LOCALIZATION AND PATH-FINDING OF A QUAD-COPTER

Christian Kaan
University of Wyoming

Overview

- The kidnapped robot problem
- Path planning
- Unmanned Aerial Vehicle

The kidnapped robot problem

- Some robots need to know where they are in their environment to work
- That is where localization comes in
  - Helps the robot to probabilistically locate itself in the environment

Localization

- Uses some known characteristics of the environment
  - In this case, the robot knows the map of the environment
- Takes some measurements and compares it to these known characteristics
- Using a Particle Filter for localization

Particle Filter

- Initialize X number of particles across the entire map
- Every time a measurement is taken, compare each particle to the measurements taken
- Each particle will have a probability of being the actual position
- Sample from particles to obtain a new set of particles

Particle Filter

- Also need to account for movement by moving particles with expected movement
- Account for noise in both movement and measurement
- Run this algorithm until a certain threshold is reached

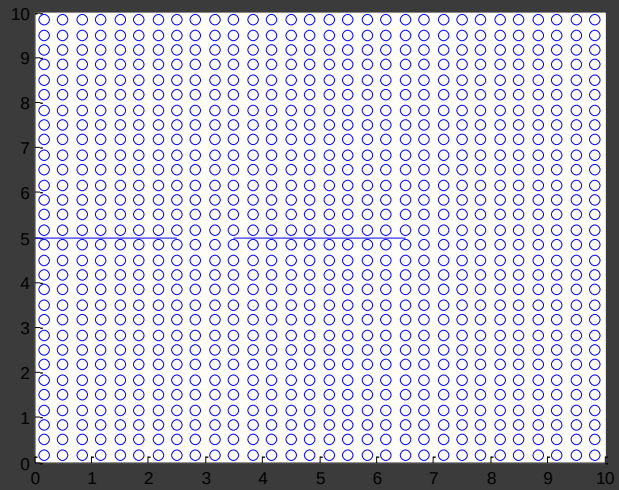
Measurements

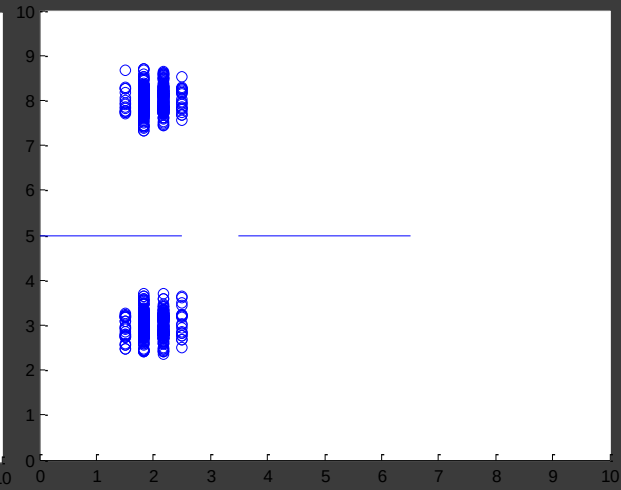
- My design uses 4 sonar sensors for measurements
  - Forward, Left, Back, and Right
- I use 4 MB1040 LV-MaxSonar-EZ4 Ultrasonic Range Finder
- Connect the four sensors to an Arduino Uno using analog input
- Transmit data to computer using an HC-06 Bluetooth module

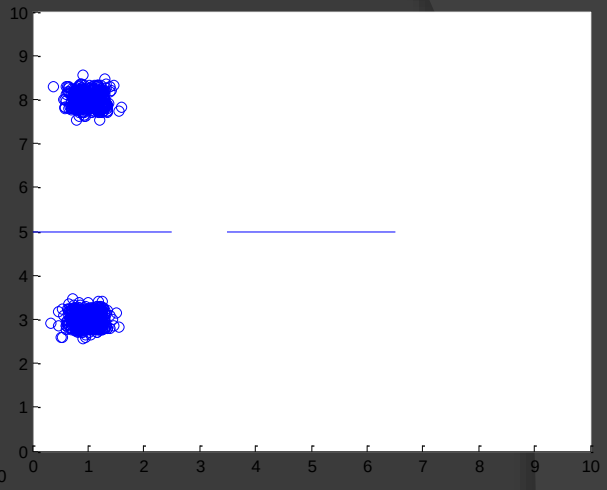
Sonar Array

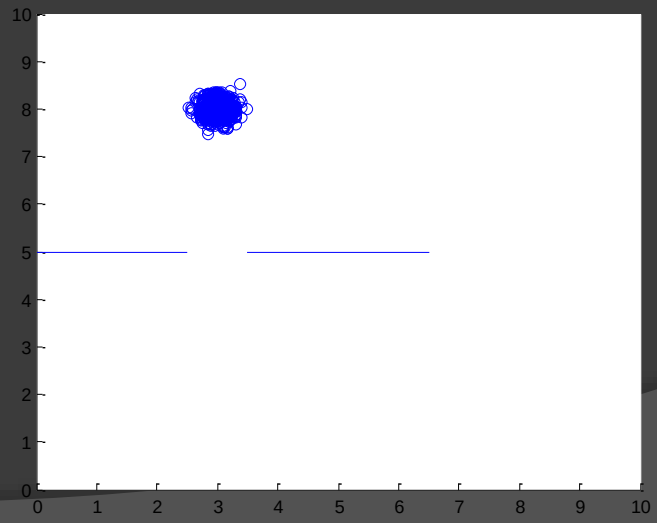
- Works very accurately
- Initially, I was trying to use all four sensors at the same time, which caused the sensors to interfere with each other.
- Now, scan with one, then turn it off.
- Repeat that process until I have all four measurements

