Programming Things: Assessment 2019-20
Coursework 1 (Individual)

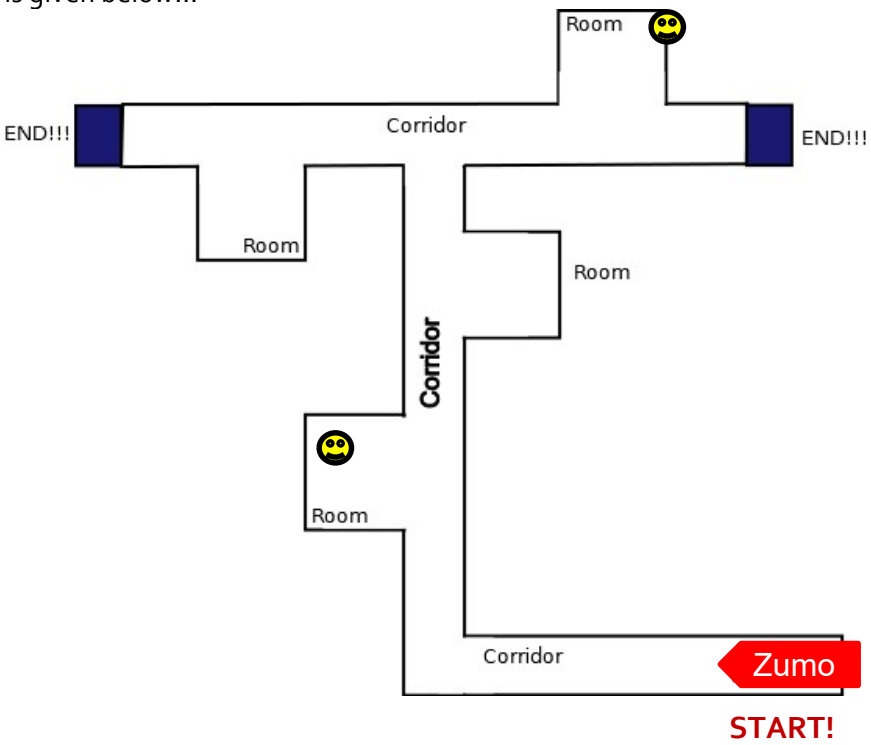
- Identify and critically assess the elements needed within a physical computing system
- Interface a programmable controller with peripheral devices such as sensors, switches, key pads, motors, lights, sound, displays and other input devices and actuators.
- Determine what types of devices are appropriate for various products and processes.
- Design and implement 'control' algorithms for the relevant hardware platforms.

For this assessment, you are required to code a Pololu Zumo 32U4 robot to perform a (simulated) search and rescue operation.

The scenario **motivating** this assignment is that your robot is trying to find/rescue people trapped on a single floor in a building which is filled with smoke. The robot moves through a corridor and people are to be discovered in rooms or in the corridor. When the robot discovers a 'person' it signals back to 'base' so that a search party can come in to rescue that person. The robot, however, continues to search, signalling as and when it finds people in other rooms. When the robot reaches the (final) end of the corridor, it turns around and returns to base (by the quickest route possible, but visiting all the locations where it has found people to confirm they have been rescued).

For the Zumo, this means you will have to 'design' and code the Zumo so that it can explore a 'maze' and find a number of 'hidden' objects. You will also need to use an XBee module with the robot so that it can communicate wirelessly with your computer.

The 'building floor' will consist of a corridor with corners and adjoining rooms. The borders will be set out with black lines on a white background. As an illustration, **one possible configuration** of rooms and corridor is given below...



This diagram is not to scale, but is meant to be indicative of relative sizes of elements of the environment the robot is moving in.

Everything is 2D: all 'walls' are black tape/lines on white paper/card.

To help the Zumo 'know' where it is, the widths of corridors, rooms, doors, etc. is important.

The rooms are 'deeper' than the corridors.
The doors are wider than the corridors.
The corridor is (just) wider than the Zumo.
*eg: corridorWidth is approx Zumo width*1.2.*

The Zumo MUST stay INSIDE the corridor.
*Room depth is at least 1.5*corridorWidth.*
*Door width is 2*corridorWidth.*
The placing of the objects is also indicative. They could be anywhere in the room or in the corridor.

