Optimal Swarm Distribution for Collective Transport

Brian Francis
RBE Dept.
bfrancis@wpi.edu

Apratim Mukherjee
RBE Dept.
amukherjee3@wpi.edu

Steven Hyland
RBE Dept.
smhyland@wpi.edu

***Abstract*—The use of collective transport by a group of robots is a well studied task with a variety of approaches in the literature. Even with basic shapes, the grasping strategy of robot swarms is often sub-optimal when considering two-dimensional manipulation. By instead distributing the load evenly among all robots actively, the issue of uneven load distribution can be mitigated. This improvement can avoid uneven wear and tear, and can minimize the incidence of robot failures. This work proposes an approach to address optimal distribution of robotic agents across any payload shape, such that each robot is taking an equal load. We achieve this in two ways, representing different moments of computation (online vs pre-computation). The first approach is using a physical model based on charge distribution, in which each robot sequentially attaches to the object in a way to equalize the force they experience from the payload's inertia. The second method is performed by structuring each configuration of the robots around the payload as a node in a graph. By applying a graph search, we are able to find the optimal configuration based on distributed geometry.**

I. Introduction and Background

Collective transport [1] by swarms of robots as a field of research has gained significant attention in recent years due to its potential applications in various fields such as material handling, home care, construction, and manufacturing. However, the effective coordination of a swarm of robots to transport an object in a collective manner is a challenging task due to the need to balance load distribution and ensure stable grasping [2]. These requirements improve many aspects of the transport task, namely in the stability of the payload and robustness to external disturbances during manipulation. In this paper, we propose an approach to address these problems by developing a pre-grasp strategy prior to the collective transport task. We have approached this problem in two ways, the first being a physics-based methodology mimicking surface charge distribution to arrange the robots. Here, the robots perform the equalizing calculation online, starting detached from the payload up until the transport task. The second method exploits a graph search algorithm to determine an optimal grasp strategy prior to starting any robot motion offline.

Much of the prior research on swarm robotics investigate the coordination and control of the robot swarm to perform various tasks such as exploration, formation control, and object transportation. In particular, the problem of collective transport is a point of focus. Groß *et al.* [3] propose a force-based method for group transport that combines both pulling and pushing forces, allowing robots to adapt their positions based on perceived forces and the object's orientation. More broadly, Berman *et al.* [4] introduce a geometric approach for designing control policies in spatially heterogeneous swarms, applied to the problem of commercial pollination. While not specifically tailored to collective transport their method takes into account the payload's geometry, mass distribution, and robot capabilities to produce predictable swarm behaviors. Wang *et al.* [5] present another force-based, decentralized method. In their work, robots estimate the object's weight distribution and the forces exerted by other swarm members, allowing them to adapt their positions and forces dynamically. Here, a leader robot determines the direction of the swarm, with the remainder of the swarm amplifying the effective force. One of the more comprehensive strategies for determining optimal agent arrangement for lifting is seen in [6] and [7], where the robots can find and optimize their grasp policy autonomously. The agents are only provided the shape of the object, and based on an estimate of the center of mass (CoM), distribute themselves evenly. Subsequently, the robot's perceived force of the payload's weight is analyzed and if the CoM is determined to be different to the estimate, the agents can re-arrange to a more optimal arrangement. While this is a seminal work in determining an optimal grasp policy for a multi-robot system, it takes many assumptions that idealize real-world scenarios, such as a relatively regularly-shaped payload, and requires the robots to *lift* the payload. It also requires the generation of thousands of potential candidates for formations, which is a considerably slow and taxing process. In [8], four different formations are analyzed by using a vector field composition method. The robots reach a quorum using global data in experiment, however, the grasp strategies aren't analyzed for optimality. That is, if a formation is considered satisfactory, the swarm proceeds with the transportation task. While this approach is practical in a laboratory environment, environmental perturbations or dynamic obstacles can render these formations inadequate. Other research focuses on pushing a load while utilizing predetermined robot positions, which ignores the challenge of distribution [2]. Our work differs by focusing on keeping the payload on the ground and pushing, and having the agents determine the optimal distribution as described above.

