Collaborative Object Transportation using Centralized Multi-Robot Systems in Warehouse Resource Distribution

Daniel Kim

Chester Szatkowsi

Arizona State University

Table of Contents

I. Abstract

In this project, the group investigated the use of multi-robot systems for cooperative transportation of objects in resource distributions settings. The specific use case of interest was the transportation of objects within a warehouse to be shipped out. The project explored two potential methods for controlling such a system. Firstly, a stochastic process for swarm robotics was modeled, analyzed, and simulated. This system consisted of a homogenous robot swarm which would transport small packages and large pallets. The robots were also capable of assembly and disassembly of large pallets from small objects. From this setup, an associating chemical reaction network was created. Analyzing the chemical reaction network and its reversibility revealed that the system would achieve the desired effect of the system and this was compared to a different potential swarm approach through a modified chemical reaction network. The system was modeled macroscopically and mesoscopically while demonstrating the discrepancies between the two methods. As the macroscopic model lacks consideration for the randomness involved in the system, the results were expected to be smooth and repeatable. The mesoscopic model, however, accounts for some of these lost phenomena resulting in notable population fluctuation and a range of potential results. Gillespie's direct method was used to simulate the mesoscopic model which displayed the greater fluctuations and possibilities in population throughout the simulation compared to the macroscopic model.

Furthermore, the path planning and following for a two-wheeled differential drive robot was explored. For path planning, the Probabilistic Roadmap (PRM) method was utilized. A simple but significant property of the method, probabilistic completeness, is described and simulated. The kinematic model for a two-wheeled differential drive robot was applied to a path-following controller and algorithm. The basic framework and important parameters were

explained. Lastly, the effects of the parameters on the system's stability and performance was simulated.

II. Mathematical Model

II.1 Swarm Control Model

The simulated swarm multi-robot system investigated a retailer shipping warehouse primarily responsible for shipping out large pallets to brick and mortar retailers but occasionally shipping out individual small packages direct to consumers. These small packages and large pallets are kept in the storage area of the warehouse and are moved by the robots to the shipment area for orders which are placed.

This system consisted of one type of robot designed to move pallets and packages throughout the warehouse from the storage to the shipping area. These robots were capable of moving small packages alone, or moving large pallets when in groups of four. Additionally, the robots were capable of combining ten small packages into a single large pallet or disassembling a large pallet into ten small packages to account for variances in orders which may call for unexpected amounts of small vs large shipments.

The multi-robot system was represented as a chemical reaction network with species representing different actions which the robots are working on and different states of the packages (Lerman). The robots may be idle and not currently interacting with any objects as represented by "a". Large pallets may be idle in the storage area represented by "c" while idle small packages are represented by "b". The action of the four robots transporting the large pallets is represented by "d". The action of a robot transporting a small package is represented by "e". Once a large pallet is delivered to the shipping area, it becomes a delivered large pallet represented by "w" and similarly a delivered small package becomes "q". The chemical reaction network has mass action kinetics with the reaction rate constants $alpha_1$, $alpha_2$, $alpha_3$, $alpha_4$, $alpha_5$, and beta. The chemical reaction network is shown below in figure 1.

$$4a + c \xrightarrow{\alpha_1} d \qquad 10b + a \underset{\beta}{\overset{\alpha_5}{\rightleftharpoons}} c + a$$

$$a + b \xrightarrow{\alpha_2} e$$

$$d \xrightarrow{\alpha_3} w + 4a$$

$$e \xrightarrow{\alpha_4} q + a$$

Figure 1. Chemical Reaction Network

The modeled system was set to begin with 50 idle robots, 300 idle small packages, and 600 idle large pallets and run for one hour. Reaction rates were estimated to be as follows: $alpha_1$= 0.08, $alpha_2$= 0.064, $alpha_3$= 0.032, $alpha_4$ = 0.016, $alpha_5$ = 0.0032, and beta = 0.00144.

The multi-robot system was first modeled as a macroscopic model using a system of multi-affine ordinary differential equations in the form x_{dot} = -MKy(x) where M contains the coefficients of each coefficient for each complex, K contains reaction rates, and y(x) is a vector of the complexes. Each of these are shown in figure 2, figure 3, and figure 4 respectively. The system was also modeled from a mesoscopic perspective using Gillespie's direct method (2347). This method takes the same initial state and reaction rates as the macroscopic model. The simulation then runs in one second increments through the hour. For each second, reaction propensities, a_p, are calculated and used to choose the next reaction based on equation (1). A time advancement tau is then calculated based on equation 2.

$$M = \begin{bmatrix} 4 & 0 & 1 & 0 & 0 & 4 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 10 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Figure 2. M Matrix

$$K = \begin{bmatrix} \text{alpha}_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\text{alpha}_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \text{alpha}_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\text{alpha}_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \text{alpha}_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\text{alpha}_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \text{alpha}_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\text{alpha}_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \text{alpha}_5 & -\text{beta} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\text{alpha}_5 & \text{beta} \end{bmatrix}$$

Figure 3. K Matrix

$$Y = \begin{bmatrix} a\bullet c & d & a\bullet b & e & d & w\bullet a & e & q\bullet a & b\bullet a & c\bullet a \end{bmatrix}$$

Figure 4. Y Matrix

$$\Pr(s_p) = a_p \,/\, \Sigma_k a_k$$

Equation 1. Reaction Propensity

$$f(\tau) = \Sigma_k a_k * \exp(-\tau * \Sigma_k a_k)$$

Equation 2. Time Interval

II.2 Path Planning and Environment Model

The environment was modeled as a bounded 2D environment with dimensions of 40 by 40 meters. A map of the environment and obstacles is represented by a binary occupancy grid. The environment is divided into a matrix of 40 by 40 cells where each cell can take on the value of 0 or 1 to represent its occupancy state. A 0 represents that there is free space for a robot to move into, whereas a 1 represents a space occupied by an obstacle ("binaryOccupancyMap"). It should be noted that in this representation, the obstacles are convex. The map of the warehouse is shown in the figure below where the obstacles are the warehouse goods, delivered goods, and the black squares which represent structural columns. The green arrows represent the general flow of traffic for the robots to minimize collisions. Figure 5 shows the resulting binary occupancy grid in which black cells represent obstacles. A Probabilistic Roadmap (PRM) will be used to generate the paths. The current map will unlikely pose a problem due to a lack of narrow passageways which can be a source of problems for this method (Bera et al).

