Cameron Guest
Student ID: 201120096
Simulating robot swarms: collaborative transport

Project Design

SUMMARY OF PROPOSAL

Inspired by the behaviour of social animals the field of swarm robotics deals with the coordination of multi-robot systems and their collective behaviours. Using simple robots with simple behaviors a system utilising swarms should be designed to be robust and scalable. With this in mind the following constraints should be adhered to:

- Each robot should be autonomous, it will act independently.
- No robot should have any access to a centralized control system.
- Each robot should only have local sensing and communication capabilities.
- Robots should act cooperatively.

The aim of this project is to simulate the occlusion based cooperative transport algorithm. The formulation of this problem is found in [1] and is as follows:

" A bounded environment contains a convex-shaped object, a goal, and a number of robots. The environment is otherwise free of obstacles. The aim is that the robots, which are initially placed in arbitrary locations, push the object to the goal."

Additionally the following assumptions are made:

- The object and goal are recognisable by the robots.
- The dimension of the object is large enough to occlude the robot's' perception of the goal when they are behind it.
- The robots can perceive the goal from any point within the environment, unless it is occluded by the object.

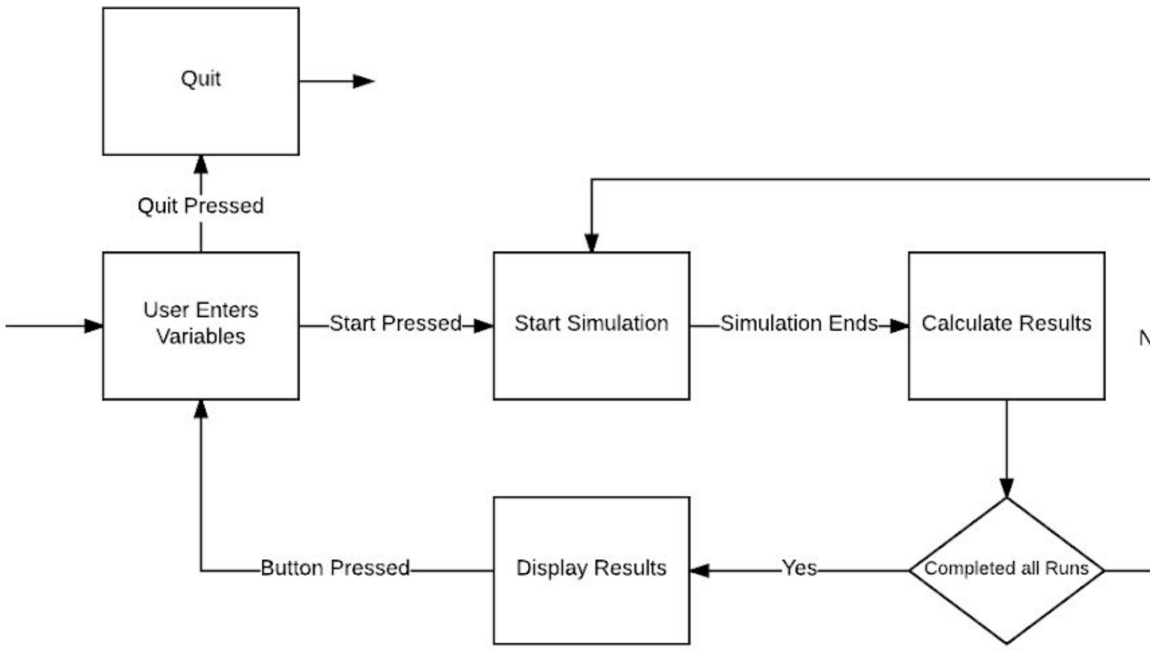
With the foundation of the problem laid out this project should also aim to include:

- A comparison of the algorithm being simulated to it being run in the real-world.
- The simulation should allow users to change parameters such as the number of robots and the size of the arena.
- An evaluation of how each parameter affects the running time of the algorithm.

My current research into the area surrounding this project has shown how the design and the analysis of systems using robot swarms is carried out. This includes methods such as micro and macroscopic analysis methods and behaviour based design methodologies.

Other transportation algorithms have been looked at as well such as the group transport method found in [3] and the pheromone communication based transport methods.

