Will Nodvik
Tara Umesh

ESP RSSI Navigation: RTES Design Project

BOM
3x Ultrasound sensors
1x Adafruit Robot
1x ESP8266

Router Issues:

Since testing was done on a home router access point we were unable to demonstrate at the demo timeslot as I was unable to get a connection to my phone/computer across. With the more powerful antennas on Netgear nighthawk the ESP8266 had no problem connecting.

Hardware Issues:

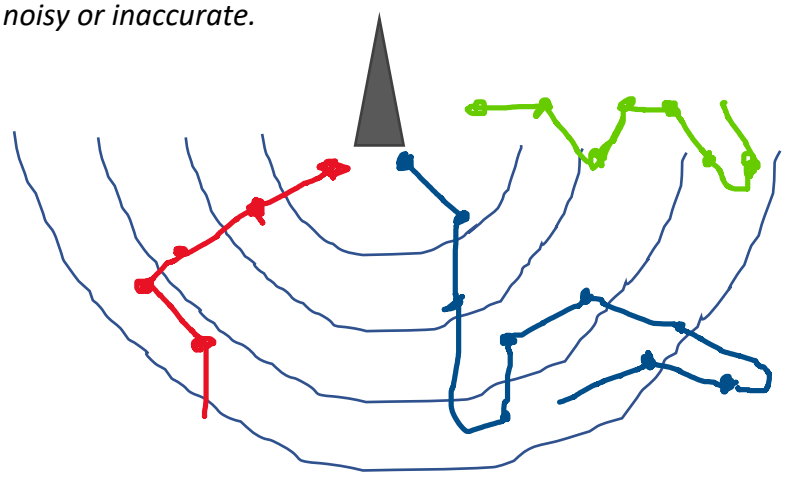
The HC-SR05 sensors used became inexplicably unreliable where they were working one moment, then broken the next. Neither was giving a reliable output when the algorithm was ran, but they work independently. There was difficulties with the turning from obstacle avoidance conflicting with the RSSI navigation but the issues seem to work themselves out with working hardware conditions, the HC-SR05 sensors themselves failed (they can be seen working sometimes in the latter half of the video).

The Algorithm:

The robot traverses the RSSI field by making 45 degree left turns. 8 turns make a full revolutions so the theoretical circle will be referred to as an octagon. The robot will compensate for these left turns depending on it's position on the octagon. Figure 1 demonstrates a robot's various paths across the RSSI field.

Obstacle avoidance is achieved with a function that returns the angle of the octagon the robot is oriented (0 = 0 degrees, 2 = 90 degrees, etc.) So if a robot makes two 90 degree left turns avoiding an obstacle, it will return 4 as the amount rotated. Right turns can be seen as a subtraction from 8 using modular arithmetic. In code this is represented by the amount of left turns the robot takes. In every state, the robot will check for obstacles and turns to a clear path before going straight. Since the next state depends on the rotation number and RSSI value it will continue independently of the obstacle avoidance.

Note: In the test runs the sonar sensors were unreliable this might be an issue with power draw from the USB bank. Perhaps a solder joint worked loose. Additionally, the right sensor was not working after several tests, so the entire algorithm operates by mostly going left to try to avoid objects on the right. Object avoidance was working independently of the RSSI sampling and turned on at one point while operating (which can be seen on one of the videos), but it was found to be too noisy or inaccurate.



Finite Automaton:

The robot is implemented with a sampling algorithm for the RSSI signal followed by a series of conditional statements. After averaging the RSSI sample the robot has one of four choices: clear ahead, marginal progress, keep turning, and out of range. It operates as a finite state machine after receiving an RSSI power lower than the defined threshold (it stays still if it's close enough to the AP). Some biases are programmed into the code to account for fluctuations in RSSI powers. The basic finite state machine is as follows: