Detecting the DOA for Robots with Wireless Antennas

Jiejie Wei*, Bumsik Kim† and Zhe Yan†
Department of ECE, Stony Brook University
NY,Stony Brook, 11790
Email: *jiejie.wei@stonybrook.edu, † bumsik.kim@stonybrook.edu‡ zhe.yan@stonybrook.edu

***Abstract*—Our project is to develop algorithms and protocols for the localization of sensors and mobile robots. The main problem is to find the direction and distance of the robot. We will use Multiple Signal Classification(MUSIC) algorithm to determine the direction of arrival(DOA) of robots, and we will estimate the distance between each of the robots based on the path loss of our received signal strength. [1] [2]**

I. Introduction

A. Background

There are many robots created to improve quality of our life, so the AI of robots can make a great contribution. What's more, we may want to easily find a specific room location even when we are not familiar with the place. We can navigate our outdoor position with GPS, but it is hard for us to know our indoor position with our smart devices. All of these problems lead to the topic on how to implement the localization of our robots.

B. Related Work

II. Overview

Signal will fade during the transmitting and power will somewhat loss no matter what kind of transmitter be used. As a result, we can estimate the distance between the receiver and transmitter if we know the properties the the wireless signal. Meanwhile, the phase of signal will shift when receiver get a signal, which related to their direction of arrival. In this case, out project decide to use the MUSIC algorithm and path loss to estimate direction and distance between multiple robots.

Due to the interaction of noise, one of the biggest challenge is to recover the original signal with considering white noise and Rayleigh fading noise. To easily understand the effect of these noise, the following figures are the simulation of some common signals when addictive noise is applied.

To simulate our signal processing, we are going to use Matlab to model signals and apply them in simulation field, then we will integrate them with MUSIC algorithm and finally get DOA location, which will be described in detail later.

III. Architecture

During our modelling stage, we separate it into three parts: path loss, fading and noise.

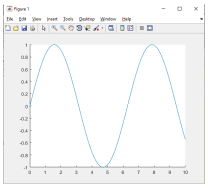

Fig. 1. sin waveform

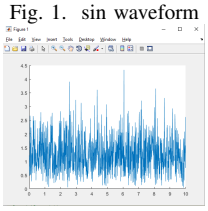

Fig. 2. Rayleigh fading noise

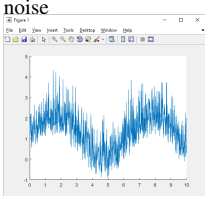

Fig. 3. sin waveform added with Rayleigh fading noise

The first step is to randomly generate a location for each robot. Each robot should have three ranges, sensory range, communication range and reject range. Sensory range is the largest range that allows each of robots to sense each other within that area. Communication range is the smaller range that allows robots to communicate with each other. Reject range is the smallest range in the case when two robots are too close, they can't sense each other, which is what we want to avoid during the simulation. In order to do this, we create three files, described as following:

- "drawCircle.m": A simple function to draw circle with parameters x,y and r which stands for the locations and radius.
- "Robot.m": A class that has properties called "sensory", "communication" and "reject" which hold values for the radius of each range. Properties "x" and "y" stands for the location of robots in graph to be drawn. This class has four methods of drawing

circles, ”drawSensory(obj)”, ”drawCommunication(obj)”, ”drawReject(obj)” and ”drawAll(obj)” which can draw three types of circle range individually and in one time.

- ”simulation.m”: The driver file that creates 8 robot object in random location in the graph. In order to avoid robots correlated with each other within the reject range, the algorithm is to keep looking for the position of a new robot until the condition satisfied and then create robot according to that position.

IV. Simulation Result

The following graphs are 4 simulation results that randomly allocate 8 robot objects:

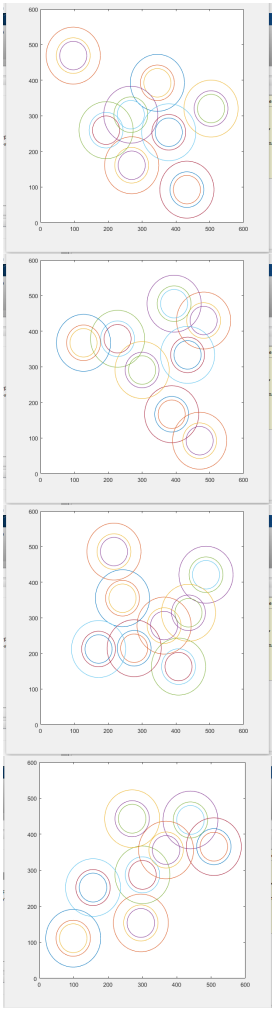

Fig. 4. Four sample simulations

V. Evaluation

VI. Conclusion

References

[1] Wikipedia. Charge-coupled device — wikipedia, the free encyclopedia, 2015. [Online; accessed 11-December-2015].

[2] sensorwiki. Temperature sensors, 2011/04/14. [Online; accessed 11-December-2015].