II. Methodology

As mentioned in section I, we proposed two methods. The first method is inspired by the ways similar charges distribute over the surface of a form. By having the robots interact like particles they can distribute themselves in approximate uniformity over any sort of surface.

Our second approach is to construct a graph consisting of the different robotic agent's configurations about the payload. In this graph, different nodes correspond to the different robot positions around the payload, after a small adjustment of their positions. A heuristic is added to find a solution if an optimal configuration is already known, or to optimize each prior configuration. This way, the graph search can be conducted prior to the collective transport task as a pre-computation step to predetermine the optimal configuration for any payload and number of robots.

A. *Artificial Potential*

Since the goal of the project was to test the ability of the agents to arrive at a payload and then arrange themselves without any initial assistance the agents first states are to simply point at the payload and then drive towards it until they are touching the payload. After an agent is in contact with the payload they announce to all the other agents that they are ready to begin. When all agents are broadcasting that they are prepared the swarm begins the force dispersion.

In order for the agents to distribute themselves around, the swarm and the payload apply imaginary repulsive forces on each agent. The force is applied along the vector connecting the entities, and the magnitudes is related to the distance via the inverse square law. The force applied on any robot F_i can be calculated via Eq 1 where d_{ij} is the distance between agent i and any given neighbor agent j, d_{ii} is the distance between an agent and the payload CoM, $\hat{v_{pi}}$ is the unit vector from the payload to the current agent, $\hat{v_{ji}}$ is the unit vector from agent j to agent i, n is the number of agents in the swarm, and coefficients a and b are parameters to control the relative strength of the force from other agents or the payload respectively.

$$F_i = \frac{b}{d_{ip}^2} \cdot \hat{v_{pi}} + \sum_{j=1}^{n} \frac{a}{d_{ij}^2} \cdot \hat{v_{ji}} \tag{1}$$

The calculation relies on the range and bearing sensors of the footbots to communicate both the distances between the agents and the angle between them. Once the agents are all within the range of the payload they disperse according to their calculated force vectors for time t_d seconds which in our experiment was set to 10 seconds. The goal is to simply have each agent face along their force vector and take a small step without moving substantially away from the payload. After t_d the agents move into a wall following behavior

In order to keep the robots in range of the payload while also distributing themselves around the shape the agents perform a simple wall following algorithm after. Since the footbots are differential drive robots we setup the wall following behavior to control the speed of the inside wheel while the outside wheel speed v_o is set to a constant value of 3. To determine which side of the robot is closest to the payload we take a summation of the proximity sensors on the left and right and select the greater value as the inside. Eq 2 calculates v_i which refers to the inner wheel velocity. The calculation uses p_s which is the highest proximity sensor reading between the robot and the payload, and c_t the turning coefficient to calculate the wheel speed required to maintain a desired offset distance o.

$$v_i = c_t * (p_s - o) + 1 \tag{2}$$

The wall following algorithm provides a lightweight way to have the agents surround a payload. The swarm performs this step for t_f seconds. After this the agents return to the force dispersion state. They take another step in the direction of the repulsion force before beginning the loop over again starting with the alignment and approach states. The system repeats this loop N times before coming to a final configuration.

B. *Graph Search*

In order to provide a contrast with an online solver as provided by the Artificial Potential method, we decided to approach the problem using a graph search approach, which was designed to behave like a pseudo-breadth-first search. This graph search has been titled *Optimal dynamic distribution of Robots using Graph Search* or *OddRuGS*. As mentioned earlier, each node in the graph corresponds to a certain distribution of agents around the payload. The payload is represented by a 2-dimensional shape with a number of vertices defining the outer edge of the payload. Each vertex is a potential 'grab point' for the agents, whom can only attach at the defined vertices. To more closely mimic the real world, each agent can only move to a vertex that is unoccupied by any other agent. The agents have no communication with each other and behave in a decentralized manner. The algorithm only requires as inputs the shape of the object (a list of vertices) and the location of the CoM. A visualization of the payload and three agents can be seen in Fig. 4*b*.