PROGRAM FLOW OF CONTROL



The flowchart above visualizes the behavior of the program through its execution. The first step is to define the variables the simulation will run with, the current parameters planned to be changeable are:

- The number of robots.
- The arena height.
- The arena width.
- The distance of the object from the goal.
- The speed of the simulation.
- Number of runs.

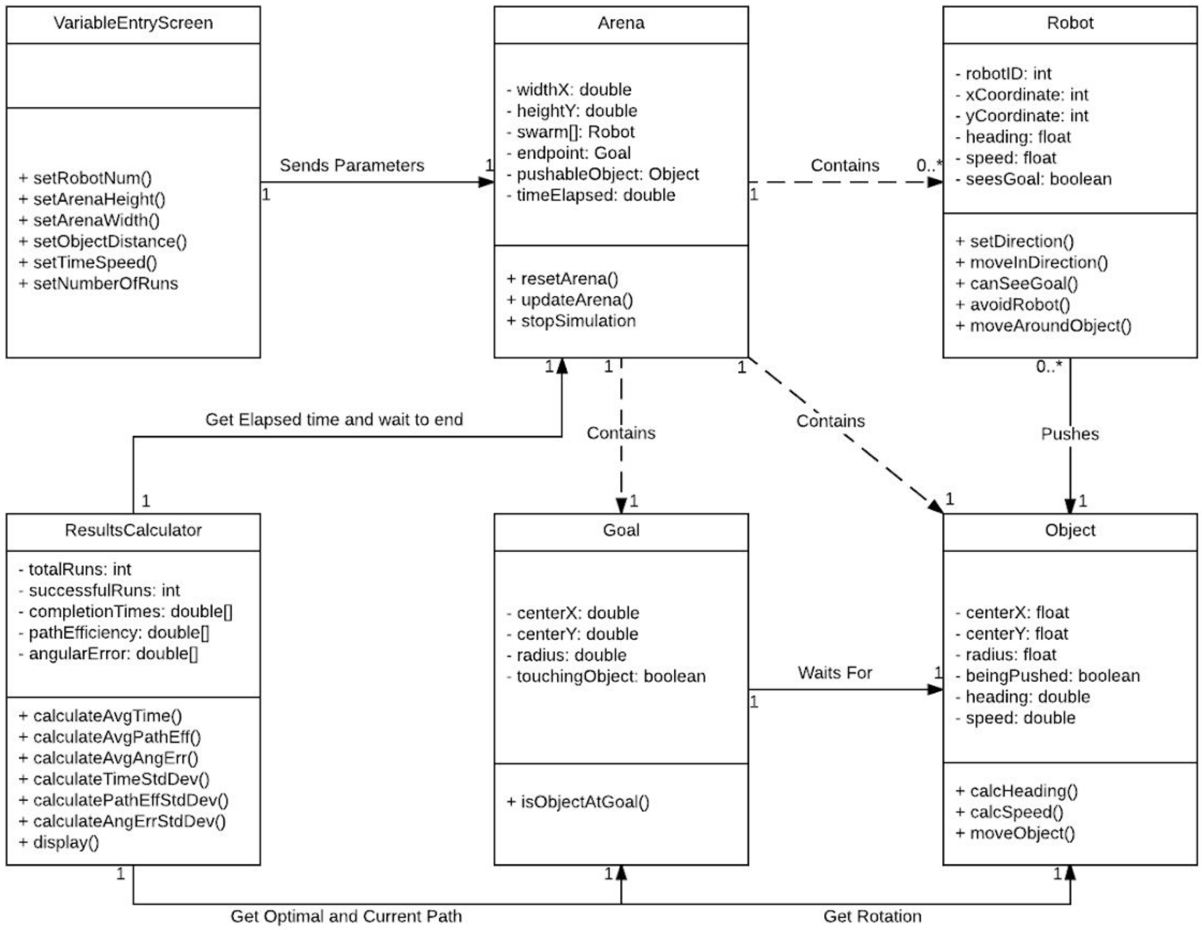
Once these have been defined and validated upon pressing the start button the simulation will begin running and has three stop conditions; the object reaches the goal, the object gets stuck in a corner or if the elapsed time exceeds 15 minutes. When any condition is met the program will calculate the:

- The number of successful trials.
- The mean and standard deviation of completion times.
- Mean and Standard Deviation of the Path Efficiency - a comparison between the path taken and the ideal straight line path .
- Mean and Standard Deviation of the Angular Error - the amount of unnecessary rotation applied to the object.

