Segregation without Computation

Peter Mitrano* , Jordan Burkland*, Michael Giancola*, Carlo Pinciroli*
*Worcester Polytechnic Institute, Worcester, Massachusetts

***Abstract*—In this paper, we demonstrate that robots which directly map a single ternary sensor readings to wheel speeds, perform no explicit communication are capable of n-class segregation. We evolve controllers with a genetic algorithm and perform a grid search over all parameters to understand the full parameter space. We then analyze the best controller, and prove that certain sub-behaviors of segregation are guaranteed. More broadly, this work demonstrates how to search for and study emergent behaviors in swarms of simple robots.**

***Index Terms*—swarm robotics, robot aggregation, robot segregation**

I. Introduction and Related Work

Many prior methods have demonstrated the ability to aggregate robots and objects in a distributed way [1], and [2] [3] have shown the ability to perform such tasks with limited computation and sensing capabilities. Segregation is the sorting of robots of different classes into clusters or groups, and is a natural extension of aggregation. Segregation can be seen as a precursor to object sorting, task allocation, or self-assembly. For example, swarms may need to split into arbitrary groups to diffuse and search different areas, or segregate by skill or capability in order to form useful heterogeneous teams. In this work we achieve segregation into an arbitrary number of groups with no messaging between robots, no complex sensors, no leaders or beacons, and no environmental cues.

In nature, we can find a few examples of aggregation and segregation. Segregation occurs in cells during embryogenesis [4], and segregation of objects can be found in ants who sort their brood [5]. These examples motivate the study of simple controllers for segregation, but our main motivation comes from the broader question of whether a complex behavior like n-class segregation can be achieved with a simple robot and controller.

We make no claims that the controllers studied here are the most practical engineering solutions, but rather that they are an example of how complex behavior can emerge from simple rules. The advantages of simple robots is that they are inexpensive, easier to implement. It is also easy to understand and construct proofs about the behavior of these simple robots and their controllers.

We also provide analysis of a specific emergent behaviors that allows the segregation to work. This analysis, while specific to the segregation behavior and controllers we found, is an example of how to make provable claims about emergent behavior of simple controllers. This is important because having guarantees on swarm behavior and well understand limitations allows one to make an informed decision about whether to deploy the controller in untested environments. If swarm robotics are to actually be used in disaster relief, as is so often proposed, it's important to know the conditions under which certain behavior is guaranteed. In these scenarios, using the simplistic robot and controllers may actually be advantageous.

A. Aggregation and Segregation

Aggregation is defined as having all robots in the swarm collect at a particular location in a distributed manner. Many swarm aggregation controllers require robots to compute bearing and distance to other robots, to sense gradients in the environment, or to otherwise communicate information. However, implementing these communication systems is difficult in practice, so methods that do not require communication or complex sensing are desirable. In [2], Guaci et al propose a new class of controllers that require no computation.

Gasparri et al. instead develop an aggregation controller which enables robots to respond to guidance commands – i.e., a human using hand gestures to indicate which direction the swarm should move [6]. They show that swarms running their controller are able to follow guidance commands and stay aggregated without colliding. A simpler controller is used by Bahgeci and Sahin [7], where robots are equipped with IR distance sensors surrounding the robot, 4 microphones, and an omni-directional speaker. The controller is a linear combination of the IR sensor values and the intensity values of the microphones. The authors show that a genetic algorithm is able to find weights for the controller such that robots aggregate. In [8], Ando et al. present an aggregation controller with very limited sensing and no memory, but allows the robots to perform computations to determine their next position. They prove that their controller is correct in theory, then run the controller in simulation.

Finally, Gauci et al. consider a binary sensor that maps directly to wheel velocities [2]. The authors demonstrate robust aggregation despite these limitations, and they also prove formally that aggregation is guaranteed and find theoretical bounds on aggregation time in simple situations. Gauci et al. also perform object aggregation, as opposed to robot aggregation, with the same restrictions on sensing and control [3].

Segregation of robots has received little attention in swarm robotics, however many researchers have focused on sorting of objects [9] [10] [11] [12]. Santos et al. shares our objective, however they assume that each class has the same number of robots and that each robot knows the position of all other robots [5]. In contrast, we adhere to the `if/elseif/else` controller architecture with a simple ternary sensor. In [13],

Groß et al. also explore segregation robots based on local interactions inspired by gravity, however they require a centralized broadcast or a consensus algorithm to agree on the source or direction of gravity.

B. Oblivious Robots

Oblivious robots is a These robots were originally proposed by Gauci et al., and have been extended to other tasks and been modified in many ways by other researchers.

Kernbach et al. propose an aggregation controller inspired by bees clustering in an optimal temperature location [14]. In their method, robots are only able to distinguish between collisions between another robot or a wall, can only sense the intensity of a light source when they have collided, and do not communicate information. Although the robots have limited sensing and communication capabilities, the authors show that robots can aggregate to an optimal light intensity location, and that the time to converge to the optimal location improves with increasing number of robots in the environment.

Oblivious robots have also been shown to aggregate around a specific object, circle in a ring around an object, and forage for obstacles [15]. In this work, the show one can construct simple cost functions to guide the evolution of controllers to perform new, yet interesting tasks. However, they also report that in attempting to evolve a controller to rendezvous the robots around an object, they accidentally and consistently evolved a controller where the robots circled around the target object. They achieved rendezvous by initializing the controller at generation zero to the controller found in [2] for simple aggregation.

