Last updated on Saturday, March 15, 2014

2014 INB860 - Assignment description

Assessment information

- Demos to tutor during the lab sessions. Demos start in **Week 06** (see detailed schedule at the end of this document)
- Submit report by the end of **Week 13 (Sunday 11.59pm)**
- Use *Blackboard* to submit your work
- Group size (from 1 to 3). Groups of size 3 have extra tasks to complete compared to groups of size 2. These extra tasks are tagged with ***

Overview

In this assignment, you will design and implement a collection of behaviors on the Lego Mindstorms robot.

You will demonstrate your programs to the tutor during lab sessions starting in Week 06. The tutor will record whether or not you have solved satisfactorily the different challenges in a criteria sheet.

The robotic arenas will be laminated white posters (size A1 or A0) with dark lines (electrical tape) and grey patches. You are only allowed to use the set of sensors contained in a single kit.

All *final messages* should be displayed for **10 seconds** on the brick. Beep each time you display a new message. In order to preserve the mental heath of everyone, set the volume of the Lego brick on “low”.

The absolute value of the angle between two consecutive branches of an intersection is constrained to be between 30 degrees and 150=180-30 degrees. This means that you can detect that you are passing an intersection by following an edge of the tape and monitoring a virtual compass.

Behavior List

Elementary skills and Matlab simulation

Behavior A1

Starting at one endpoint of a polygonal line (a chain of straight segments), follow the line and beep at each segment junction (one beep for each junction). Stop when reaching the other endpoint of the polygonal line.

Behavior A2

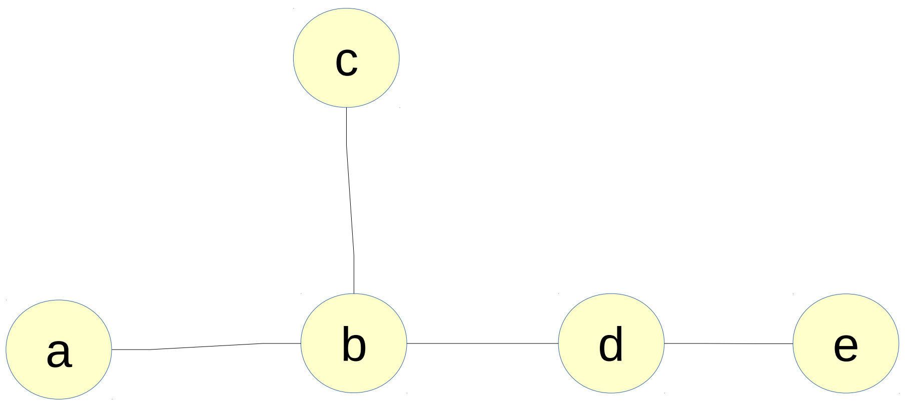
While performing the previous behavior, display your virtual compass value. When the robot starts the compass value should be reset to zero. Initially, negative angles should correspond to the right of the robot (therefore positive angles are to the left of the robot). Full mark for maximum error less than 5 degrees, half-mark for maximum error less than 10 degrees. No mark for larger errors.

Behavior A3 (Matlab simulator)

Drive randomly for a fixed number *n* of steps (for instance, *n*=12) starting at a leaf of the tree (leaf and *n* chosen by the tutor). After the *n* steps, backtrack to the starting place. Stop when arriving back at the starting position.

Behavior A4 (Matlab simulator)

As in the previous behavior, drive randomly for a fixed number of steps *n* starting at a leaf of the tree. After *n* steps, return to the starting place using the shortest route. That is, without making unnecessary detours. For example in the Figure below, if the robot random walk is “abcbde”, the robot should return to “a” with the route “edba” (skipping “C”)



Drawing 1: Simple Tree

Behavior A5 (Matlab simulator)

Explore a tree network driving along the segments. While exploring the network you should print on the console a label number for the current intersection (you are naming the intersections with numbers). If this is the first time you visit an intersection you should display "New node *k*" where *k* is the node label. If the node is already known (visited), you should display "Known node *k*" where *k* is the node label that you assigned to the node during its first visit. Stop when you have visited all the nodes. Display a final message "*n* nodes found" where *n* is the total number of nodes in the tree (including leaves).

we don't care about the angle, only care about how many branched we have at an intersaction.
using fprintf to print the message into console.

