Distributed Mapping

Divya Ratnasami (w1v8)
James Park (e2d9)
Beichen Zheng (j2y8)
Howard Zhou (q6c9)
Haokun Chen (t1k0b)

Abstract

The objective of this project is to build 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 the robot starts its journey it will be initialized with it's starting coordinate so that all robots share the same coordinate system.

Low Level Overview

Robot States:

The robot can be in multiple states and this influences the actions it can perform.
A robot can be in the roaming state, busy state, waiting state, or joining state.

1. Roaming State: It is in the roaming state if there are no other robots within its communication radius. In this state it is mobile and it will be completing its given path.
2. A robot can be in the joining state if there are robots within its communication radius. In this state, the robot will stop its motion (therefore its task). It will exchange it's IPs with the robots in its proximity to form a network and confirm that all the neighbours within the network are consistent.
3. After this the robot will be in the busy state, and will follow the map merging protocol. When it is in this state it will refuse connections from new robots that want to join it.
4. The other state a robot can be is in the waiting state, here the robot is within the communication radius of a robot in a busy state. Here it will still stop its motion and continue to poll the busy robot until the busy robot comes out of the busy state, that is the robot in the busy state will return it's status to the robot in the waiting state. If the robot in the busy state dies it will obviously not be able to respond back to the waiting robot so this robot will stop waiting and go into the roaming state.

Basic Inter-Robot Communication:

After further investigations since the last office hour, we planned to use BlueTooth to exchange locations and IP between robots to decide if they should start communicate. Once robots are in the same proximity, they will start merge maps and assign tasks through WIFI. The reason we proposed to use WIFI for more heavy load communication is because the existing firmware of NFC and Bluetooth on Raspberry Pi are lacks of necessary functionalities which is summarized in the table below. Besides, we also provide an alternative to WIFI which is ad-hoc topology of localized wifi net. Please kindly advise if our primary approach is acceptable (Bluetooth for Bootstrap and Wifi with WWW connection for major communications)

	Pros	Cons	Planned Usage
NFC	Limit the communication	-Short Range(4-cm)	No long considered

	range by design	-Slow (1 kbytes) -Easily interrupted when multiple devices are in proximity	
BlueTooth	-Better range and speed than NFC	- Master-Slave pairing limits communication between two devices only per time. -Required to implement Go-Wrapper to existing Python library otherwise majority of the program will be in Python	Broadcasting the location and IP of the robot to its proximity.
Wifi	-flexibility, speed, access to TCP/IP stacks and other golang library	-Dependence on WIFI infrastructure	Once IPs are exchanged through bluetooth, robots will use wifi to merge maps and collaborate on task assignment.
Ad Hoc Wifi	-Independent from WIFI Infrastructure	-Need to wrap C firmware library -Static IP Address Required -Depends on topology, an access point may need to be configured	Alternative to Wifi

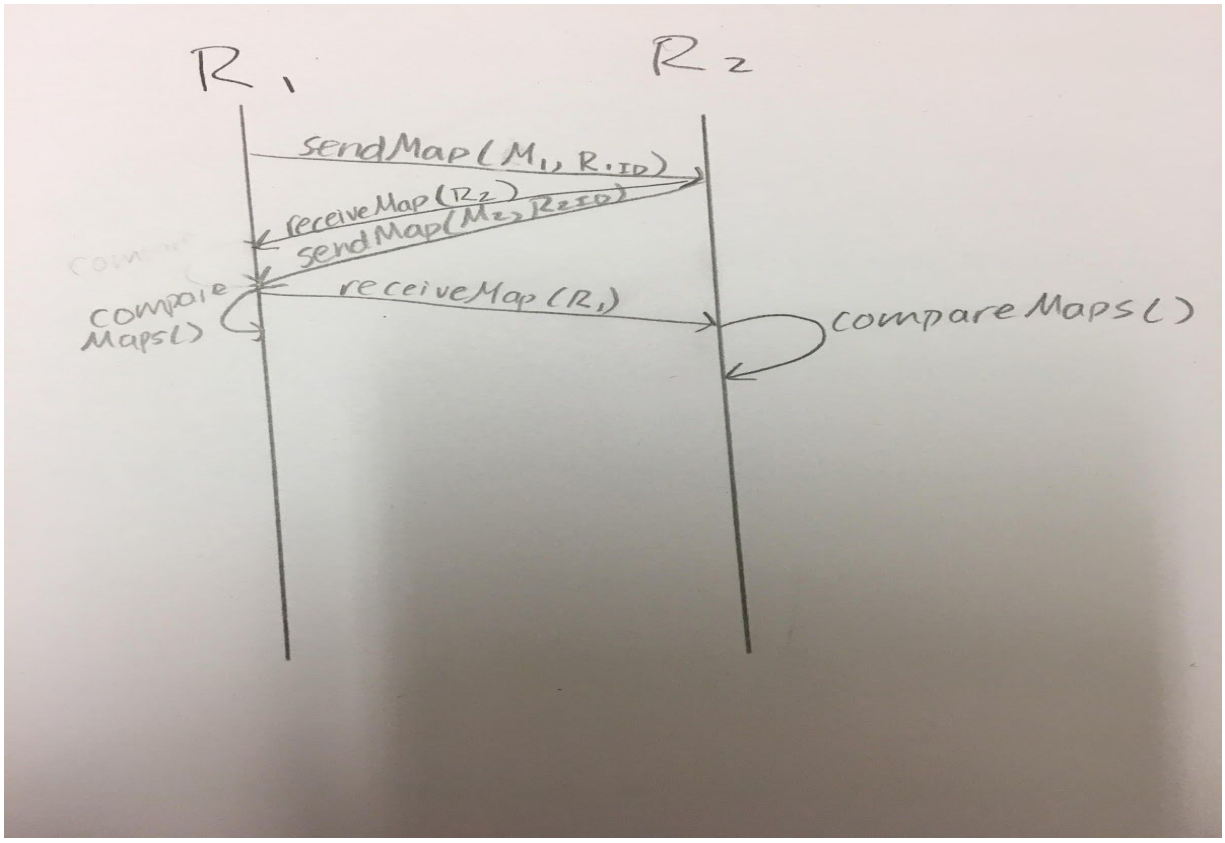
Determine Robot Physical Coordinate

Due to the poor accuracy of civil GPS(4-7m), we need alternative ways to determine the physical location of the robots. One proposed method is that all robots should start at the same point, the origin, and record down the number of rotations of their wheels to calculate its current location in reference to the the original point. However, such approach has significant cumulative errors due to mechanical inaccuracy. Therefore, we propose that if it is allowed to put some reference points such as a beacon or special object at the known location of the room to be explored. Whenever the robot pass through the reference point, it will recalibrate its coordinate system based on the beacon's location.

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. Based on this information it will figure out the coordinates that haven't been traversed and compute paths for each robot to take.

Map merging procedure



Map Merging

Our aim is to obtain an uniform and most updated map for every robot in the proximity. The robots will broadcast a map package which will consist of the Robots ID, the robots map, and a end tag which as its name suggest will be placed at the end of the package. A robot sends back an acknowledgment that it received the robots map if it received end tag. Only when the robot which sent it’s the to the network. Therefore, they will broadcast their own map to each other and merge the newly received maps with the current map in their memory. 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 => Beichen
March 9th	Final Project Proposal => Beichen
March 14th	Set up development environment => Haokun, Howard
March 28 th	Finish up all low level protocols => James, Divya
March 31st	Finish debugging => BeiChen, Haokun, Divya, Beichen
April 4th	Prepare for demo => BeiChen, Haokun, Divya, Beichen

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.