II. Methodology

A. Problem Formulation

We consider a collection of differential-drive robots all executing the same controller in a homogeneous, two-dimensional, obstacle-free environment. We also introduce the concept of kinness. Two robots are kin if they are of the same class (denoted by color in our experiments), otherwise they are non-kin. The robots are equipped with a single forward-facing line=of-sight sensor which can distinguish between the presence of a kin robot, a non-kin robot, and the absence of a robot. This sensor is assumed to have infinite range (we consider non-infinite range in Section V-E). We assign $I = 0$ to the state when no robot is seen, $I = 1$ to the detection of a kin robot, and $I = 2$ to the detection of a non-kin robot. We allow for any number of classes, but the robots need not distinguish between different non-kin classes. They need only to detect if a robot is of the same class or not. The objective is to segregate robots into clusters, ideally such that all the robots of the same class are packed into one cluster with no non-kin robots.

The controllers we use in our experiments are of the same form as in [2]. The controller maps each of the three sensor values to a set of wheel speeds. Pseudo-code for the controller is shown in Algorithm 1. This controller has six parameters:

Algorithm 1 Controller Design

```
if I = 0 then
    set wheel speeds to v_{l_0}, v_{r_0}
else if I = 1 then
    set wheel speeds to v_{l_1}, v_{r_1}
else
    set wheel speeds to v_{l_2}, v_{r_2}
end if
```

$$[\, v_{l_0}, v_{r_0}, v_{l_1}, v_{r_1}, v_{l_2}, v_{r_2} \,]$$

Keeping with the form used in [2], we let these parameters range from -1 to 1. In simulation, we then scale this parameter to range from $-20\,\mathrm{cm\,s^{-1}}$ to $20\,\mathrm{cm\,s^{-1}}$.

B. Simulation Environment

We use the ARGoS simulation environment to search for controller parameters and evaluate them, which has the advantage of allowing us to run trials much faster than on real robots [16]. In our simulations, we use a range-and-bearing sensor to implement the theoretical line-of-sight sensor. In our analysis, we describe the sensor as infinitely thin, but in practice it must have some small finite angle. We explore the performance effect of various beam angles in Section V-D. In our simulations, we consider robots to be connected in a cluster if the gap between them is $5\,\mathrm{cm}$ or less. This detail is required for the cluster metrics. We now describe two approaches for finding these parameters, genetic algorithms and grid search.

C. Evolving Segregation

[7] [15] [17] show that evolutionary strategies are effective for finding emergent behavior. In order to quickly search for parameters to achieve segregation, we use a simple genetic algorithm to evolve controller parameters. The genetic algorithm we used is unmodified from the example MPGA code[1] provided with the ARGoS simulator. The mutation strategy is to mutate each of these 6 parameters with some probability p (0.05 in our experiments). If a parameter is selected to be mutated, a uniformly random number from -1 to 1 is picked for the new value. Selection is performed by picking the two lowest cost individuals, and the next population is formed by crossing the parameters of these two individuals. The original two parents are always kept in the population so that the current lowest cost individual is never removed until a lower cost genome is discovered.

D. Cost Functions

As with any optimization algorithm, it is important to have a cost function that accurately assigns cost to behaviors. The cost metric that we used (Equation (1)) is based on the cluster metric $c^{(t)}_{\text{gauci}}$ used by [2], which is the proportion of the number

[1] http://www.argos-sim.info/examples.php

of robots in the largest cluster to the total number of robots at some time step t.

$$c_{\text{total}} = \frac{1}{n} \sum^{n} \sum_{t=0}^{T-1} -t c_{\text{gauci}}^{(t)} \tag{1}$$

Let n be the number of classes of robots, and T be the total number of time steps in the trial being evaluated. We apply this cost function for each class of robots and sum the cluster-metric cost for each class to get the total cost. The negative sign is present because the cluster metric is 0 when no robots are connected and 1 when all robots in a class are connected, so the negative sign assigns more connections a lower cost. We used this cost function in grid search and all following experiments.

In our experiments, we found that this cost function correctly assigns cost in most scenarios, but it is not ideal because it considers a straight line of robots to be one cluster. For some tasks such as task allocation, we might care about how tightly the swarm is packed in its clusters.

E. Grid Search

In order to exhaustively search the space of possible controllers, we conducted a grid search of the 6-dimensional parameter space. Due to limited computational resources, we were only able to search with a resolution of 7 values per parameter, which means in total we evaluated $7^6 = 117649$ parameter sets. For each parameter set, we ran 1 trial in 36 different initial configurations, with 180 seconds for each simulation. These initial configurations consisted of uniformly random placement, clusters, and lines of robots distributed throughout the environment. We chose to include some structure configurations (clusters and lines) because we discovered that they varied significantly in performance from uniform random configurations, and by explicitly evaluating on structure configurations we can speak more confidently about the ability of the controller to succeed in any configuration. Otherwise, we would need to have far more trials with uniformly random robot placement to make the same claims. We were able to parallelize across the 36 configurations and run at 75x real time, so it took approximately 4 days to evaluate all the controllers.