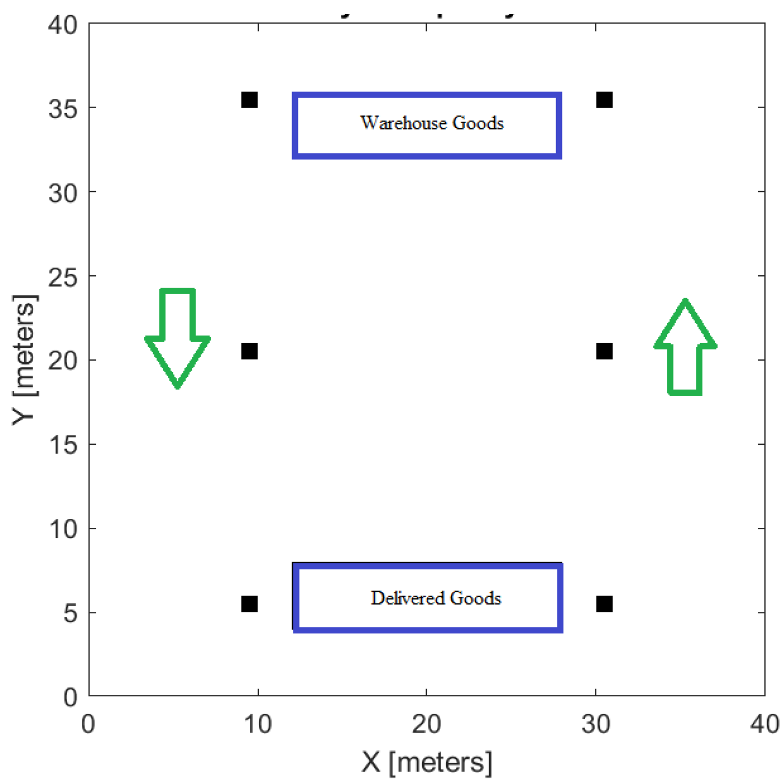

Figure 4. Map of Warehouse with Traffic Flow

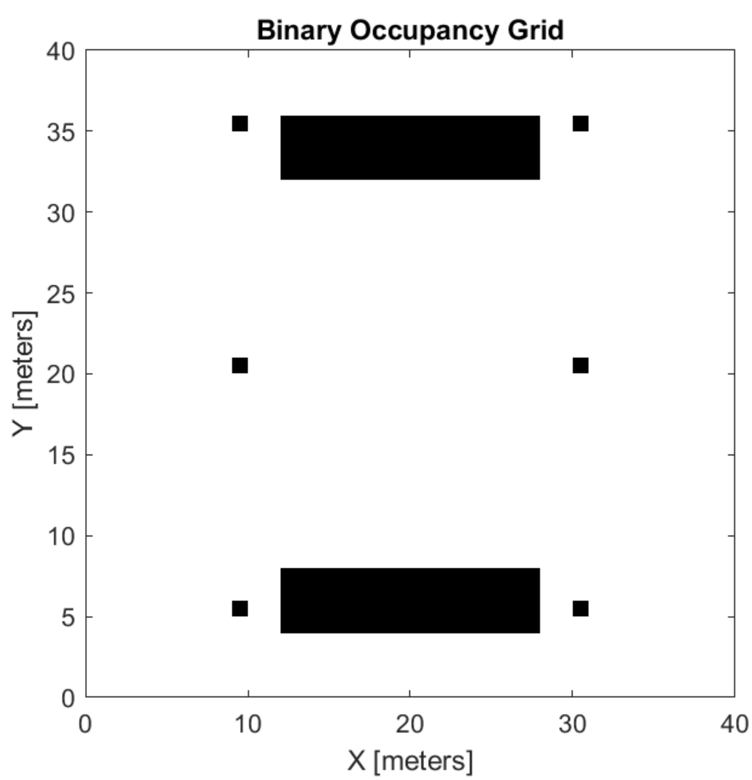

Figure 5. Binary Occupancy Grid of Warehouse

In the proposed multi-robot system, a central controller will utilize the binary occupancy grid to generate paths for each of the robots to complete the necessary tasks. Thus, every robot must be able to receive control commands from the central controller to execute the tasks.

II.3 Kinematic Model of a Two-wheel Differential Drive Robot

The multi-robot system consists of homogenous nonholonomic robots with two wheels with a separate motor for each wheel. The control inputs to each robot are its linear velocity *v* and angular velocity ω ("Differential-Drive Vehicle Model"). The robot moves in a 2D environment with coordinates x(t) and y(t) with heading orientation of θ, the angle between the robot heading and the positive x-axis in the counterclockwise direction. The kinematic equations are described in the following equations 3, 4, and 5:

$$\dot{x}(t) = v(t)\, cos[\theta(t)]$$

Equation 3. Robot velocity in the X-direction

$$\dot{y}(t) = v(t)\, sin[\theta(t)]$$

Equation 4. Robot velocity in the Y-direction

$$\dot{\theta}(t) = \omega$$

Equation 5. Robot angular velocity

From the control inputs of linear velocity and angular velocity, the velocities to drive the left and right wheels (v_L and v_R) for a robot with track width l are determined by the following equations (Inglett and Rodriguez-Seda):

$$v_L = v(t) + \frac{l}{2}\omega(t)$$

Equation 6. Velocity of robot's left wheel

$$v_R = v(t) - \frac{l}{2}\omega(t)$$

Equation 6. Velocity of robot's right wheel

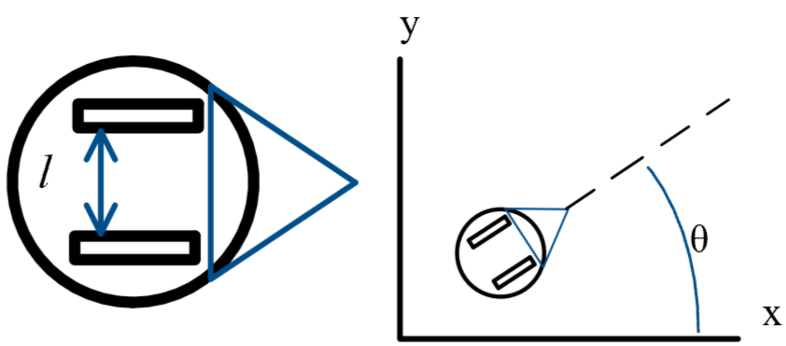

Figure 6. Robot Geometry and Defined Coordinates

The kinematic laws can be extended to a group of robots attached to a pallet. The robots would be configured as displayed in the figure below. In this configuration each robot becomes a single wheel for the pallet which is treated as a larger single unit. The robots on the same side will be given the same velocity commands to replicate a two-wheel differential drive robot.

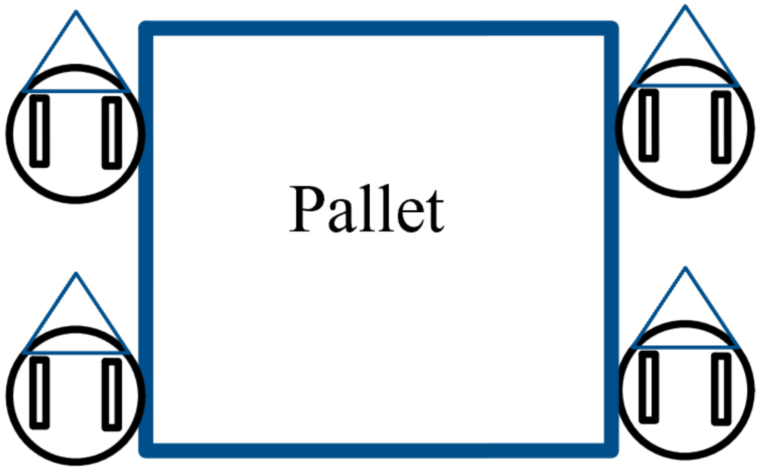

Figure 7. Pallet unit with four robots attached

III. Theoretical Analysis

III.1 Swarm Control Theory (Mesoscopic vs Macroscopic)

In this approach, the two models produce similar results but with a key distinction; a macroscopic model does not account for any of the randomness involved in the control of the robotic swarm. As such, the results are smooth and repeatable because they are not dependent on the random fluctuations of species populations. A mesoscopic model is used to capture some of these lost phenomena as it is based on the stochastic formulation of the system which accounts for the statistical correlations and fluctuations (Lachowicz 57). It should be noted that the macroscopic model discrepancies are expected to be greater with smaller populations such as the modeled system with a population of only 1000. Consequently, the mesoscopic model can be expected to have notable fluctuations and unsteadiness leading to a range of potential final results through different runs of the model.

