Conveying Directional Information Through Regional Division in Swarms

Chris Cedron, Clayton Dembski, Floris van Rossum

I. ABSTRACT

Our project aims to explore the feasibility and usefulness of communicating directional information using LED rings in order to have a swarm of robots all visit a target location. To do this we set up a number of trials in ARGoS using the Swarmanoids footbot. Our idea is loosely based off of AdaBoosting, in this machine learning technique, a complex distribution of points is described by using simpler, less robust identifiers, then combining them to form a more complex and robust identifier. Rather than separate sets of data, we use multiple weak guide robots to form one strong force that points towards the goal. We can do this by using the footBots LED ring and then interpret that ring through the omnidirectional camera.

II. INTRODUCTION AND RELATED WORK

While there are multiple, already established, methods to get a swarm of robots to converge at or go towards a particular point, our team set out to develop a new one. Using basic communications through an LED ring around each robot, the swarm is capable of transmitting the location of the point of interest. (POI) This system begins when one robot finds the goal and then signals to the rest of the swarm the region that the POI is in relative to itself. Then as more robots reach the goal, the relative position of the POI begins to be more clearly defined and makes the task of finding it simpler for future robots. This method is similar to the way that AdaBoosting works in machine learning. Where in the boosting technique the data region is separated using a multitude of weak classifiers eventually forming a complex and robust classifier. We use relative position to split the field into POI is here and POI isn't here, once enough robots have managed to find the goal, the position of the POI is fairly well defined and robots can more easily find the goal. This kind of navigation does not require any kind of information transfer between individual robots and therefore is not subject to packet loss, data corruption or interference with wireless transmissions. Furthermore, because LEDs are relatively simple and robust pieces of technology, and cameras are fairly standard on swarm bots, this implementation should be lower maintenance than other forms of data transmission.

III. PROBLEM STATEMENT

The purpose of this project is to outline a model for the discovery of a point of interest, by a decentralized robotic swarm, within a finite space. This model acts as a search around a space, where every single entity searching for a particular object, needs to personally find the object, without being directly passed the location. This situation could occur if direct communication between two robots is not possible, or limited. Or if the exact location of the point of interest is not known to a swarm robot, just the general location. This model could also be used as a decentralized recruitment algorithm, where all robots locate a specific point of interest, and then approach that point.

IV. METHODOLOGY

As opposed to the foraging models presented in the previous section, this research project will not use a pheromone trail to inspire foraging behavior. Rather, this project presents a method of trail forming behavior in the following manner.

A. ARGoS and Buzz

The foundation for this experiment was provided by the ARGoS simulation software. ARGoS is a multi-physics robot simulator which can simulate large-scale swarms of robots of any kind efficiently [1]. ARGoS allows for the implementation and coding of robotic swarms through a programming language called Buzz. Buzz allows for the programming of individual robots as well as the entire swarm in an efficient manner. All experiments and trials for this project were run in the ARGoS simulator, and coded using Buzz programming language. Version control was provided using a GitHub repository.

B. *Foot-bot*

The robot entity used in ARGoS for this paper was the Foot-bot. The simulated robot originates from a swarm robotics project that was active between 2006 and 2010 [2]. The Foot-bot boasts 24 light sensors, 24 short range proximity sensors, a 12 LED ring, all around the robots perimeter, and an omni-directional camera for colored blob detection.

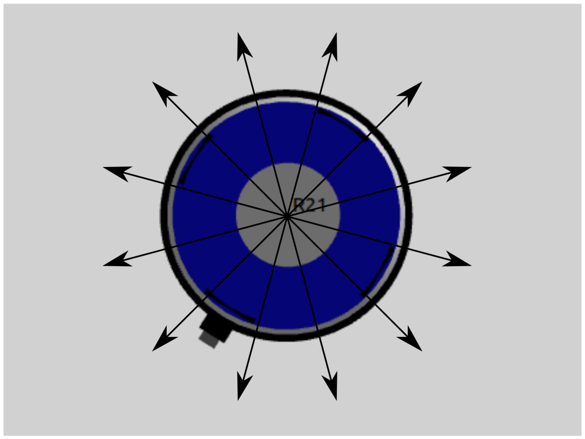

Fig. 1: 12 rays representing the LED ring

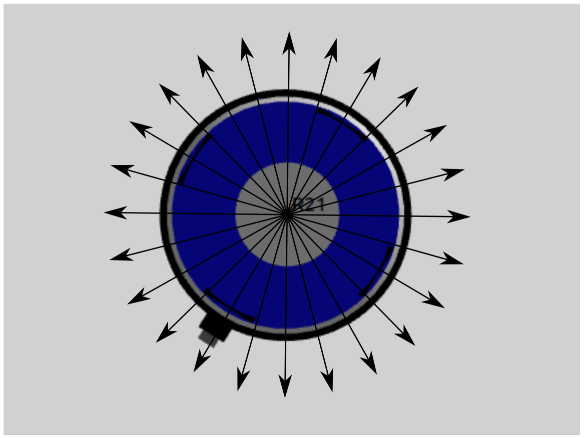

Fig. 2: 24 rays representing the proximity and light sensors

C. *LED Rings*

The LED rings around each robot was utilized to display information for other unit for these experiments. With the 12 LEDs, the presented model allows each robot to outwardly express information, such as a specific direction. More specifically the LED ring allows the robot to be split into two distinct halves, and light each in a different color.

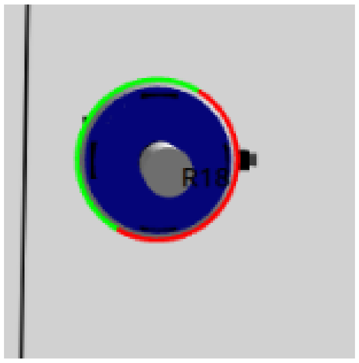

Fig. 3: Top view of the two-color LED ring

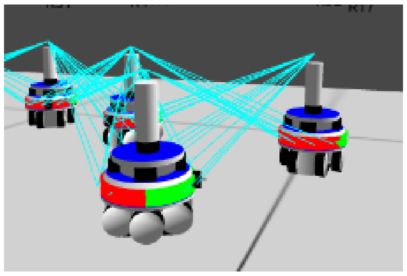

Fig. 4: Side view of the two-color LED ring

D. *Color Detection*

The robots were able to use the Omni-directional Blob Detection Cameras to see colors surrounding them within the cameras range. The range is given by 0.289m*tan(), where the is user specified aperture. Each color in line of sight within this range is logged in the robot as a blob, which contains an R.G.B value between 0 - 255, a distance, and an angle. This allows a robot to register and interpret the LED rings of the other robots.

