International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

Self-disassembly of swarm robots with collision avoidance

V.Nikith Varma19BEC0658 B.Tech [ECE]
Vellore Inst. Of Tech., Vellore.

M.Ajay Kumar19BEC0118 B.Tech [ECE]
Vellore Inst. Of Tech., Vellore.

Shaik Shajahan19BEC0667 B.Tech [ECE]
Vellore Inst. Of Tech., Vellore

.

***Abstract*— In this work, we offer a system of large-scale robot units capable of autonomously forming a variety of user-specified forms. We choose a subtractive rather than a cumulative technique, which is different from previous publications on the same issue. A system is formed by an initial dense, immobile arrangement of robots, and it is up to each individual robot to determine whether or not it is part of the requested shape. Robots that aren't part of the user-selected shape disappear, leaving just those robots in place.**

I. INTRODUCTION

The field of research known as "swarm robotics" focuses on how vast numbers of robots may work together using just a few globally applicable principles. Swarm robots takes its cues from insect communities, which may accomplish goals that would be impossible for a person to do on their own. What a swarm truly is and how it functions is crucial information. According to the literature, a swarm is "a large number of locally interacting people with shared objectives," which indicates that the goal is to create systems with swarms of robots that interact with one another to achieve certain common goals, much as natural swarms do.

The swarm robots may use either an additive or a subtractive approach to the job at hand. A group of robots work together until they have fused into the proper form and completed the assignment using an additive method. As an example, think of 3D printing, where layers of material are added and fused together in strategic locations to produce the desired object. The goal of the subtractive method is to have the remaining robots construct the appropriate shape, without the assistance of the pre-assembled robots. A excellent example of a subtractive method is cutting a paper template to the required form. Natural systems use both additive and subtractive strategies.

The additive method, also known as the self-assembly technique, begins with the seed module, which specifies the coordinate system's origin and orientation in addition to the module's form or structure. The remaining modules locate with respect to the seed, and the shape is built up in successive layers. This method, also known as "guided growth," has been used in mobile collectives, where robots have complete freedom of movement, and in self-reconfigurable systems, where robots are limited to moving along docking sites. Many additive self- assembly methods have been investigated since the additive method is more well-known and more extensively used than the subtractive method; nevertheless, the robot's mobility is strongly constrained by parallelism, thus it has to be very efficient.

precise. As a result of these causes, the period required for form creation is rather lengthy.

The subtractive method, also known as the self-disassembly method, is the one that has seen the least use so far. This method relies on the robots being first organized in a dense, large group while they are idle. Coordinate systems are formed when one or more seed robots get the ball rolling, and the robots that aren't needed for the shape or structure leave the field. Commonly, a 2D rectangular lattice is used to assemble these robot units.

A high degree of motion parallelism and a low level of necessary motion accuracy are two benefits of the subtractive technique that may make it more appealing than the additive approach as the number of swarm robots grows. The self-disassembly may be used on a fleet of identical robots, such as modular robots, that can be organized in a regular lattice. In this scenario, the robot modules may take use of directional communication to quickly and accurately assign coordinates. It's important to remember that these solutions have only been tried out on very few modules so far.

Here we provide a model for the programmable self-disassembly technique, which employs just local sensing and interactions to completely automate a group of mobile robots operating in free space. Through distance sensing and multilateration, the robots construct a continuous coordinate system that can accommodate for localization mistakes on a smaller scale. After the coordinate system has been formed, the robots disassemble themselves using a decentralized consensus mechanism. In the end, the robots that aren't part of the form will walk away from it on their own without altering it, all with the help of a single light. Our theoretical findings reveal which shape classes can be produced using our method, and how the shape class influences the self-disassembly process. After verifying our method on the MATLAB platform with a total of 1,225 robots, we report experimental findings showing great reliability and accuracy despite the noise and faults inherent in large-scale collectives.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

II. SELF-DISASSEMBLY ALGORITHM

A. *Robot Implementation*

Here, we detail how the robot was really built. Each individual robot was programmed to detect an external light source, do calculation and localization, and move about. There is no need to worry about robots colliding because to their built-in communication capabilities. These robots are set up in a predetermined coordinate system on a hexagonal grid. These robots have the default form of a 'o' since that's what the MATLAB plot function produces.