Because the search space is six-dimensional, we chose to visualize it by plotting every pair of parameters against each other. For example, we consider how the cost changes as v_{l_0} and v_{r_0} change. We note that these graphs, shown in Appendix A, show very similar patterns to the same plots presented by [2]. As an example of reading these plots, we can tell from the plot of parameters 2 and 3 ($I = 1$) that there were no good controllers where the left and right wheel speeds were equal and negative (dark squares in the upper left), and that the best controllers had slightly unequal values close to 1 (lightest squares in the bottom left). These plots also visualize how there are some sharp discontinuities where performance changes dramatically.

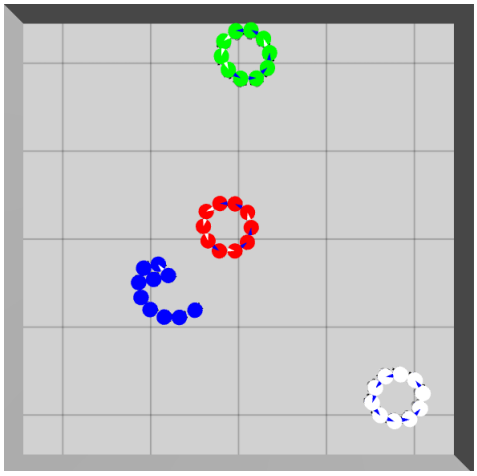


Figure 1. the emergent behavior is rings of kin. These rings expand and drift over time.

III. The Emergent Behavior

After running 10 generations with 10 genomes per generation, the lowest cost controller found was [0.5, -0.5, -0.2, 0.6, 0.6, 0.2]. Similarly, after running the grid search, the best controller was [1, -2/3, 1/3, 1, 1, 0]. The controller found via grid search outperforms the one found via a genetic algorithm, however they both exhibit the same emergent behavior. For both of these controllers, the resulting behavior is to turn away from kin but turn the opposite way when they see nothing or non-kin. This behavior causes the kin robots to zig-zag in a line towards their kin, and when multiple robots execute this behavior the kin robots form rings. An example of this can be seen in Figure 1. Rarely, the robots form shrimp-like shapes which can also be seen in Figure 1.

It's important to note that although we originally hoped that segregated clusters would be tightly packed, in our comprehensive grid search we never found any controller that was capable of this. Another important emergent behavior is that the rings expand over time. When we placed many kin robots together in a large area, they would quickly form a ring and this ring would slowly expand in radius, seemingly infinitely. Explaining or proving this formally is left for future work.

The dynamics of the robots is far more complex than merely forming rings. A complete description of the observed behavior is beyond the scope of this paper, but we encourage the reader to view the supplementary videos at https://goo.gl/z8UAuB. For example, we notice that the rings, once formed, expand over time. If a ring is disturbed by non-kin robots that trying to pass through, the ring is disrupted and eventually reforms.

IV. Controller Analysis

We found from grid search that the parameters [1, -0.6667, 0.3333, 1.0, 1.0, 0.0] best achieve segregation as defined by our cluster metric cost function. The actual wheel speeds, which are scaled by 0.2 to convert to $\mathrm{m\,s^{-1}}$ are therefore [0.2, -0.1333, 0.06667, 0.2, 0.2, 0.0]. Now that we have found

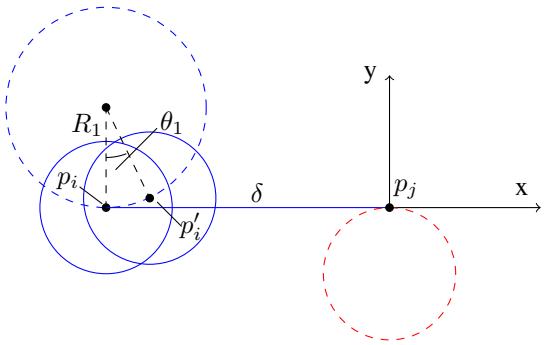


Figure 2. The configuration of a robot aggregating to another kin. R_1 is the radius of the robots path around the ICC.

parameters that achieve what we consider segregation, we define the conditions under segregation is guaranteed. We first define the conditions under which a kin robot will aggregate with a static kin or ring of kin. We then show that a robot also aggregates to a non-kin. However, we show that aggregation to kin is always faster than aggregation to non-kin. These three proofs partially explain why segregation occurs.

In all of these proofs we will consider the controller found via grid search since it is the best controller. We also assume the controllers are in sync, which means that each motion is executed for the same Δt starting at the same point in time, and that all robots have a fixed positive interwheel distance W and radius r.

