Evolving and Controlling Perimeter, Rendezvous, and Foraging Behaviors in a Computation-Free Robot Swarm

Matthew Johnson
Air Force Research Laboratory
26 Electronic Parkway
Rome, NY 13441
matthew.johnson.151@us.af.mil

Daniel S. Brown
Air Force Research Laboratory
26 Electronic Parkway
Rome, NY 13441
daniel.brown.81@us.af.mil

ABSTRACT

Designing and controlling the collective behavior of a swarm often requires complex range, bearing sensors, and peer-to-peer communication strategies. Recent work studying swarm of robots that have no computational power has shown that complex behaviors such as aggregation and object clustering can be produced from extremely simple control policies and sensing capability. We extend previous work on computation-free swarm behaviors and show that it is possible to evolve simple control policies to form a perimeter around a target, rendezvous to a specific location, and perform foraging. We also demonstrate that simple manipulations of the environment can be used to control, these collective behaviors. The robustness and expressiveness of these behaviors, combined with the simple requirements for control and sensing, demonstrate the feasibility of implementing swarm behaviors at small scales or in extreme environments.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*multiagent systems, coherence and coordination*; I.2.9 [**Artificial Intelligence**]: Robotics—*intelligent vehicles*; I.2.6 [**Artificial Intelligence**]: Learning—*parameter learning*

Keywords

swarm robotics, evolutionary algorithms, computation-free robot, controlling collective behaviors

1. INTRODUCTION

Flocks of birds, schools of fish, and colonies of ants, bees, and termites exhibit a remarkable robustness and resilience, despite the limited capabilities of each individual. Recently research into bio-inspired swarm robotics has been gaining popularity due to the low-cost, robust, redundant, and distributed nature of swarms [3]. Potential applications for robot swarms include search and rescue, construction, and chemical spill clean-up, as well as nano-medical applications such as finding tumors [11]. Many of these applications would benefit from simple, cheap, disposable swarms of robots that can accomplish these tasks quickly and without much human supervision.

There has been much recent work on different swarming algorithms and technologies; however, many still require localization, mapping, complex coordination algorithms, and precise identification of neighboring robots' orientations and relative positions. This often results in swarm behaviors that are interesting but extremely difficult to implement on actual robotic platforms. For swarm applications in the nano-medical field, developing collective behaviors that use extremely simple controllers and sensors is especially important if these behaviors have an hope of being implemented on nano-robots [12].

Recently, Gauci et al. have shown that swarms of robots with minimal computational power and memory can still collectively solve canonical multi-robot problems such as aggregation [7], and simple object clustering [6]. Researching the capabilities of simple swarms is important for several reasons: (1) simple robots are cheaper and more disposable, (2) simple control algorithms are easier to transition from simulation to actual robots [7, 9], and (3) even teams of complex robots may need a "Plan B" consisting of simple robust algorithms that require only the most basic capabilities in case of malfunctions and failure.

This paper extends the work of Gauci et al. by investigating what behaviors swarms of computation-free robots can achieve in an environment with a small number of targets. We utilize the evolutionary model proposed in [7] to learn simple robot controllers that lead to global behaviors that include: forming a perimeter around a target, rendezvousing with a target, and foraging. Additionally we show that simple manipulations of the environment can be used to control these behaviors.

2. RELATED WORK

Trianni et al. evolved a neural network based controller that performs aggregation using swarms of s-bots [14]. However, each s-bot uses eight infrared proximity sensors, three microphones, three sensors for detecting connections on the body and a gripper sensor. Baldassarre et al. evolved a controller that aggregates a group of robots and then moves them towards a light source [1]. Their controller utilizes a neural network that takes in eight infrared proximity sensors readings, four directional light source sensors readings, and

.

BICT 2015, December 03-05, New York City, United States
Copyright © 2016 ICST
DOI 10.4108/eai.3-12-2015.2262390

four directional sound sensors readings as control inputs. Gauci et al. introduced the concept of robots that can't compute [5]. They showed that a simple reactive controller could be used to allow a swarm of computation-free robots with a single line-of-sight sensor to perform aggregation [7] and clustering [6].

Other work has looked at controlling collective behaviors. Examples include controlling collective transport using a simple signals [13, 2], using termite-inspired stigmergic control to build complex structures [15], and controlling a small subset of a swarm to cause global switches between stable collective motion patterns [4]. However, none of this work considers the extreme conditions of a single line-of-sight sensor and zero computation.

3. PROBLEM FORMULATION

This paper investigates what collective behaviors are possible given a swarm of extremely simple robots operating in a simple environment. For our experiments we consider a circularly bounded 2D environment that is homogeneous and contains no obstacles. Throughout this space n circular robots are randomly distributed and randomly rotated such that each robot faces a random direction. Robots learn to interact with immovable targets, movable objects, and other robots to achieve global behaviors. All entities (robots, targets, and objects) are rigid such that no two entities can occupy the space at the same time.