B.Implementation and Algorithm

Starting with a setup in which all of the robots are in a densely packed group, the primary objective of this method is to produce a user-specified shape as the ultimate outcome of the swarm robots. In this case, we check to see whether the user-selected shape's ultimate dimensions are feasible inside the initial set-up of swarm robots. In the first step, three robots are selected at random to serve as the seed robots, and they are localized using the starting configuration to serve as the coordinate system's origin and initial orientation. Following this, the algorithm will execute in three stages:

I Self-Assembly, where robots in the desired shape remain stationary while those outside the shape move away from it; (ii) Transition using a Consensus Algorithm, where robots collectively determine when all robots have localized in order to transition to the next step; and (iii) Self-Disassembly, where robots in the desired shape remain stationary while those outside the shape move away from it.

i.Coordinates are formed in a decentralized manner.

Using multilateration, a robot may determine its position relative to three or more previously-located neighbors. If the robot has n neighbors within d1 of its current location, then. . . locations p1, and,dn distant from it. . . When given a,pn, this robot I estimates its location as:

When a robot has at least three neighbors who all match the following conditions, it starts thinking about localizing: (a) the minimum angle of these three neighbors is more than 35%; and (b) two of these neighbors are within 1.25 body lengths and the other is within 2.5 body lengths. First, the robots must not be localized relative to their almost-collinear neighbors; if they are, the best-guess location might be ambiguous in the other direction. The second requirement guarantees the

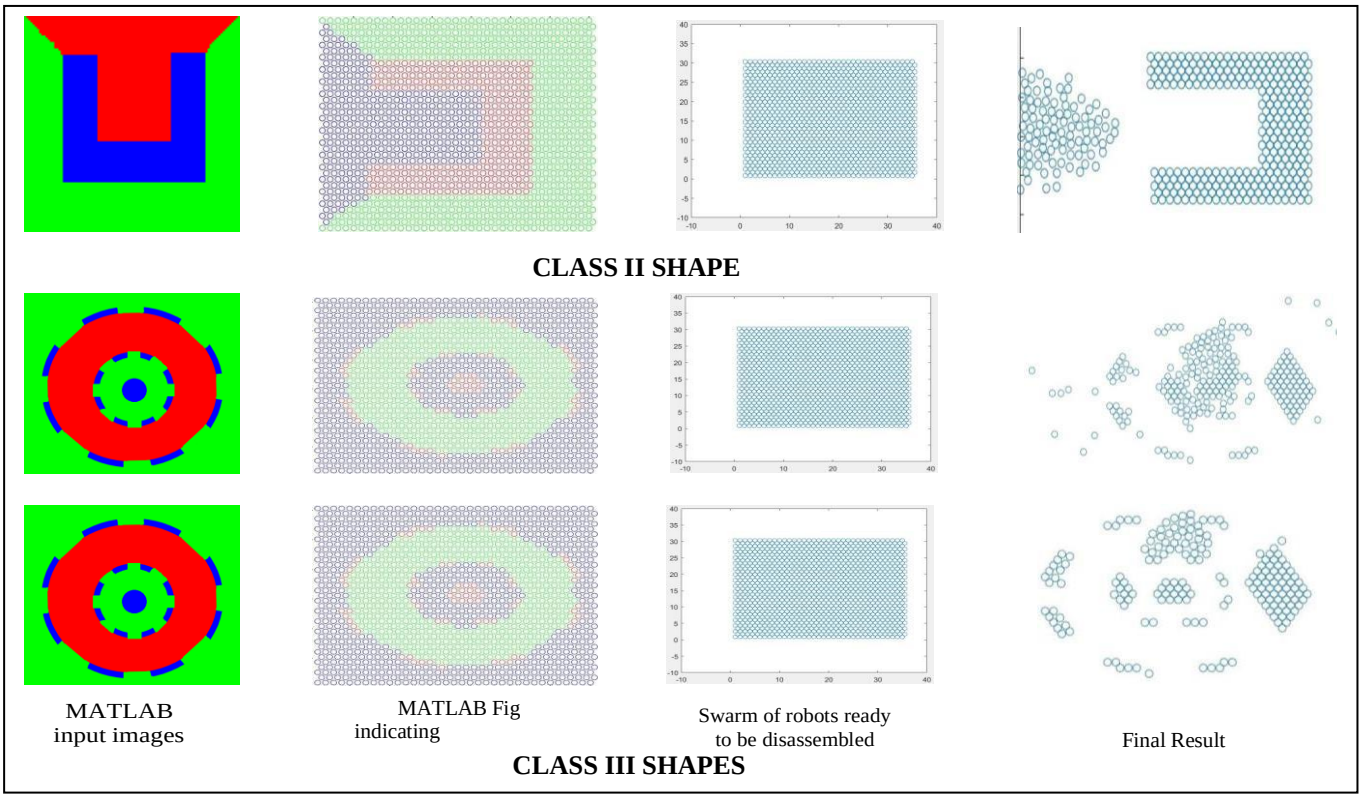
$$\mathbf{p}_i^* = \arg\min_{\mathbf{p}_i} \sum_{j=1}^{n} \frac{\left| d_j - \|\mathbf{p}_i - \mathbf{p}_j\| \right|}{d_j},$$