The formulas used in the calculations for path efficiency and angular error are specified in [1]. Once all runs have been completed the program will display the results and wait for a button press before returning to the starting screen.

CLASS DIAGRAM

With a general overview of how the program should run a more detailed diagram can be drawn describing how each individual class behaves.



There are currently six classes identified for use in the program a brief summary of each class is given below:

- The arena class represents the area in which the simulation will take place. It contains the robot, goal and object classes and is responsible for drawing the current state of the algorithm to the screen.
- The goal class is an area in the arena which waits for the object to touch it. Once this happens this run of the simulation is considered complete and the simulation is either stopped to allow the results to be shown or restart until the required number of runs is completed.
- The object class represents the item which is to be moved by the robots. It contains methods which calculate its trajectory based on how many robots are pushing it and from which directions.

- The robot class is a singular robot in the swarm. It looks to see if it has line of sight towards the goal, if so it travels around the object for a set amount of time, if not it will head in the direction of the goal pushing the object on the way.
- The variable entry screen is the window the program opens to and sets the parameters which the simulation will run with.
- The results calculator takes variables from all entities in the arena and uses them to calculate the simulation results.

PSEUDOCODE

OCCLUSION BASED TRANSPORT METHOD

The occlusion based transport method is used by the robots to push the object to the goal. Whenever a robot doesn't have line of sight with the goal we can assume that as there is nothing else in the arena, the object is occluding the goal. When this is the case we simply head towards the goal until we once again have line of sight.

The algorithm is as follows:

1. Find The Object
2. Approach The Object
3. If Goal is occluded
    a. Move towards goal pushing Object on the way
    b. Repeat A until Object no longer occluded, go to 4.

4. Move around the object and go to 1.

CALCULATING PUSHABLE OBJECTS DIRECTION AND SPEED

The velocity and heading of The Object or a Robot can be expressed in the form of a vector. Using trigonometry we can split this vector into separate x and y components, more easily allowing us to update the position of an entity.

1. For each robot pushing object
    a. Get heading and force applied.
    b. Separate the vector into x and y components.
        i. X = VcosH
        ii. Y = VsinH
        iii. Where V is the velocity and H is the robot's heading.