A. Aggregation with Kin

Imagine an isolated kin robot, i which has sensed another kin or a static ring of kin (kin entity), j. This scneario is depicted in Figure 2. We can define the conditions under which robot i is guaranteed to be closer to the point where the imaginary sensor beam meets the kin entity. This point is chosen as the origin. When robot i starts at p_i with sensor reading $I = 1$, it then executes $v_{l_1} = 0.06667$ and $v_{r_1} = 0.2$ and arcs with radius R_1 counterclockwise and ends up at position p'_i. The condition we need to satisfy is shown in Equation (2).

$$\|p'_i - p_j\| < \|p_i - p_j\| \tag{2}$$

We can expand Equation (2) and derive a simple condition which describes for which values of r, r_j, W, Δt aggregation is guaranteed. The full derivation can be found in Theorem 1, and the result is shown in Equation (3). The variable r_j denotes the radius of the kin entity, whereas r still refers to the radius of the single robot i.

$$-\sqrt{r^2 + 2rr_j}\sin\left(\frac{2\Delta t}{15W}\right) - W\cos\left(\frac{2\Delta t}{15W}\right) + W < 0 \tag{3}$$

One can substitute in the parameters of their robot into this equation, and if the condition holds true then the behavior described here is guaranteed.

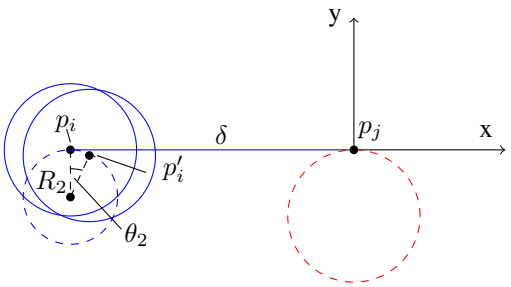


Figure 3. The configuration of a robot aggregating to a non-kin. R_2 is the radius of the robots path around the ICC.

Table I
AGGREGATION IS GUARANTEED FOR MANY DIFFERENTIAL DRIVE SWARM ROBOTS. WE USED $\Delta t = 0.1$ HERE, AND ASSUME $r_j = r$

Robot	r (m)	W (m)	Eq (3)	Eq (4)	Guaranteed
foot-bot[2]	0.085	0.14	-0.013	-0.020	Yes
Khephera IV[3]	0.07	0.1054	-0.014	-0.021	Yes
E-Puck 2[4]	0.035	0.053	-0.013	-0.020	Yes
Kilobot[5]	0.0165	0.030[6]	-0.009	-0.014	Yes

B. Aggregation with Non-Kin

This scenario and proof follows the same pattern as with kin, but the resulting condition is slightly different. We can again expand Equation (2), this time with different coordinates for p'_i. The full derivation can be found in Theorem 2, and the result is shown in Equation (4).

$$-\sqrt{r^2 + 2rr_j}\sin\left(\frac{2\Delta t}{10W}\right) - \frac{W}{2}\cos\left(\frac{2\Delta t}{10W}\right) + \frac{W}{2} < 0 \tag{4}$$

C. Segregation

So far all of the behaviors we have discussed are actually just aggregation behaviors. Here we partially explain why segregation occurs by showing that kin robots aggregate faster than the non-kin robots. To show that non-kin aggregate more slowly than kin, we show that the displacement during their kin step (the arc left with speeds $0.06667, 0.2$) is greater than the displacement during their non-kin step (with speeds $0.2, 0.0$). This proof can be found with Theorem 3. Note that we only consider the magnitude of the step, not the direction, so in theory the longer kin step could aggregate less than the shorter non-kin step, but this partially justifies why kin aggregate faster then non-kin.

D. Application to Real Swarm Robots

The above conditions do not hold true for all scenarios (such as those with large Δt), but we show that they do hold true for a number of popular robots in Table I.

[2]ARGoS file footbot_entity.cpp
[3]Khephera IV User Manual, page 66.
[4]http://projects.gctronic.com/epuck2/e-puck2-flyer.pdf
[5]https://www.k-team.com/mobile-robotics-products/kilobot/specifications
[6]only approximate

Ultimately, we can use these geometric proof to make useful assertions about the behavior of an individual in our swarm and of the swarm as a whole. While we have not shown that segregation is guaranteed in all cases, this may serve as a building block for more general claims.

V. Experimental Results

A. Comparing Cost Functions

We ran grid search using both of our cost functions.

B. Scalability Study

In this experiment we investigate how segregation behavior scales with the number of classes and number of robots in the environment. We varied the number of classes from 1 to 25 and ran 100 trials of robots uniformly randomly distributed.

a) Fixed number of robots per class: We first considered having 10 robots for each class. However, this means that in the trial with 25 classes there are 25 times the total number of robots than in the 1-class trial. This means that occlusion of robots is more likely and so the cost increases with more classes. The results of this are plotted in Figure 5.

b) Fixed total number of robots: Another scenario is to consider a fixed number of robots and split them into more and more classes. We choose 100 robots because we are still able to get 4 robots per class at 25 classes. As you can see in Figure 4, the cost no longer increases as the number of classes increases. However, the cost for just a few classes is much higher. This is expected, because when there are many robots of a single class, our controller forms very large sparse rings where robots are too far apart to be considered clustered.

Ultimately, we can say that our controller scales well to many classes of robots, but not very well to many robots.

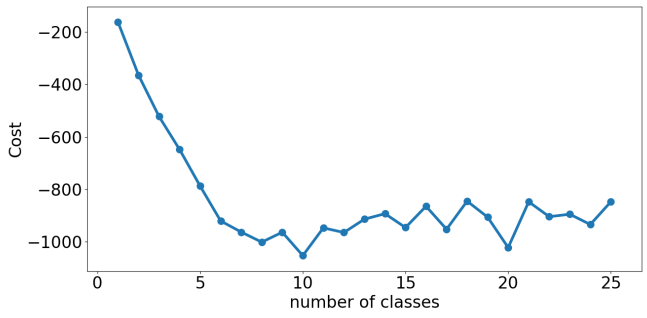


Figure 4. The average cost with 100 robots divided into N classes. More classes are lower cost partially because kin robots stay close enough to be considered clusters.

