MAE 598: Multi-Robot Systems

# **Object Rearrangement by Individual Robots and Discovery**

Aniket N Prabhu; Pancham Kamdar; Sanket Kavathekar

1. **Abstract (Aniket N Prabhu)**

Exploration of hazardous locations on Earth as well as in outer space pose numerous challenges to explore such as presence of harsh conditions. Apart from methods such as Simultaneous Localization and Mapping (SLAM), other ways of analyzing such places is via sample collection of objects such as soil, rocks, minerals, and other elements that constitute the environment. Due to the risks associated with traversing such environments, deployment of centralized robots can be expensive as they are often costly to manufacture, maintain and in the worst case, replace.

The redundancy of the decentralized robotic swarms, however, can be exploited for such tasks as their maintenance and manufacturing is relatively cheap, and so they are easily replaceable or repairable in case of malfunctioning or destruction. Moreover, swarm systems often use local sensing with which the individual units can coordinate with each other and function without needing a central supervising unit. This task of sample collection can be accomplished through a swarm behavior known as “Object Clustering and Assembly” which comes under a behavior group called “Spatial Organization” **[1]**.

In this project, we have tried to apply this behavior by simulating food foraging robots, that follows a behavior of distributing the population in walkers and beacons using task allocation. We have presented two algorithms, inspired by ant colony behavior **[2]**, which uses a simple concept: each robot can dynamically take on one of two roles, that of a stationary environment beacon (like a pheromone) or a wandering robot (walker). The two algorithms using cardinality, differ in when the beacon role is chosen and what information the beacon emits and similarly the walker. The project uses a stochastic model of multirobot systems with a swarm size of 30-50 robots, with species definition and task allocation. We have analyzed the change in species based on the interactive behaviors, the ability to initiate random walk, the amount of food transfer and rate based on the path formation and theoretical analysis of robot interactions, etc. We try to show that both algorithms can effectively construct paths between food and nest.

1. **Mathematical Model (Sanket Kavathekar; Pancham Kamdar)**

The Multi Robot behavior studied in this project is the foraging of robot swarms to find a target and trace the path to the nest from the target. To form a workable mathematical model and simulate the model we had to make certain assumptions. The objective of the overall model is to transfer food from food site to nest site and increase the transfer rate by formation of pheromone connected path.

Assumptions:

* The study area used in the model is bounded by walls to make it a definitive model. All the robots operate within this control space. This is to prevent the robots from wandering off into the infinite space and getting lost.
* There are two sites of interest, a food site and a nest site. The food site is assumed to be an infinite source of food. This is to make sure that the simulation goes on for as long as we desire for our experiments.
* The robots are epucks with limitless batteries and have additional turret nodes, namely emitter and receiver, in addition to its default emitter and receiver nodes so that the robots can communicate with the food source, the nest site, and with each other omnidirectionally. Additional nodes were installed because the epuck’s default nodes were directional..
* We have decided to perform the experiment with a swarm of 16 robots.
* The change from the biological model we have used is the use of virtual pheromone values to avoid the computational power requirements due to odometry and vision related tasks.
* We have two sets of values representing food and food pheromone. Food values can only be carried by a walker robot and a food pheromone and nest pheromone value

Based on our project model, below is our mathematical mean field models, below are the equations. We have used Gillespie’s Direct methodology to form a system of equations forming a mathematical representation of our project. In **[2]**, we have two species, walkers and beacons, that travel between the food site (F), and the nest site (N).

Wm Ba … with probability Ki,Kj

Wm is the walker that is foraging to find food,

Ba is the beacon that is stable without the food pheromone value,

Ki is the probability of Walker turning into a beacon, Kj is the probability of walker turning into beacon

At the start of the simulation or scenario, we start with all walker robots and with probability Ki, a certain number of walkers turn into beacons. Further with probability of walkers encountering other beacons, they turn into beacons or remain a walker depending on the populations. This is all constituted for using probability Ki.

Wm + F  Wn …with rate constant α1

Wn is the walker with positive food value going towards the nest,

F is the food site,

When the foraging walker encounters the food site, it gets the positive food value and turns into the walker searching nest site.

Wn + Ba Bb …with rate constant β1

Bb is the beacon with positive food pheromone value

When the Walker with positive food value encounters a stable beacon, it gives that beacon a positive food pheromone value.

