Assignment Report

Task1: The first task was to get the Zumo to be driven down the corridor using the W, A, S, D keys and the space bar to stop. We had previously completed this task in a tutorial exercise so I took the code that I had from the arm and adapted it to work for this assignment. I managed to get this working and I was able to get the Zumo navigating around the maze. I looked around and managed to find some code that allowed me to click the W, A, S, D buttons multiple times for movement rather than one long continuous movement. It works by constantly checking for a key press.

Task2: The second task was to get the Zumo to automatically keep within the corridor by using the reflectance sensors. I first started by calibrating the reflectance sensors by using the examples included in the Arduino library. Once the sensors were calibrated it could now detect the corridor and the lines so I started by looking at the code that we had done in a tutorial exercise which was the boundary checking and the line-following examples. I realised that the reflectance sensor has 6 sensors and to keep it in the corridor I needed to use sensor 0 and 5. I introduced a threshold for the reflectance of 400 so if the reading is over 400 we know that a black line is detected. So if the 0 sensor detected the black line it would detect a reflectance number over the threshold and we know that we have hit the wall of the corridor on the left and we need to reverse and turn the opposite way to the wall we’ve detected. I did this by using a simple if statement, then again copied the code for the 5’th sensor and it worked correctly.

Task3: In this task I needed to get the Zumo to recognise that it has reached a corner by detecting the wall in front. The way I completed this task is by adapting the code I already had from task 2. So instead of using sensor 0 and 5 to detect the wall, I changed the code to use sensors 1 and 4. Once the sensors detected the line I then call a function to turn the Zumo. Once the key ‘C’ has been typed it will then switch back mode to the manual driving ready to continue through the maze. Once the mode has been switched back to manual drive task2 will also be enabled again.

Task4: For this task the Zumo needs to be able to search a room for a person. When the Zumo is outside a room the 'R' key is pressed the mode of the robot changes to the room scan function. I then manually turn in the direction of the room. The way that I decided whether the room is a left or right is by measuring how many times the ‘A’ and ‘D’ keys have been pressed. The higher the number on the counter means that it’s most likely that side of the room. The room scan function then starts an automated scan.

Task5: For this task I used the NewPing library that is available from the examples in the Arduino editor. I had previously got this working in class where it was measuring distances up to 2 metres. For this task all I had to do was adapt some of the coding to move the Zumo robot around. I first got it turning left, right, left and then back right. Originally I only had it turning left to right once but I found that it wasn’t accurate enough when finding objects. Once an object has been found a string is printed out saying the room has been found and the object is in there. A simple IF statement was sufficient to get this working. I used a Boolean called item\_found and set that to true if an item was present in a room. I originally set the CM of the ultrasonic sensor to 20CM but realised it wasn’t accurate enough to had to increase that value to 30CM. I have also noticed that the object has to be of a certain colour to reflect enough back to be picked up by the receiver.

I made use of all the libraries that we had previously used in class. This included the ZumoMotors.H to get the motors to work and this allowed the Zumo to move around. I used the ZumoReflectanceSensorArray.h to keep in the maze. I adapted the maze solver/line follower code that I had completed in class. Pete mentioned in class that it would be a good idea to include different modes within this assignment and I did that by using a switch statement. The switch statement allowed me to switch between functionality and implement some basic error handling. When thinking about storing the path which the Zumo robot has taken I originally had split ideas. I was thinking about implementing an array of strings that would hold chars for the different actions, ‘L’ or ‘R’ being the direction and ‘R’ or ‘T’ being a room or a turn. I ran into several problems with this and realised I could do it a much easier way. I then thought about doing a struct data structure which I had previously implemented in my C++ module last year with Jan. I was able to store all the data in individual nodes, store the nodes in an array and easy iterate through the array to retrace the steps of my path. I didn’t manage to get the Zumo to automatically come back though I had an idea on how to complete that section I just ran out of time. I was going to use the line follower code from the example and use the 0 and 5’th sensor to follow the edge of the line until it hit a room. I was then going to work out if that specific room has a person in it from storing the bool value in my struct, if so it would then go in and do a scan. If there wasn’t a person in the room I would set the motors to go forward until it picked up the line again on the 0 or 5’th sensor. I would repeat that until I got to the last room, I was intending to use the array like a stack and just decrement the room counter after each time I’d gone past one.