In exploring the configuration space, we took the approach of a 'random walk' for each agent. That is, at each time step every agent can decide to either stay in its current position, or move to the next closest vertex to its left or right. We have set the chance for *staying* to *0.2* or *20%*. This way, the robots 'walk' away from their initial configuration in a random pattern. This approach was selected over a random sampling of possible configurations and a discretized walk because it was the simplest to implement, given the time constraints of the project. Furthermore, this approach should have the fastest convergence to a 'good enough' solution since the configuration space does not need to be fully explored to determine a valid configuration.

This algorithm is different than a traditional graph search in a few ways. Primarily, the goal node is unknown to both the user and to the algorithm. Additionally, depending on the shape of the payload, i.e. a circle, there are any number of

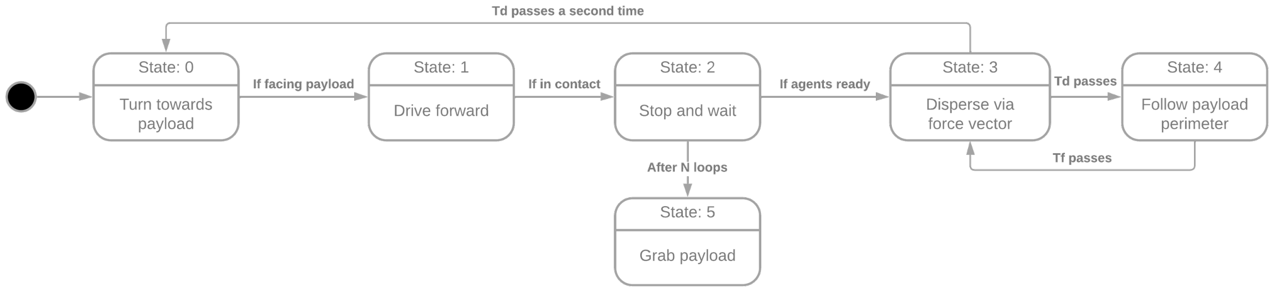


Fig. 1. Artificial Potential State Machine

valid final configurations. Therefore, the goal node or the stop condition of the algorithm is determined by minimizing the current node cost. While the node cost or heuristic can be calculated in a number of ways, we selected to calculate the distance between the payload's CoM and the centroid of the support polygon formed by the agents. This way, we can guarantee that the CoM is actually supported by the robots. An additional common consideration of graph search algorithms is to ensure that previously explored nodes (in this case, robot configurations around the payload) aren't revisited. This ensures a faster convergence to an optimal solution. This was also implemented in the OddRuGS algorithm.

A key difference between this method and the Artificial Potential method is that here, the robotic agents are assumed to be pre-attached to the payload. This simplifying assumption is valid since this approach is used as during pre-computation before collective transport. The OddRuGS algorithm could be improved in a number of ways, primarily by improving the heuristic and the sampling; these considerations are covered in section VI.

III. Experimental Setup

In order to test our distribution methodologies we tasked the swarm with finding a configuration of robots that offered the most stable control of their payload. To test the artificial potential, the agents will start from a short distance away from the payload, and then arrange themselves around the object. For the graph search, the agents were pre-attached to the payload and resemble a worst-case scenario with each agent initialized at subsequent vertices next to one another. We tested the approaches with 3, 5, and 10 agents in order to confirm the scalability of the approach to handle a varying number of robots. Additionally for each swarm size, we tested a variety of shapes which included two non-convex shapes for the artificial potential method. All our experiments for the artificial potential method were run in the ARGoS simulator as shown in Fig. 2, while the graph search method was tested in numpy grid, with the results being observed in a continuously updating matplotlib plot.