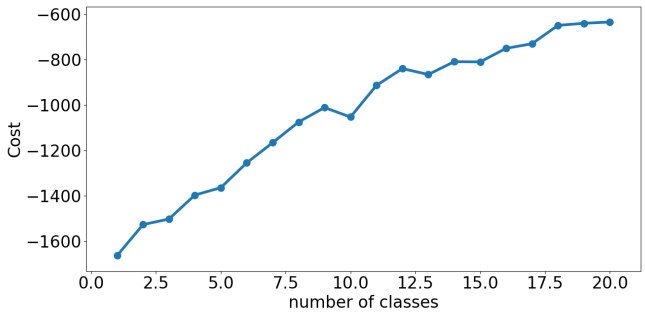


Figure 5. The average cost with N classes, 10 robots per class. More classes are higher cost because other robots obstruct your view making sensing kin less likely.

C. The Effect of Implementation Details of the sensor

We found in our experiments that the implementation details of the line-of-sight sensor have a significant effect on the behavior of the controller.

Initially, our method for determining sensor state from our simulated range-and-bearing sensors was to consider all the robots within some small angle in front of the robot and pick the closest one. This is very similar to what would be provided by a real-world camera that uses colored skirts on each robot and picks the largest blob as the robot to be detected. This sensor implementation works well and was used in all our genetic algorithm and grid search experiments. However, we found later that if the robots instead always prefer to react to kin over non-kin, you can form larger rings more quickly and robustly. For example, if there are two robots within the field of view of your sensor and the non-kin robot is closer, you will ignore it and execute the $I = 1$ state which will drive you towards the farther away kin robot. Exploring exactly which of the various implementation details have what effect on cost is left for future work.

D. The Effect of the Beam Angle

On a real robot, there must be some finite beam angle to the theoretically line-of-sight sensor. We ran 100 trials in simulation with uniformly random initial distributions of 40 robots with various half beam angles. Figure 6 shows the results, as well as a diagram showing how we define half beam angle. The best half beam angle we tested was $15°$, and angles smaller or larger became progressively worse. We found that at lower beam angles, it was possible for a robot to become stuck in groups of two or three where the robots spent all their time looking at each other and not peeking around them to find kin. At larger angles, we suspect the behavior fails because larger beam angles cause the rings to enlarge faster, which in turn causes the rings to be so large that they are not considered a cluster anymore and so the cost rises.

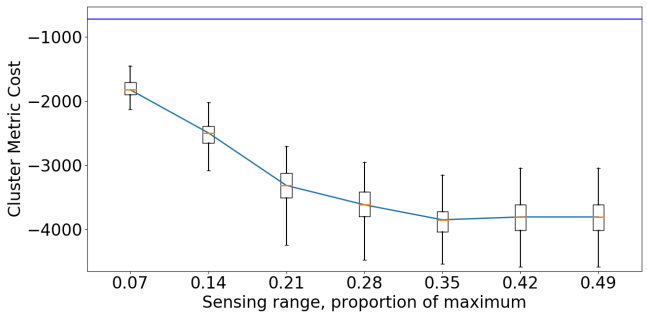


Figure 7. Segregation is robust to very finite sensor beam ranges. The blue bar indicates the worst (highest) attainable cost in which all robots are isolated from any kin.

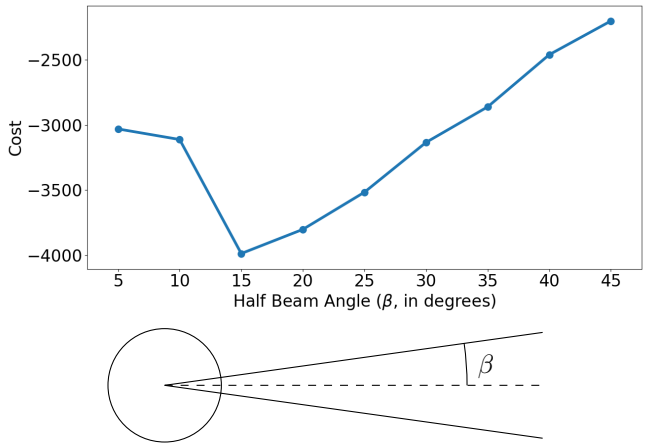


Figure 6. A 15° degree half beam angle is best for segregation. Lower cost is better.

E. The Effect of Beam Length

We also consider what happens if our theoretically infinite sensor now has finite range. We use 15° half beam angle and the same experimental setup as with the beam angle experiments. Analogously to [2], we consider the maximum range of the sensor as the diagonal length of the square in which the robot are initially distributed. In all our experiments, this square was $5\,\mathrm{m}$ on each side, so we consider a range of $7.07\,\mathrm{m}$ to be effectively unlimited. We report the costs for beam ranges as a fraction of this maximum range. As shown in Figure 7, a beam range of 35% of the theoretical maximum performs just as well as an infinite sensor. Below this, the performance degrades. However, even a beam range of 7% of unlimited is more effective than zero range at segregation.

