, in which grouping is always performed whenever possible.

B. Dynamic Planning via Bitmask-based Dynamic Programming

To determine the energy-optimal route for collecting all target objects, we adopt a *Dynamic Planning* framework based on bitmask-encoded dynamic programming (commonly known as BitDP [15]).

Let N denote the total number of objects to be collected, and S the set of already collected objects ($0 \leq |S| \leq N$). Each subset S is represented as an N-bit integer bitmask, where the i-th bit is set if object i has been collected. For example, when $N = 4$ and $S = \{0, 1, 3\}$, the bitmask is 1011. We define the dynamic programming (DP) state as $dp[S][i]$, which represents the minimum energy required to collect all objects in set S and end at object i.

The computation of the dp array is performed in ascending order of the number of collected objects. First, the minimum energy for all states in which exactly one object has been collected is computed, and the smallest value is recorded in the dp array. Next, using that result, the minimum energy for all states with two or more collected objects is computed, and similarly, the smallest value is stored in the dp array. This process is repeated until all the objects are collected, always recording the minimum energy in the dp array. By iterating this procedure, the energy values stored in the dp array are guaranteed to be minimal.

Algorithm 2 Dynamic Programming with Dynamic Grouping

Initialize $dp[S][j] \leftarrow \infty$
for $j = 0$ to $N - 1$ **do**
  $mask, circle \leftarrow$ FINDSWEEPGROUP$(j, 0, 0)$
  $dp[mask][j] \leftarrow D_{\text{start, circle.center}} \cdot P_0 / v_{\text{max},0}$
end for
for $S_{\text{current}} = 0$ to $2^N - 1$ **do**
  for $i = 0$ to $N - 1$ **do**
    if $i \notin S_{\text{current}}$ **then**
      continue
    end if
    $M_{\text{current}} \leftarrow mass(S_{\text{current}})$
    for $j = 0$ to $N - 1$ **do**
      if $j \in S_{\text{current}}$ **then**
        continue
      end if
      $mask, c \leftarrow$ FINDSWEEPGROUP$(j, S_{\text{current}}, M_{\text{current}})$
      $S_{\text{new}} \leftarrow S_{\text{current}} | mask$
      $v_{\text{max,current}} \leftarrow v_{\text{max},0} - K_v \cdot M_{\text{current}}$
      $dp[S_{\text{new}}][j] \leftarrow \min(dp[S_{\text{new}}][j], dp[S_{\text{current}}][i] + E_{\text{trav}(i \to j)})$
    end for
  end for
end for

The routes recorded in the dp array are updated through energy comparisons as follows. At a certain point, suppose that for the state in which the set S_{current} of objects has been collected and the robot is currently at object i, the smallest energy consumption is stored in $dp[S_{\text{current}}][i]$. Then, let object j be the next possible collection target. Based on Algorithm 1, other objects in the grouping circle will also be collected. In that case, not only j-th bit but also other bits are set in the same time, which is the unique point of our method from the original bitDP. Thus, new subset is computed as below expressing the set of all the objects in the grouping circle as $mask$;

$$S_{\text{new}} = S_{\text{current}} | mask \tag{2}$$

$dp[S_{\text{new}}][j]$ is calculated in each for loop in Algorithm 2, and updated if smaller value is found;

$$dp[S_{\text{new}}][j] \leftarrow \min(dp[S_{\text{new}}][j], dp[S_{\text{current}}][i] + E_{\text{trav}(i \to j)}) \tag{3}$$

Here, $E_{\text{trav}(i \to j)}$ denotes the energy required to travel from object i to object j, and is defined as:

$$E_{\text{trav}(i \to j)} = (P_0 + K_E M_{\text{current}}) \frac{D_{i,j}}{v_{\text{max,current}}} \tag{4}$$

$$v_{\text{max,current}} = v_{\text{max},0} - K_v \cdot M_{\text{current}} \tag{5}$$

where $D_{i,j}$ is the Euclidean distance between objects i and j, M_{current} is the currently transported load, P_0 is the