Particle Filter









Path Planning

- Once the robot knows where it is, it has to figure out what to do
- In other words, plan its path
- The most commonly used algorithm for this is A* (A-star)

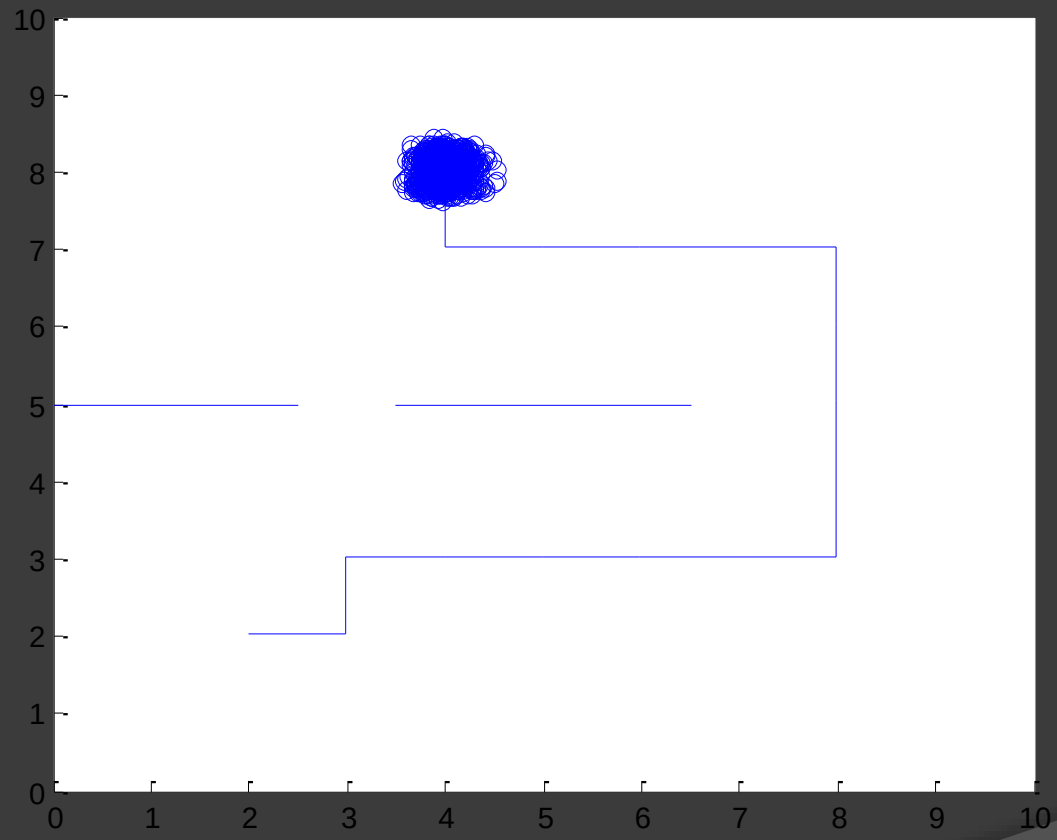
A*

- Uses the location of the robot to find the most cost efficient path to the goal
- Cost is determined by a heuristic function
  - This heuristic can include many variables
  - Distance to the goal is a common heuristic to use
- The heuristic I used involves the distance to the goal and if the robot gets close to a wall

A*

- The algorithm works by checking all possible paths next to the starting point
- It then calculates the heuristic for each possible point and orders these in a queue
- It then takes the lowest cost point and expands it
- Does not look at points that have been expanded already

A*



Unmanned Aerial Vehicle

- Unmanned Aerial Vehicles are very interesting to use because they can travel where many wheeled and legged robots can’t travel
- I used the 3DR Iris made by 3D Robotics

Unmanned Aerial Vehicle

- Very noisy movements without proper controls
- For very good control, need a much more advanced sensors than I had
  - For instance, using cameras which can determine not only distance but pose

Unmanned Aerial Vehicle

Unmanned Aerial Vehicle

- Unfortunately, due to the noise and the lack of a more advanced sensors, I was unable to create a stable autonomous flight controller
- Tried to do the algorithm with manual control, but the noise was still exceedingly large, as seen previously

Future Work

- Try to create a better controller with the sensors I have
- A better option would be to use a better set of sensors for the algorithm

Components and Parts Cost

- 3DR Iris - $800
- Spare Propellers - $100
- Sonar Sensors - $120
- Arduino Uno - $60
- Project Enclosure - $5
- HC-06 Bluetooth module - $5
- Total - $ 1090

Special Thanks to:

- Volpi and Cupal Senior Design Fund
- Wyoming NASA Space Grant Consortium
- Faculty and Staff of the ECE Department

Thank You

- Questions?