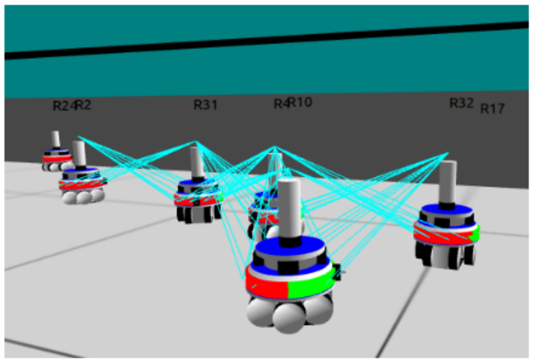

Fig. 5: Blob detection rays

E. *Using the LED Ring*

The novelty of this project comes when an individual robot locates, or is otherwise informed of a Point Of

Interest (POI) within the given map. For our convergence tests, the robot was supplied with this point, and for the foraging tests, the robot found this point when within a threshold of a supplied light source. Once the

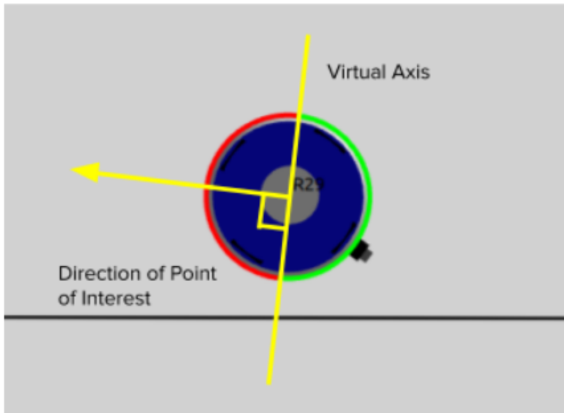

Fig. 6: The virtual axis of a robot

position is noted, the robot then uses its LED ring to create two semicircles, one of color red and one of color green. Where these semicircles meet is a virtual axis (6: yellow dashed line), dividing the two colors. This line is always perpendicular to the virtual vector(6: Yellow arrow) created by the stored position and the robot. In essence, the robot divides the space into two areas, where the point is and where the point isnt. For simplicity's sake in ARGoS, we set up the robots to store the POI once found, and use the global positioning system to keep track of where the robot is relative to the POI. In a real system, after the POI was found and the virtual axis was set, the axis could be kept track of using a well tuned gyro, so global tracking data would not be needed. From this two color split, other robots in range are attracted to green and repulsed from red via virtual forces, subject to the robots innate Inertia. The net force imposed onto a robot within range of a lighted ring is a unit vector defined as:
$\sum F_x = \alpha \sum \frac{cos(\theta_{red})}{D_{green}} + \beta \sum \frac{cos(\theta_{red})}{D_{red}} + Inertia$
$\sum F_y = \alpha \sum \frac{sin(\theta_{green})}{D_{green}} + \beta \sum \frac{sin(\theta_{red})}{D_{red}}$
$\sum \theta = tan^{-1}(\frac{\sum F_x}{\sum F_y})$
Where θ, F_x, and F_y are relative to the robot and $alpha$, $beta$ are the attraction and repulsion force respectively. When a robot has a virtual force imposed onto it, it will rotate to a threshold of the specified relative angle, then, once there, drive in the direction of the force.

F. Convergence Testing

To validate the idea that a LED ring with two colors combined with the omnidirectional camera could cause the appropriate and expected result in a robotic swarm, we first did convergence testing. The aim of this was to see if the robots would converge to a specific point in space only by following the virtual forces imposed on it. The question we wanted to answer was, given a Point of Interest (POI), if the Foot-bots pointed their LED ring at that point and used the colored LEDs from other robots to guide them, would the robots arrive at the POI. Although every robot knew the location they were travelling to internally to correctly display their light ring, the robot did not use this location in its navigation. For this experiment, robots were placed randomly in a finite simulation in ARGoS, and given adequate time to converge to the selected point.

G. Food Finding Experiments

The second experiment was an attempt to solve the problem described in the Problem Statement with the presented model. For this experiment, a finite space was used, and a single light was placed inside, marking the point of interest. Foot-bots were placed in random locations in this field based on a gaussian distribution. For this experiment, the robots had a number of different states as shown in Figure XXX below. The

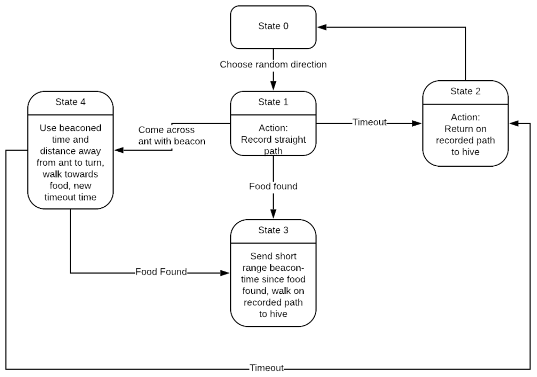
State 0
Choose random direction
State 1
Action: Record straight path
Timeout
State 2
Action: Return on recorded path to hive
State 4
Use beaconed time and distance away from ant to turn, walk towards food, new timeout time
Come across ant with beacon
Food found
State 3
Send short range beacon-time since food found, walk on recorded path to hive
Food Found
Timeout

Fig. 7: State Diagram for Food Finding Trials

robots start in the random walk state, where the robots perform some type of random walk. In the case of this research project, the robots random walk took the robot in a straight line until obstacle avoidance caused the robots to change direction. If a robot found the Point of Interest (POI) or saw a colored blob, the robots behavior would deviate from the random walk. If the robot sees a colored blob, it will calculate the virtual forces according to all the colors around the robot (Using the LED Ring). If the robot detects the Point of Interest, the robot stores its location, and turns on its

LED display, as shown in LED Rings. Once the robot has detected the location of the POI, it will continue random walking forever. The experiment is considered finished once all of the robots have found the Point of Interest.