power consumption which is required for the robots to move, and K_E, K_v are coefficients of energy and velocity reduction by M_{current}, respectively. The parameters K_E and P_0 are adjusted so that Eq. (4) represents the mechanical work performed by the robots. Specifically, K_E is defined as the product of the dynamic friction coefficient of the object against the ground μ_{object}, gravitational acceleration g, and the initial maximum output velocity $v_{\text{max},0}$. Similarly, P_0 is defined as the product of the dynamic friction coefficient of the robot against the ground μ_{robot}, g, the robot mass M_{robot}, and $v_{\text{max},0}$. Eq. (5) represents the relationship in which the maximum achievable velocity of the robot decreases according to the carried load M_{current}, while Eq. (4) expresses the energy consumption when the robot travels at the maximum velocity computed from Eq. (5).

By repeating this process until all objects are collected, the planner determines the route with the minimum energy consumption for the task.

About the computational cost, brute-force search has $O(N! \times N_c)$ complexity, our DP-based method achieves $O(2^N \times N^3 \times N_c)$ due to the *findSweepGroup* shown in Algorithm 1 is called in the innermost loop.

IV. Simulation study

A. Simulation setup

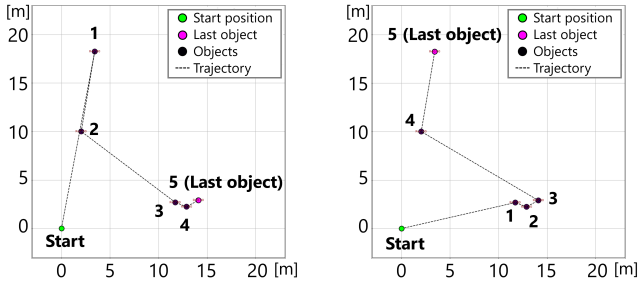
We randomly generated a total of 40 object placement environments in 20 × 20 meters; 10 different object placement environments for each total number of objects N; 5, 10, 15, and 20 (with $O(2^N \times N^3 \times N_c)$ complexity, the planner handles up to $N = 20$). Furthermore, for each environment, we conducted simulations under two different initial robots' positions; lower left (0, 0) and the center (10, 10). Also, we prepared four different maximum output velocity settings during full load transport, corresponding to 25%, 50%, 75%, and 100% of the no-load velocity.

These velocity settings were realized by adjusting the coefficient K_v in Eq. (5). In addition to the proposed method, we prepared three baseline algorithms and compared the results across the following four approaches:

- Ours (Em-WG): Energy minimization with grouping
- Em-WoG: Energy minimization without grouping
- Dm-WG: Distance minimization with grouping
- Dm-WoG: Distance minimization without grouping

Note that Dm-WoG corresponds to the standard Traveling Salesman Problem (TSP).

We set the parameters as following; $v_{\text{max},0} = 1.5\,\text{m/s}$, $R_{\text{max},0} = 3.5\,\text{m}$, $M_{\text{Robots}} = 100.0\,\text{kg}$ as a total mass of the twins with single robot mass is 50.0 kg, $M_{\text{object}} = M_{\text{total}}/N\,\text{kg}$ with the total mass of the objects $M_{\text{total}} = 50.0\,\text{kg}$, the dynamic coefficient of the objects and the robot against the ground $\mu_{\text{object}} = 2.0$ and $\mu_{\text{robot}} = 0.35$ respectively, $K_R = 0.075$, meaning the twins get closer for 0.75 m in case of carrying all of the objects.



(a) Energy-minimization (b) Distance-minimization

Fig. 4: Effect of minimizing energy on the path

The experiments were conducted on a machine equipped with an AMD EPYC 7313 16-Core Processor and 256 GB RAM. All solutions were obtained in an average of 3.4 seconds up to 8.5 seconds under $N = 20$.