VI. Conclusion

In this paper, we demonstrate that oblivious robots are capable of n-class segregation. We use a simple controller design consisting of a 6-tuple. This controller is invariant to the number of classes, so any given controller can work for any number of classes. To quickly find a quality controller, we evolved one using a basic genetic algorithm, but we also performed a grid search to make claims about the full parameter space. We investigated the effect of sensor implementation details and the number of robots and classes on performance. We find that robust segregation is possible, although not guaranteed. Instead, we prove that kin robots will aggregate, non-kin robots will also aggregate, and kin robots aggregate faster than non-kin robots.

References

[1] N E Shlyakhov, I V Vatamaniuk, and A L Ronzhin. Survey of Methods and Algorithms of Robot Swarm Aggregation. *Journal of Physics: Conference Series*, 803:012146, January 2017.

[2] Melvin Gauci, Jianing Chen, Wei Li, Tony J. Dodd, and Roderich Gro{\\ss}. Self-organized Aggregation Without Computation. *Int. J. Rob. Res.*, 33(8):1145–1161, July 2014.

[3] Melvin Gauci, Jianing Chen, Wei Li, Tony J. Dodd, and Roderich Gro{\\ss}. Clustering Objects with Robots That Do Not Compute. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '14, pages 421–428, Richland, SC, 2014. International Foundation for Autonomous Agents and Multiagent Systems.

[4] Eduard Batlle and David G. Wilkinson. Molecular Mechanisms of Cell Segregation and Boundary Formation in Development and Tumorigenesis. *Cold Spring Harbor Perspectives in Biology*, 4(1):a008227, January 2012.

[5] V. G. Santos, L. C. A. Pimenta, and L. Chaimowicz. Segregation of multiple heterogeneous units in a robotic swarm. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1112–1117, May 2014.

[6] A. Gasparri, A. Priolo, and G. Ulivi. A swarm aggregation algorithm for multi-robot systems based on local interaction. In *2012 IEEE International Conference on Control Applications*, pages 1497–1502, October 2012.

[7] E. Bahgeci and E. Sahin. Evolving aggregation behaviors for swarm robotic systems: a systematic case study. In *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005.*, pages 333–340, June 2005.

[8] H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita. Distributed memoryless point convergence algorithm for mobile robots with limited visibility. *IEEE Transactions on Robotics and Automation*, 15(5):818–828, October 1999.

[9] A. Vardy. Accelerated Patch Sorting by a Robotic Swarm. In *2012 Ninth Conference on Computer and Robot Vision*, pages 314–321, May 2012.

[10] Owen Holland and Steve Hoddell. Collective sorting and segregation in robots with minimal sensing, 1998.

[11] Tao Wang and Hong Zhang. Collective Sorting with Multiple Robots. pages 716–720. IEEE, 2004.

[12] Owen Holland and Chris Melhuish. Stigmergy, Self-organization, and Sorting in Collective Robotics. *Artif. Life*, 5(2):173–202, April 1999.

[13] R. Gro{\\ss}, S. Magnenat, and F. Mondada. Segregation in swarms of mobile robots based on the Brazil nut effect. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4349–4356, October 2009.

[14] Serge Kernbach, Ronald Thenius, Olga Kernbach, and Thomas Schmickl. Re-embodiment of Honeybee Aggregation Behavior in an Artificial Micro-Robotic System. *Adaptive Behavior*, 17(3):237–259, June 2009.

[15] Matthew Johnson and Daniel Brown. Evolving and Controlling Perimeter, Rendezvous, and Foraging Behaviors in a Computation-Free Robot Swarm. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (Formerly BIONETICS)*, BICT'15, pages 311–314, ICST, Brussels, Belgium, Belgium, 2016. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[16] Carlo Pinciroli, Vito Trianni, Rehan O'Grady, Giovanni Pini, Arne Brutschy, Manuele Brambilla, Nithin Mathews, Eliseo Ferrante, Gianni Di Caro, Frederick Ducatelle, Mauro Birattari, Luca Maria Gambardella, and Marco Dorigo. ARGoS: a Modular, Parallel, Multi-Engine Simulator for Multi-Robot Systems. *Swarm Intelligence*, 6(4):271–295, 2012.

[17] Marco Dorigo, Vito Trianni, Erol Şahin, Roderich Gro{\\ss}, Thomas H. Labella, Gianluca Baldassarre, Stefano Nolfi, Jean-Louis Deneubourg, Francesco Mondada, Dario Floreano, and Luca M. Gambardella. Evolving Self-Organizing Behaviors for a Swarm-Bot. *Autonomous Robots*, 17(2-3):223–245, September 2004.