The food finding experiments were also split with two distinct purposes. The first set of experimental trials were aimed at applying the model to the problem statement. Following these experimental trials, we started optimizing the parameters listed below.

- Continuous Turning: Robot turning that allows for turning in continuous curves along the path.
- Omni-directional Aperture: The distance that the omni-directional camera can see
- Linear Robot Speed: The linear speed of the robot
- Turning Speed: The turning speed of the robot
- Inertia Factor: A parameter changing the effect of repulsion and attraction forces on the direction of the robot
- Repulsion Factor: A scaling factor on the force generated bu the red LEDs
- Attraction Factor: A scaling factor on the force generated by green LEDs
- Number of Robots: The number of robots in the experiment

V. RESULTS

A. Convergence Results

To validate the model, the initial experiment was to test for the convergence of the Foot-bots if they were all given a POI and could all be influenced by each other's virtual forces. The following data was collected for the convergence trials. Multiple trials were run with different seeds and different locations around the map. The data collected was the distance of each robot to the Point of Interest as well as the time-step when the data was collected. All data was then inserted into one model as shown below, 8.

B. Control Results

To allow for the verification of our food finding model, we started off by collecting control data for the food finding experiments. Our control experiments were made up of 25 randomly seeded trials. For each trial, the robot was only able to use our method of random walking to find the light– they were not influenced by other virtual forces, and would walk in a straight line until encountering an object. From these control trials 12, we can see that the baseline random walk is slow: after 8000 time steps, half of the 40 robots have converged. It is also evident that this model is logarithmic in its growth: as time goes on, there is a smaller and smaller probability a robot will reach the POI.

By averaging our data, we can get a model of our growth:
Indicator Robots = 9.94000325 log(x) -67.38994838
which yields the following graph 13:

C. Food Finding Results

1) Initial Trials: The first set of trials for food finding was, once again, proof of concept based. We wanted to test both the forces individually so that we could analyze their effects on the system. We did 2 sets of trials, one with aperture 88, and one with aperture 70, which corresponds to a vision range of 0.289m*tan(88) = 8.276m, 0.289*tan(70) = 0.794m respectively. Both sets were made up of 25 runs of red only, and 25 runs of green only trials. We then looked over the results and decided to execute a set of 25 runs with both red and green at an aperture of 70. Each simulation was run for 30 minutes, which corresponded to 8000 timesteps in ARGoS, using the time parameters we specified. For all of these simulations we ran with an attraction force and repulsion force of 1/-1 respectively. The robots were using discrete turning, were not taking into account obstacle avoidance when following blobs and had no inertia.

2) Optimized Trials: The optimization trials were performed on one random seed (123), and one parameter was changed at a time. In total, there were 17 different trials, however, some trials were discarded due to the model not producing the desired behavior, or some trials are a duplicate with a different seed to verify results.

D. Videos

- Initial convergence test: https://www.youtube.com/watch?v=p05i5vZeQlA
- Initial convergence test: https://www.youtube.com/watch?v=EoyFn7X1E44
- Final convergence test: https://www.youtube.com/watch?v=0r2cxE1sSqo
- Optimization trial ID 11: https://www.youtube.com/watch?v=fcvAmq7f2es
- Optimization trial ID 7: https://www.youtube.com/watch?v=086vrdFAgiI