B. Effect of energy minimization

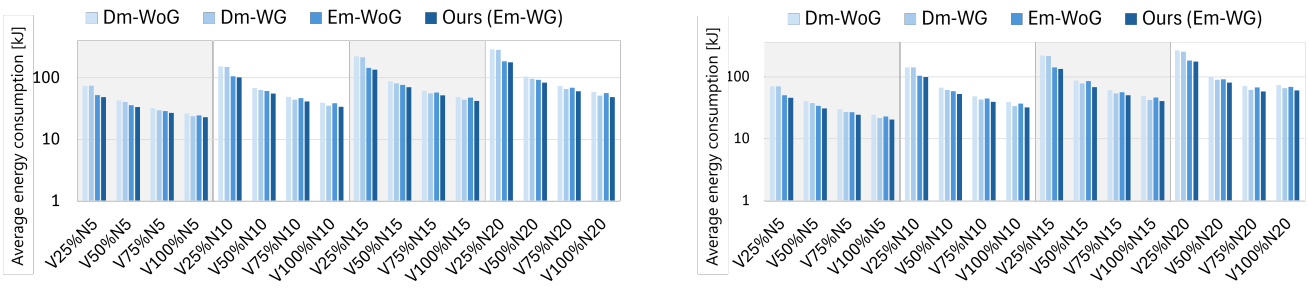
In order to validate the pure effect of the energy minimization, we compare the results of Em-WoG and Dm-WoG algorithms, both without grouping. The robot's starting position is set to $(0, 0)$, and the simulation result for $N = 5$ objects is shown in Fig. 4.

In the distance-minimization case, the robot collects nearby objects first (Fig. 4b), whereas in the energy-minimization case, it prioritizes collecting distant objects (Fig. 4a). This behavior indicates that the energy-minimization strategy tends to increase the travel distance when the robot carries no load, and conversely shortens the distance traveled while transporting objects.

As a result, the energy consumption in the energy-minimization case is reduced by approximately 40% compared to that of distance minimization. These findings demonstrate that the energy-minimization approach is effective in deriving more energy-efficient routes compared to conventional distance-based planning.

C. Simulation with Grouping

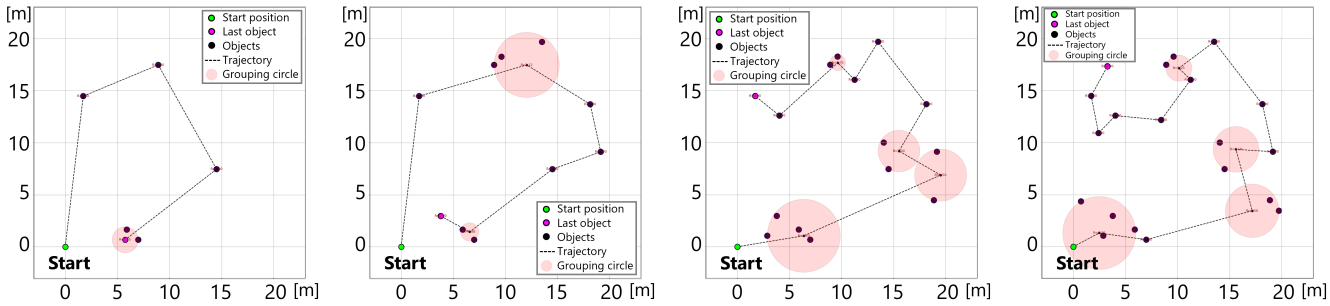
Next, we conducted simulations including cases with grouping enabled. For each number of the objects $N \in \{5, 10, 15, 20\}$, the Fig. 5 reports the average energy consumption. The path of the twins are shown in Fig. 6 and Fig. 7 for each number of objects, starting from (0, 0) and (10, 10) respectively with $K_v = 1.0$. Red translucent circles are grouping circles, where the robots collect the objects in the circle all at once. We also calculated the average minimum travel time (equivalent to the minimum task execution time). The energy and the travel time are computed over 10 different object placement scenarios per maximum output velocity setting. The travel time is calculated under the assumption that the robot always moves at the current maximum velocity $v_{\text{max,current}}$, and thus reflects the theoretical minimum time required to complete the task.