The success of the methodology were measured by two metrics that indicate how well distributed the weight of the payload is between the agents and how efficient the distribution methodologies are. The first metric looks at how close to the CoM the centroid of the support polygon is. That is, the smaller the distance between the CoM and the centroid, the better that particular configuration is. The second metric is how long each methodology takes to generate a viable loading configuration. Once the CoM of the payload is encompassed by the support polygon of the robots, the configuration is deemed a minimally viable configuration. The sooner a methodology can converge to a viable solution, the sooner the agents can start transporting their payload. Together, these metrics give an indication of how optimally and how rapidly the distribution methodologies work. We also looked at the success of the methodologies over different sized swarms to test the scalability of the method.

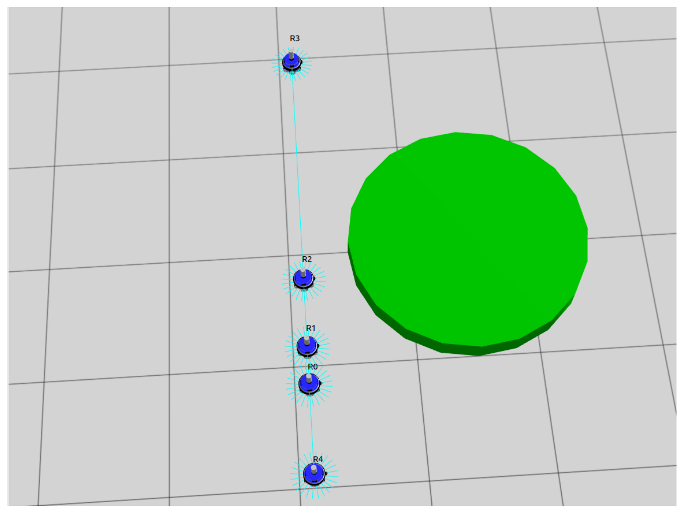


Fig. 2. ARGoS Experiment Initial Configuration

IV. Results

A. Scalability

The first test was to compare how the methodology scales as the number of agents increases. To do this, swarm sizes of 3, 5, and 10 were tested for all of the shapes. The final solutions

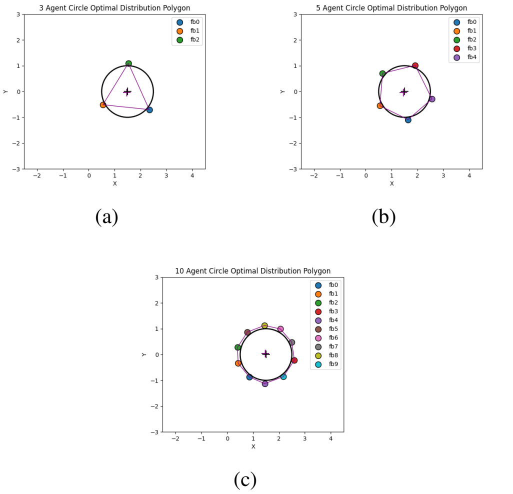


Fig. 3. Grasping polygons determined via artifical potential

for a circle determined by artificial potential are shown in Fig. 3 and for graph search in Fig. 4.

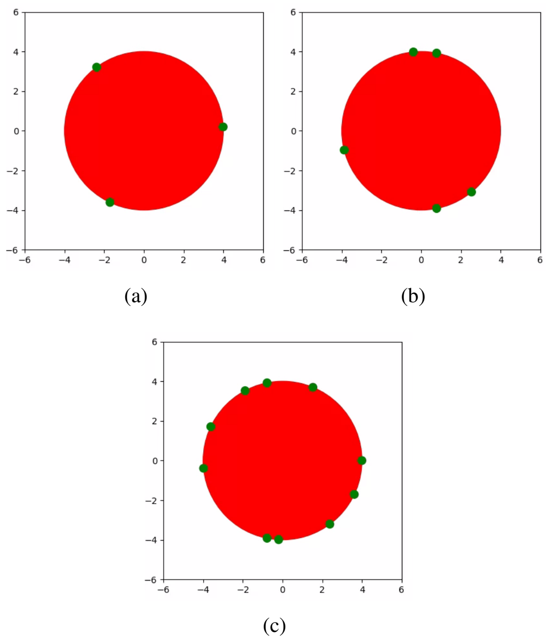


Fig. 4. Grasping polygons determined via graph search for circular payload

For artificial potential, the grasping polygon around a circle takes the shape of a normal polygon. Because of this, at any scale, the artificial potential is able to comfortably arrange around the payload. However the shape of the grasping polygon also becomes more closely aligned with the true shape of the payload. Because of this, greater sized swarms perform better when the object is more difficult to arrange around. In Fig. 6 the three agent swarm is able to find a viable solution over the length of an experiment, but with ten agents the swarm centroid is significantly closer to the true CoM. This trend is exhibited over all of the shapes, and for each test shape

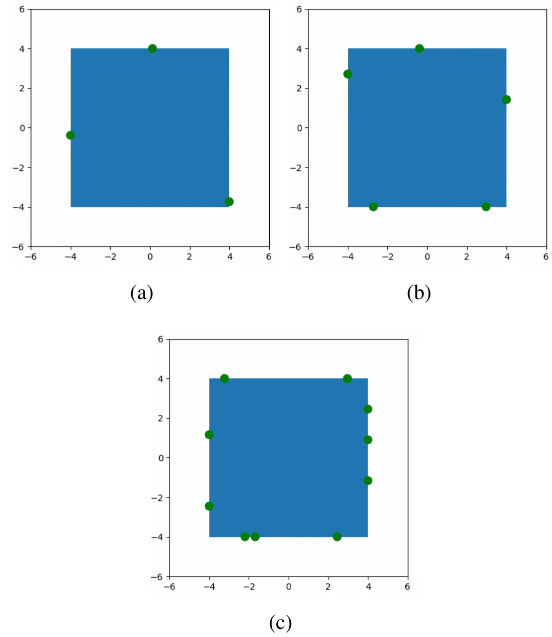


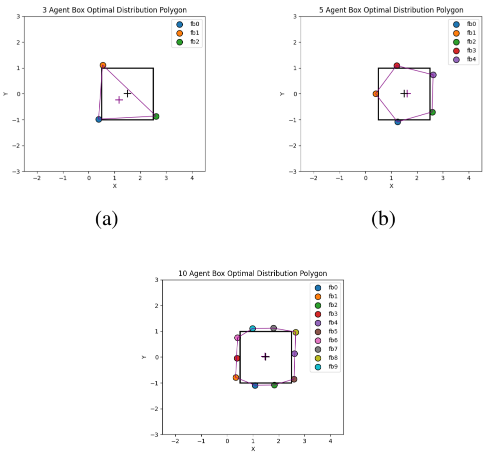
Fig. 5. Grasping polygons determined via graph search for square payload

as the swarm size increased the swarm finds a better solution and faster. There is a carrying capacity limitation as the agents interfere with each other if there are too many robots in close proximity. So long as the shape is large enough for all the footbots to be in contact with the payload this interference is largely negligible to the distribution.

For the graph search, we see similar trends as the potential field method. Interestingly, the algorithm determines an optimal solution to not necessarily be equally-spaced agents around the payload. The reason behind this has already been covered in section II-B. Due to time constraints, only two shapes were tested for the graph search, a circular and a square payload, seen in Figs. 4 and 5, respectively. For the circular payload, the agents are mostly evenly distributed. For the square, the agents also attempt to evenly distribute and closely mimic the results from the artifical potential method.