APPENDIX

Theorem 1. *An isolated robot will aggregate to a kin entity.*

Proof. Use the differential drive forward kinematics model to determine the coordinates of p_i'.

$$\theta_1 = \Delta t\omega = \Delta t\frac{v_{r_1} - v_{l_1}}{W} = \Delta t\frac{0.2 - 0.06667}{W} = \frac{2\Delta t}{15W} \tag{5}$$

$$R_1 = \frac{W}{2}\left(\frac{v_{r_1} + v_{l_1}}{v_{r_1} - v_{l_1}}\right) = \frac{W}{2}\left(\frac{0.2 + 0.06667}{0.2 - 0.06667}\right) = W \tag{6}$$

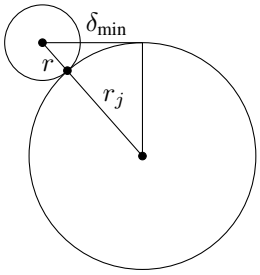
We can define the coordinates of p_i, p_j, and p_i'.

$$p_i = \begin{bmatrix} -\delta \\ 0 \end{bmatrix} \tag{7}$$

$$p_j = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{8}$$

$$p_i' = \begin{bmatrix} -\delta + R_1\sin(\theta_1) \\ R_1 - R_1\cos(\theta_1) \end{bmatrix} \tag{9}$$

We also define the minimum possible δ, which is acheived when the robots and tangent.



$$\delta_{\text{min}} = \sqrt{(r + r_j)^2 - (r_j)^2} = \sqrt{r^2 + 2rr_j + r_j^2 - r_j^2} = \sqrt{r^2 + 2rr_j} \tag{10}$$

Show that $\|p_i' - p_j\| < \|p_i - p_j\|$ (Equation (2)) for all $\delta > \sqrt{r^2 + 2rr_j}$.

$$\begin{aligned}
\|p_i' - p_j\| &< \|p_i - p_j\| \\
\sqrt{(p_{ix}' - p_{jx})^2 + (p_{iy}' - p_{jy})^2} &< \sqrt{(p_{ix} - p_{jx})^2 + (p_{iy} - p_{jy})^2} \\
\sqrt{(-\delta + R_1\sin(\theta_1) - 0)^2 + (R_1 - R_1\cos(\theta_1 - 0)^2} &< \sqrt{(-\delta - 0)^2 + (0 - 0)^2} \\
\sqrt{\delta^2 - 2\delta R_1\sin(\theta_1) + R_1^2\sin(\theta_1)^2 + R_1^2 - 2R_1^2\cos(\theta_1) + R_1^2\cos(\theta_1)^2} &< \delta
\end{aligned}$$

Apply $\sin^2 + \cos^2 = 1$

$$\begin{aligned}
\sqrt{\delta^2 - 2\delta R_1\sin(\theta_1) + 2R_1^2 - 2R_1^2\cos(\theta_1)} &< \delta \\
\delta^2 - 2\delta R_1\sin(\theta_1) + 2R_1^2 - 2R_1^2\cos(\theta_1) &< \delta^2 \\
-\cancel{2}\delta\cancel{R_1}\sin(\theta_1) - \cancel{2}R_1^{\cancel{2}}\cos(\theta_1) + \cancel{2}R_1^{\cancel{2}} &< 0 \\
-\delta\sin(\theta_1) - R_1\cos(\theta_1) + R_1 &< 0
\end{aligned}$$

We assume $\theta_1 \in (0, \pi/2)$, so if the above condition holds for the smallest $\delta = \delta_{\text{min}}$, it holds for greater δ.

$$-\sqrt{r^2 + 2rr_j}\sin\left(\frac{2\Delta t}{15W}\right) - W\cos\left(\frac{2\Delta t}{15W}\right) + W < 0$$

□

Theorem 2. *An isolated robot will aggregate to a non kin entity.*

Proof.

$$\begin{aligned}\theta_2 &= \Delta t\omega = \Delta t\frac{v_{r_2} - v_{l_2}}{W} = \Delta t\frac{0.2 - 0.0}{W} = \frac{2\Delta t}{10W} \\ R_2 &= \frac{W}{2}\left(\frac{v_{r_2} + v_{l_2}}{v_{r_2} - v_{l_2}}\right) = \frac{W}{2}\left(\frac{0.2 + 0.0}{0.2 - 0.0}\right) = \frac{W}{2}\end{aligned} \tag{11}$$

We can define the coordinates of the new p_i'.

$$p_i' = \begin{bmatrix} -\delta + R_2\sin(\theta_2) \\ -R_2 + R_2\cos(\theta_2) \end{bmatrix} \tag{12}$$

Show that $\|p_i' - p_j\| < \|p_i - p_j\|$ for all $\delta > \sqrt{r^2 + 2rr_j}$. This follows the same steps as the Theorem 1.

$$\begin{aligned} \|p_i' - p_j\| &< \|p_i - p_j\| \\ \sqrt{(-\delta + R_2\sin(\theta_2) - 0)^2 + (-R_2 + R_2\cos(\theta_2 - 0)^2} &< \sqrt{(-\delta - 0)^2 + (0 - 0)^2} \\ \sqrt{\delta^2 - 2\delta R_2\sin(\theta_2) + R_2^2\sin(\theta_2)^2 + R_2^2 - 2R_2^2\cos(\theta_2) + R_2^2\cos(\theta_2)^2} &< \delta \\ \sqrt{\delta^2 - 2\delta R_2\sin(\theta_2) + 2R_2^2 - 2R_2^2\cos(\theta_2)} &< \delta \\ -2\delta R_2\sin(\theta_2) - 2R_2^2\cos(\theta_2) + 2R_2^2 &< 0 \\ -\delta\sin(\theta_2) - R_2\cos(\theta_2) + R_2 &< 0 \end{aligned}$$