2. Add all X and Y values to get force and direction of object.

3. Convert back into vector $F = (X^2 + Y^2)$, $H = \tan^{-1}(y / x)$

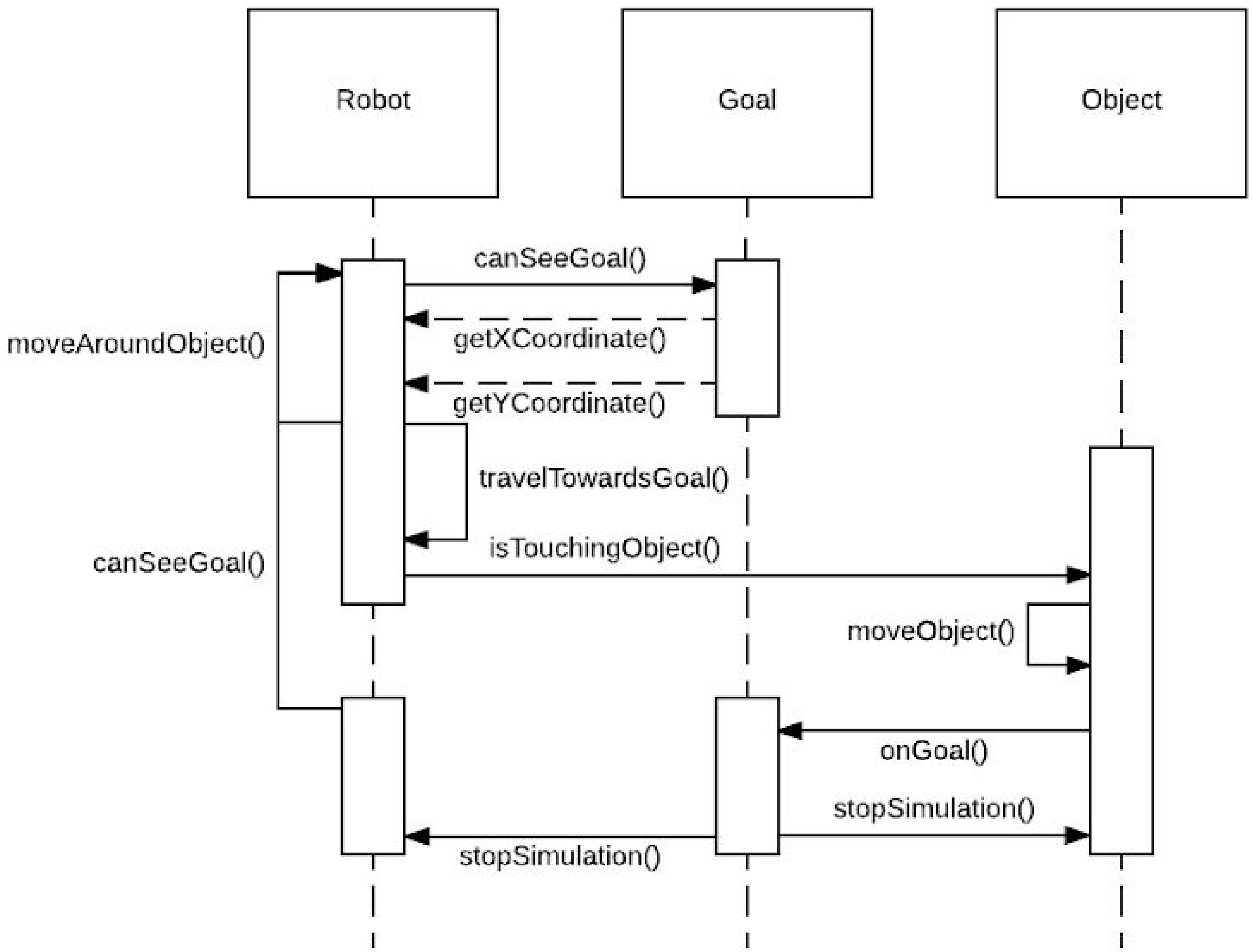
CALCULATING THE PATH EFFICIENCY

The path efficiency is one of the factors used to judge how well the algorithm is working. It's a measure of the most optimal path: a straight line, against the path actually taken by the robots. The formula in [1] defines the path efficiency as **'PE = S_{MIN}/S'.**

In order to use this equation we will have to take multiple readings of the pushable objects position and sum their distances.

