Cosc 4P78 Project: ”Martini Bot”

Michael Alsbergas, Peter Wilson

April 19, 2016

1 Instructions:

To activate the martini bot, simply turn both power switches to the on position. The power to motors is a toggle switch at the front on top of the robot, while the power to the other hardware is on the large battery pack attached to the back of the robot.

Once the robot has been powered on, it will remain in an ”idle” state until it receives a signal from the target remote. The remote will send the signal when it is initially turned on by plugging in the battery. The signal can be resent by pressing one of the reset buttons on the remote.

When the robot has received the signal, it will begin its search for the target remote. After the robot has found its target, by being within a martini passing range, it will return to its ”idle” state and can be reactivated by pressing reset on the remote.

2 Problem:

The problem task that this robot was designed to solve is when a person wants a made martini but does not want or is not able to retrieve it themselves. Potentially, a solution to this problem could be used to deliver all manners of payloads.

This problem, from the eyes (rangefinders) of a robotic solution, is locating and moving towards a specific and unknown point without having any prior knowledge of the surrounding area aside from some core assumptions.

3 Solution:

The hardware on the robot itself includes an ultrasonic rangefinder, an XBee radio transceiver, 2 DC motors and wheels equipped with encoders, a Romeo arduino chip to control it all, and of course the appropriate amount of batteries to power it all.

Similarly, the target remote is made of the same kind of ultrasonic rangefinder, the same kind of XBee radio transceiver, an AT mega arduino chip to control it, and a 9V battery as the power supply all contained within a clear plastic case to keep it together.

Our initial solution using this hardware was to use the 2 ultrasonic's on the devices to communicate with each other so that the robot could determine the distance to the target. Combining that and the distance traveled using the encoders, we could use trigonometry to determine the direction of the target. Once both the distance and direction were found, it was a simple matter of going to the target. Yet, since we're using ultrasonic's, we had to build our robot with the assumption that there would be no obstacles between it and a straight line to the target since the ultrasonic's could not read each other if there was one.

The problem was then coordinating the ultrasonic's so that they triggered at the same time, at least roughly. That's where the XBees are used. The transceivers would be used to send a signal to notify the other of when to send/receive an ultrasonic pulse. The robot would send an activation signal to the remote and would then listen for a pulse from the remote. The XBees are also used to have the remote send the activation signal to the robot while it's in its idle state.

However, we quickly discovered that even when using the XBees for timing, the ultrasonic's would still not determine an accurate distance between each other. Not only were the received readings inaccurate, but they were also erratic and inconsistent. Yet, we noticed that all the readings obtained using the 2 together were always less than the actual distance. With this, we rethought our solution.

The robot would spin so that it and its ultrasonic were facing a direction. Then, it would use the ultrasonic to take a number of readings in front of it to determine a baseline. Then, it sends a radio signal to the remote telling it to emit a series of non-stop pulses. The robot would then take another set of readings in the same direction and then send a stop radio signal to the remote. If the second set of readings had the same average of the first, the robot would rotate and try the next direction. But, if the robot was facing the target at the time of the second set of readings, the values and their average would be considerably smaller due to the interference given off by the target, which means that it's facing the target.

Since the robot now has a general direction in which the target might be in and a rough approximation of the distance, all it needs to do is move forward. Because of the assumption of the unobstructed straight line, the robot can move forward with confidence. The robot will move half of the approximated distance, and if that movement/distance is small enough, the robot will know that it reached its target and will become idle to wait for its next target. Since the robot only receives a rough approximation of the direction, it may lose track of the target. When this happens, it simply begins its process of turning around until it locates the target again.

Due to the Romeo's motor control ability to have a DC motor spin in reverse, it allows the robot to actually rotate on the spot rather than turn on a pivot. The encoders no longer become essential in our final solution; however they are still helpful in moving forward a calculated distance as well as controlling the turning angle of the robot. Neither of these parts are truly essential, but their use is recommended to solve other potential problems around the robot's movement (such as hitting a nearby wall while pivoting).

4 Remaining Problems:

The only remaining problems are the possibility of false positives and a maximum range. When used in an environment that has other sources of ultrasonic interference, the robot may get confused and begin to move in the wrong direction. This has been noticed when the robot is facing baggy clothing which tend to absorb ultrasonic waves. Also in the robotics lab where there are a lot of metal stool legs, it appears that some of the ultrasonic interference from the target is reflected off the legs and is registered by the robot as the interference it is looking for. We have also been unable to coax it to go through a metal door frame. This may be because either the metal deflects the pulses too much or the robot just picks up interference pulses bouncing off walls and objects behind it. To help reduce the chances of getting obscurely high values during the robot's test readings, we programmed it to ignore all readings above 3000cm (100ft). So, if the target is beyond this range, the robot will ignore it. This is also assuming that the max range of the ultrasonics and the XBees are greater than this distance.

5 Resources:

The XBees print to the serial and do not need/have any dedicated libraries to use. The rangefinders need the ultrasonic library to use normally. We also modified the ultrasonic library for the sake of the target to emit a steady stream of pulses without needing to read in any echo data.