We assume $\theta_2 \in (0, \pi/2)$, so if the above condition holds for the smallest $\delta = \delta_{\text{min}}$, it holds for greater δ.

$$-\sqrt{r^2 + 2rr_j}\sin\left(\frac{2\Delta t}{10W}\right) - \frac{W}{2}\cos\left(\frac{2\Delta t}{10W}\right) + \frac{W}{2} < 0$$

□

Theorem 3. *The distance moved in the kin step is always greater than the distance moved in the non-kin step.*

Proof. We compare the arc lengths derived from the wheels speeds.

$$\begin{aligned} \theta_1 R_1 &> \theta_2 R_2 \\ \frac{2\cancel{\Delta t}}{15\cancel{W}}\cancel{W} &> \frac{\cancel{2}\cancel{\Delta t}}{10\cancel{W}}\frac{\cancel{W}}{\cancel{2}} \\ \frac{2}{15} &> \frac{1}{10} \end{aligned}$$

□

A. Grid Search Images

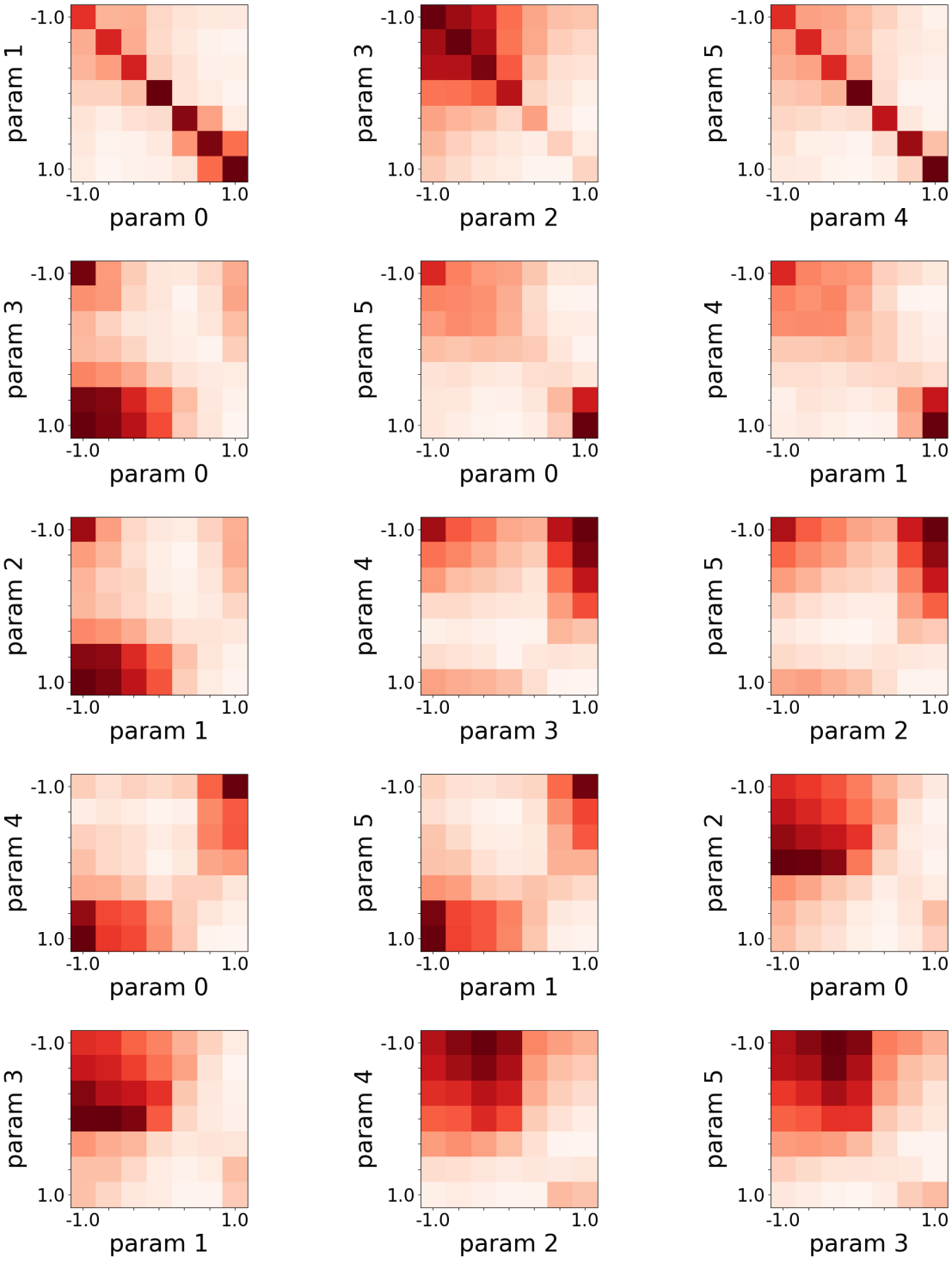


Figure 8. Each grid cell shows the cost of the best controller with the x-axis and y-axis parameters at that given cell value.