(a) Initial position: (0 , 0)

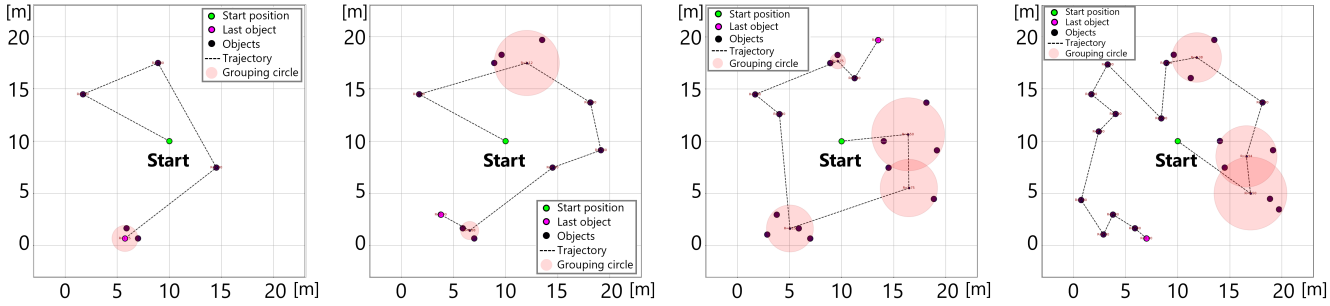
(b) Initial position: (10 , 10)

Fig. 5: Consumption energy comparison



(a) $N = 5$ (b) $N = 10$ (c) $N = 15$ (d) $N = 20$

Fig. 6: Path by proposed method for each N from the initial position at (0, 0)



(a) $N = 5$ (b) $N = 10$ (c) $N = 15$ (d) $N = 20$

Fig. 7: Path by proposed method for each N from the initial position at (10, 10)

1) Effect on the path: Regardless of the initial positions, the higher the maximum output velocity, the more frequently the routes diverged. Moreover, the paths tend to be different with the larger number of the objects because of decision of the grouping circle. The difference above is shown in Fig. 6 and Fig. 7.

This is because when the maximum output velocity is large (i.e., small K_v), the decrease in velocity with respect to carried load is softened, as shown in Eq. (5). Consequently, $v_{\max,\text{current}}$ in Eq. (4) remains relatively large, and the impact of travel distance $D_{i,j}$ on E_{travel} is reduced. As a result, the planner is more likely to choose more distant objects, making the route more sensitive to the initial position.

2) Effect by N: The larger number of the objects, the greater the reduction in both energy consumption and travel time of the proposed method compared to any baseline. As the number of objects increases, the object density becomes higher, making it more likely for multiple objects to fall within the current grouping radius $R_{\max,\text{current}}$ as shown in Fig. 6 and Fig. 7, though $R_{\max,\text{current}}$ decreases according to M_{current} making the planner difficult to perform grouping. This facilitates grouping and reduces the need for the robot to travel individually to each object, thereby decreasing both energy consumption and time.

3) Statistic analysis: The proposed method showed 10–25% lower energy consumption (normalized by N and K) compared to all baselines. Paired t-tests confirmed significant improvements, with large effects for Em-WoG and Dm-WoG ($d_z > 1.33$) and a strong effect for Em-WG ($d_z = 0.78$), highlighting the impact

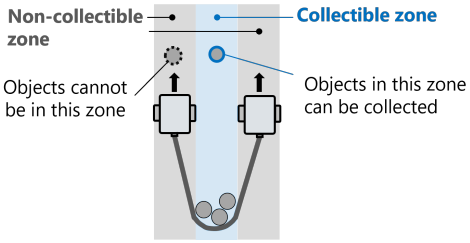


Fig. 8: Planning target in the system

of the grouping strategy. Significant negative method–output-rate interactions ($p < 0.001$ for methods without grouping) suggest that the proposed method Em-WG achieved its largest energy-saving benefit under $K = 0.5$, with diminishing but still significant advantages as the available output increased.

It should also be noted that in some cases, baseline Dm-WG yields shorter travel times than the proposed method. However, this occurs because Dm-WG does not incorporate the impact of transported mass M_{current} on $v_{\text{max,current}}$ into its cost function. As a result, it may select routes that are faster in theory but impose higher mechanical impact on the robot during execution.

V. Toward Realistic Application

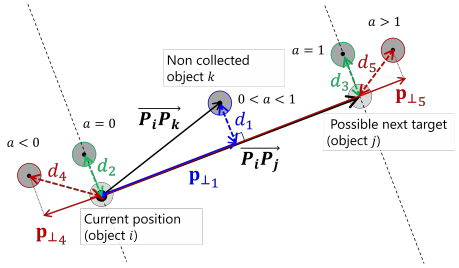
Up to this point, both the objects and the robot twins have been assumed as point masses in the planning phase, under which the effectiveness of the proposed method has been demonstrated. In this section, we examine whether the method is applicable to more realistic scenarios where both the robots and the objects have finite physical dimensions.