The robots in this triangle have had their distances calculated, and they are generally close to the mark. Assuming b) has been met, the robot will wait up to three iterations before checking to see if it can acquire more neighbors who meet c). As a result, the multilateration process becomes more consistent as each robot uses a larger pool of neighbors for averaging purposes.

i.Algorithmic Consensus for a Smooth Transition

Once all robots have been localized, the robot swarm must move on to the next phase, in which the robots that don't belong in the desired shape disengage. To achieve this, a basic consensus algorithm based on rumors is used.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362



An agreed upon value is broadcast by all robots. For a robot that cannot determine its location, this consensus value is always 0. The consensus value for a robot with localization capabilities is calculated by adding 1 to the lowest value among its neighbors. When the value of a robot reaches a certain threshold, the robot group concludes that agreement has been established.

Collision-Avoidant Self-Disassembly i.

All robots that don't fit within the user-defined form must leave the configuration for it to be completed, and they do so primarily via collision avoidance, phototaxis, and anti-phototaxis. When a moving robot gets too near to its neighbors, it does collision avoidance. Robots may be programmed to move in a phototaxic or anti-phototaxic fashion, respectively, depending on the shape you give them. If the robot detects an immobile neighbor within two body lengths, it will take precautions to prevent a collision. This reduces the likelihood that moving robots would collide with stationary robots inside the form, but at the expense of a slower rate of self-disassembly. A robot will use phototaxis or anti-phototaxis if it detects no other localized robots moving within two body lengths.

Class B. Realistic Contours

Robots that aren't part of the shape being formed must back away from it so as not to mess it up. We use phototaxis and anti-phototaxis, which are quick and robust against sensory and actuation noise in robots. However, this does need

Leaving the shape must be possible for all or almost all of the robots using any of these two motions, which places a limitation on the system. Below, we detail the many shapes that may be solved by self-disassembly given this limitation. We express the forms as polygons and categorize them into three groups.

In the first category are the forms that can be resolved with just anti-phototaxis. Well-known "star-shaped polygons" may be described as follows. At least one point within the polygon has a ray that, when cast, would hit the perimeter of the polygon precisely once. As long as the light source is positioned in such a manner, any robot located outside the form may retreat in a straight line (i.e., engage in anti-phototaxis) without coming into contact with it. In most cases, there is a linear-time method for finding out how many points within a polygon meet this criterion.