III.2 Deficiency Zero Theorem

With the system goal of delivering all packages in the warehouse, it was important to confirm that this would be accomplished. This can be confirmed by examining the reversibility of the chemical reaction network. In the modeled chemical reaction network, it is seen that the system is neither reversible nor weakly reversible. This means that there is not a pathway from a reaction's product back to its reactants for all reactions. In the modeled system, this is the case for the idle small packages and large pallets. There does not exist a path to return back to this state. As such, it can be expected for these species to die out in the system thereby accomplishing the goal of the system. If, however, the model were modified to follow the chemical reaction network shown in figure 8, additional investigation would be required.

$$\mathbf{4a} + \mathbf{c} \underset{\beta_1}{\overset{\alpha_1}{\rightleftharpoons}} \mathbf{d} \qquad \mathbf{e} \underset{\beta_4}{\overset{\alpha_4}{\rightleftharpoons}} \mathbf{q} + \mathbf{a}$$

$$\mathbf{a} + \mathbf{b} \underset{\beta_2}{\overset{\alpha_2}{\rightleftharpoons}} \mathbf{e} \qquad \mathbf{10b} + \mathbf{a} \underset{\beta_5}{\overset{\alpha_5}{\rightleftharpoons}} \mathbf{c} + \mathbf{a}$$

$$\mathbf{d} \underset{\beta_3}{\overset{\alpha_3}{\rightleftharpoons}} \mathbf{w} + \mathbf{4a}$$

Figure 8. Modified Chemical Reaction Network

This system allows for robots to cancel transportation of objects as well as return delivered objects back to the storage area. While this could be physically allowed, these are situations which do not serve the intended purpose of the desired system. As this system is weakly-reversible, further investigation is required to understand the population behavior. For this, the Deficiency Zero Theorem can be used (Shi). This theorem states that for a system with mass action kinetics which is weakly reversible and has a deficiency of zero, then for any rate constants the system will have one positive steady state which is asymptotically stable. Deficiency, δ, is found as the number of complexes, C, minus the number of linkage classes, l, minus the network rank s. These values for the system are shown below in figure 9.

$$\mathbf{C = 10}$$
$$\mathbf{l = 5}$$
$$\mathbf{s = 5}$$
$$\boldsymbol{\delta} \mathbf{= C - l - s = 0}$$

Figure 9. Deficiency Numbers

As the example chemical reaction network is weakly reversible and has mass action kinetics as well as a deficiency of zero, it has a single asymptotically stable equilibrium with all positive entries.

III.3 Path Planning using Probabilistic Roadmap (PRM)

Path planning using Probabilistic Roadmap (PRM) has been proven to be more effective than both uninformed and informed planners in dynamic simulations (Bekris and Kavraki). Additionally, a useful property of this approach is probabilistic completeness which ensures that any random path can be generated when sampling is sufficiently large (Bera et al. 463). The probabilistic roadmap method works by randomly sampling N number of nodes in the free space of the binary occupancy grid. The algorithm then connects each node to every other node using straight lines with a maximum length of R. The lines represent potential paths that could be selected to connect the initial location A to the desired location B ("Path Planning"). The method can be easily proven as probabilistically complete. When N approaches infinity, all of the unoccupied space will be sampled by the nodes. Thus, as long as R is greater than 0, any direction will be on a potential path and a connected path from A to B can be found.

III.4 Path Following with a Two-wheeled Differential Drive Robot

Based on the generated path from the probabilistic roadmap, an agent or a group of agents can follow the path based on the path-following control laws derived for a two-wheeled differential drive robot ("Path Following"). The controller works by computing the linear and angular velocities of the agent. This could be done by measuring changes in the position and orientation of the agent based on the camera information. The controller would then compare the actual velocity with the desired linear velocity and adjust the control input to the robots accordingly. On the other hand, angular velocity is adjusted based on how far the robot is looking (Park and Kuipers). By increasing the distance that the controller looks ahead, the agent may follow a smoother trajectory at the cost of inaccuracies when the path is nonlinear. Inversely, if the controller does not look far enough, the trajectory might become unstable and oscillatory ("Pure Pursuit Controller"). Lastly, the controller is susceptible to becoming stuck in a loop where the turn radius is not small enough to reach the destination point. This can occur when the desired velocity is too large.

IV. Validation in Simulations

IV.1 Swarm Control Simulation

The macroscopic model was used first to model the system. The system discussed in section IV.1 was solved with MATLAB's ode45 producing the distribution in figure 10. It was seen that the robots can be expected to successfully deliver all packages as the number of idle packages and pallets reached zero while the number of delivered packages and pallets reached a maximum and remained steady. As the simulation continued, the available idle packages and pallets decreased causing the amount of packages and pallets in transport to decrease while more robots returned to their idle state.