B. Precision

Artificial potential was also able to achieve a high degree of precision in creating an optimally distributed grasping arrangement. For both the square and the circle the swarm came within just a few centimeters or less of the optimal distribution. Fig. 9 and 10 show the increasing accuracy over the experiment with the agents coming to a viable solution typically under 10 seconds and an optimal one within 30 seconds. The stepwise shape of the plot comes from the iterative nature of the motion. Each redistribution takes the swarm closer to the optimal arrangement, and the horizontal periods come from the swarm preparing to rearrange. We can also observe the saturation in Fig. 10, where 5 and 10 agents hit their minimum viable solution almost at the same time. This can be used to determine the maximum number of robots



(a) (b)

(c)

Fig. 6. Grasping polygons determined via artificial potential for a square payload

that can be used to distribute the forces around the CoM of an object evenly.

Conversely, the graph search method had much more variability in the optimal solutions. The noisiness seen in Figs. 7 and 8, representing the circular and square payloads, respectively is due to the 'random walk' mentioned previously. The random walk means that worse nodes may be explored at every time step, such that a monotonically decreasing error is not seen. However, a viable solution is found more rapidly as the number of agents increases, as expected. Additionally due to our random walk approach, the optimal solution is not always the same: at times, the algorithm will get stuck at a local minimum. Interestingly, for the square payload, the trial with 5 agents resulted in a worse final configuration than for 3 agents. This case represents the algorithm hitting a local minimum. The algorithm was set to run for 1000 total time steps, and increasing this number improved the odds of converging to a better solution. However, this of course also increases computation time.

