CS12020:
Robotics Assignment

Dimitris Tsakiris (*dit5@aber.ac.uk*)

November 2019

This assignment is only intended for students working with the robot shield in their practical sessions. If you have been allocated a different practical group, then there is a different assignment for you.

1 Introduction

For this assignment you are required to integrate a number of the functions that you have developed during the course of the semester and extend them with some new code. The end result of this work should be a control system that makes your robot perform a short dance, drive it along a closed path on the floor and navigate along this path while avoiding obstacles.

The robot controller is to be developed in four stages. The details of the required behaviours are given in section 2 (Functional Requirements).

You are also required to write a report on your work. For each stage of the assignment, you should describe and comment on the **techniques** you used, any **problems** encountered and **how well** the robot performs its tasks. Some stages list additional information that you should also include in your report. See section 3 for more details on the report.

While getting your robot to complete the task is the overall aim of the assignment, there are a significant number of marks available for making sure that you have an appropriate breakdown of your program (into functions), and for ensuring that you have used sensible algorithms and appropriate, informative comments. Please don't under-estimate the importance of making your code clear and well structured.

This assignment in total is worth **50%** of the marks for the module. You are likely to need to spend between sixteen and thirty hours of your own time to complete this assignment.

2 Functional Requirements

This section describes the functionality your program should implement. There are **four** stages in the development.

The marks listed for each component are for a perfect implementation of that component. Implementations of components that are less than perfect will receive appropriately lower marks. You are required to use the Arduino IDE for this assignment.

2.1 Stage 1: Let's dance [15%]

For this part of the assignment, you will need to integrate into the controller, and use, the motion functions that you developed during the practicals. After pressing the ***left*** pushbutton, your robot should be able to perform the following tasks in sequence:

(i) ***EEPROM reading:*** Your program should store in the EEPROM, and retrieve from it, the values of the following robot calibration variables:

- motion calibration: servoLeftStop (0), servoRightStop (1), servoLeftOffset (2), servoRightOffset (3), wheelDiameter (4),
- sensor calibration: sensorLeftThreshold (10), sensorMiddleThreshold (13), sensorRightThreshold (16)

Task: When your program starts, these data should be read from the EEPROM in the variables above. At some appropriate time during the execution of your program, the corresponding values should be stored in the EEPROM.

(ii) ***Straight-line movement:*** Functionality: The robot should be able to move straight ahead and backwards. ***Task:*** The robot should move straight forward for at least 50 cm, and then backward for the same distance. It should be able to end up at almost the same spot where it started.

(iii) ***In-place turns:*** Functionality: The robot should be able to turn in-place, making 90, 180 and 360 degree turns both clockwise and counter-clockwise. ***Task:*** The robot should turn 90 degrees clockwise, then 90 degrees counter-clockwise, and end up at almost the same orientation where it started.

(iv) ***Dance:*** The robot should be able to perform some dance moves of your choosing for a period of **20 seconds**, while flashing the LEDs in some sequence of your choosing while dancing.

For this part of the assignment, as well as the standard items, you must also include in your report a description of the mechanisms that you used in order to:

a) time the period for which your robot dances

b) flash the LEDs in sequence (if you implemented that)

This part of the assignment is worth a total of **15%** of the marks for this assignment.

2.2 Stage 2: Follow a path [25%]

For this part of the assignment, you will develop a ***path-following behaviour*** for your robot. You will need to use the motion functions that you developed during the previous stage, and to trigger them, as appropriate, based on input from the ***LDR sensors***. This sensory input will be used to detect the black and white regions in the test environment, which will contain a black line (not necessarily straight – approximately 50mm wide) printed on a white background (see Figure 1 for an example). The robot should be able to:

(i) ***Calibrate*** its sensor data, in order to ***discriminate*** reliably ***black from white features*** on the floor. You will be allowed to place the robot on a white, then on a black, area of the floor, to collect sensor data. Some ***averaging*** scheme for the sensory data is expected, in order to reduce the effects of noise. These data should be used to devise a way for the robot to distinguish black from white areas on the floor.

(ii) ***Move forward along the black line*** for 50 cm. You will be allowed to place the robot on the black line, and press the ***right*** button to initiate this movement behaviour.

(iii) After completing (ii), the robot should turn around (180 degree in-place turn) and move in the opposite direction, along the path, for 50 cm.

(iv) After completing task (iii), the robot should move ***backwards along the line*** (i.e. maintaining its original orientation, as it is at the end of (iii)), for 50 cm.

(v) Re-acquisition of the path: During the movements (ii), (iii) and (iv) above, if the robot deviates from the black line, it should be able to re-acquire the black line and move on it. This requires using sensory information.

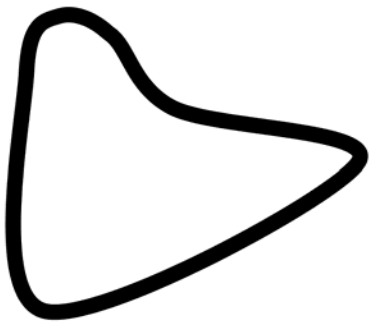
Figure 1: Example path to be followed by the robot during Stages 2 and 3

Signalling and ***reporting*** with the LEDs: While following the path, the robot should flash the yellow LED. If lost (not detecting the path) the robot should flash the red LED until it re-acquires the path.