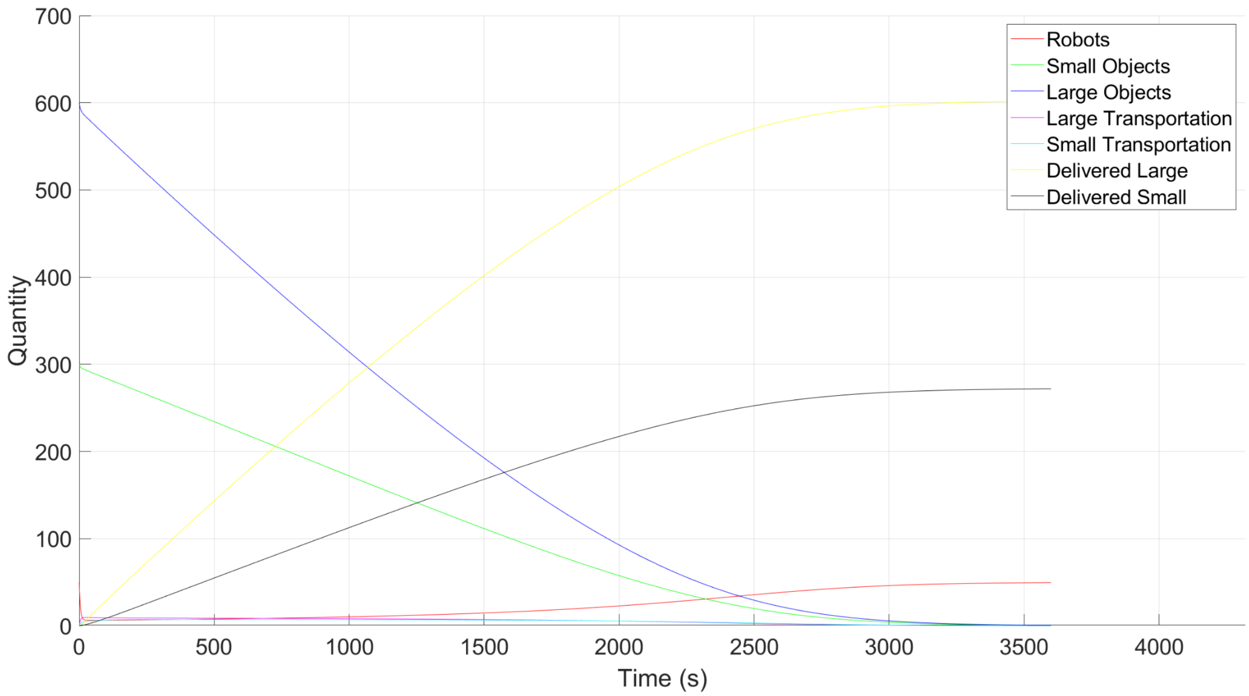

Figure 10. Swarm Control Macroscopic Simulation

As discussed in section III.1, the macroscopic model does not account for any of the randomness involved in the control of the robotic swarm so a mesoscopic model was used to capture some of the lost phenomena. Gillespie's direct method was implemented in MATLAB and used to simulate the model. Figure 11 depicts the mesoscopic model overlaid on top of the macroscopic model demonstrating the fluctuations the mesoscopic model had compared to the macroscopic model. Additionally, the mesoscopic model can be seen to capture the results of state fluctuations which the macroscopic model lacks. For example, in this simulation of the mesoscopic model, the robots are seen to disassemble more large pallets into small packages compared to the purely deterministic approach. In another run of the simulation shown in figure

12, the opposite is seen. Figure 13 presents a number of simulation runs overlaid to demonstrate the potential range of results possible with the mesoscopic model.

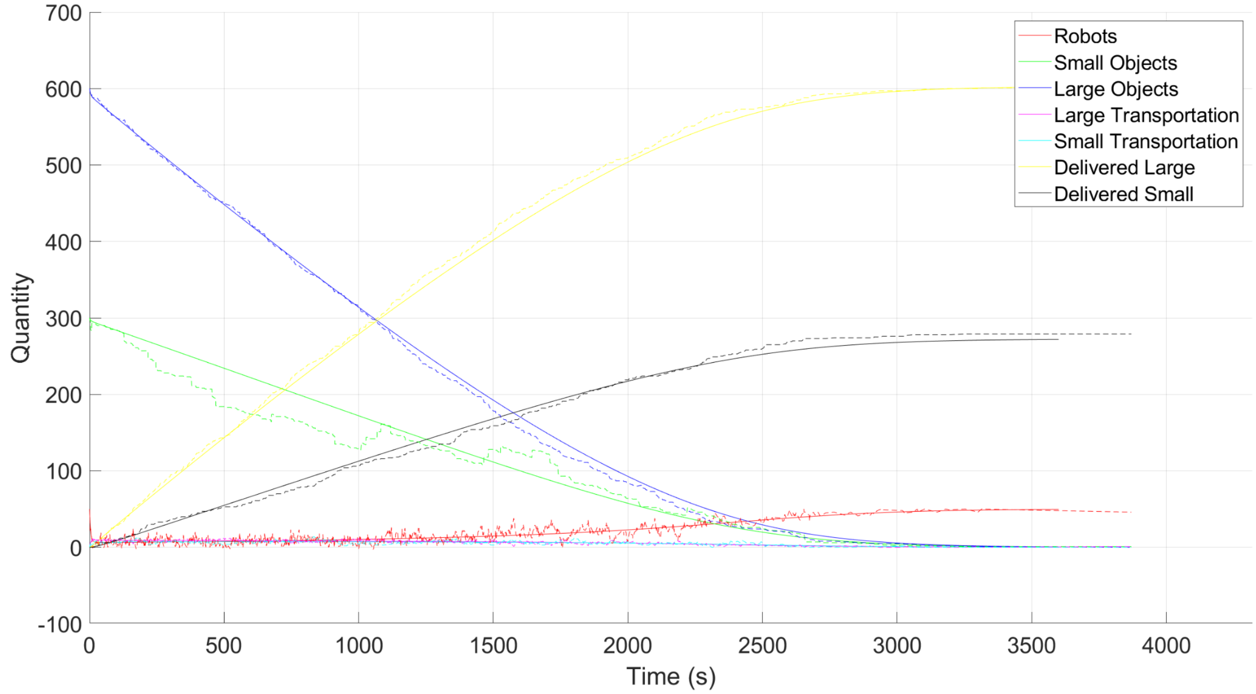

Figure 11. Swarm Control Model Simulation Two(Macroscopic Solid vs Mesoscopic Dashed)

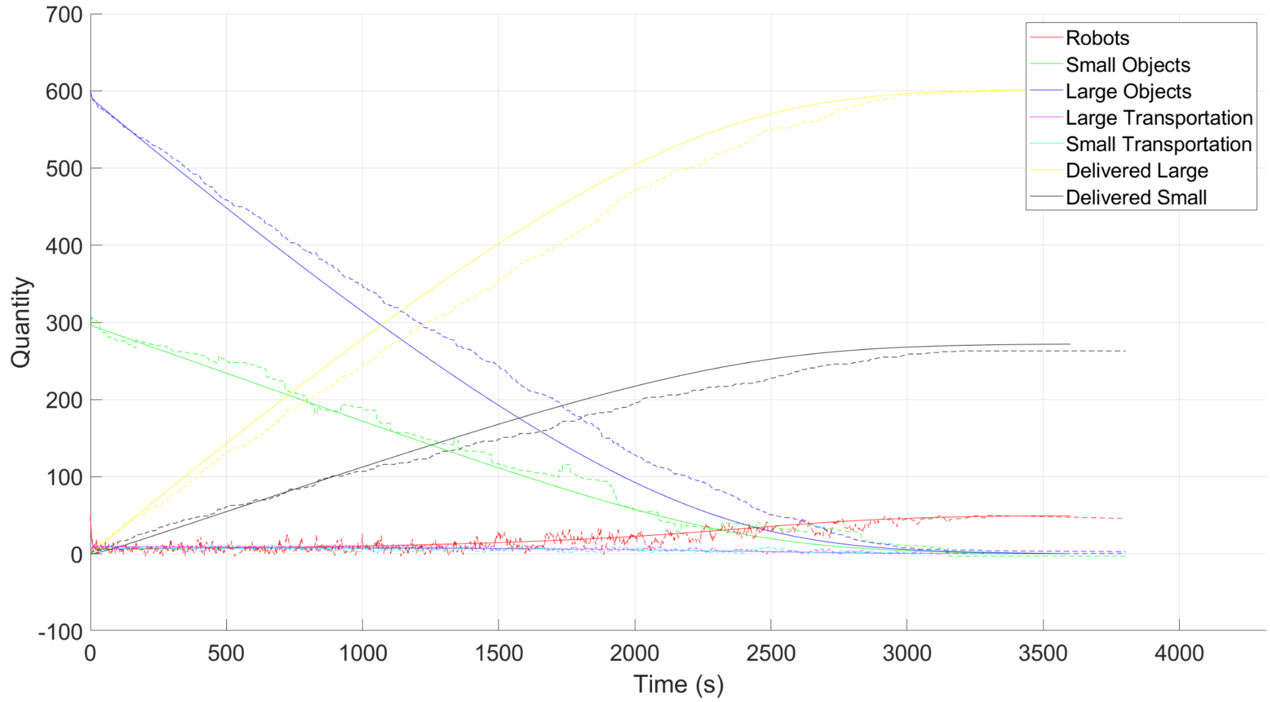

Figure 12. Swarm Control Model Simulation Two (Macroscopic Solid vs Mesoscopic Dashed)