Wn + N  Wm … with rate constant α2

N is the nest site. Here when the walker is positive food value reaches the nest, it empties the food into the nest and turns back into the foraging walker. Now this foraging walker will follow the first equation and can turn into a beacon or remain a walker.

Wm + Bb  Wo … with rate constant α3

Where, Wo is the walker that is directed to the food site.

When the foraging walker comes in contact with the beacon having positive food pheromone value, it is directed towards the food site to accelerate the process.

Thus, this whole process is repeated till the food is completely transferred from food site to nest site. But since we have assumed an infinite food source to have a better control over the simulation sample size, this process will go on till we stop it or till a user-defined finite time.

These assumptions mimic the realistic scenarios; the ants collectively work together to find and transfer food from a random food site. The assumption of having two species is that beacons represent the biological food pheromones which in reality are the food olfactory signals left in the path the ants follow back to the nest. And walkers represent actual ants that follow these signals to find a way back to the nest or towards food. Due to limitations in robot sensing capabilities we have used exchanging of real-valued floating point numbers instead of olfactory senses and visual identifications. One limitation that the system has over the biological model is that the encounter rate, which in the simulation depends on sensor range and velocity of the robot.

In the research work by N. Britton et al. They have studied the decision of a new home for honeybees. In **[4]**, they have used the spread of information or a belief model for the mathematical introduction of the honey bee communication. The analogy is drawn between those ignorant of the information and susceptible, those spreading the information and infectives, and, in some models, those no longer spreading the information and those immune to the disease. We have based our model somewhat on the same principle analogous to believers and nonbelievers as walkers and beacons. And the belief transfer and conversion being the cardinality algorithm. The probabilistic approach of choosing a nest site in the honeybee inspired robots can be related here with the decision of walker and beacon in the first step.

Similarly, Dr. D. Lee, in his study based on resource-based task allocation has divided the mathematical modelling on the basis of Task matrix, similar to our task allocation of foraging and stable transfer of pheromones. In **[5],** They have further explained the robot task interaction by simulating the bidding scenario where their approach consists of bidding, winner update, task trade and list update. Based on the resource the task is allocated to the robot, like our model.

1. **Theoretical Analysis (Sanket Kavathekar; Pancham Kamdar; Aniket Prabhu)**

By clear observation of the chemical reaction networks formulated above, we can analyze certain aspects of the system. The system of CRNs is not strongly connected as the path cannot be traced back from one equation to another.

We have theoretically 7 species of robots interacting with each other. Whereas based on the CRNs we have 9 complexes. (Considering Wn + N  Wm Ba as one CRN). At t = 0 all the robots will be walkers (Wm).

M =

[1 0 1 0 0 0 0 0 0 1 0

0 1 0 0 1 0 0 0 0 0 0

0 0 1 0 0 0 0 0 0 0 0

0 0 0 1 1 0 1 0 1 0 0

0 0 0 0 0 1 0 1 0 1 0

0 0 0 0 0 0 1 0 1 0 0

0 0 0 0 0 0 0 0 0 0 1] Mϵ RSxC

K =

[ ki, -kj, 0, 0, 0, 0, -a2, 0, 0]

[ -ki, kj, 0, 0, 0, 0, 0, 0, 0]

[ 0, 0, a1, 0, 0, 0, 0, 0, 0]

[ 0, 0, -a1, 0, 0, 0, 0, 0, 0]

[ 0, 0, 0, 0, b1, 0, 0, 0, 0]

[ 0, 0, 0, 0, -b1, 0, 0, 0, 0]

[ 0, 0, 0, 0, 0, 0, a2, 0, 0]

[ 0, 0, 0, 0, 0, 0, 0, -a3, 0]

[ 0, 0, 0, 0, 0, 0, 0, -a3, 0] Mϵ RCxC

We have attached the MATLAB program for the analysis of above model. Although, this model needs the rate constant values and probability theory to get all the actual values. So, the MATLAB program is the framework for the model.

A Braitenberg controller is implemented for obstacle avoidance of the e-puck robots **[7]**. The controller can be implemented by either the Proximal control paradigm or the Distal control paradigm. In our controller we have implemented the Distal controller paradigm.