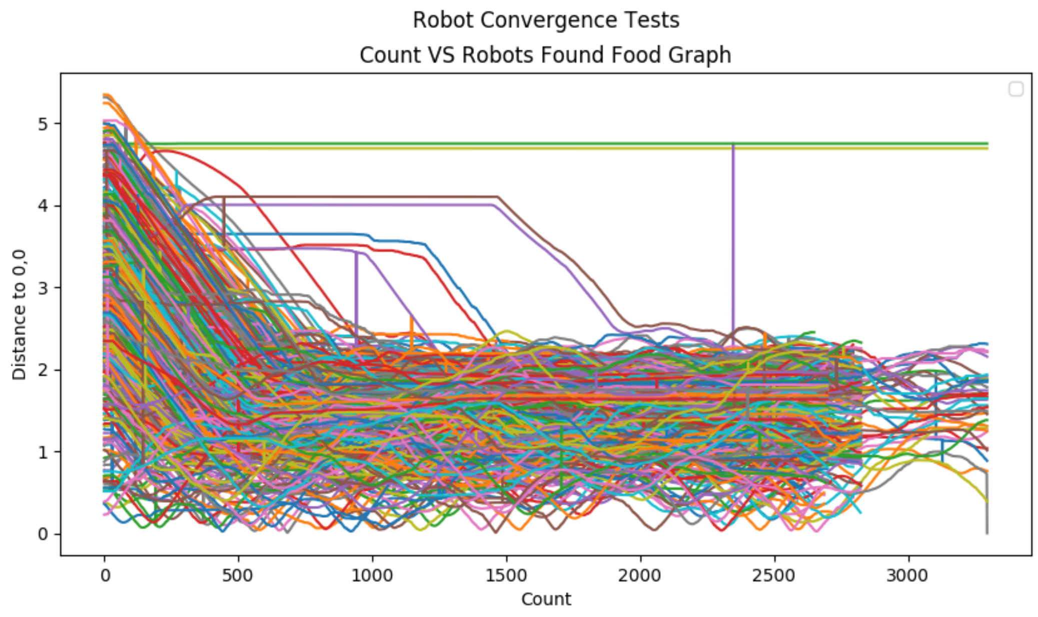

Fig. 8: Distance of robots to POI in convergence testing

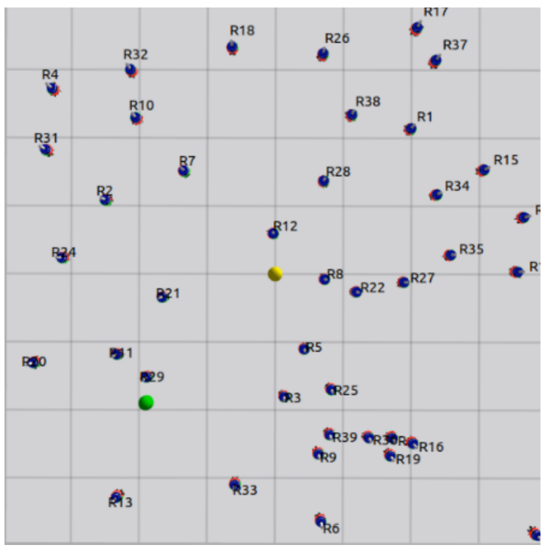

Fig. 9: The initial state of convergence testing

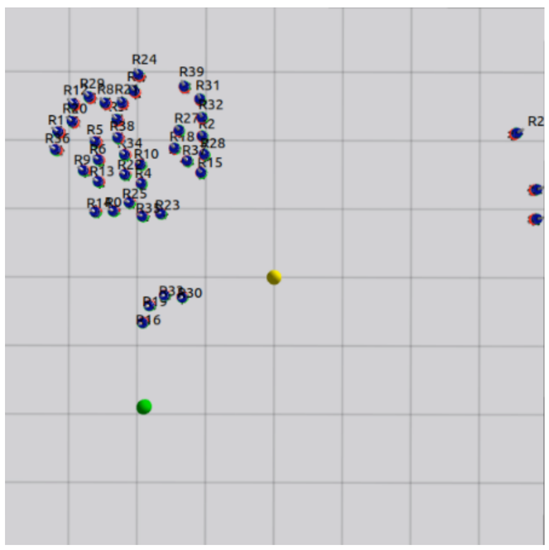

Fig. 10: The final state of convergence testing

VI. Discussion

A. Convergence

During the convergence tests we see that the robots all approach the POI quickly. Because all of the robots have their virtual axis set up to impose a virtual force on other robots towards the center from the start, any robot can average out all of the blobs and will get an accurate virtual force vector pointing towards the POI. Upon closer inspection of the data it is clear that the robots do not end up touching the POI, this is because when a robot is close to the POI, the repulsive forces from the red lights next to and across from it push the robot away. The distance the robot ends up from the POI, therefore, is the steady state from the repulsive forces and the attractive forces, a low energy state, where the forces are in equilibrium. If we were to reduce the effect the red blobs have at close distances, we would see the robots pack closer together.

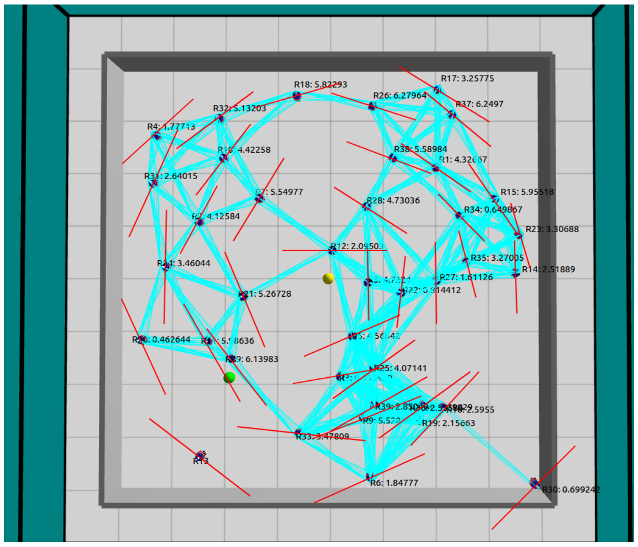

Fig. 11: Drawing of Virtual Divisions Across Each Robot relative to (0,0)

B. Control Behavior

For the control, the time it takes for all robots to find the food is logarithmic. This is explained as the probability that each robot finds the food is independent of one another. This means, to start, there is a high likelihood that at least one of the 40 will find food, but this quickly dwindles with time. The probability that an individual robot finds food in a timestep is P_f. The number of robots that have found the food at a time step is given by N. The probability any robot will find food in a given time step is therefore given by:
P=1-(1-P_f)$^{40-N}$, where
N = 9.94000325 log(x) -67.38994838
and x is the current time step.

C. Presented Model vs. Control

1) Green Behavior: Looking at the 88 aperture trials for green, we see that, for the most part, the Foot-Bots find the POI extremely quickly. This, however, is subject, first off, to that fact that the range of these cameras for these trials are not realistic. In a real system, it would be a stretch for the robots to quite see so far. Second off, and more importantly, we see a few stragglers. If not all the robots find the food source within 2000 steps of the first robot finding the food, we can see that there are large swathes of time before any more robots find the food. Looking at the 70 aperture trials, a much more reasonable detection distance, we notice the same effect from the 88 trials exacerbated: more frequently, no more robots find the light source for long swathes of time. Watching the simulation at aperture 70 trials, we see why this behavior happens. When the robots find the light source, while they remain close to the light source, they attract all other robots to themselves and the light source. However, as these robots diverge from the light source, they continue to pull all other robots along with them. This leaves them all in a cluster, moving with the indicator robot. This means that these robots, which have not found the food, will not venture out to find the food, but will follow the indicator robot as it random walks, until this indicator robot happens to random walk back into the range of the POI. This leads to a boom or bust system, where it is highly effective if all robots can find the source quickly, and highly ineffective if they take significant time to reach the indicator bots.

2) Red Behavior: The behavior of the indicator robots is almost the antithesis of the green only indicator robots. With a high aperture, and therefore a long sight range, we can see that barely any robots actually notice the food source, after only one notices the food. This is easily explained. When only one robot that has found the POI, its range is great enough that it repulses all other robots away from the POI. The behavior in the second trial set, with an aperture of 70, is significantly better. As the range is less, this allows for indicator robots to diverge far enough from the POI that they no longer repulse all robots away from the POI. This allows, through random walking, for other robots to find the POI. When enough robots have found the POI, then repulsion alone can become useful. If the indicator robots are aligned in such a way that their virtual forces point at, or near one another, the repulsions away from the POI will cancel out, and the leftover force will point at the POI.

D. Variations and Parameters

1) Aperture: The optimized model included many different parameters and variations. Varying the aperture parameter resulted in faster POI detection time, however, was more unrealistic. As the aperture is raised, the robot can see further robots. Although distance scaling was implemented, where blobs further away give less force, a high aperture allows a robot to gain too much information. If the robot has vision that extends across the finite space, then there is likely a better model for food finding than the one presented in this paper. Therefore, once a decently low aperture of 70 was found, it was used for the remainder of the optimization trials, although a higher aperture would likely work better.