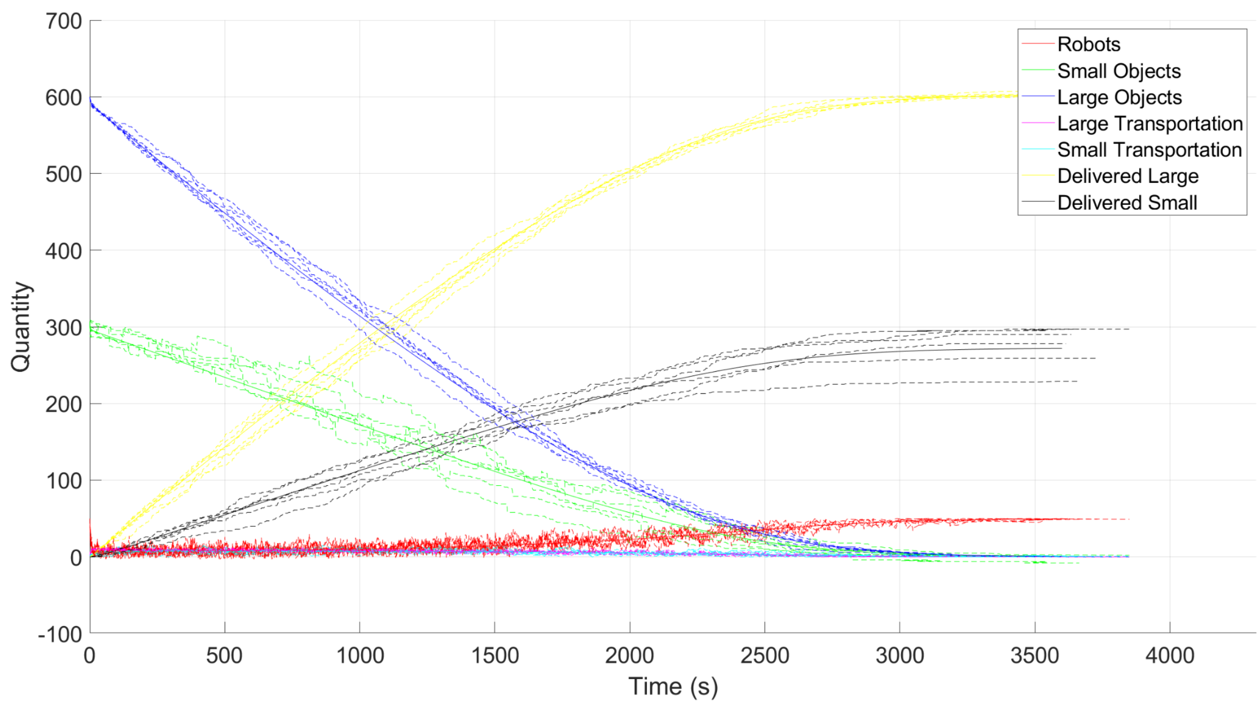

Figure 13. Series of Swarm Control Model Simulations (Macroscopic Solid vs Mesoscopic Dashed)

IV.2. Path Planning

For this simulation, consider a standard 48” by 40” pallet being transported by 4 robots from the warehouse goods to the delivered goods (from the top to the bottom of the map in figure 14). Thus, an additional zone of obstacles was added to the binary occupancy grid to ensure that the robots would move in a relatively counterclockwise direction from an aerial view of the warehouse.

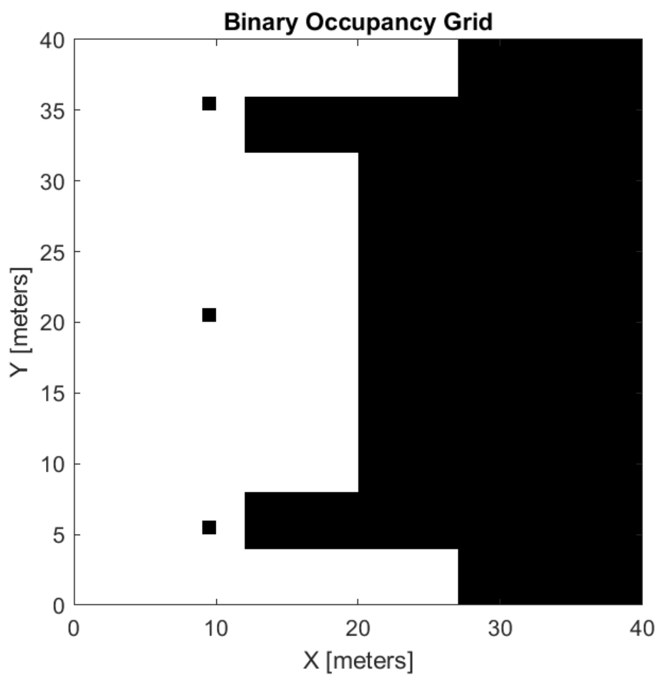

Figure 14.

To account for the size of the pallet, the map is “inflated.” In other words, every unoccupied cell that is in contact with an occupied cell will become occupied. The inflated map is shown in the figure below.