In Proximal control, the robot’s sensors are directly interfaced with the motor controls. The sensor inputs enable the motor speeds to respond directly. The e-puck is a two-wheel-differential robot with eight sensors, resulting in a weighted matrix that can convert the sensor inputs to motor speeds. The rows correspond to the number of distance sensors and the columns correspond to the number of motors. The weighted matrix consists of a two - dimensional array, where each row is a specified item list of left and right motor speeds respectively. This array is then used in the obstacle avoidance algorithm. The weights of the matrix are empirically determined. The Braitenberg controller implements the control algorithm below:

*speed[i] += matrix[j][i] \* (1.0 – (sensorsValue[j] / 512));*

The *speed[]* array is a two item which represents the speed of the left and the right motor, the *matrix[]* array holds the values in the weighted matrix, and the *sensorValue[]* array holds the sensor values. The 512 integer included in the algorithm helps normalize the sensor values. The *i* and *j* values are values from a nested loop used to iterate over the sensor values and the weighted matrix.

In Distal control, the sensor values are given to a set of rules which determine the motor speeds. In our controller, this is the code we implemented in our Step function:

*left\_obstacle = psValues[5]>80.0 or psValues[6]>80.0 or psValues[7]>80.0*

*right\_obstacle = psValues[0]>80.0 or psValues[1]>80.0 or psValues[2]>80.0*

*# Using info on obstacle to actuate wheels:*

*MAX\_SPEED = 6.28 # ~2\*pi rad/s*

*# Initializing motor speeds at 50% of MAX\_SPEED:*

*leftSpeed = 0.5 \* MAX\_SPEED*

*rightSpeed = 0.5 \* MAX\_SPEED*

*# Robot will turn right:*

*if left\_obstacle:*

*leftSpeed += 0.5 \* MAX\_SPEED*

*rightSpeed -= 0.5 \* MAX\_SPEED*

*# Robot will turn right:*

*if left\_obstacle:*

*leftSpeed += 0.5 \* MAX\_SPEED*

*rightSpeed -= 0.5 \* MAX\_SPEED*

Properties of the model that we have validated below in section IV are:

1. Decentralized control: All swarm members function independently without the need of an external supervisor.
2. Communication through local sensing: Robots communicate locally without the need to store excess information or to be in contact with a server.
3. Minimal Computation: Robots rely on IR sensors and serial communication (bluetooth) to traverse and transport food thereby making the operation computationally inexpensive as opposed to the use of odometry or computer vision based operations.
4. **Validation in Simulations (Aniket N Prabhu; Pancham Kamdar)**

Apart from simulating the stochastic models in MATLAB, we have also performed simulations in a 3D robotics simulator by Cyberbotics called “Webots”. The controller programming is done in Python 3. We have used the GCTronic’ e-puck robot for our simulations due to its simplicity. The robot has 8 IR sensors which can function as light sensors as well as distance/proximity sensors, 9 LEDs, 8 red, 1 yellow and one green, two differential wheel motors with an encoder each, and inbuilt directional emitter and receiver devices that communicate via bluetooth. We also have utilized the e-puck’s turret slot to install additional emitter and receiver nodes in order to enable the robot to communicate omnidirectionally. The e-puck has a “food” variable initialized to zero to imitate an actual food target. Two robots have been defined as FOOD and NEST sources. The FOOD robot (blue) has an emission range of 0.15m and it broadcasts a byte-encoded “food” string to the e-pucks within its range. The NEST robot (green) has an emitter node with a range of 0.15m to inform the e-pucks of its function as the nest, and a receiver node to receive “food” from the e-pucks carrying food.

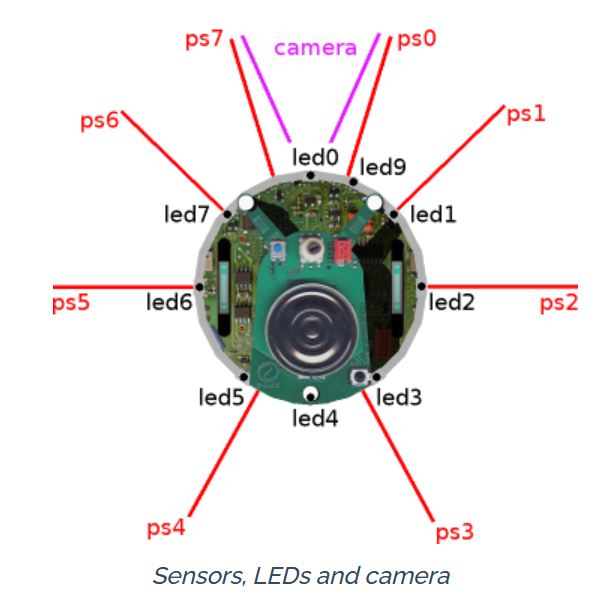


Fig. 1. This diagram shows the direction of emission of the IR sensors (labelled ‘ps0’ to ‘ps7’, the location of the LEDs (labelled ‘led0’ to ‘led7’), and the aperture of the camera [7].