We define simple robots as agents that are memoryless, cannot perform computations and have limited input/output capabilities. Specifically we look at robots that are only equipped with a line of sight sensor and two wheels for differential drive. The line of sight sensor can only detect the presence or absence of objects and outputs a trinary value where $s = 2$ corresponds to a target or object in line of sight, $s = 1$ corresponds to a robot in line of sight, and $s = 0$ corresponds to nothing in line of sight.

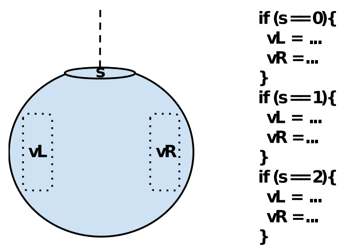
Simple robots are reactive in nature because they cannot remember past input or actions. As a result simple robot controllers can be to a sequential series of if-statements that assign left and right wheel velocities based on current sensor readings. This controller can be represented as a set of six wheel velocities

$$V = [v_{l0}, v_{r0}, v_{l1}, v_{r1}, v_{l2}, v_{r2}]$$

where v_{l0}/v_{r0} are the velocities of the left/right wheel when there is no robot in the sensors current line of sight, v_{l1}/v_{r1} are the respective velocities when a robot is within the line of sight of the sensor and v_{l2}/v_{r2} are the respective velocities when a target or object is within the line of sight of the sensor. Velocities are normalized such that $v = [-1, 1]$ where 1 corresponds to a wheel spinning forward at full speed and -1 corresponds to a wheel spinning backwards at full speed.

4. BEHAVIORS

This section explores several global behaviors learned using evolutionary optimization techniques. We discover global behaviors by optimizing a universal robotic controller according to a behavior dependent fitness function. Each potential robot controller is evaluated by running a swarm simulation and calculating the fitness function at every time



```
if (s==0){
  vL = ...
  vR =...
}
if (s==1){
  vL = ...
  vR =...
}
if (s==2){
  vL = ...
  vR =...
}
```

Figure 1: Robot representation and corresponding controller.

step to generate a fitness score, which is given by

$$U(V) = \sum_{t=0}^{T-1} tu(t)$$

where T is the number of time steps in the simulation and $u(t)$ is the fitness function. Multiplying the fitness function by the time step rewards controllers that achieve desired behavior quickly. The robot controllers are optimized using the average fitness score over multiple simulations to reduce the effect of noise.

All simulations are run on the Enki 2.0 robot simulator, which is able simulate hundreds of robots in a 2D environment in faster than real time [10]. For our experiments the simulation physics are updated 100 times per second and the robot controller is updated 10 times per second. Robots are simulated using Enki's e-puck model which have a diameter of 7.4 cm, inter-wheel distance of 5.1 cm, and weight of 152 g. Targets and objects are simulated as cylinders with a diameter of 10 cm using Enki's physical object model. Objects have a mass of 35g and a coefficient of friction of 0.58. Targets have a sufficiently large mass and coefficient of friction to ensure that are immobile.

Robot controllers are optimized using the Covariance Matrix Adaption Evolution Strategy (CMA-ES) [8]. This genetic optimization technique uses the variance of each gene to generate mutations between generation. Earlier work by Gauci et al. has shown that CMA-ES can effectively optimize simple robotic controllers [6]. CMA-ES optimizes across all real numbers, which can result in genes out of normalized range. To avoid this we constrain genes by applying the following sigmoid function

$$v = \frac{1 - e^{-x}}{1 + e^{-x}}$$

where x is a gene optimized by CMA-ES. For our experiments we utilized the following CMA-ES parameters: population size of 13, initial step size, $\sigma(0) = 0.72$, and starting controller of $V = [0, 0, 0, 0, 0, 0]$.

4.1 Aggregating to a Target

We first investigate what is possible when a single stationary target is placed in the environment. In this behavior robots are initially distributed randomly throughout an environment and over time form a group around a randomly positioned target. Our fitness function rewards global behaviors that minimize the total distance between each robot and the target. Let $p_i(t)$ represent the position of robot i at time step t and p_{target} represent the position of the target.

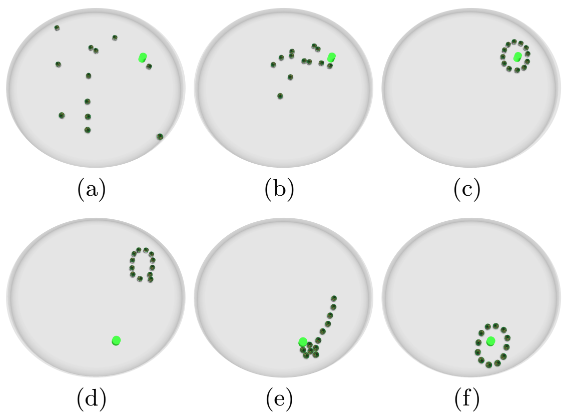


Figure 2: Snapshots of perimeter formation around a dynamic target at several time instances.

Then the rendezvous fitness function is given by

$$u_{rendezvous}(t) = -\sum_{i=0}^{n-1} \|p_i(t) - p_{target}\|_2^2 .$$

This fitness function rewards solutions where the robots are close to the objects locations.

4.1.1 Perimeter Formation

Using the procedure described above with the rendezvous fitness function we evolved the following controller

$$V = [1.0, 0.37, 1.0, 1.0, -1.0, 0.83]$$

that results in the robots forming a perimeter around a target. Robots gravitate towards a target or other robots by using the v_0 velocities to scan the area. Then a combination of the v_0 and v_2 velocities move the lead robot towards the target, while the other robots simply follow the leader using v_1. The perimeter formation emerges once the v_0 velocity can not turn the lead robot far enough in a time step to see the target.

Experiments with the controller show that behavior is robust to changes in the location of the target mid-simulation. When a target is moved to a new location mid-simulation, the entire swarm quickly moves to the location and reforms the perimeter as shown in Figure 2. Controlling behavior using the environment removes the need to broadcast information to the swarm and simplifies control logic.

4.1.2 Rendezvous

We are also interested in having every member of the swarm gather as close as possible to the target, rather than just circle around it. Rendezvous is an important behavior for swarms because it sets the stage for more complicated behaviors by assembling a group of robots to a specific desired location. We first tried to find a controller for the rendezvous problem using the fitness function described above; however, all trials resulted in controllers in which robots would form a circle around a target.

To solve this problem we seeded the starting controller with an aggregation solution from Gauci's et al. earlier work [7]. Using the seeded optimization strategy we evolved the following controller

$$V = [-0.72, -1.00, 1.00, -1.00, 0.99, 1.00]$$

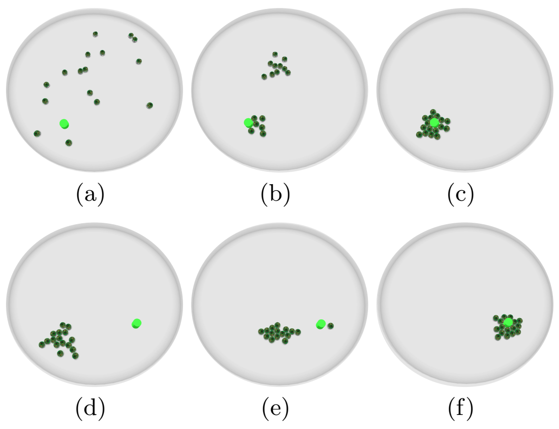


Figure 3: Snapshots of rendezvous to dynamic target at several time instances.

which results in the rendezvous behavior. Using this controller robots scan the area using the v_0. If a robot encounters a target while scanning it moves toward it with v_2. If another robot is in view the scanning robot begins to form a cluster using the v_0 and v_1 velocities. Over time the clusters merge into a single cluster around the target. This behavior is also robust to changes in the location of the target mid-simulation as shown in Figure 3.

4.2 Foraging

In this behavior objects and robots are distributed randomly throughout the environment and the robots must gather the objects to a specified target location. Earlier work by Gauci et al. found an computation-free controller for clustering objects, we extend there work by showing that this controller can be used for foraging, i.e. clustering objects to a specific location [6]. The clustering fitness function rewards global behaviors that minimize the total distance between each object and center of the cluster of objects. Let $o_i(t)$ represent the position of object i at time step t and $\overline{o}(t)$ represent the center of the object cluster. Then the fitness function is given by

$$u_{clustering}(t) = -\sum_{i=0}^{m-1} \|o_i(t) - \overline{o}\|_2^2$$

where m is the number of objects. Using the clustering fitness function we evolved the following controller

$$V = [0.72, 1.00, 0.40, 0.31, 0.53, -1.00]$$

which causes the robots to circle around the objects and slowly nudge them into a central point as they pass.

The foraging behavior occurs when we place one or several fixed targets in the environment. Figure 4 shows the classic foraging problem where there is a “nest” location (shown in green) where all of the items must be gathered. Figure 5 shows an alternative foraging scheme where multiple stationary targets are placed in the environment. The convex hull of these targets defines the region into which the objects will be harvested. Similar to the previous behaviors, the foraging behavior can be controlled simply by changing the location of the targets. The robots will then move the items to the new desired location, as shown in Figure 5.