Behavior A6

Drive straight until you detect a black line. When you reach the line, orient the robot perpendicularly to the line. Full mark for an error less than 5 degrees, half-mark for an error less than 10 degrees. No mark for larger errors.

Behavior A7

Given a rectangular frame, drive the robot to the center of the frame, then stop. The robot is deemed to be at the center of the frame if the distance between its light sensor and the actual center of the frame is less than 2cm. The sides of the frame used for the testing of this behavior will be at least as wide as an A4 sheet of paper (21cm).

Tree and grey patches

The environment is a network **without cycles** of dark segments, and grey patches.

Behavior B1

Move along a straight dark line until you reach an intersection or the segment ends. When a stopping condition is reached, beep and display the message "*# branches = n*" where *n* is the number of branches of the intersection. If the robot has reached a leaf (that is, the segment ends), you should display "number of branches = 1" (a node leaf has exactly one branch).

Behavior B2

Starting at a leaf node (chosen by the tutor), explore all nodes of the network. Stop when you have explored all the nodes. Report the number of nodes found. You should not visit a node more times than needed (for example, visiting a leaf twice is wrong).

Behavior B3

Same as with previous behavior, but if a grey patch is found, beep and return directly (shortest route) to the starting node and stop.

Behavior B4

Explore the tree network driving along the segments. While exploring the network you should display at each intersection the number of branches on one line of the brick, and on another line you should display a label number for the current intersection (you are naming the intersections with numbers). If this is the first time you visit an intersection you should display "*New node n*" where *n* is the node label. If the node is already known (visited), you should display "*Known node n*" where *n* is the node label that you assigned to the node at the first visit. Stop when you have visited all the nodes. Display a final message "*n nodes found*" where n is the number of node in the tree (including leaves!).

Behavior B5

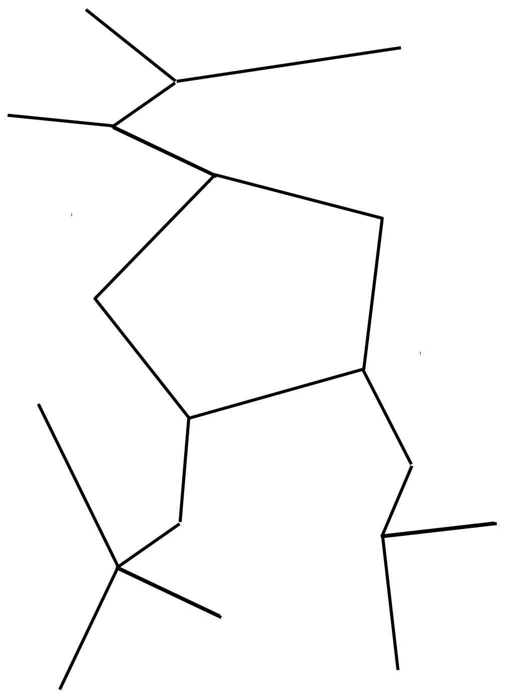
While performing the previous behavior, create a text file to represent the tree incidence matrix with the following format; each line corresponds to a node. First number is the node label, the remaining numbers on the line are the labels of its neighbors.

*** Behavior B6 ***

Explore the tree network driving along the segments like in the previous behaviors. When you find a grey patch (grey patches are only located at leaf nodes), beep, pause the motors and display for 5 seconds the message "*patch detected at node n*" where *n* is the label of the leaf node. After the 5 seconds, resume the exploration until you have visited all the nodes or you come across a second patch. Beep and display a message if you detect a second patch. Once you have explored the whole tree or come across a second patch, beep and display a message "returning to first patch". Then return to the leaf where the first patch was detected, and display a final message "first patch was here". The tutor will remove the patch after its initial detection. This implies that you have to record in a map where the patch was.

Flower

The environment is a “flower network” that is made of a cycle and subtrees attached to this cycle. The whole network has a unique cycle. An example of such a network is shown in the Figure below.