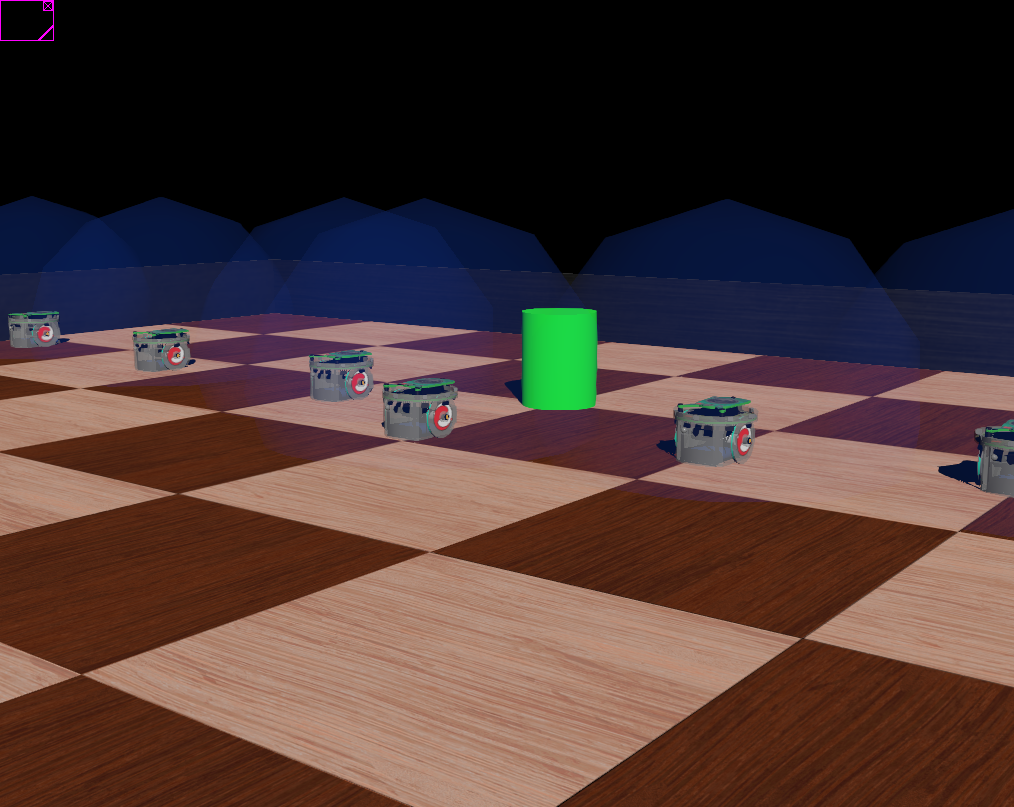


Fig. 2. This figure shows multiple epucks along with the NEST (in green). The blue spheres around each e-puck are additional nodes created to visualise their emitter’s range (which was infinite by default in the simulator, but manually set to 0.17m)

In our experiment, we initially start from the nest site with the robots navigating the field randomly based on the Braitenberg based collision avoidance controller and search for the FOOD robot. As walkers, they have their green LED turned ON. As soon as the simulation starts, alongside their random walks, a random floating-point value is drawn from a uniform distribution in the semi-open range, [0.0, 1.0), which decides which robot shall continue to be a walker and which robot will become a beacon after 30 seconds have passed in the simulation time.

The food robot constantly emits the message, “food”, informing any robots in its range of its function as the food source. Once a wandering e-puck receives this message, it sets its food variable to 1, indicating that it is now “carrying food”, and begins looking for the nest. On the other hand, the NEST robot constantly broadcasts the message, “nest” to inform the e-pucks of its function as the nest. If the e-puck detects the NEST while having its food variable = 1, it will transmit a message, “food received”, to the emitter and turn its food variable back to zero. The NEST on the other hand has a counter, food\_received, to count the amount of food returned to it, and every time it receives the “food received” message from the e-puck, it increments its food\_received counter by 1.