Class 2: Contours that respond to both phototaxis and anti-phototaxis are included in this category. This set of polygons is defined by the following characteristic: For each point outside the polygon, there exists a unique point within it such that any ray thrown from that point either does not contact the perimeter of the polygon, or does so precisely twice. Since phototaxis is more reliable on robots, we give it preference over anti-phototaxis if a ray does not cross the perimeter at any point along the ray. If a ray has two points of contact with the boundary, phototaxis may be performed from locations along the path from the light source to the first intersection, and anti-phototaxis can be performed from locations beyond the second junction.

The robots in a Class 3 polygon cluster in response to the light source, using phototaxis and anti-phototaxis strategies.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

Class	Shape	Positive Error	Negative Error	Mean Error
I	Triangle	+0.245%	0	+0.245%
	Spanner	+4.73%	0	+4.73%
	5 point Star	+0.245%	0	+0.245%
II	U Shape	+7.67%	0	+7.67%
III	Shield 1	+13.469%	0	+13.469%
	Shield 2	+8.163%	0	+8.163%

Each phototaxis movement requires the other to cross its path. Because of the limitations imposed by our setup, self-disassembly into Class 3 polygons is not achievable in most cases. Some robots may be able to use collision avoidance to escape from blocked areas, however this will depend on how deep the blocks are.

III. SIMULATION EXPERIMENTS

A. Simulation Setup and Protocol

We ran 7 trials with 3 different Class 1 shapes, 1 different Class 2 shapes, and 1 different Class 3 forms to ensure the accuracy of our software. To ensure that all robots were able to conduct anti-phototaxis, the Class 1 forms consisted of a 5-point starfish, a spanner, and a triangle with the light source positioned in the center. U-shaped Class 2 structures had the light source positioned above the cavity so that the robots within could use phototaxis and the robots outside could use anti-phototaxis. Class 3's outline took the form of a shield, with phototaxis and anti-phototaxis robots positioned such that they must go in opposing directions to attain their goals. The trials allowed us to evaluate the algorithm's speed and precision.

The following details the procedure followed throughout the experiment. Initially, the robots were set up in a hexagonal grid. There were 1225 robots spread over 35 rows and 35 columns. Across the board, the initial arrangement of seed robots was determined at random.

All experiments began with the light source being positioned properly. No outside help was used during the self-disassembly process. After 200 repetitions of self-disassembly had begun for the class 1 forms, the experiment was halted. Class 2 forms are allowed 500 iterations before the program exits, but class 3 shapes are limited to 2000.

The B. Result Figure depicts the beginning and end points of both tests, as well as the coordinates that each robot thinks it is on, with different colors indicating whether or not the robot thinks it is a part of the

phototaxis or anti-phototaxis to get out of the form. Snapshots of the starfish and U shapes disassembling themselves over time are also shown in the figures. Full-length videos documenting the starfish, spanner, and U shape experiments' respective self-disassembly processes are included in the appendices.

We used the measures of both positive and negative error to conduct our analysis. Error is expressed as a percentage, which may be computed by dividing the number of robots by the total number of robots, then multiplying the result by 100 to account for any robots that were left over after the configuration was finalized. There were 3 extra robots in the class 1 forms of triangles and 5-pointed stars. Predictability of this outcome suggests that more iterations should have been performed to prevent it. In the Spanner shape, the wrong placement of the light source, which should have been in the center of the shape but was instead placed along the boundary of the surface, prevented some robots from completing the anti-phototaxis movement because they were blocked by the robots that were already in the Spanner configuration.

We counted a total of 94 robots and determined that this number was too high for a U-shaped class 2 robot formation. However, it should be emphasized that all 94 robots were already engaged in the phototaxis movement and that none of them were within 10 units of the primary U-shaped structure. This study estimated a margin of error of 7.67%. Fifty to one hundred additional iterations would have caught this mistake before it was committed.