***** Behavior C1 *****

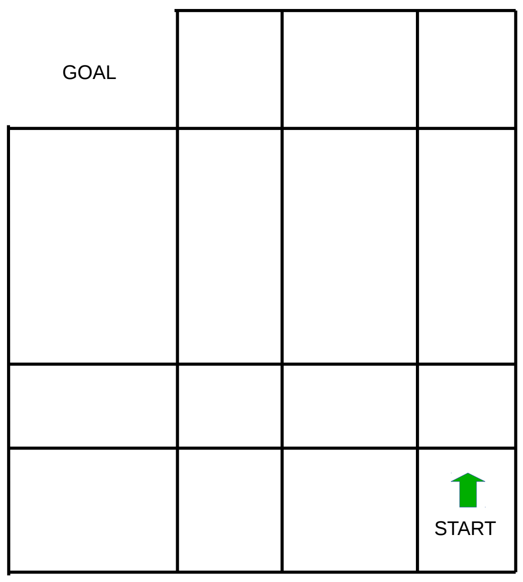
Explore the flower network until you find the find the cycle. Move inside the cycle and beep triumphantly.

Dead reckoning
sum up distance*tan(compass)

closing loop

Irregular grid

The environment is an irregular grid similar to the Figure below. The aim of the robot is to get to the goal position (the cell with no West and North borders) starting from the cell at the South-East corner of the grid. The robot will wake up looking North and has to navigate inside the grid.



Behavior D1

Go to the goal visiting the minimum number of cells. Stop when reaching the goal and display the number of cells visited (including the start and goal cells). Time bonus!?

Behavior D2

Go to the goal (traveling inside the grid) using the "cheapest path". The cost of traversing a cell is defined as follows. Let *(w,h)* denote the width and height of a cell. If the cell contains a grey patch at its center, the cost of traversing the cell is *w*+*h*, otherwise if the cell is empty the cost is only max(*w*,*h*). As you travel, you need to beep and display the width and height of the current cell as soon as it is known.

Pattern recognition

The environment is a set of polygonal lines (patterns). One endpoint will be covered with a grey patch, the other endpoint will be naked.

Behavior E1

Starting facing a pattern, continue in a straight line until you reach the pattern. Go the the grey patch, beep and log the compass direction to a text file while traveling from the grey patch to the other endpoint.

Behavior E2

Record (learn) up to six different training patterns in a different text files. Upload the files to a PC and plot the training patterns in Matlab figures.

Behavior E3

For this behavior, you have first to learn some training patterns. When presented with a test pattern, report the index of the most similar learned pattern. You have to use dynamic programming to compare the patterns. *There will be a lecture on dynamic programming and edit distance.*

***** Behavior E4 *****

Same as previous behavior, but this time the classification should be scale invariant. That is, if I make a copy of learned pattern number 3 with its size doubled, and present this resized copy, you should still classify it as pattern number 3.

Demonstration Schedule

Behavior	a1	a2	a3	a4	a5	a6	a7	b1	b2	b3	b4	b5	*b6*	*c1*	d1	d2	e1	e2	e3	*e4*
Week	6	6	6	6	7	7	7	9	9	10	10	10	11	11	12	12	13	13	13	13
Marks	3	4	4	4	4	4	4	6	4	3	3	4	4	6	4	8	2	3	8	4

Deliverables

On top of demonstrating the robot behaviors to the tutor, you should submit via Blackboard a zip file containing

1. A README.TXT file containing a table of contents of the zip file you are submitting
2. A report limited to 6 pages of text (be concise!)
   - explaining concisely your approach for each behavior
   - describing the performance and limitations of the behaviors you have implemented
3. A statement of completeness (list which behaviors you have attempted, whether they meet the specifications and whether there are any limitations)
4. All c files with proper commenting.
5. A report limited to 9 pages (pdf format)

Draft Marking Scheme

- Report: 20 marks
- Code quality (readability, simplicity, structure, genericity, in line documentation): 20 marks
- Each completed challenge: as indicated in the schedule table

Final Remarks

- Do not underestimate the workload. Start early. You are strongly encouraged to ask questions during the practical sessions.
- Email questions to f.maire@qut.edu.au

Have fun while learning!