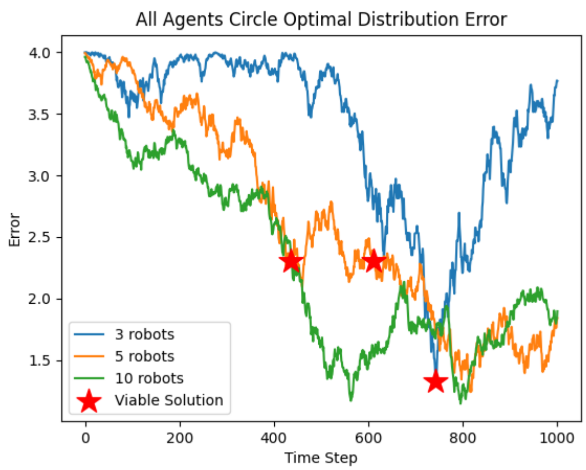


Fig. 7. Graph search CoM error over time for circular payload

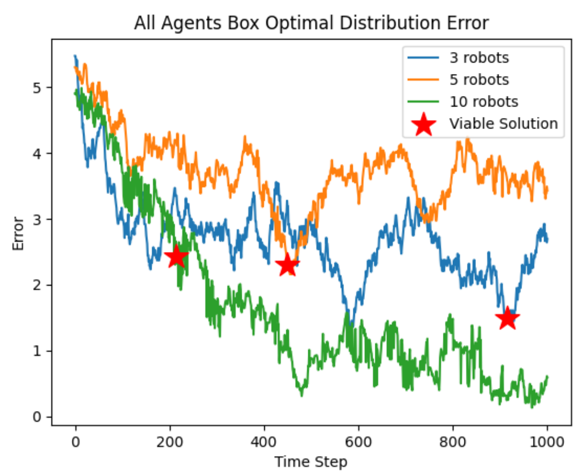


Fig. 8. Graph search CoM error over time for box payload

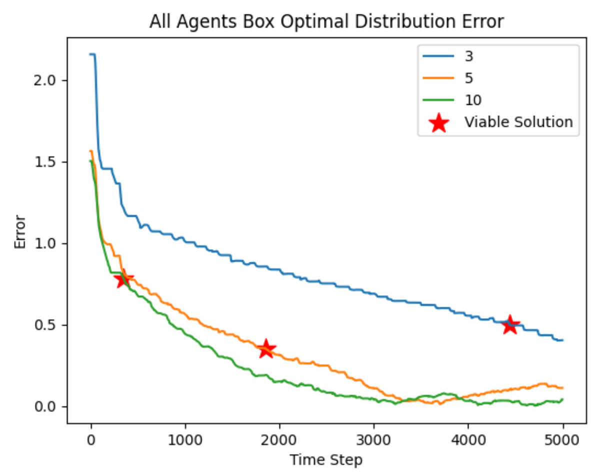


Fig. 9. Artificial potential CoM error over time for box payload

C. Non-Convex Obstacles

We also tested out the artificial potential method on non-convex obstacles, as seen in Fig. 11 and 12. The polygons used for this experiment were still symmetrical, but non-standard shapes, which makes robot distribution across them a non-trivial task.

We observe similar performances for robot placement as the box and cylindrical payloads, with the agents better approximating the shapes as the swarm size goes up. The 3 agent swarms are able to arrange themselves in a configuration that produces a viable support polygon over the duration of the experiment, but they converge to an acceptable solution much quicker when the number of agents are increased to 5 and 10, with the CoM error going down significantly quicker.

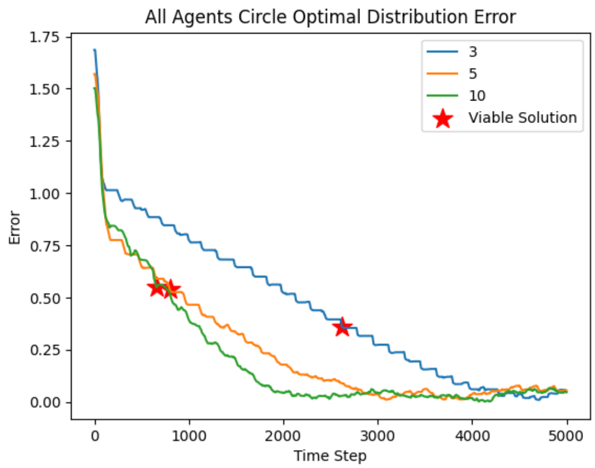


Fig. 10. Artificial potential CoM error over time for a cylindrical payload

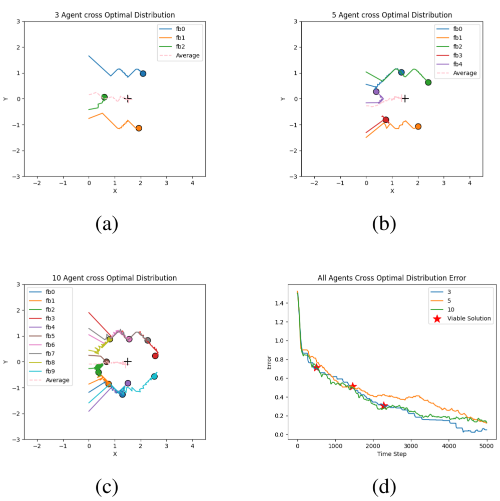


Fig. 11. Final robot locations and errors determined via artificial potential for a cross payload

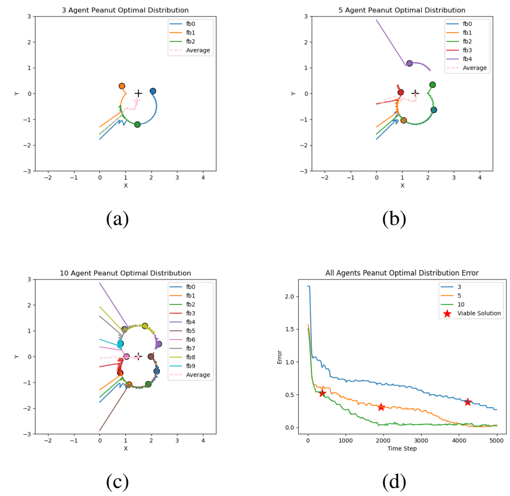


Fig. 12. Final robot locations and errors determined via artificial potential for a peanut payload

V. Comparison and Conclusions

The graph search performs notably worse than the artificial potential method. This can be attributed to the methods for constructing the graph described previously. However, the ability to pre-compute an optimal distribution is useful for freeing computational resources during collective manipulation for other tasks. A potential improvement would be to combine both methods: to pre-compute a feasible solution, and then to send the agents to their respective positions after which the potential field method could finish optimizing the initial solution.