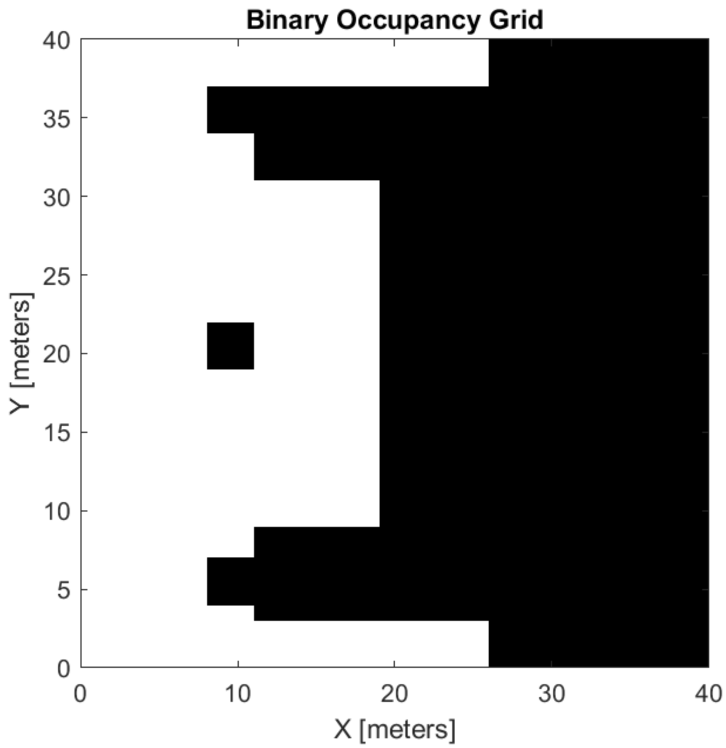

Figure 15. Inflated Binary Occupancy Grid

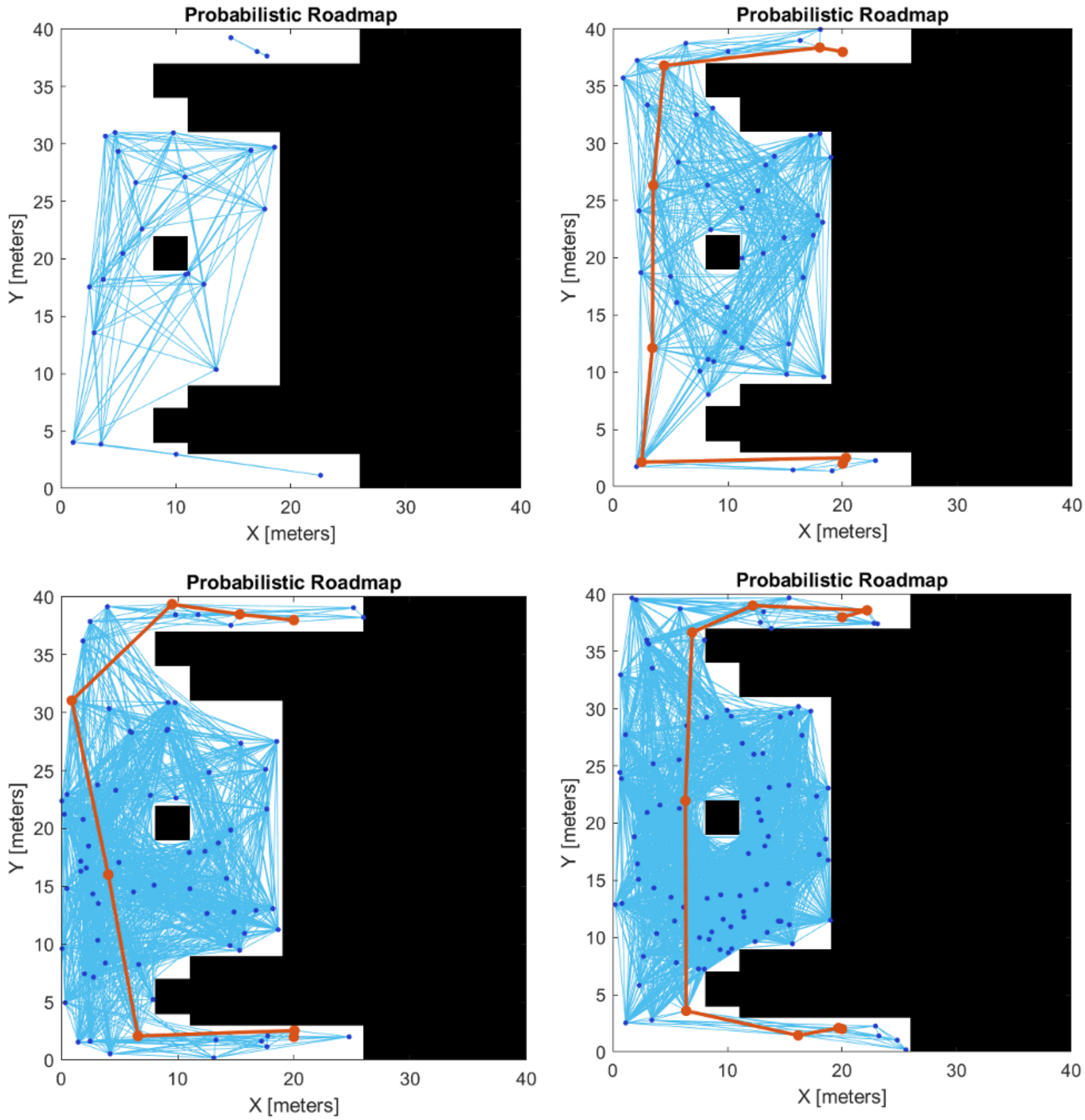

Figure 16. Probabilistic Roadmap for 25 (top left), 50 (top right), 75 (bottom left), and 100 Nodes (bottom right)

With a maximum connection distance of 20 meters and 25 nodes, the probabilistic roadmap cannot reliably generate a path from A to B. As seen in figure 16, the number of nodes allow for more paths to be created and improve the likelihood of a path that will connect points

A and B. In figure 16, a probabilistic roadmap with 25, 50, 75, and 100 nodes was generated. The current simulation uses a fairly standard sized warehouse and pallet and was able to quickly generate a reliable path with around 100 nodes with a max connection distance of 20 meters.

IV.3 Path Following

The current simulation will be a continuation of the scenario found in the previous path planning section. The path is defined by a set of waypoints which are indicated by the diamonds on the plot. From the simulations, the robot's performance is highly dependent on controller parameters such as the maximum desired velocity and look-ahead distance. When the desired velocity is too large, the oscillations in the controller are amplified, preventing the robot from remaining on the path. The look-ahead distance also greatly impacts the unit's ability to remain on path. The effects can be observed in the simulation.

V. References

Bekris, Kostas e, and Lydia e Kavraki. "Informed and Probabilistically Complete Search for Motion Planning under Differential Constraints." Rutgers University, Computer Science Department, Rice University, 2008, people.cs.rutgers.edu/~kb572/pubs/informed_planning_dynamics.pdf.

Bera, Titas, et al. "Analysis of Obstacle Based Probabilistic RoadMap Method Using Geometric Probability." IFAC Proceedings Volumes, vol. 47, no. 1, 2014, pp. 462–69. Crossref, doi:10.3182/20140313-3-in-3024.00245.

Berman, Spring. "MAE 598 MRS Berman - Aug 30 and Sept 1 2021." ASU Canvas, 2021, asu.instructure.com/courses/98516/files/36691879?module_item_id=6649566.

Berman, Spring. "MAE 598 MRS F21 HW2 Solutions." ASU Canvas, 2021, asu.instructure.com/courses/98516/files/folder/Assignments?preview=41807688.

Berman, Spring. "Notes on Modeling Multi-Robot Systems as Chemical Reaction Networks (CRNs)." ASU Canvas, 2021, asu.instructure.com/courses/98516/files/36004793?module_item_id=6497033.

"binaryOccupancyMap." MathWorks Help Center, The MathWorks Inc., 2015, https://www.mathworks.com/help/nav/ref/binaryoccupancymap.html.