INTERACTION CHART

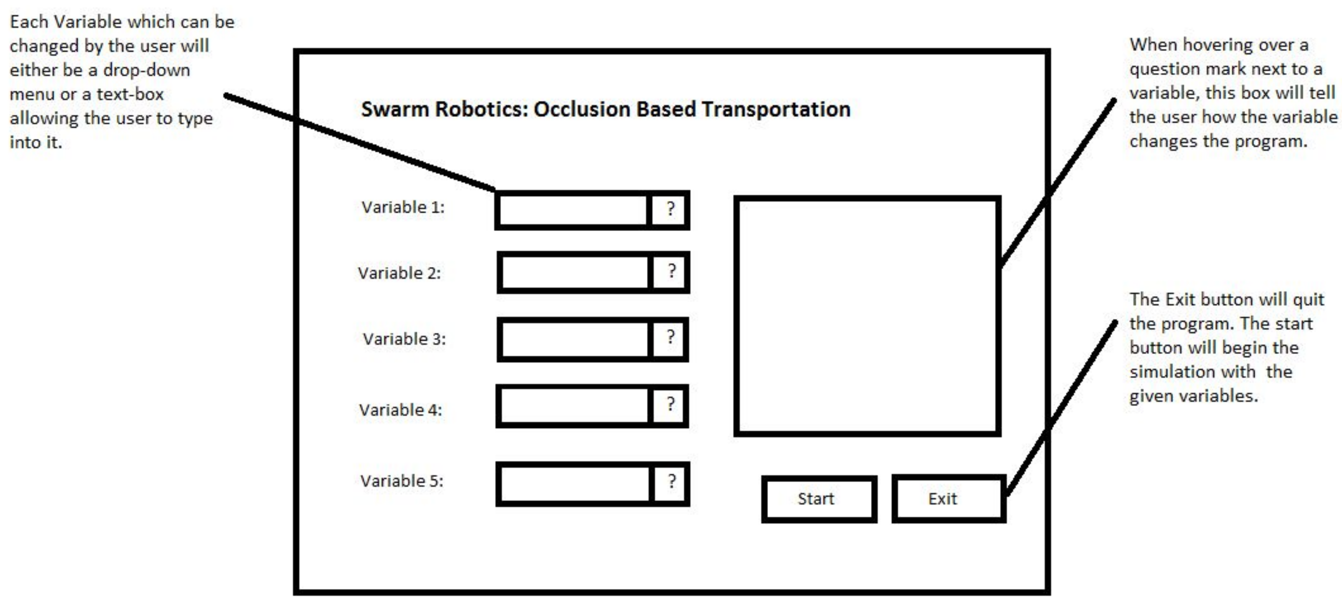
This interaction chart describes the behaviour between the three entities within the arena. The robot is on a looping behaviour, everytime it sees the object it will move around it until it can't see the goal anymore. The interactions will only cease once the onGoal method returns true otherwise these interactions will continue looping until one of the exit conditions are met.



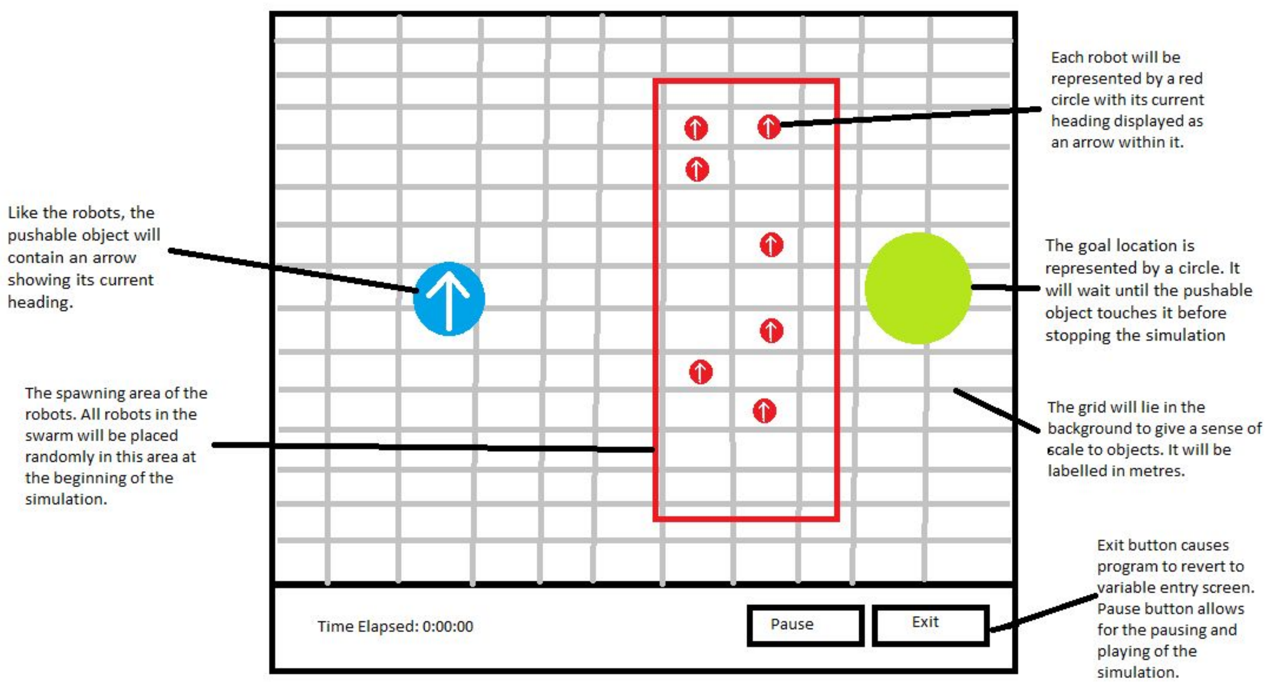
Note that all objects already exist when these interactions start however before any interactions can begin the arena must first add each item to itself as it will keep track of where each entity is.