After 30 seconds have passed, the robots with their drawn random value below 0.5 turn into beacons, turn off their green LED and turn on their yellow LED.

Fig. 4. Walker with green LED (Left) and Beacon with yellow LED (Right)

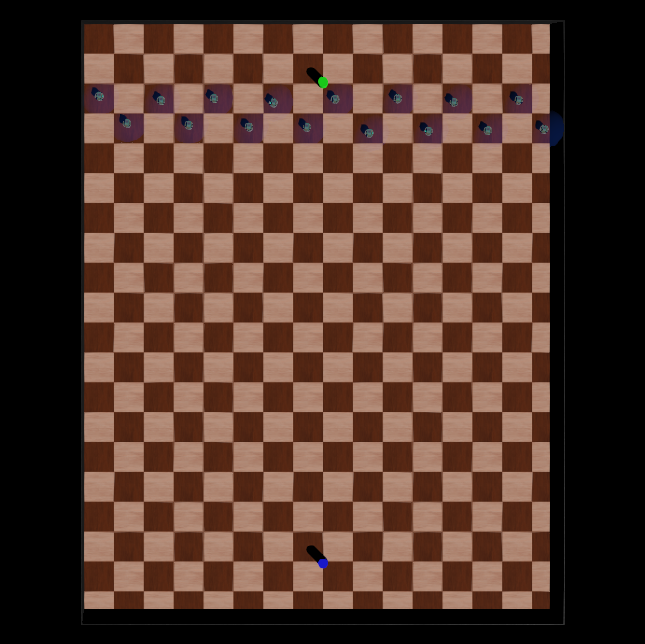


Fig. 5. Top view of our simulation. This is a 10m x 8m board (each tile 0.5m x 0.5m) used for our simulations with 16 e-puck robots (the blue spheres engulfing each bot is their approximate emitter range of 0.17m). The green cylinder near the e-pucks at the top is the NEST robot while the blue cylinder at the bottom is the FOOD robot.

1. **Future work:**

Our future work includes devising a method for the epucks to sense multiple robots in its vicinity, either walkers or beacons, and then with a certain probability, either continue its current role, or switch to another role. Currently, its IR sensors only detect objects to avoid while traversing the terrain, but a way to distinguish between objects needs to be created without using computationally expensive nodes such as the camera due to the purpose of the project being able to operate inexpensively. More specifically, if a walker is not carrying food and detects less than two beacons within a finite time frame, it would switch its role from being a walker to becoming a beacon. On the other hand, if a beacon were to detect more than three beacons in its vicinity, it will switch to being a walker with probability, “p”, or if both of its pheromones deplete beyond their thresholds.

1. **Conclusion**

Thus, our project aims to successfully transport objects from one place to another using robotic swarms and without the need of computationally intensive devices, thus achieving a decentralized, inexpensive transport system operating through local sensing. Our project also aims to study the behaviour of population transition using mean-field models. With this, our project aims to decentralize all the properties that define a multi-robot control architecture to be decentralized according to the definition provided by Dr. James McLurkin, namely, Communication, Computation, Sensing, Actuation and Coordinates.

The files and codes needed to perform the simulation are provided separately.

**REFERENCES**

[1] Melanie Schranz, Martina Umlauft, Micha Sende, and Wilfried Elmenreich. Swarm Robotic Behaviors and Current Applications. Frontiers in Robotics and AI, 2020

[2] Nicholas R. Hoff III et al. ‘Two Foraging Algorithms for Robot Swarms Using Only Local Communication’, Harvard University in Cambridge Massachusetts, USA

[3] M. Ullah et al. ‘Deterministic modelling and stochastic simulation of biochemical pathways using MATLAB’, IEE Proc.-Syst. Biol., Vol. 153, No. 2, March 2006

[4] N. F. Britton et al., ‘Deciding on a new home: how do honeybees agree?’, The Royal Society, May 2002

[5] Dong-Hyun Lee, ‘Resource-based task allocation for multi-robot systems’, Robotics and Autonomous systems, May 2017

[6] [Introduction to Robotics, Lab #3: Obstacle Avoidance](http://correll.cs.colorado.edu/?p=974) - Braitenberg controller algorithm

[7] [Webots documentation: GCTronic' e-puck](https://www.cyberbotics.com/doc/guide/epuck)

[8] Spring Berman. MAE 598 MRS - Lecture 3 - September 4, 2020