VI. Future Work

While both methods converge to a solution, they could be improved in a number of ways, and tested further on more complex shapes to determine how well the algorithms generalize to other unknown shapes. As mentioned previously, artificial potential can be improved by combining the two methods, using precomputed approximate optimal locations for the robots to start and augment the swarm trial-and-error. Also, the algorithm can be made continuous instead of a step-wise method, which would speed up the algorithm since the robots can test out multiple configurations continuously instead of stopping and rearranging. Another notable test could be added in the form of a resiliency check, where the swarm size changes dynamically during the process. Failures can be induced in certain robots, disrupting the support polygons, and the adaptability of the algorithm can be studied. We can also introduce malicious agents into the swarm, which apply forces that are detrimental to the movement of the object, and observe how well the robots compensate.

For graph search, the main drawbacks are the exploration of new configurations and the heuristic used to define the cost of each configuration. While the 'random walk' is a simple implementation, it is slow to explore and can potentially miss better, lower cost configurations. By changing to a discretized movement pattern, where every possible configuration is tested, OddRuGS could guarantee that the final solution is the global optimum. However, this of course has the drawback of a longer convergence time. As for the heuristic, the distance between CoM and centroid is a good approximation of an ideal configuration. However, this implementation combined with the random walk, means that the algorithm often gets stuck in local minima, where no neighboring nodes have a smaller cost than the current while a more optimal solution much further from the current node may exist. This could be improved greatly by instead applying a force-distribution heuristic. This could be designed in a physics-based way, in which the cost of a node corresponds to the maximum difference between the portion of the payload each robot is supporting.

VII. Github Links to Code

- Artificial Potential

- Graph Search

References

[1] Michael Rubenstein, Adrian Cabrera, Justin Werfel, Golnaz Habibi, James Mclurkin, and Radhika Nagpal. Collective transport of complex objects by simple robots: theory and experiments. pages 47–54, 05 2013.

[2] Akanyeti O Tuci E, Alkilabi MHM. Cooperative object transport in multi-robot systems: A review of the state-of-the-art. *Frontiers in Robotics and AI*, page 5:59, 2018.

[3] Roderich Groß and Marco Dorigo. Towards group transport by swarms of robots. *International Journal of Bio-Inspired Computation*, 1, 01 2009.

[4] Spring Berman, Vijay Kumar, and Radhika Nagpal. Design of control policies for spatially inhomogeneous robot swarms with application to commercial pollination. In *2011 IEEE International Conference on Robotics and Automation*, pages 378–385, 2011.

[5] Zijian Wang and Mac Schwager. Multi-robot manipulation without communication. In *International Symposium on Distributed Autonomous Robotic Systems*, 2014.

[6] J. Sasaki, J. Ota, E. Yoshida, D. Kurabayashi, and T. Arai. Cooperating grasping of a large object by multiple mobile robots. In *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, volume 1, pages 1205–1210 vol.1, 1995.

[7] Youhei Kume, Yasuhisa Hirata, Kazuhiro Kosuge, Hajime Asama, Hayato Kaetsu, and Kuniaki Kawabata. Decentralized control of multiple mobile robots transporting a single object in coordination without using force/torque sensors. volume 3, pages 3004–3009, 01 2001.

[8] Jonathan Fink, Nathan Michael, and Vijay Kumar. Composition of vector fields for multi-robot manipulation via caging. In *Robotics: Science and Systems*, volume 3, 2007.