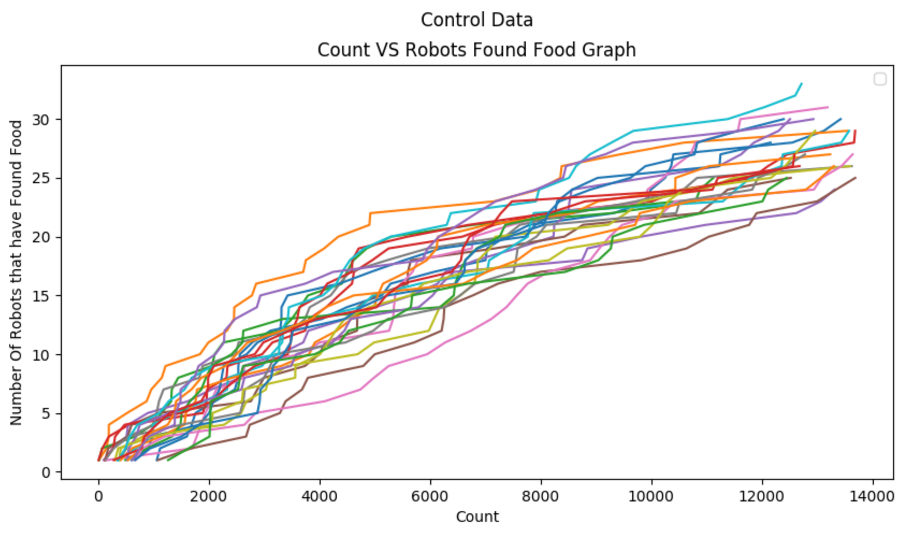

Fig. 12: Control Trials

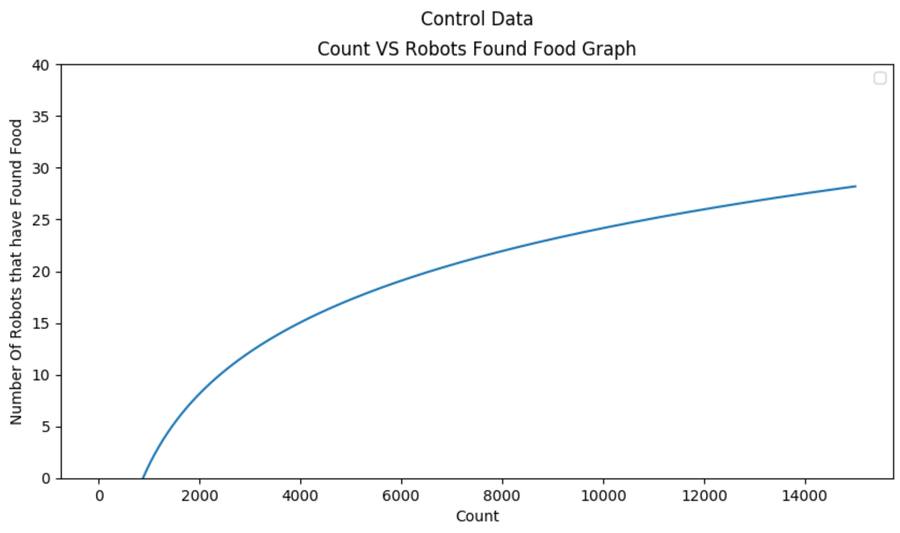

Fig. 13: Food Finding Data Trend

2) Continuous and Discrete Turning: Upon the implementation of this model in Buzz, we found two possible ways of turning. Continuous turning uses the gotop() function to set a linear and angular speed of the robot to approach a virtual point. This virtual point is continually adjusted as the robot moves. This allows the robot to move in continuous curves as opposed to discrete straight lines. Discrete turning cannot turn and travel linearly at the same time. Once the robot detects that the virtual force is outside a threshold as compared to its yaw angle, it will rotate, otherwise it will travel forward. Discrete turning is more simple to implement, and therefore was used in the initial trials before it was found that continuous turning proved more effective. This can be attributed to the fact that continuous turning allowed for the robot to continuously approach its destination as opposed to turning while its environment was still changing.

3) Obstacle Avoidance: Obstacle avoidance was always enabled for all trials when the robots did not see any blobs, or had found the POI. However, for some trials, if the robot saw a blob and used the virtual forces to determine its direction, it would not use obstacle avoidance. The initial trials, and some of the optimized trials did not have obstacle avoidance enabled while being influenced by the blobs. This results in a faster POI detection time, but is not as realistic.