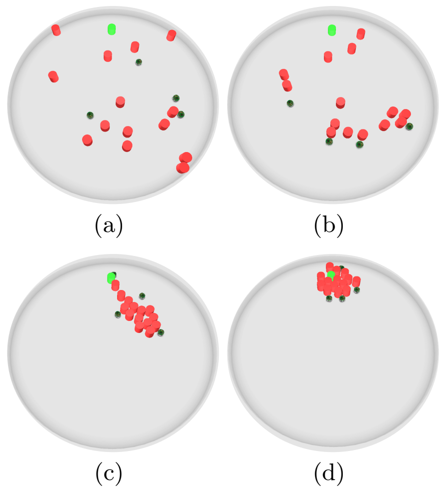


Figure 4: Snapshots of foraging at several time instances.

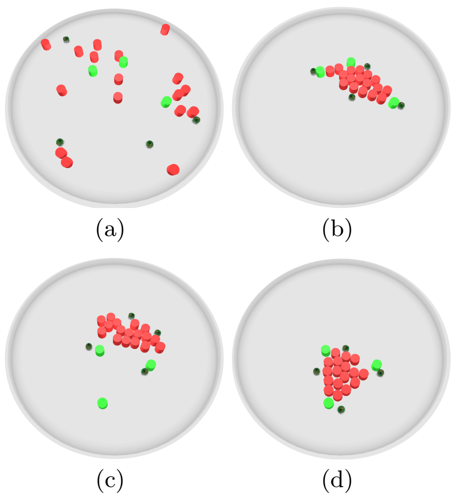


Figure 5: Snapshots of dynamic foraging into specified convex hull at several time instances.

5. CONCLUSIONS AND FUTURE WORK

A large amount of research has been dedicated to developing multi-agent systems that perform complex behaviors. We show that swarms of robots that can't compute can perform complex behaviors such as rendezvous to a desired location, simple perimeter monitoring of a desired location, and foraging in changing environments. Our results demonstrate that complex behaviors can be evolved from simple interactions between agents and that these behaviors can be controlled during execution by simply changing the environment. We note that these controllers are so simple that they could simply be hardwired, requiring no computational capabilities. We believe that this research is an important step towards swarm behaviors that can be easily implemented in hardware and produced at small, maybe even nano-scale. In the future we plan to apply these behaviors to actual robots, explore virtual targets and other forms of control uisng the environment, and more rigorously explore the space of possible behaviors given our computation-free assumptions.

6. REFERENCES

[1] G. Baldassarre, S. Nolfi, and D. Parisi. Evolving mobile robots able to display collective behaviours. *Artificial Life*, 9(3):255–267, 2003.

[2] A. Becker, C. Ertel, and J. McLurkin. Crowdsourcing swarm manipulation experiments: A massive online user study with large swarms of simple robots. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2825–2830. IEEE, 2014.

[3] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, 2013.

[4] D. S. Brown, S. C. Kerman, and M. A. Goodrich. Human-swarm interactions based on managing attractors. In *Proceedings of the 2014 ACM/IEEE International Conference on Human-Robot Interaction*, pages 90–97. ACM, 2014.

[5] M. Gauci, J. Chen, T. J. Dodd, and R. Groß. Evolving aggregation behaviors in multi-robot systems with binary sensors. In *Distributed Autonomous Robotic Systems*, pages 355–367. Springer, 2014.

[6] M. Gauci, J. Chen, W. Li, T. J. Dodd, and R. Groß. Clustering objects with robots that do not compute. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems*, pages 421–428. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[7] M. Gauci, J. Chen, W. Li, T. J. Dodd, and R. Groß. Self-organized aggregation without computation. *The International Journal of Robotics Research*, pages 1145–1161, 2014.

[8] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[9] N. Jakobi. Half-baked, ad-hoc and noisy: Minimal simulations for evolutionary robotics. In *Fourth European Conference on Artificial Life*, pages 348–357. MIT press, 1997.

[10] S. Magnenat, M. Waibel, and A. Beyeler. Enki: The fast 2d robot simulator, 2007.

[11] S. M. Moghimi, A. C. Hunter, and J. C. Murray. Nanomedicine: current status and future prospects. *The FASEB Journal*, 19(3):311–330, 2005.

[12] A. A. Requicha. Nanorobots, nems, and nanoassembly. *Proceedings of the IEEE*, 91(11):1922–1933, 2003.

[13] M. Rubenstein, A. Cabrera, J. Werfel, G. Habibi, J. McLurkin, and R. Nagpal. Collective transport of complex objects by simple robots: theory and experiments. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-Agent Systems*, pages 47–54. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[14] V. Trianni, R. Grob, T. Labella, E. Sahin, and M. Dorigo. Evolving aggregation behaviors in a swarm of robots. In *Advances in Artificial Life*, pages 865–874. Springer, 2003.

[15] J. Werfel, K. Petersen, and R. Nagpal. Designing collective behavior in a termite-inspired robot construction team. *Science*, 343(6172):754–758, 2014.