A. Collectible/Non-collectible zone

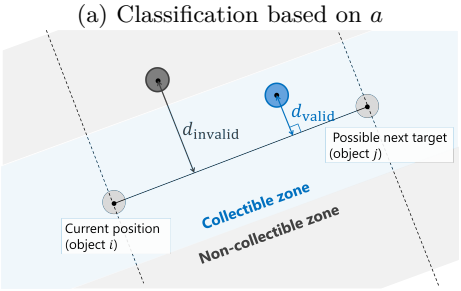
One notable feature of the tethered mobile robot twins is that if another object lies along the trajectory to the next target, the robots can also collect it. Conversely, if an object is located directly in the robots' path, the planner must generate a trajectory that avoids passing over it. To incorporate this concept into the proposed method, we first define two regions as illustrated in Fig. 8: a *collectible zone* and a *non-collectible zone*, corresponding to the possible trajectory of the flexible tool and the robots themselves respectively.

Based on the above concept, we introduce the following two constraints into the algorithm before the energy calculation in Algorithm 2. 1) For any uncollected object at a given step, the object is regarded as collected only when the entire object, with radius r_{obj}, enters the *collectible zone*. Partial entry into the zone does not trigger collection. 2) If any uncollected object k lies even partially within the *non-collectible zone*, the path to object j is considered invalid, and the planner searches for an alternative object.

To determine the relative position of the foot of the perpendicular $\mathbf{p}_\perp$ dropped from any uncollected object k



(a) Classification based on a



(b) The shortest distance from the object k to $\overline{\mathbf{P}_i\mathbf{P}_j}$

Fig. 9: Geometric object classification

to the line segment connecting the current robot position at object i to a potential next target object j, we employ a method based on scalar projection.

Let $\overline{\mathbf{P}_i\mathbf{P}_j} = \mathbf{P}_j - \mathbf{P}_i$ be the vector from the current object $\mathbf{P}_i$ to the target object $\mathbf{P}_j$, and let $\overline{\mathbf{P}_i\mathbf{P}_k} = \mathbf{P}_k - \mathbf{P}_i$ be the vector from $\mathbf{P}_i$ to the uncollected object $\mathbf{P}_k$. Then, the scalar projection coefficient a is defined as:

$$a = \frac{\overline{\mathbf{P}_i\mathbf{P}_j} \cdot \overline{\mathbf{P}_i\mathbf{P}_k}}{\|\overline{\mathbf{P}_i\mathbf{P}_j}\|^2} \tag{6}$$

Using this coefficient a, the foot of the perpendicular $\mathbf{p}_\perp$ is given by:

$$\mathbf{p}_\perp = \mathbf{P}_i + a\,\overline{\mathbf{P}_i\mathbf{P}_j} \tag{7}$$

The geometric location of the foot $\mathbf{p}_\perp$ can be classified based on the value of t as shown in Fig. 9a:

Case 1	$0 < a < 1$: $\mathbf{p}_\perp$ lies within $\overline{\mathbf{P}_i\mathbf{P}_j}$.
Case 2,4	$a \leq 0$: $\mathbf{p}_\perp$ lies on the extension of $\overline{\mathbf{P}_i\mathbf{P}_j}$ on or beyond $\mathbf{P}_i$.
Case 3,5	$a \geq 1$: $\mathbf{p}_\perp$ lies on the extension of $\overline{\mathbf{P}_i\mathbf{P}_j}$ on or beyond $\mathbf{P}_j$.