"Differential-Drive Vehicle Model - MATLAB." MathWorks Help Center, The MathWorks Inc., 2019, www.mathworks.com/help/robotics/ref/differentialdrivekinematics.html.

Gillespie, Daniel T. "Exact Stochastic Simulation of Coupled Chemical Reactions." The Journal of Physical Chemistry, vol. 81, no. 25, 1977, pp. 2340–61. Crossref, doi:10.1021/j100540a008.

Inglett, J. Evan, and Erick J. Rodriguez-Seda. “Object Transportation by Cooperative Robots.” SoutheastCon 2017, 2017. Crossref, doi:10.1109/secon.2017.7925348.

Lachowicz, Mirosław. “Microscopic, Mesoscopic and Macroscopic Descriptions of Complex Systems.” Probabilistic Engineering Mechanics, vol. 26, no. 1, 2011, pp. 54–60. Crossref, doi:10.1016/j.probengmech.2010.06.007.

Lerman, Kristina, et al. “A Review of Probabilistic Macroscopic Models for Swarm Robotic Systems.” Swarm Robotics, 2005, pp. 143–52. Crossref, doi:10.1007/978-3-540-30552-1_12.

Park, Jong Jin, and Benjamin Kuipers. “A Smooth Control Law for Graceful Motion of Differential Wheeled Mobile Robots in 2D Environment.” 2011 IEEE International Conference on Robotics and Automation, 2011. Crossref, doi:10.1109/icra.2011.5980167.

“Path Following for a Differential Drive Robot.” MathWorks Help Center, The MathWorks Inc., 2021, https://www.mathworks.com/help/robotics/ug/path-following-for-differential-drive-robot.html.

“Path Planning in Environments of Different Complexity.” MathWorks Help Center, The MathWorks Inc., 2021, https://www.mathworks.com/help/robotics/ug/path-planning-in-environments-of-difference-complexity.html.

“Pure Pursuit Controller.” MathWorks Help Center, The MathWorks Inc., 2021, https://www.mathworks.com/help/robotics/ug/pure-pursuit-controller.html.

Shi, Zoey. “An Introduction to the Chemical Reaction Network Theory.” University of Washington, 5 June 2020, sites.math.washington.edu/~morrow/336_20/papers20/zoey.pdf.

Ullah, M., et al. “Deterministic Modelling and Stochastic Simulation of Biochemical Pathways Using MATLAB.” IEE Proceedings - Systems Biology, vol. 153, no. 2, 2006, p. 53. Crossref, doi:10.1049/ip-syb:20050064.

VI. Appendix

Appendix A1. Team Work Distribution

	Daniel Kim	Chester Szatkowski
I. Abstract	X	X
II.1 Swarm Control Model		X
II.2 Path Planning and Environmental Model	X	
II.3 Kinematic Model of a Two-wheel Differential Drive Robot	X	
III.1 Swarm Control Theory (Mesoscopic vs Macroscopic)		X
III.2 Deficiency Zero Theorem		X
III.3 Path Planning using Probabilistic Roadmap (PRM)	X	
III.4 Path Following with a Two-wheeled Differential Drive Robot	X	
IV.1 Swarm Control Simulation		X
IV.2 Path Planning	X	
IV.3 Path Following	X	
V. References	X	X
A1 Team Work Distribution		X
A2 Stochastic Simulation MATLAB Code		X
A3 Path Planning Simulation MATLAB Code	X	
A4 GitHub Link		X

Appendix A2. Stochastic Simulation MATLAB Code


```
%% MAE 598 Final Project
% Daniel Kim , Chester Szatkowski
% Fall 2021

%% Clearing
clear all;
clc;
%% ODE Numerical Solving
species = 7;
tspan = [0 3600]; % seconds
x0 = [0.05, % initial robot population fraction
    0.3, % initial small object population fraction
    0.6, % initial large object population fraction
    0, % initial large transportation population fraction
    0, % initial small transportation population fraction
    0, % initial delivered large object population fraction
    0]; % initial delivered small object population fraction

Ntot = 1000;

M = [4 0 1 0 0 4 0 1 1 1; 0 0 1 0 0 0 0 0 10 0; 1 0 0 0 0 0 0 0 0 1; 0 1 0 0 1 0 0 0 0 0;↙
0 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 0 0];


alpha1 = 0.08; % rate for robots working to move a large object into transportation phase
alpha2 = 0.064; % rate for robot working to move a small object into transportation phase
alpha3 = 0.032; % rate for large object to be delivered and 4 robots to be freed
alpha4 = 0.016; % rate for small object to be delivered and a robot to be freed
alpha5 = 0.0032; % rate for a robot to take 10 small objects to make a large object
beta = 0.00144; % rate for robot to disassemble a large object into 10 small objects

k = [alpha1, alpha2, alpha3, alpha4, alpha5, beta];
K = [alpha1, 0, 0, 0, 0, 0, 0, 0, 0, 0;
    -alpha1, 0, 0, 0, 0, 0, 0, 0, 0, 0;
    0, 0, alpha2, 0, 0, 0, 0, 0, 0, 0;
    0, 0, -alpha2, 0, 0, 0, 0, 0, 0, 0;
    0, 0, 0, 0, alpha3, 0, 0, 0, 0, 0;
    0, 0, 0, 0, -alpha3, 0, 0, 0, 0, 0;
    0, 0, 0, 0, 0, 0, alpha4, 0, 0, 0;
    0, 0, 0, 0, 0, 0, -alpha4, 0, 0, 0;
    0, 0, 0, 0, 0, 0, 0, 0, alpha5, -beta;
    0, 0, 0, 0, 0, 0, 0, 0, -alpha5, beta];

[t,x] = ode45(@(t,x) odefun(t,x,M,K), tspan, x0);
figure(1)
hold on
set(gca,'Fontsize',20);
grid on
plot(t,Ntot*x(:,1),'r'); %robots
plot(t,Ntot*x(:,2),'g'); % small objects
plot(t,Ntot*x(:,3),'b'); % large objects
```

```
plot(t,Ntot*x(:,4),'m'); % transporting large objects
plot(t,Ntot*x(:,5),'c'); % transporting small objects
plot(t,Ntot*x(:,6),'y'); % delivered large object
plot(t,Ntot*x(:,7),'k'); % delivered small object


%% Gillespie

m12 = M(:,2) - M(:,1); % reaction 1
m34 = M(:,4) - M(:,3); % reaction 2
m56 = M(:,6) - M(:,5); % reaction 3
m78 = M(:,8) - M(:,7); % reaction 4
m910 = M(:,10) - M(:,9); % reaction 5
m109 = -m910; % reaction 6

%Reaction vectors
R = [m12 m34 m56 m78 m910 m109]';

c1 = alpha1/Ntot; % reaction 1
c2 = alpha2/Ntot; % reaction 2
c3 = alpha3; % reaction 3
c4 = alpha4; % reaction 4
c5 = alpha5/Ntot; % reaction 5
c6 = beta/Ntot; % reaction 6


t = 0;
tfinal = 3600;
N = Ntot*x0';
tvec = t;
Nvec = N/Ntot;

while t < tfinal

    % Reaction propensities
    a(1) = c1*N(1)*N(3); % reaction 1
    a(2) = c2*N(1)*N(2); % reaction 2
    a(3) = c3*N(4); % reaction 3
    a(4) = c4*N(5); % reaction 4
    a(5) = c5*N(2)*N(1); % reaction 5
    a(6)= c6*N(3)*N(1);  % reaction 6

    asum = sum(a);
    j = min(find(rand<cumsum(a/asum))); % index of the next reaction
    tau = log(1/rand)/asum;
    N = N + R(j,:); % simulate the reaction occurring;
                    % update the integer population counts
    t = t + tau; % time of the next reaction
    tvec(end+1) = t;
    Nvec(end+1,:) = N/Ntot;
end
```

```
plot(tvec,Ntot*Nvec(:,1),'r','LineStyle','--'); %robots
plot(tvec,Ntot*Nvec(:,2),'g','LineStyle','--'); % small objects
plot(tvec,Ntot*Nvec(:,3),'b','LineStyle','--'); % large objects
plot(tvec,Ntot*Nvec(:,4),'m','LineStyle','--'); % transporting large objects
plot(tvec,Ntot*Nvec(:,5),'c','LineStyle','--'); % transporting small objects
plot(tvec,Ntot*Nvec(:,6),'y','LineStyle','--'); % delivered large object
plot(tvec,Ntot*Nvec(:,7),'k','LineStyle','--'); % delivered small object
xlabel('Time (s)');
ylabel('Quantity');
xlim([0 tfinal*1.2])
set(gcf, 'Position',  [100, 100, 1800, 900])

legend('Robots', 'Small Objects', 'Large Objects', 'Large Transportation', 'Small↙
Transportation', 'Delivered Large', 'Delivered Small');
%xlim([0 tfinal])
%}

function V = gma(D,k,L)
%
% Supplementary Matlab files for the paper:
%
% " Deterministic Modelling and Stochastic Simulation of Pathways using Matlab"
% by M.Ullah, H.Schmidt, K-H.Cho and O.Wolkenhauer
%
% This function implements the generalized law of mass action
%
% We would appreciate a citation if these files are used by others.

if nargin < 3
    L = -D.*(D < 0);
    i2 = L > 1;
else
    i2 = L>0 & L~=1 ;
end
i0 = L==0;
M1s = ones(size(k));
V = @Vfn;
    function xd = Vfn(t,x)
        X = x(:,M1s);
        X(i0) = 1;
        X(i2) = X(i2).^L(i2);
        xd = D*(k.*prod(X)).';
    end
end
function dxdt = odefun(t,x,M,K)
% x = [x(1); x(2); x(3); x(4); x(5); x(6); x(7)];
y = [x(1)*x(3); x(4); x(1)*x(2); x(5); x(4); x(1)*x(6); x(5); x(7)*x(1); x(2)*x(1); x(3)↙
*x(1)];

dxdt = -M*K*y;
```