For this part of the assignment, as well as the standard items, you must also include in your report:

- some recorded data which show how the **LDR** sensor values vary during a typical run of your robot following the path. Comment on the variation of these data.
- After completing movements (ii) and (iii) above, does the robot return approximately to its initial position? What is the average error? Were you able to improve it?

This part of the assignment is worth a total of **25%** of the marks for this assignment.

2.3 Stage 3: Detecting Obstacles [15%]

For this part of the assignment, your robot must be programmed to ***detect obstacles*** in front of it with its ***IR transmitter*** and ***receiver pair***, while it is following a path, as in the previous stage. This behaviour is initiated by pressing the left button while the robot performs the path-following behaviour. When the robot encounters an obstacle on its path, it should stop and flash all LEDs for 1 sec, then turn around (180 degree in-place turn) and move in the opposite direction along the path.

This part of the assignment is worth a total of **15%** of the marks for this assignment.

2.4 Stage 4: Counting junctions [20%]

For this part of the assignment, you will need to use the path-following behaviour developed during the Stage 2. The path to follow will be laid out using a black line (not necessarily straight) printed on a white background, as before. However, now this line encounters junctions formed by perpendicular segments of the same width, placed at irregular intervals along the black line (see Figure 2).

Functionality: The robot should be able to drive along the black line. It also needs to cross successfully the junctions, and count them as it passes over them, in order to *detect when it arrives at the desired location*.

Task: You will be allowed to place the robot on a junction, and press **both** buttons to initiate this behaviour. The robot should move forward along the black line, crossing and counting the junctions it encounters, until it arrives at the 3rd junction from its start. There, it should turn around (180 degree in-place turn) and retrace its steps until getting to the initial junction.

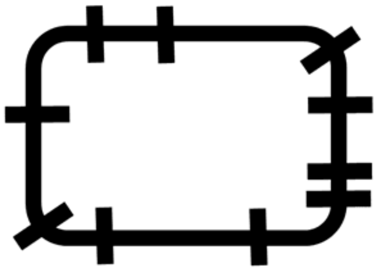

Figure 2: Example path with junctions, to be followed by the robot during Stage 4

Signalling and ***reporting*** with the LEDs: While following the path, the robot should flash the yellow LED. When detecting a junction, and while crossing it, the robot should flash the green LED. If lost (not detecting the path) the robot should flash the red LED until it re-acquires the path. When it returns at its initial junction, it should **report** the number of junctions it encountered by flashing all LEDs an appropriate number of times.

This part of the assignment is worth a total of **20%** of the marks for this assignment.

3 Report [25%]

Write a report of **up to 1000 words** describing what you did. Include any problems or complications that were encountered, a list of the stages you believe you have completed and give your assessment of the mark you would award yourself for the work (based on the assessment criteria mentioned in section 5 , below). You must submit your report as a **PDF** file.

For each of stages 1, 2, 3 and 4, you should be sure to include in your report:

- a short description of how your program is supposed to work

- any *extra* information specified in the functional requirements for that stage

This part of the assignment is worth a total of **25%** of the marks for this assignment.

4 What you must submit

You are expected to submit a single **ZIP** format archive file. Other archive formats, such as 7zip, tar or rar, are ***not acceptable***. The **ZIP** file must contain the following items:

1) The code you have created (**75%** of total marks).
   We will be looking for useful commenting of the code, suitable naming of identifiers, layout and indentation of the code, modular code (i.e. use of functions), and good coding practice such as defensive programming.
   Your code must be included within the ZIP archive as an **Arduino IDE folder** containing **.ino** files.

2) A report of up to 1000 words, as described in section 3 (**25%** of total marks). You must submit your report as a **pdf** file.

The **ZIP** archive should be named ***`xxx_cs12020.zip`***, where you should change ***xxx*** to your user ID.

5 Assessment Criteria

This assignment will be assessed according to "Appendix AA - Assessment Criteria for Development" of the student handbook: http://impacs-inter.dcs.aber.ac.uk/images/editor-content/Documentation/Handbooks/Appendices/AppendixAA.pdf

You are advised to complete each part as much as possible. If you have time remaining and want to impress, then go for it by presenting a more in-depth analysis of the data captured in the various stages and by tuning your robot's behaviour to be as precise and fast as possible.

Note: This is an individual assignment and must be completed as a one person effort by the student submitting the work. To avoid any possibility of plagiarism, your attention is drawn to section D of the Department's Student Handbook, available at: https://impacs-inter.dcs.aber.ac.uk/images/editorcontent/Documentation/Handbooks/handbook-cs-2019-2020.pdf

6 Submission

This assignment must be submitted online via the link on the module pages on Blackboard. The **deadline** for this assignment is **13:00** on **Wednesday 11th December 2019**.

You should submit your assignment as a single **ZIP** file - **all other file types will be ignored**.

Your ZIP archive may contain **pdf** and **.ino** files only - **all other file types will be ignored**. In particular, please do not submit Word files (**.doc** or **.docx**).

7 About the path to follow

Drawings of parts of the path to follow are downloadable from Blackboard as pdf files.