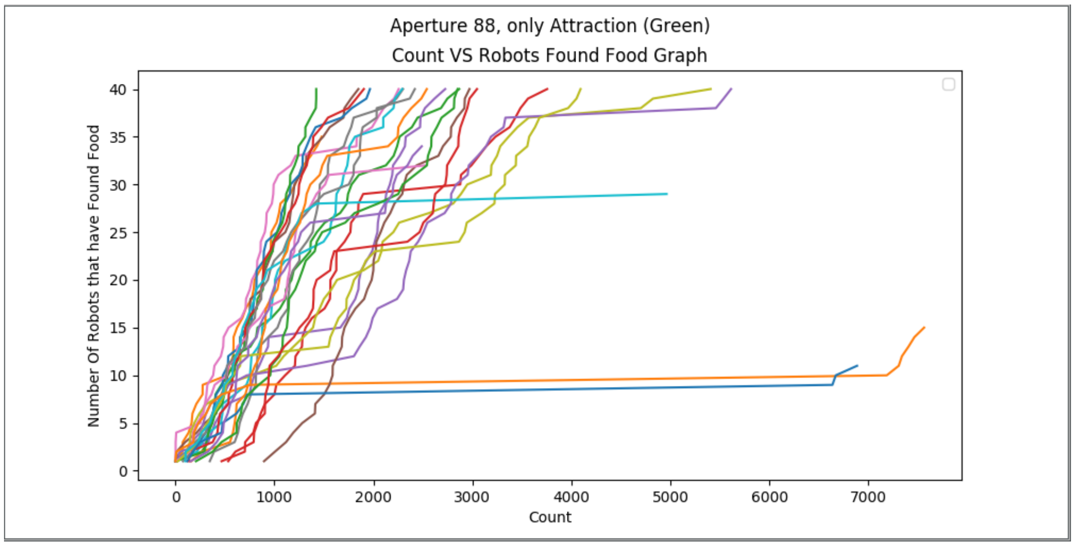

Fig. 14: Aperture 88 on only attraction

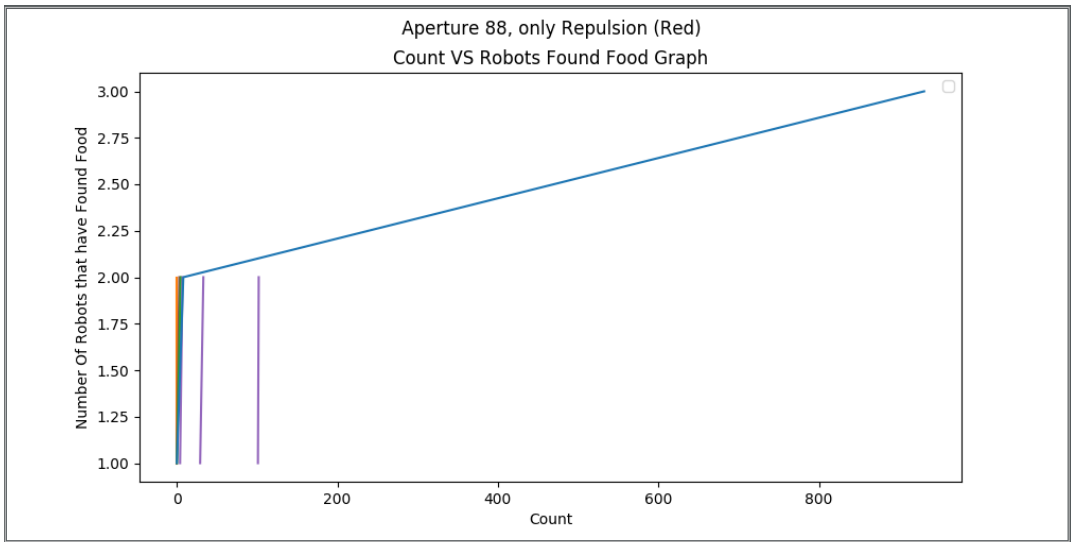

Fig. 15: Aperture 88 on only repulsion

4) Number of Robots: Based on the results of Trial 11, 13 and 14, decreasing the number of robots in this simulation reduces the effectiveness of this model. Trial 13 and 14, for 30 and 20 robots, respectively, both need more time for the last few robots to find the POI. This is due to the fact that less robots means there is less directional information available on the field for the last few robots to use to establish the approximate location of the POI. Additionally, based on observation, a robot obtains a better sense of direction of the POI with more robots providing directional information.

5) Inertia Factor: The inertia factor describes how much the blob forces affect the steering of the robot. The higher the inertia factor, the more a robot maintains its current direction. A higher inertia factor will cause the robot to ignore the virtual force vector established by the LEDs of other robots. The inertia factor creates a force in the current yaw orientation of the robot, effectively, bringing the total virtual force vector in line with the yaw. An inertial factor of around 0.01 was found to produce behavior according to the model. However, this parameter could use more tuning to produce better results and allow the robots to discover the POI faster. An inertia factor of 0.5 was used in Trial 4, however, this was found to reduce the directional steering of the robots in response to the lights too much. As seen in Trial 4, the inertial factor was found to be much less effective at bringing the robots to the POI

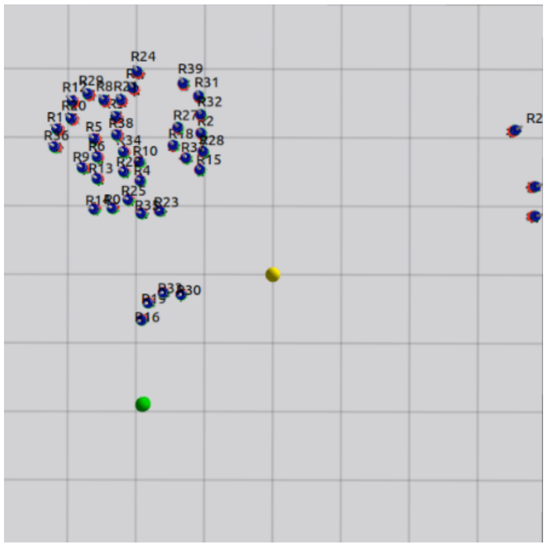
Fig. 16: Aperture 70 on only attraction

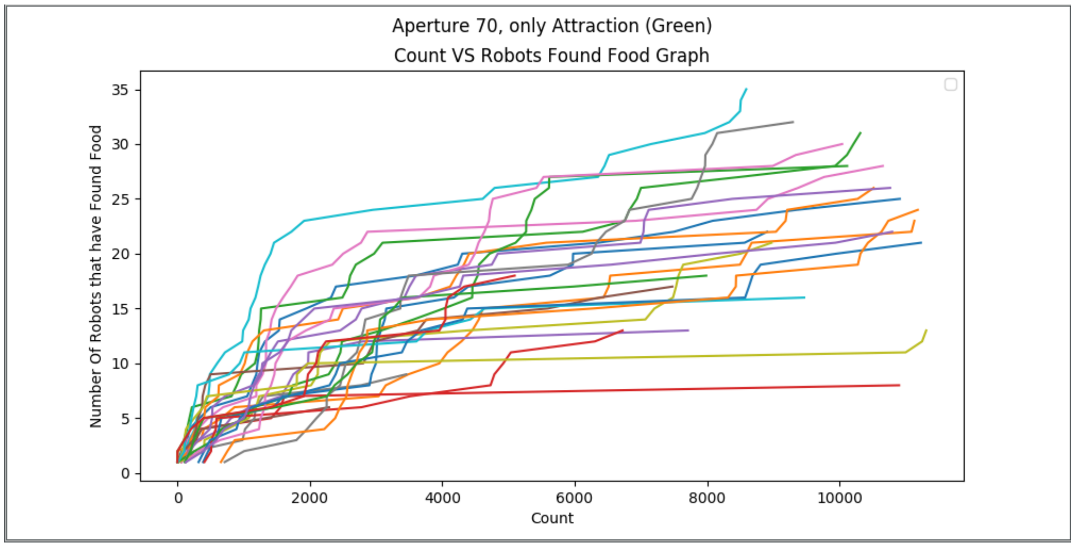

Fig. 17: Aperture 70 on only repulsion

than an inertial factor of 0.01. This is also expected because the inertial factor brings in the behavior from the control, which is much more inefficient. Therefore, increasing the inertial factor too much results in a longer time for robots to find the POI.