Because disassembly took significantly longer than expected, we had to run the simulation twice for the Shield shape in class 3 shape. During the first run, we expected the simulation to take no more than 30 minutes (or 1500 iterations), but it ran much longer than expected. As a result, many errant robots remained in the configuration after the simulation had ended.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

Class	Shape	Number of Iterations	Time Taken
I	Triangle	200	260.23 secs
	Spanner	200	380.86 secs
	5 point Star	200	479.93 secs
II	U Shape	400	654.92 secs
III	Shield 1	1500	1838.66 secs
	Shield 2	2000	2867.96 secs

The positive mistake rate was 13.469 percent on the approximately 165 robots we observed that had been partially reconfigured. It was clear that this arrangement required a lot more time and iterations, so we set the maximum number of iterations to 2000. Approximately 48 minutes into the simulation, 100 robots from two opposing motion sets, phototaxis and anti-phototaxis, collided while attempting to cross each other. This resulted in a deviation of 8.16 percent.

First, we need to know how many robots aren't included in the final configuration; next, we can divide that number by the total number of robots in the configuration; and last, we can multiply that result by 100 to obtain the error value as a percentage. The negative error achieved is zero across all three form classes and all the simulated shapes, suggesting that both phases of the simulation process went smoothly and the robots' coordinates were delivered accurately. Therefore, there are no missing robots in the final arrangement.

In Fig. 1 (b), we can see qualitatively that the coordinate system construction technique was successful in all three circumstances. All the forms have a rotation applied to them, but because it is global, it does not significantly affect the user's original shape or the coordinate system.

After around four to five seconds, the coordinate system stabilized for forms across all categories. We also found that edge robots, with the fewest neighbors, were taking more time than expected to locate themselves.

How long it takes to disassemble anything depends on how many different form classes it falls into. Class 1 forms often have shorter simulation times, and most configurations may be run in about 10 minutes. In comparison to other forms, disassembling a triangle takes the least amount of time at around 5 minutes. This is due to the very simple form.

When compared to the spanner form, the 5-point star takes the most time to dismantle. The time required to complete the task grows exponentially with the number of edges since the robots' range of motion is limited and multiple robots must go through the collision avoidance system to ensure there is no collision; this procedure must be repeated at each iteration.

Robots experiencing phototaxis are limited to one direction of motion in all three spatial dimensions when the class 2 form is a U configuration. Because of this, running the simulation will take much longer. It takes roughly 11 minutes to tear the U form apart. It should be noted that if additional iterations were allowed for complete disassembly of errant robots, the time required for disassembly would be significantly longer in this case.

Class 3 robots take a long time to disassemble because they undergo both phototaxis and anti-phototaxis, and because clusters of photo and anti-photo robots need to pass through each other. It took the robot cluster in the simulation around 48 minutes longer than expected to disassemble.

Note that the disassembly was not flawless and that a significant number of mistakes were made in Class 3. After the disassembly process begins, the small cluster of robots (phototaxis robots) moving in opposition to the robots belonging to a larger cluster (anti-phototaxis robots) forms a wall-like structure, preventing the movement of the rest of the robots from both clusters of robots to pass through, which accounts for a large portion of the time and error. To investigate this phenomenon, we built a separate shape consisting of a circle of stationary robots and a small cluster of 15 phototaxis motion robots. Shape robots that don't move in any other directions hide the shape's top and bottom, as well. Anti-phototaxis robots populate the remaining space in the configuration. For the reason that immobile robots are lined up in a circular column:

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

Class	Shape	Positive Error		Negative Error		Total Error	
		Our Result	Gauci Paper Result [2]	Our Result	Gauci Paper Result [2]	Our Result	Gauci Paper Result [2]
I	Triangle	+0.245%	-	0	-	+0.245%	-
	Spanner	+4.73%	+1.97%	0	-5.62%	+4.73%	7.59%
	5 point Star	+0.245%	+1.82%	0	-9.39%	+0.245%	11.21%
II	U Shape	+7.67%	+3.52%	0	-4.70%	+7.67%	8.22%
III	Shield 1	+13.469%	-	0	-	+13.469%	-
	Shield 2	+8.163%	-	0	-	+8.163%	-