TASKS

Your aim is to get the robot to perform certain tasks:

Task 1: Manual control of the Zumo.

The Zumo can be driven down the corridor from a **GUI* (see comments below)** e.g. using w, a, s, d and 'stop' 'buttons' or a text field. **You** are controlling the Zumo at this point. Communication is via the xBee communication channel (not over a USB cable).

Task 2: Autonomous control of the Zumo

The Zumo automatically keeps within the corridor by using the line sensors to turn away from the walls (this is an adaptation of the boundary checking and line-following examples looked at in the tutorials). This means you only start the Zumo moving; after that, it is navigating itself along the corridor. It stops when it encounters a 'wall' in front of it (i.e. comes to a corner). If Zumo hits one of the side walls of the corridor, it modifies its path to keep within the walls of the corridor.

Task 3: Turning Corners

At the end of task 2, the Zumo recognises that it has reached a corner and stops. It should then return manual control to the human navigator (you) by:

1. Sending a message using the XBee indicating that fact. The messages received from the Zumo should appear in a text area in the GUI.
2. It then deactivates the autonomous behaviour from task 2 (which is keeping the Zumo between the corridor walls); this allows the (human) controller to turn the robot. When that turn is complete, the human navigator (you) signals that by sending another keypress (eg 'C' or 'c' for complete).
3. This then reactivates the autonomous behaviour of task 2, so that the Zumo can drive itself down the corridor again.

Task 4: The Zumo turns autonomously around a corner

This is an extension of Task 3. When the Zumo recognises it is at the end of a corridor (such as at the end of task 2), it sends a message through the XBee that appears in the text area on the GUI, the same as for task 3. The user can then press 'L' or 'R' to rotate the Zumo 90° to the left or 90° to the right. To complete the rotation and the movements, wheel position encoders can be used to measure the space covered by each wheel. Additionally, the on board gyroscope may be used to sense the rotation. Once the turn is complete, the Zumo should automatically return to the autonomous behaviour of task 2, so that the Zumo can drive itself down the corridor again.

Task 5: The Zumo searches a room.

The (human) navigator will first stop the robot (outside the room) and then signal that the robot is about to enter a room by sending an appropriate button press or text field data (e.g. "Ro" for room and 'R' or 'L' for right/left). The Zumo will then use the rotation algorithm from task 4 to rotate towards the room to perform a search. An appropriate message should appear in the GUI, giving that room a number and identifying whether the room is on the left or right of the corridor. The Zumo should also record that information. The Zumo should then (autonomously) move into the room to perform a scan of the room, using the proximity sensors, for objects. Depending on the size of the room, proximity sensors may require the Zumo to roam the room to complete the search. If an object is detected, the Zumo reports that back using the XBee link. This report needs to be seen inside the GUI you have created. The message should identify which room the object is in. After the scan is complete, the Zumo should return back into the corridor autonomously, using the wheel encoders to identify its position.

Task 6: The T-junction

In the T-junction, you can turn either way and search the remaining corridor and rooms as in Tasks 4 & 5. However, depending which way you have turned, when you reach the end of the corridor, the Zumo should stop and wait until it is told to turn around and retrace it's steps (using the 'B' key to initiate a 180 degree turn). The Zumo turns around and then goes to search the other half of the corridor. It should navigate the half of the corridor just searched, autonomously, and ignore any instructions to turn into rooms or back down the main corridor. These instructions must be sent though, so that the robot knows it has passed

those points. After the T-junction has been passed, the Zumo should be allowed to search the rooms in the second half of the corridor.

Task 7: The return journey. At the end of the second half of the T-junction corridor, the Zumo should again stop and wait until it is told that it has reached the end by receiving a keypress ('H'). This time the Zumo should recognise that is due to return home. It should 'optimise' the return route and navigate its way back to the start without intervention by the user. It should avoid rooms that were empty and 'check' locations that had people in. If a room is still occupied, that should be signalled over the XBee. The robot's LEDs should turn on and a tone should be played on the buzzer – to provide a 'follow me' guide – which should turn off when the robot gets back to base.