6) Robot Speed: Increasing and decreasing the robot speed had positive impacts on the effectiveness of the model at finding the POI. Specifically, if the speed of a particular state is tuned to the speed of another state. For example, a slower turning speed for robots that were blob following proved more effective. This is because this allowed the robots to slowly converge to the position of the POI and created a more robust model. With a too high turning speed the robots directly approached a green light instead of the direction in which it was pointing. This caused obstacle avoidance to trigger until the two robots separated.

E. Other Behavior

With the correct parameters this model can create more specific and interesting behavior. As seen in the image below, two behaviors were created, herding and flocking. Herding is a behavior where a robot that has not yet found the POI, pushes the robot with the LEDs displaying to the POI. This can also occur with more than one robot, as they push the robot with the LED display to the POI. Flocking is when a robot with the LEDs on, acquires a flock of robots that have not found

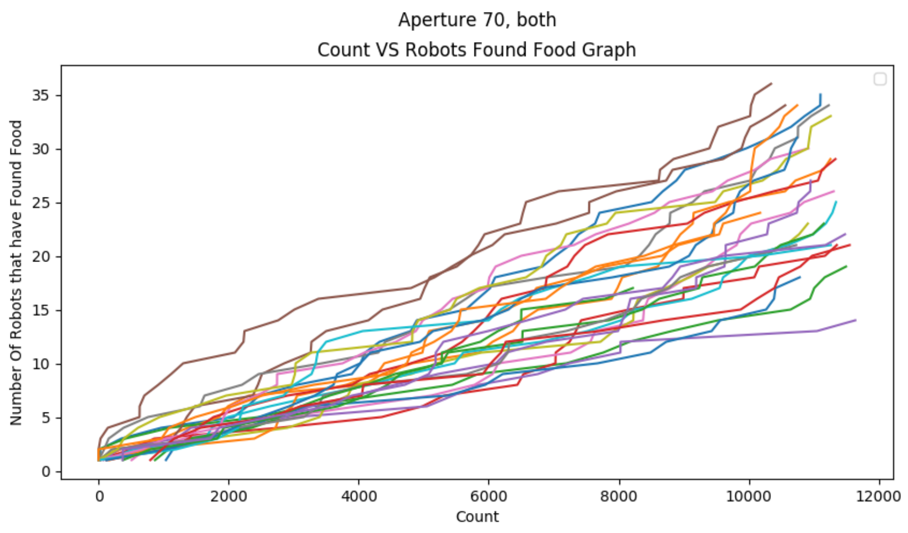

Fig. 18: Aperture 70 with both repulsion and attraction

ID	Turning type	Obstacle Avoidance	Aperture	Inertia Factor	Turning speed	Linear Speed	Red scaling factor	Green scaling factor	Number of robots
3	Discrete	N	70	0.01	50	-	-1	1	40
4	Discrete	N	70	0.5	50	-	-1	1	40
5	Continuous	N	70	0.01	4	4	-1	1	40
6	Continuous	N	70	0.01	4	4	-1	1	40
7	Continuous	N	70	0.01	4	4	-0.2	1	40
8	Continuous	Y	70	0.01	1	4	-0.2	1	40
9	Continuous	Y	70	0.01	1	4	-1	1	40
10	Continuous	Y	70	0.01	1	4	-1	1	40
11	Continuous	Y	70	0.01	1	4	-1	1	40
12	Continuous	Y	70	0.01	1	4	-1	1	40
13	Continuous	Y	70	0.01	1	4	-1	1	30
14	Continuous	Y	70	0.01	1	4	-1	1	20

Fig. 19: Table of Parameters

the POI. These robots that have not found the POI are too held back to push the robot with the LEDs on, but they follow the robot actively. The follower robots appear very dynamic and in a flock due to the fact that they are trying to see the green light, but avoid the red light. Both of these behaviors were created by tuning the parameters of the presented model. Specifically, object avoidance for robots that are being steered by blobs is disabled and those robots have faster linear and angular speed as compared to robots that have found the POI. This results in robots following the LED robot and leads to herding and flocking. The herding behavior provided some very interesting results as the robot that had not found the POI was pushing the robot that had found the POI to the POI directly. This means that the robot that had the LED displayed was effectively unable to move, and completely under the control of the pushing robot. This behavior can be described as