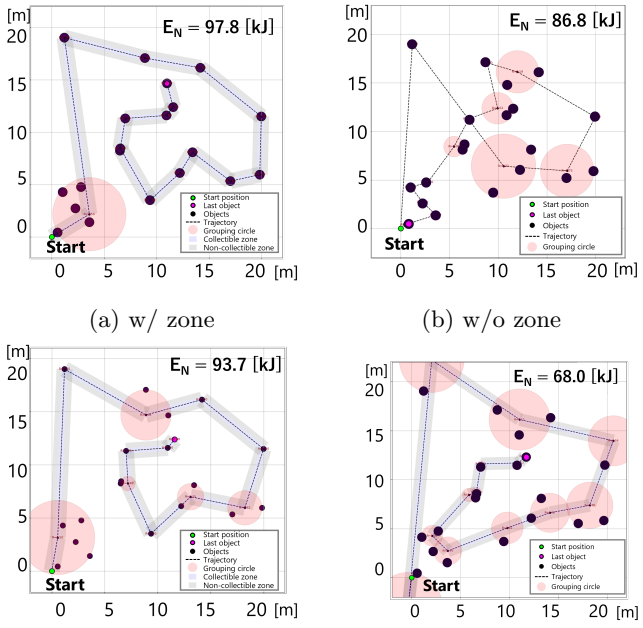
The shortest distance d from object k to $\overline{\mathbf{P}_i\mathbf{P}_j}$ is determined as follows:

- For Case 1, d is the distsnce between $\mathbf{P}_k$ and $\overline{\mathbf{P}_i\mathbf{P}_j}$.
- For Cases 2 and 4, d is the distance to $\mathbf{P}_i$.
- For Cases 3 and 5, d is the distance to $\mathbf{P}_j$.

If this distance d satisfies the following condition:

$$\frac{H}{2} - r_{\text{obj}} < d \leq \frac{H}{2} + H_r + r_{\text{obj}}, \tag{8}$$

then object k is considered to be located in the *non-collectible zone* as shown in Fig. 9b. In such cases, the path connecting objects i and j is deemed invalid, and object j is excluded from the set of possible next targets.



(a) w/ zone (b) w/o zone

(c) w/ zone ($r_{\mathrm{obj}} = 0$) (d) w/ zone (Em-adapG)

Fig. 10: Effect of collectible/non-collectible zone

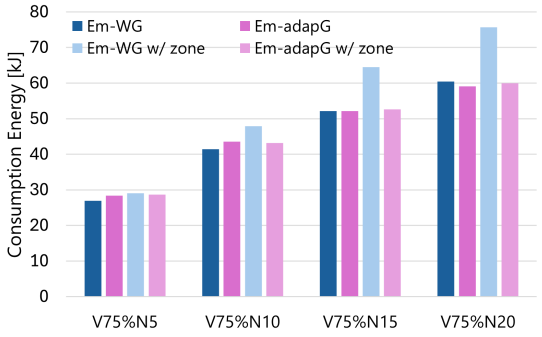


Fig. 11: Adaptable grouping

A route is infeasible if, from state S_{current}, every candidate target causes a non-collectible zone violation or the object radius r_{obj} cannot fit within the sweep circle, leaving no valid path forward. We conducted simulations under the fixed K_v at 1.0 and the entire field at 20 m × 20 m with six different object radii ranging from 0.0 m to 0.15 m, and for each case, we evaluated how many out of ten object configurations resulting in succeeded in generating a path. Here, the width H of the collectible zone is set to 0.50 m, representing the span of the flexible tool during transport between the objects.

The number of configurations that the planner succeeded in generating a path correlates more strongly with the number of objects than with their radii; the planner successfully generated a path in 8 to 9 configurations in $N = 10, 15$ while 5 to 6 in $N = 20$ in any R_{obj} In order to varify the effect of collectible/non-collectible zone constraints, we applied the same constraints to the baseline Em-WoG, resulting in a greater number of configurations where the path is generated than the proposed method for all numbers of the objects. This suggests that grouping increases the likelihood of violating the non-collectible zone constraint.

To verify this, we conducted simulations with varied initial grouping radii: increased ($R_{\mathrm{max},0} = 4.0, 4.5$) and decreased ($R_{\mathrm{max},0} = 3.0$). The results indicate that smaller initial radii lead to more success in path generation. This is because the proposed method does not often opt to avoid grouping, even when such a choice would be more favorable.

