Embedded Systems Report

Lecturer: Niall O'Keeffe

Student: Ciunas Low Bennett.

My project is a room mapping robot. The robot is a small tank like vehicle. There was a small amount of hardware needed to get my project working.

This involved an Ultrasonic sensor, a Transceiver, two Dc motors a H-bridge and a lithium ion battery. Wiring was a little tedious due to the small size of the robot.

The most challenging part was the software. A short description of the algorithm I implemented follows.

The robot is controlled by multiple state machines. There most important one is the one that runs in the main loop of the program. With many smaller state machines running inside functions and other parts of the code. I am going to describe the functionality of the main state machine for now.

Firstly the robot needs to calculate the size of the room it does this using the ultrasonic sensor. For my robot to work it must be placed in one of the corners of a room facing parallel to one of the sides of the room. When the robot is placed in the correct place it may be turned on. It sends out an ultrasonic pulse from the front of the vehicle. It measure the returning pulse width. This will give us a larger value depending on how far it is from an object. A function is called to transfer this returning value into a number corresponding to 30cm squares. The robot now turns 90 degrees to be parallel with the wall perpendicular to the previous result. Again it measures the distance and turns it into a number corresponding to a 30cm square. A 2D array is created from these values, the size and width of the 2D array corresponds to the length and breadth of the room.

A function is called to calculate what maneuvers the robot has to perform. These preset maneuvers are calculated depending on the size of the room. And are created using the values of the row and column that I created the 2d array with.

The robot is programmed to move in the room by covering each square in a pattern. It moves along the column until it reaches the highest element and then it will move up one row and then move back down the columns until it gets to element zero, then up another row etc. Until it has covered all the elements of the array.

I divided up these maneuvers into different processes. So the process for moving along the columns is a separate one than moving from row to row and also avoiding an obstacle. I created a buffer that I load the maneuvers into. Each maneuvers is equal to moving from one square to another in the 2d array. Except when the robot is turning.

Therefore after the robot has calculated the size of the room. Because it is at element [0] [0] of the array and facing along the columns. A state machine uses these details to fill the buffer with values that will make the robot move along the columns until it has reached the end element of the columns, it will then turn right so it is facing along the along the row. This will empty the buffer, and now another process is calculated. Because the robot is no longer facing along the column and it is now in the final element of the column [0] [last element]. Another state machine calculates that the robot needs to move up one element to the next row. It now fills the buffer with the maneuvers to move up a row and turn to face down the column. If there is no obstacles in the room the robot will continue doing this until it has been to every element in the array. Depending on which row it is in the robot will have to turn different directions.

To pick up obstacles the robot uses the ultrasonic sensor. The ultrasonic sensor will only be on when the robot is moving forward this is to stop any flags being set when the robot is turning. Such as when it is in the final element next to the wall. A flag is created to be set to one whenever the robot picks up an object closer than 30 cm.

There is different protocols in how this flag is used depending in where the robot is and what maneuvers it is performing. When the robot is moving along the columns because it will pick up an object before it moves into the square it is in. The flag is only executed after the robot has performed its current move. If the robot turns and wants to move to another row an object could be straight in front of it. In this case the flag would have to be acted on instantaneously. Otherwise the robot would drive into the object.

There is two maneuvers that I have created for avoiding an object. More would be needed to be able to deal with every situation but I didn’t have the time to work on move maneuvers. If the robot comes across an obstacle while moving along the columns it will move around it by turning to the left moving one element forward turning right moving forward twice, then turning right moving forward and then turning once again to face the same way as previous. If the robot is travelling the opposite way it merely does the same but changes which way it turns, so it doesn’t go into any squares it has not checked already.

In the case of the robot turning to move to another row and finding an obstacle in its way. The robot will turn right move forward one square turn left and then move forward and turn left. Thus avoiding the obstacle.

This is a very basic overview of how the robot works. And the State machines and functions have been simplified.

Thanks for reading.

Ciunas Low Bennett.