A central-place foraging algorithm using Swarmies in the Gazebo simulated environment

Manuel G. Meraz

Abstract-- The Moses Biological Computational Lab observes the swarming behaviors of natural complex systems with an emphasis on foraging. Foraging behaviors arise out of necessity in order for these organisms to sustain themselves, and as such the strategies that are used by these organisms can be modeled into a generalized algorithm using stochastic methods with parameters that can be optimized for the given environment. The central-place foraging algorithm (CPFA) is one of these algorithms; inspired by the seed-harvester ants of New Mexico, the narrow window of time in which they have available to them forces the ants to evolve an optimal search strategy biased towards the quantity of food collected. This foraging strategy has previously been implemented in the iAnt robots developed here, as well as in the ARGoS simulator. As the lab continues to study the CPFA and its derivatives, the need to move from simulated robots living in ideal environments, to more chaotic and realistic environments begins to arise as a stepping stone towards application. The Gazebo simulator utilized by the robot operating system (ROS) framework was the next obvious choice using Swarmies, and the increased complexity highlights the importance of a robust collision avoidance strategy. Initial testing shows us that the collection efficiency per robot in Gazebo decreases as compared to ARGoS.

I. Introduction

The cost of robot components have reached a state of affordability, designing and building robots en masse has become something that, while previously impractical, is now completely feasible. These low-cost robots would ideally be capable of working together as a decentralized unit to explore unmapped environments using efficient communication patterns to optimize foraging behaviors.

However, the problem with low-cost robots is that their sensors are noisy and unreliable for precise navigation and in a simulated environment, each robot is an exact clone of each other. In physical robots, while most of the robots are built with the same hardware, they differ enough such that they will scale differently. It’s possible that one robot has a loose wheel that generates noise in the odometry of the robot, which then inevitably falls off, while another robot’s battery is nearly drained and power output is decreased.

One step closer towards a physical robot, is the environment provided to us by the Gazebo simulated environment, which simulates how physical robots behave much more accurately than the ARGoS simulated environment.

The goal of this project was to start the migration process from the ARGoS simulator as the standard tool to model and collect data for swarm algorithms, to the Swarmie robots in Gazebo where the software can easily be translated to work in their physical robot counterparts.

II. Related Work

A. Low-cost *Robots*

The Kilobots [1] are an example of low cost robots that can be mass manufactured to study swarm behaviors. The iAnt robot platform developed here gave us early insights into how an earlier derivative of the CPFA [2] behaves in physical robots as compared to a simulated environment.

We learned that the simulated experiments that used an agent based model showed slightly better scaling than the real robots. Which is not surprising because real robots have more difficulty with avoiding each other, physical hardware limitations, imperfect localization and the real possibility that physical robots confuse each other for other objects [3].

B. Swarm Foraging Algorithms

The CPFA can be optimized up to a certain point, but the problem with the CPFA is that is not scalable. As the number of robots increases in the swarm, the central collection zone inevitably becomes congested and traffic jams become inevitable, thus time spent avoiding collisions increases. There is a threshold for the number of robots where robots inevitably end up spending more time avoiding collision with each other, and considerably less time spent foraging.

To solve this problem, the multi-place foraging algorithm (MPFA) came to be as a natural derivative of the CPFA [4]. The MPFA solves the congestion problem by adding in the necessary quantity of collection zones for optimal foraging rates. As such, the algorithm is able to scale up to satisfy any search area.

The MPFA is not just the CPFA with multiple nests; the MPFA can be implemented with both static and dynamic collection zones. The dynamic collectionzones will place themselves in the location required to minimize the travel time of the foraging robots, and maximize the search time the robots are in. These foraging algorithms, while useful, do not tell on their own tell us how well they work because then this begs the question, *relative to what?*

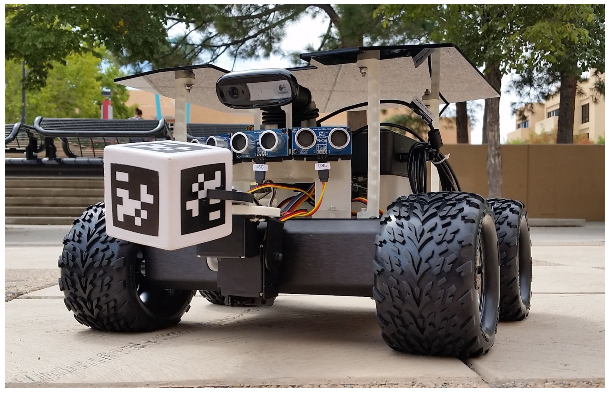
The answer to this is a distributed deterministic spiral search (DDSA). The purpose of the DDSA is to serve as a benchmark for swarm foraging algorithms, that other swarm foraging algorithms can compare themselves to. Now the CPFA, MPFA, and any other algorithm within this category can systematically be compared to and studied relative the DDSA.