This tendency also appears in configurations where the path is generated. Fig. 10 shows examples of generated paths. Fig. 10a shows the result with finite-sized objects, and Fig. 10b shows the case without any zone constraints. Comparing the two, the former results in minimal grouping, and the robot visits one object at a time. In contrast, Fig. 10c shows the result under the zone constraints but with point-mass objects. Compared to Fig. 10a, more aggressive grouping is observed, indicating a strong relationship between grouping behavior, object size, and path generation possibility.

B. Adaptable grouping

To address the infeasibility of path generation, we implemented an *Adaptive Grouping* strategy under energy minimization (Em-adapG). Our original method, Em-WG, selects only one circle with a sweep angle chosen from $360/N_c$ candidates. Since the planner prefers circles including the maximum number of objects, the maximum radius $R_{\mathrm{max,current}}$ is chosen in most cases. In contrast, Em-adapG retains the top C candidates with smaller estimated future energy based on the remaining travel distance, following these steps (the robots are at target i):

- Step 1 Generate $360/N_c$ sweep angles and four radii $wR_{\mathrm{max,current}}$ with $w = \{0.0, 0.5, 0.75, 1.0\}$ (72 candidates for $N_c = 24$) around the next target j.
- Step 2 Sort all candidates by energy consumption $E_{\mathrm{stage},k}$.
- Step 3 Apply a Pareto filter to remove dominated candidates (larger $E_{\mathrm{stage},k}$ and larger added mass).
- Step 4 Retain the top C candidates with the smallest estimated future energy $E_{\mathrm{est},k}$.

$E_{\mathrm{est},k}$ is defined as:

$$E_{\mathrm{est},k} = E_{\mathrm{stage},k} + \lambda\, E_{\mathrm{unit}}\, L_{\mathrm{rem}} m_k \tag{9}$$

where m_k is the mass of objects in candidate k, E_{unit} is the energy to transport unit mass per unit distance, L_{rem} is the estimated remaining travel distance calculated using the Beardwood–Halton–Hammersley theorem [16], and $\lambda = 0.20$. Small λ leads planner keep more grouping options.

Simulations under the same conditions as in Section V-A showed that the planner successfully generated feasible paths in all configurations. This is because it can select not only whether to form a grouping circle but also

adjust its radius, making pathfinding easier than Em-WG under zone constraints. We compared the total energy consumption with the original method under $K_v = 0.5$ (Fig. 11). Em-adapG with zone constraints outperformed Em-WG with the same constraints, but not in the unconstrained case, and sometimes performed worse for small N. This occurs because Em-adapG tends to form unnecessary grouping circles for one or few objects, whose centers may be far from the actual objects, resulting in longer paths. However, with zone consideration, the planner can form smaller and more effective groups (Fig. 10d), allowing multiple objects to be collected simultaneously and reducing travel distance. To further improve Em-adapG, the grouping method should allow the target object itself to serve as the circle center, not only positions on the rim.

However, an important advantage of applying both collectible and non-collectible zones is that the generated routes better reflect the physical constraints of the flexible tool. As shown in Fig. 10b, the path of Em-WG w/o zone crosses itself multiple times, whereas the routes with zone constraints avoid such intersections. This indicates that the flexible tool and the robots are less likely to interfere with uncollected objects, making the routes more feasible for real-world deployment.

VI. Conclusion and Future Work

In this study, we addressed a path planning problem for minimizing the energy consumption of the tethered robots twins for collecting multiple objects using a flexible tool such as a tether. We formulated an energy-efficient collection route by leveraging dynamic programming, where the state is defined in terms of accumulated energy consumption. Furthermore, we introduced a dynamic objects grouping that determines the area for collecting objects all at once based on the current transported load. This allows the planner to account for the mechanical influence of heavy loads on the robots. The formulation of travel energy incorporates the robot's maximum output velocity as a variable. This enables to reduce both energy consumption and travel time.

We conducted simulations under various parameter conditions demonstrating the robustness and superiority of our method across different scenarios. We also conducted simulations under settings closer to real-world scenarios. As a result, while some cases imply that the grouping strategy should be improved, in other cases, the generated paths were favorable for maneuvering the flexible tool, resulting in more practical trajectories.

Our model focuses on global transportation costs. Therefore, the detailed robot motions such as acceleration or turning are beyond our study. Validation through physical simulation and real experiments is left for future work, which may reveal the need for local planning.