Appendix A3. Path Planning Simulation MATLAB Code


```
%% Cite MATLAB code thingy
% https://www.mathworks.com/help/robotics/ug/path-planning-in-environments-of-difference-
complexity.html
% https://www.mathworks.com/help/robotics/ug/path-following-for-differential-drive-robot.
html

%% Initialize Video
%myVideo = VideoWriter('myVideoFile'); %open video file
%myVideo.FrameRate = 10;  %can adjust this, 5 - 10 works well for me
%open(myVideo)

%% Creating a map

warehouseMap = readmatrix('map.txt'); % Matrix where 1 represents an obstacle and 0
represents a free space

map = binaryOccupancyMap(warehouseMap,1); % 1 cell per meter
% show(map)

robotRadius = 0.2; % robots are assumed to be circles of radius 0.2 meters

mapInflated = copy(map);
inflate(mapInflated,robotRadius); % to account for robot's dimension and ensure no
collisions occur

% show(mapInflated)

prm = mobileRobotPRM; % define a path planner

prm.Map = mapInflated; % assign map to path planner
prm.NumNodes = 150; % number of randomly sampled nodes in free space
prm.ConnectionDistance = 10; % maximum line that connects any two nodes; increasing can
increase computation time

startLocation = [20, 38];
endLocation = [20, 2];

path = findpath(prm, startLocation, endLocation); % solve the path

show(prm) % display nodes, connections, and final path

%% Path Following for a two-wheeled differential-drive robot
% Robot will follow the path generated by the PRM in previous section

% initialize robot's path and initial orientation
robotInitialLocation = path(1,:);
robotGoal = path(end,:);

initialOrientation = 0; % (angle between robot heading and +x-axis (CCW)

robotCurrentPose = [robotInitialLocation initialOrientation]';
```

```
% Initialize the robot with kinematic equations for motion
% The robot's inputs are linear and angular velocities

% define robot parameters; wheel radius = 0.05m, track width = 0.25m, and
% inputs are speed and heading angle
robot = differentialDriveKinematics("TrackWidth", 0.25, "VehicleInputs",↙
"VehicleSpeedHeadingRate");

% plot desired path
% figure
% plot(path(:,1), path(:,2),'k--d')
% xlim([0 40])
% ylim([0 40])

% define path following controller

controller = controllerPurePursuit;
controller.Waypoints = path;
controller.DesiredLinearVelocity = .6;
controller.MaxAngularVelocity = 2;
controller.LookaheadDistance = .3;

goalRadius = 0.2;
distanceToGoal = norm(robotInitialLocation - robotGoal);

% Initialize the simulation loop
sampleTime = 0.1;
vizRate = rateControl(1/sampleTime);

% Initialize the figure
% figure

% Determine vehicle frame size to most closely represent vehicle with plotTransforms
frameSize = robot.TrackWidth/0.8;

while( distanceToGoal > goalRadius )

    % Compute the controller outputs, i.e., the inputs to the robot
    [v, omega] = controller(robotCurrentPose);

    % Get the robot's velocity using controller inputs
    vel = derivative(robot, robotCurrentPose, [v omega]);

    % Update the current pose
    robotCurrentPose = robotCurrentPose + vel*sampleTime;

    % Re-compute the distance to the goal
    distanceToGoal = norm(robotCurrentPose(1:2) - robotGoal(:));

    % Update the plot
```

```
    hold off

    % Plot path each instance so that it stays persistent while robot mesh
    % moves
    plot(path(:,1), path(:,2),"k--d")
    hold all

    % Plot the path of the robot as a set of transforms
    plotTrVec = [robotCurrentPose(1:2); 0];
    plotRot = axang2quat([0 0 1 robotCurrentPose(3)]);
    plotTransforms(plotTrVec', plotRot, "MeshFilePath", "groundvehicle.stl", "Parent",↙
gca, "View","2D", "FrameSize", frameSize);
    light;
    xlim([0 40])
    ylim([0 40])
    set(gcf, 'Position',  [100, 100, 1000, 850])
    %pause(0.01)
    %frame = getframe(gcf);
    %writeVideo(myVideo, frame);

    waitfor(vizRate);
end
%close(myVideo);
```

Appendix A4. GitHub Link

https://github.com/ChesterSzatkowski/Fall21-MAE598-Multi-Robot-Systems-CS-DK