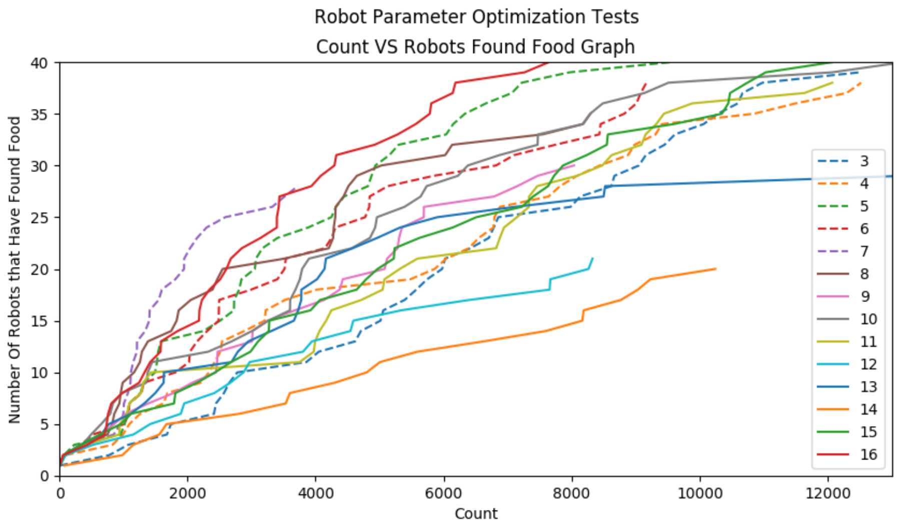

Fig. 20: The results of the optimization trials

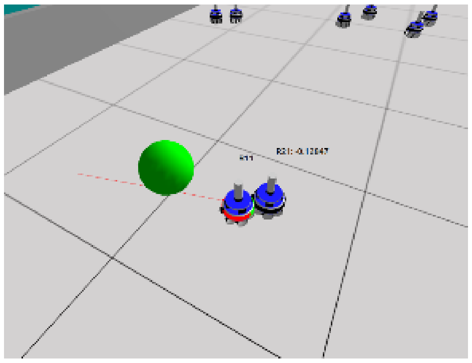

Fig. 21: Herding

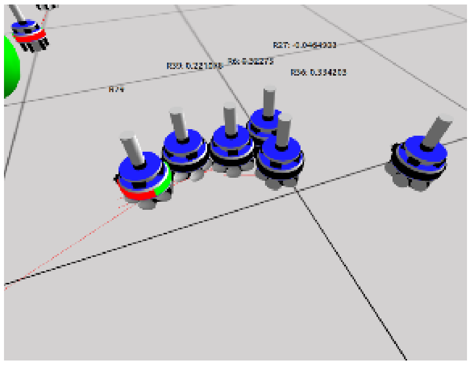

Fig. 22: Flocking

a guide dog, where the robot that has not found the POI is effectively blind, but the robot with the LEDs on brings the robot to a specific location. This could be useful if a robot was unable to move due to a motor malfunction or other error, it could use its LED ring to steer towards a particular location using other robots.

F. Weaknesses and Errors

The model provided in this research paper has a number of weaknesses that need to be addressed. First of all, while this model is a decentralized solution, it still uses the GPS location of the POI in every robot. This is unrealistic in a real-world environment, where a definite location is not known. This paper considered this aspect out of the scope of the project. There are a number of solutions that can replace this to provide a more realistic viewpoint such as using a gyroscope to save the direction, or using odometry, or the LEDs of other robots to reinforce each other. Additionally, none of the trials performed in the experiments had any sensor or mechanical noise. This would have undoubtedly increased our error and reduced the effectiveness of this model.

Many small errors and bugs were encountered in the creation of this model. For example, due to the behavior of the omnidirectional camera on the foot-bot, whenever a robot was parallel to the virtual axis of another robot, as shown in **??**, the parallel robot would experience a force to the direction of the POI. This is seen in the image below by the red ray on the ground below R13. There are possible fixes for small bugs such as these, but this would result in a more complex and slower model. Another bug in the implementation of ARGoS allows for a foot-bot to see LEDs through a robot. This bug has serious implications on our model and only occurs when two robots are close. However, this bug will result in a robot behaving in the opposite

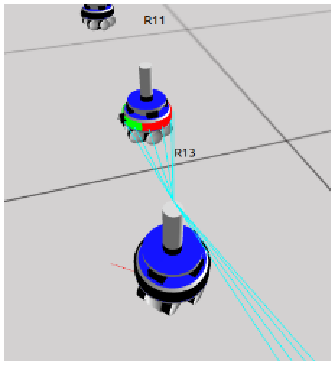

Fig. 23: R13 parallel to the virtual axis of R11

way. The robot moves away when it is supposed to be close, and moves closer when it is supposed to move away.

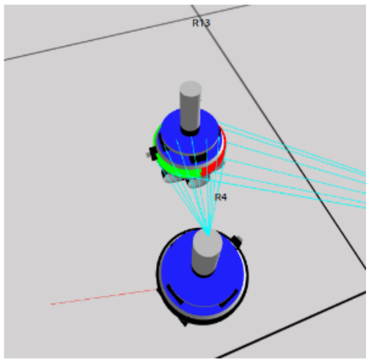

Fig. 24: R4 looking through R13

G. Future Ideas

This experiment is far from comprehensive, there are a handful of variations we would have liked to have tested. For example, using regions of interest rather than points. This may lead to some interesting behaviour as the robots wouldnt all be pointing at the exact same spot. Adding noise to our model would also have been interesting as it would allow us to test the robustness of this system. Increasing the number of colors we display on the LED ring may have also provided interesting results because it would have allowed us to provide more information as to where the POI is; in that same vein, if we were to use a running average of blobs and take into account prior timesteps we believe we may have seen more stable behaviour. Finally, adding variants of random walk may have provided us with better performance depending on the walking algorithm

VII. Conclusion

In conclusion, the directional information provided by regional division in individual robots in a swarm allows for the convergence of robots to a POI. Footbots were able to navigate to a light source in the ARGoS simulation software based on the information provided by the LED rings of other robots. Through the careful manipulation of environment variables and other model parameters the efficiency of the model was increased to provide results much better than the control. This model has the possibility for creating new swarm behavior as shown in the provided behavior in flocking and herding. Overall this project met expectation and has the capability to be expanded upon in the future.

VIII. Acknowledgements

We would like to acknowledge and thank WPI for providing us the opportunity of this class, Jayam Patel and especially Professor Pinciroli for being incredibly responsive throughout the whole process, and for going above and beyond by giving us the resources we needed to succeed even before we asked about them.

References

[1] ARGoS: Large-scale robot simulations, https://www.argos-sim.info/. April 29, 2019

[2] Swarmanoid, http://www.swarmanoid.org/project_description.php.html. April 29, 2019

[3] C. Pinciroli and G. Beltrame, Buzz: A Programming Language for Robot Swarms, IEEE Software, vol. 33, no. 4, pp. 97100, Jul. 2016. https://ieeexplore.ieee.org/document/7498536

[4] N. Hoff, A. Sagoff, R. Wood, and R. Nagpal, Two Foraging Algorithms for Robot Swarms Using Only Local Communication. .https://ieeexplore.ieee.org/document/5723314

[5] Hrolenok, Brian, et al. Collaborative Foraging Using Beacons. ACM Digital Library, International Foundation for Autonomous Agents and Multiagent Systems, 10 May 2010, dl.acm.org/citation.cfm?id=1838193.