III. Methods

A. The Swarmie

Swarmies are small robotic vehicles measuring approximately 30 cm x 20 cm x 20 cm. Each Swarmie is equipped with sensors, a webcam, GPS system, and Wi-Fi antenna. They operate autonomously and can be programmed to communicate and interact as a collective swarm [6].

The simulated swarmies and the physical swarmies have the same geometry and sensors as the physical swarmies.



B. Robot Simulation

We implemented the CPFA using the Gazebo robot simulator that comes along with ROS. Robot simulation is an essential tool in every roboticist's toolbox. A well-designed simulator makes it possible to rapidly test algorithms, design robots, perform regression testing, and train AI system using realistic scenarios. Gazebo offers the ability to accurately and efficiently simulate populations of robots in complex indoor and outdoor environments. It provides a robust physics engine, high-quality graphics, and convenient programmatic and graphical interfaces.

The parameters for the robots that are used are informed by the Swarmies. The major difference between the simulated and physical robots is that the positional data from odometry in the physical robots is much less accurate than the simulated robot, which has its pros and cons. The good part of this is that it allows us to focus more on the logical portion of the algorithm since we can assume localization is not an issue. Meaning we spend less time thinking about how to account for the robot's positional data and more on higher level state machines with a higher certainty. The downside to this is that when we transfer the code to the physical robots, it will not work the first time. We will need to add in additional functionality to account for the localization error.

Targets are cubes that have dimensions of approximately 3.2 cm^3. The robot's default forward and backward movement speed is about 0.5 m/s with a viewing distance of up to 1 meter due to the camera being angled down towards the ground.

C. Experimental Setup

The robots were set up in a barriered square arena with a 15 m^2 area. 3 robots were spawned facing the central collection zone at, which has a square shape and area of 1 m^2, at a distance of 1.32m away from the dead center. 256 targets were dispersed using a power law distribution, or partially clustered, with 1 cluster of 16, 4 clusters of size 64, 16of size 4, and 64 single targets. All experiments last 30 minutes, with a total of 15 experiments that were ran.

IV. Results

The purpose of the trial runs were to compare the DDSA with the CPFA and see how they performed. These results are just preliminary and do not mean anything as the way the algorithms were coded are not fundamentally following the same structure. The CPFA's complexity is much higher than the DDSA's and as such it needs more development time, which I was not able to completely finish.

Trail	CPFA	DDSA
1	59	92
2	57	81
3	52	102
4	67	84
5	59	83
6	48	85
7	47	81
8	46	62
9	41	97
10	51	86
11	49	105
12	71	82
13	45	107
14	58	98
15	61	86

The mean for the CPFA ended up as ~54 and the mean for the DDSA ended up being ~88 targets collected. The major cause for the difference in the two algorithms is the obstacle collision algorithm that is currently implemented. The CPFA has a very sensitive collision avoidance algorithm whereas the DDSA was able to have it be minimized in order to avoid collisions. With the DDSA this could be done because the paths for all the robots are predetermined and because localization in the simulator is very accurate, we can guarantee that as long as the predetermined paths do not collide, then the robots do not need obstacle avoidance. Ideally both algorithms would utilize the same obstacle avoidance algorithm to see a more accurate comparison.

References

1. M. Rubenstein, C. Ahler, and R. Nagpal. Kilobot: a low cost scalable robot system for collective behaviors. In Proceedings of the IEEE International Conference on Robotics and Automation, pages 3293–3298, 2012.
2. J. P. Hecker, M. E. Moses, Beyond pheromones: evolving error-tolerant flexible and scalable ant-inspired robot swarms Swarm Intelligence, US:Springer, vol. 9, pp. 43-70, 2015.
3. Hecker, J. P., Letendre, K., Stolleis, K., Washington, D., and Moses, M. E. (2012). Formica ex Machina: Ant Swarm Foraging From Physical to Virtual and Back Again. Swarm Intelligence, 7461:252–259.
4. Q. Lu, J. P. Hecker, and M. E. Moses. The MPFA: A Multiple-Place Foraging Algorithm for Biologically-Inspired Robot Swarms. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2016
5. G. Matthew Fricke, Joshua P. Hecker, Antonio D. Griego, Linh T. Tran, Melanie E. Moses, "A distributed deterministic spiral search algorithm for swarms", *Intelligent Robots and Systems (IROS) 2016 IEEE/RSJ International*

Conference on, pp. 4430-4436, 2016, ISSN 2153-0866.

6. https://github.com/BCLab-UNM/Swarmathon-Robot