INTERFACE DESIGN

There are currently two planned screens for the simulation; the first is the screen where the user can enter the variables they wish to run the simulation with.



The second is the screen in which the simulation plays out.



Finally is the screen which appears after the simulation has finished a run.

EVALUATION DESIGN

As mentioned in my specification document there will be three forms of evaluation:

Experimental:

Scalability is an important factor to consider when designing a system using swarms, it is a measure between the the growth of the complexity of the problem in relation to the problems size. For this problem we will use the running time as a measure of complexity.

To define the size of the problem we will use the two variables - number of robots and the distance of the object from the goal. As described in [2]:

“An algorithm A is scalable if there exists a constant c > 0 such that: $\mathbf{scalability_A(n) = O(log^c\ n)}$**.”**

Therefore in order to satisfy our scalability condition the algorithm should have a complexity of at the most O(n log n). This can be tested by increasing the size of our chosen variables by the same factor e.g. if the number of robots doubles, the distance from the goal should also double. The average time taken is then recorded and the simulation is run again with increased parameters.

Comparison:

As mentioned in the summary of proposal section, a comparison between the simulated algorithm to it's real-world implementation is to be included. To be consistent with [1], the performance of the algorithm will be measured using the following criteria.

- The number of successful trials.
- The mean and standard deviation of completion times.
- Mean and Standard Deviation of the Path Efficiency - a comparison between the path taken and the ideal straight line path .
- Mean and Standard Deviation of the Angular Error - the amount of unnecessary rotation applied to the object.

We must also make sure that the initialization of the arena is the same. Any large discrepancies between the results will prompt a further investigation into what could've caused such a change.

Self:

A self evaluation on the project will be carried out to identify the strengths and weaknesses on parts of the project that would be impractical to analyse mathematically. The current plan is to perform self-analysis based on the following criteria:

- Have the project objectives been met satisfactorily .
- What unforeseen changes had to made to the program and why.
- What changes could have been made to the project to improve its effectiveness.
- What challenges were faced and how were they overcome.

EXTENSIONS

This design outlines the core components of this system which should be developed in order to consider this project a success. However if given enough time the following extensions to the program may be considered:

- Introduction of obstacles to the arena. Described in [1] is a method for navigating around static obstacles placed within the arena. Using a moving robot as a temporary goal the swarm can move the selected object around obstacles using the temporary goal robot as a guide.
- Multiple objects in the arena. Currently the swarm has to move a single object to the goal however the introduction of multiple objects could lead to a deadlock in the system as there are not enough robots to push each object at the same time. To solve this problem an extension to the algorithm will be required.
- Creation of an automatic grapher. Whilst the program is running we can measure variables such as time and distance of the object from the goal and display this in a graphical format once the object has reached the goal. This would provide an easier way to view how each variable affects the running time of the algorithm.

ETHICAL USE OF DATA

All data being gathered by the simulation is synthetic and does not use or store any persons data. The only outside source of data in this project are the results printed in [1], however this is a published journal available for free use and the data here is also synthetic. For these reasons I see no ethical problems with this project.

BIBLIOGRAPHY

[1] J. Chen, M. Gauci, W. Li, A. Kolling, R. Groß

Occlusion-Based Cooperative Transport with a Swarm of Miniature Mobile Robots

EEE Transactions on Robotics, Volume: 31 Issue: 2, 2015

[2] S.-H. Teng. Scalable Algorithms for Data and Network Analysis. Foundations and Trends

in Theoretical Computer Science, vol. 12, no. 1-2, pp. 1–274, 2016.

[3] D. Milner

Swarm Robotics Algorithms: A Survey

2007

REVIEW AGAINST PLAN

So far no major changes to the plan have been made, the contingency time added to the Gantt chart has offset the minor delays caused by other projects.