While this study focuses on a specific scenario of two robots collecting multiple objects using a flexible tool, the underlying framework is extensible. Both the grouping radius and maximum output velocity were modeled as linearly decreasing functions of the transported load. By modifying these equations, the approach can be extended to tasks such as aerial delivery in disaster response, where a drone carrying a large payload adjusts its delivery sequence based on recipient group sizes. As the payload decreases, the drone's maximum velocity increases, enabling more efficient routing. Moreover, the proposed method remained effective under different starting positions, suggesting applicability to scenarios such as planetary rovers operating around a lander. Future work will aim to extend the method to a wider range of mission-critical applications.

References

[1] Su, Y., Jiang, Y., Zhu, Y., and Liu, H., "Object gathering with a tethered robot duo," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2132-2139, 2022.

[2] Kim, S., Bhattacharya, S., and Kumar, V., "Dynamic simulation of autonomous boats for cooperative skimming and cleanup," *Proc. ASME Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf.*, Paper no. V06BT07A066, 2013.

[3] Bhattacharya, S., Heidarsson, H., Sukhatme, G. S., and Kumar, V., "Cooperative control of autonomous surface vehicles for oil skimming and cleanup," *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 2374-2379, 2011.

[4] Nakamura, H., and Ishigami, G., "Towards collaborative object transportation control by tethered mobile robot twins," *Proc. ICAM*, 2024.

[5] Wojnar, T., and Stołtny, B., "An analysis of rope tension forces while towing a sailplane," *Saf. Def.*, vol. 8, no. 2, pp. 27-35, 2022.

[6] Sinibaldi, M., Bulian, G., and Francescutto, A., "A nonlinear dynamics perspective on some aspects of towing operations relevant to safety and energy efficiency," 2013.

[7] Cha, H., Lee, C., Xie, C., Lu, Q. C., Eun, J., and Cheong, T., "An exact A*-based tree search algorithm for TSP with sequence-and-load dependent risk," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 9, pp. 10817-10834, 2024.

[8] Lysgaard, J., and Wøhlk, S., "A branch-and-cut-and-price algorithm for the cumulative capacitated vehicle routing problem," *Eur. J. Oper. Res.*, vol. 236, no. 3, pp. 800-810, 2014.

[9] Xiao, Y., Zhao, Q., Kaku, I., and Xu, Y., "Development of a fuel consumption optimization model for the capacitated vehicle routing problem," *Comput. Oper. Res.*, vol. 39, no. 7, pp. 1419-1431, 2012.

[10] Zacharia, P., Drosos, C., Piromalis, D., and Papoutsidakis, M., "The vehicle routing problem with fuzzy payloads considering fuel consumption," *Appl. Artif. Intell.*, vol. 35, no. 15, pp. 1755-1776, 2021.

[11] Lu, Y., Hao, J. K., and Wu, Q., "Solving the clustered traveling salesman problem via traveling salesman problem methods," *PeerJ Comput. Sci.*, vol. 8, Art. no. e972, 2022.

[12] Anaya Fuentes, G. E., Hernández Gress, E. S., Seck Tuoh Mora, J. C., and Medina Marín, J., "Solution to travelling salesman problem by clusters and a modified multi-restart iterated local search metaheuristic," *PLoS ONE*, vol. 13, no. 8, Art. no. e0201868, 2018.

[13] Pustilnik, M., and Borrelli, F., "Clustering heuristics for robust energy capacitated vehicle routing problem (ECVRP)," *arXiv preprint arXiv:2403.13906*, 2024.

[14] Gaon, T., Gabay, Y., and Weiss Cohen, M., "Optimizing electric vehicle routing efficiency using k-means clustering and genetic algorithms," *Future Internet*, vol. 17, no. 3, Art. no. 97, 2025.

[15] Held, M., and Karp, R. M., "A dynamic programming approach to sequencing problems," *J. Soc. Ind. Appl. Math.*, vol. 10, no. 1, pp. 196–210, 1962.

[16] Beardwood, J., Halton, J. H., and Hammersley, J. M. (1959). The shortest path through many points. Mathematical Proceedings of the Cambridge Philosophical Society, 55(4), 299–327.