This makes room inside the form for two channels that the robots may travel through. These slits need at least 5 or 6 robot widths to navigate. Multiple simulations showed that both channels were blocked most of the time, and that this behavior persisted even after lowering the number of tiny cluster robots to 10. Blocking of channels ended after we increased the minimum width of both channels to 7 and decreased the number of robots in the smallest cluster to 6. In order to avoid the construction of movement-restricting walls, it is essential to ensure that the combined length of the robots making up the tiny cluster does not surpass the breadth of the whole region across which it passes. Experimental results are not presented because they would reveal that realizing such shapes with the methods presented by us is extremely difficult due to this constraint in class 3, and that successful simulation of such shapes is outside the scope of this paper.

IV. Conclusion

Here, we describe and model a self-disassembly algorithm for a large-scale robot swarm. Our findings show that our collision avoidance algorithm outperforms that of Gauci Paper [2]. The simulation results we present here indicate that this type of self-disassembly algorithm has a good chance of being used for shape formation in modular robots and programmable materials due to its ability to achieve a broad class of shapes with high efficiency and accuracy.

V. Future work

In order to handle a category of shapes for which phototaxis and anti-phototaxis alone are insufficient for self-disassembly, we plan to further develop this algorithm by introducing a stochastic motion component in the near future. Including a LIDAR sensor on the robots is another way to improve the collision avoidance algorithm. Incorporating a LIDAR sensor into the robots would allow them to move at a higher speed, shortening the time it takes to disassemble the object.

Acknowledgment

Prof. Dr.G.Venugopal served as our guide and provided invaluable input on this project.

References

[1] M. H. Wagdy, H. A. Khalil and S. A. Maged, "Swarm Robotics Pattern Formation Algorithms," *2020 8th International Conference on Control, Mechatronics and Automation (ICCMA)*, 2020, pp. 12-17.

[2] Melvin Gauci, Radhika Nagpal and Michael Rubenstein " Programmable Self-Disassembly for Shape Formation in Large-Scale Robot Collectives".

[3] Y. Khaluf, E. Mathews and F. J. Rammig, "Self-Organized Cooperation in Swarm Robotics," *2011 14th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, 2011, pp. 217-226.

[4] M. Dorigo, "SWARM-BOT: an experiment in swarm robotics," *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005.*, 2005, pp. 192-200.

[5] R. Arnold, K. Carey, B. Abruzzo and C. Korpela, "What is A Robot Swarm: A Definition for Swarming Robotics," *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2019, pp. 0074-0081.

[6] A. S. Anoop and P. Kanakasabapathy, "Review on swarm robotics platforms," *2017 International Conference on Technological Advancements in Power and Energy (TAP Energy)*, 2017, pp. 1-6.

[7] A. Foroutannia, M. Shoryabi, A. A. Anaraki and A. Rowhanimanesh, "SIN: A Programmable Platform for Swarm Robotics," *2021 26th International Computer Conference, Computer Society of Iran (CSICC)*, 2021, pp. 1-5.

[8] MARCO DORIGO, GUY THERAULAZ, VITO TRIANNI "Swarm Robotics: Past, Present, and Future" PROCEEDINGS OF THE IEEE | Vol. 109, No. 7, July 2021.

[9] G. Beltrame, E. Merlo, J. Panerati and C. Pinciroli, "Engineering Safety in Swarm Robotics," *2018 IEEE/ACM 1st International Workshop on Robotics Software Engineering (RoSE)*, 2018, pp. 36-39.

[10] H. Wei, Y. Chen, M. Liu, Y. Cai and T. Wang, "Swarm Robots: From Self-assembly to Locomotion," in *The Computer Journal*, vol. 54, no. 9, pp. 1465-1474, Sept. 2011