# Distributed Mapping

## Abstract

The objective of this project is to get 100% and the 4% EC's and we will achieve a very high mark by building a distributed system in which the peers will work together to map out the area they are in. Each node in the network is a mobile robot that will be surveying the area it is placed in. It will follow a gossip protocol communicate its findings with other nodes, that is only when they come within a certain range of the robot.. The robots will exchange their maps, resolve any conflicts they view with the maps they received and its local one, and coordinate with each other to explore undiscovered areas. The robots will stop surveying the area when their battery power goes below a certain value, this will resemble real life situation because a robot would eventually to get its power source recharged.

## High Level Overview

The aim of this project is to implement a distributed system where the nodes work together to map out the area they are placed in. The topology will be time-dependent, that is a nodes will only be connected to each other if they are within a certain range of each other, as the nodes are mobile this would lead to a different network as time progresses. This will not include a centralized server. Each node in this network is a robot assigned an ID that will be a completing task which is the path it will explore and recording whether there are walls or free-space along it's path. If during it journey it comes into contact with other robots it will stop its motion, connect with these robots within its communication radius and will broadcast it's map to them and the other robots will do the same.

Upon receiving the maps the robot will merge it's local map with each map given by it's fellow robots. It will check if there are any conflicts, that is Robot A recorded that there is a wall at coordinate (2, 7) but Robot B recorded that there wasn't a wall there, here the latest recording will take precedence. Since each robot follows this procedure in theory they should have consistent maps. Upon map exchange each robot will compete in computing the next task to do and who ever solves computes this the fastest will assign the tasks for the other robots. Upon task assignment they will continue exploring and repeat this procedure if they bump into other robots. Emulating a real-life situation the robots will eventually stop exploring when their energy goes below a certain value. That they will be assigned a certain initial energy (to reflect battery power) and it will change inversely to the steps taken.

A robot can fail at any point during its journey and our system will be equipped to deal with these. All the information of the robots local map and its current task will be stored on disk so if the robot re-connects it will be able to pick up it journey where it left off.
When starting its journey the robot will have no prior knowledge of the space it is about to map, if there are other robots in its allowed communication radius at this time they will communicate with each other and decide on an agreed origin and note down which robot shares this coordinate system. If there is a robot starting alone at some other time it will maintain a local coordinate system until it reaches another robot with a prior starting time upon which it will do a coordinate transformation to be on the same page as the robot that started earlier. The goal is to have one common coordinate system but still have the robots unaware of the area they are about to survey at the start.

# Low Level Overview

### Basic Inter-Robot Communication

Bluetooth would be used for inter-robot communications because it is faster and has more readily available hardware and firmware to interface with. However, since the Bluetooth has a good range of up to 100m(outdoor), we need to implement "soft range threshold" which means that while the two robots are within the range of Bluetooth discovery, they will exchange the their location coordinates; if their distance is less than the maximum distance, let's say 2 m then they will "pretend" that they do not see each other.

### Task Allocation Algorithm

This algorithm will basically look at the current map it is given and the number of robots needed to assign tasks to and find the coordinates that haven't been traversed in this map. After maps are successfully merged, will individually calculate a list of potential tasks to do and flood their decision to each others. If there are conflicting choices, whoever issued first takes the task.

### Map Merging

Our aim is to obtain an uniform and most updated map for every robot in the proximity. Therefore, they will broadcast their own map to each other and merge the newly received maps with the current map in their memory. They will keep broadcasting until the received map is identical to its own map. If conflicts arises, keep the mapping with the latest timestamp assuming each obstacles/wall on the mappings is timestamped. In this project, we will assume that obstacles positions are fixed (chairs don't move, doors don't close).

#### Map Merging Conflict Resolving Protocol

i. Upon exchange each robot will check for any conflicts between its current map and the one it received. A conflict is defined as a coordinate (spot) being labeled as a wall or free space. To resolve this, each robot will take the lastest measurement as the right one.

**Stop Condition**

At the beginning of the program, each robot will be assigned a certain amount of energy, when the energy level dips under a certain threshold, the robot will then stop roaming.

**Handling Dead Robot**

If one robot disconnects while exchanging information with the group in the proximity, rest of the robots will restart the map exchange and merging. The disconnected robot itself will revert its map to the last successfully merged version and start exploring on its own.

**New / Revived Robot**

If new robots joins the network or recovered from any form of system crash, it will firstly try to locate on the file system to see if there are any log files about its last interrupted exploration(Null for a new robot) and resume its exploration based on the log file.

**Server Task**

Currently ,all raspberry pi robots are expected to connect to the server(on Azure) with WIFI and constantly upload their current maps. Server will be only responsible for displaying robots' local map and will not facilitate any communication to or between robots .

# Timeline

| | |
|---|---|
| March 2nd | Initial Project Proposal |
| March 5th | Gather all hardware, figure out communication protocols |
| March 9th | Final Project Proposal |
| March 14th | Set up development environment |
| March 28 th | Finish up all low level protocols |
| March 31st | Finish debugging |
| April 4th | Prepare for demo |

# Additional Features

**Fully automated robot with wheels:**

Due to time and money constraints, it will be quite challenging to finish autonomous robot with navigation capabilities before April. We will try our best to finish the bulk of our distribution logic before the end of March to be able to make autonomous robots.

**Gas/Energy Station**

When a robot roams around the map it would spend its gas/battery level. There will be designated area that will replenish a robot's battery level to some extent enabling the robot to roam longer.