MECH 452 – Mechatronics Engineering

Laboratory #2 – Navigation by Contact with the Robot

In this lab you will be introduced to the operation of the LynxBot mobile robot illustrated in **Fig. 1**. The task is to program your robot to navigate a test arena with bump switches as sensors.

Part One – Familiarization (with the BotBoarduino)

On your Stack (not the robot), upload ***M452Introswitch*** and confirm its operation:

a) Green LED flashing, 1st button press to start
b) Green/yellow/red LED flashing, pauses on red for 500 *msec*
c) 2nd button press to stop, green/yellow/red LED flashing, no pause on red

Make a copy of ***M452Introswitch***, rename it as ***M452Lab2LynxSwitch*** and run it on your LynxBot after modifying it such that:

- Pushbutton **PBB** acts as the 1st button press (use **Pin d8**, as in **IntroSwitch**)
- Bumper switch acts as the 2nd button press (use **Pin d13,** need to declare a new variable)

Fig. 4 gives the pin layout of the BotBoarduino and identifies its key features. Run ***M452Lab2LynxSwitch*** and confirm that the program ***doesn't*** work. Identify and fix the problem. Demonstrate your working program to an instructor before proceeding to **Part Two**.

Part Two – Calibration (of the LynxBot)

a) Go to the course **onQ** website, select the program ***M452Lab2UpDown*** and download it to you workstation. Upload the program to your robot. Sequence of operations is:
   - Green LED flashes once program is loaded and ready to go
   - Open the Serial Monitor window, speed (not ***delta***) will be displayed
   - Press **PBB** to start the program, green LED on
   - Once started, press **PBB** to toggle increment step size between 1 and 5, then press **PBA** and **PBC** to increment and decrement the speed, respectively
   - Press front bumper to stop the program, green LED flashes, press **PBB** to repeat

   Put robot on pedestal (to begin, ***INIT_SPEED*** = 155, ***increment*** = 1 and ***STOP_SPEED*** = 155):
   - To get the deadzone, reduce speed by 1 (with **PBC**) until wheels stop, then continue until wheels restart, this gives upper and lower values of the deadzone
   - To get ***maxforward*,** with ***INIT_SPEED*** = 230, ***STOP_SPEED*** = deadzone, reduce speed by increments of 5 (with **PBC**), then by increments of 1 (with **PBB**), until motor stops
   - To get ***maxbackward***, same procedure as for ***maxforward***, but ***INIT_SPEED*** = 70**,** increase speed by increments of 5 (with **PBA**), then by increments of 1, until motor stops

   You should repeat the above measurements at least three times, to get a measure of repeatability.

b) Go to the ***onQ*** website, select the program ***M452Lab2Forward*** and download it to your workstation. Upload the program to your robot. Sequence of operations is:
   - Green LED flashes when ready to go
   - Press **PBB** to start the program, green LED goes on, robot goes forward for 1 sec
   - Yellow LED goes on, robot pauses for 5 sec
   - Red LED goes on, robot reverses for 1 sec
   - Green LED goes on, robot stops
   - Green LED flashes (2 sec after the stop), operation can be repeated

If robot fails to go forward, but does go backwards, wait for flashing green LED and repeat the operation.

Fig. 1. Test setup for **Part 3** of **Lab #2**.

Set ***STOP_SPEED*** to the midpoint of the deadzone. Check that your ***unloaded*** battery state is between 12 and 13 V. Put the robot on the floor and run the program. With your measuring tape, measure the distance travelled in 1 sec. Set your speeds by increasing ***delta*** from 0 to ***Maxforward*** by increments of 15. Perform three trials for each ***delta*** (recording forward and backwards distances for a given trial). Speed will be the average of the three trials. Remember that each time you change ***delta***, you will have to re-upload the program. Generate a plot similar to that in **Fig. 2**. You must show your completed **Fig. 2** to an instructor before proceeding to **Part Three**. You will be asked for the values for ***STOP_SPEED***, and the upper and lower deadzone.

Table 2. Calibration data

Comment	Pulse Counts Part a)	Pulse Counts Part b)	delta	Speed (m/s)
start point	230			
max forward	214			
		206	60	0.99
		191	45	0.88
		176	30	0.62
		161	15	0.28
upper for stop		147		0.00
deadzone		146	0	0.00
lower for stop		145		0.00
		131	-15	-0.34
		116	-30	-0.66
		101	-45	-0.93
		87	-60	-0.93
max backward	78			
start point	70			

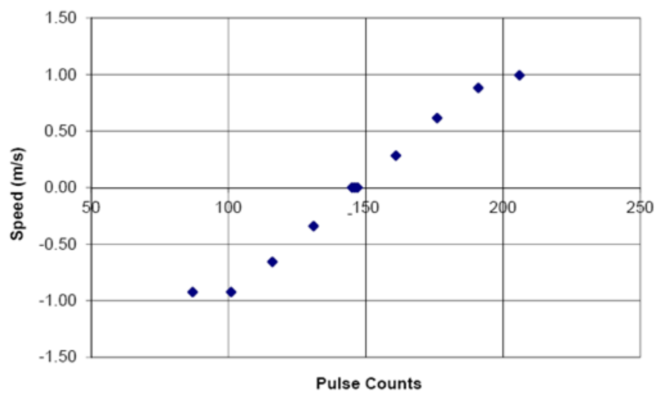


Table 1. Raw speed data

	Distance (cm)						Average Speed (m/s)	
	1st test		2nd test		3rd test		m/s=distance cm/100	
delta (+/-)	forward	back	forward	back	forward	back	forward	back
60	99	-92	99	-92.5	100	-93	0.99	-0.93
45	88	-92	88	-92.5	89	-93	0.88	-0.93
30	61	-66.5	62.5	-65	61	-65.5	0.62	-0.66
15	29	-34	28	-33.5	28	-35	0.28	-0.34

Fig. 2. Sample results, yellow cells measured (others calculated), ***M452Lb2TableBlank*** onQ.

Part Three – Navigation Task

Test arena is shown in **Fig. 1**. Make a copy of ***M452Lab2Forward*** and rename it ***Group#Lab2BumpTurn***. Initially, your robot will be placed in the home position (rear bumper between red lines, touching ***and*** square to wall). Modify the program such that:

- robot moves forward and when opposite wall is touched by a bump switch, robot reverses an appropriate distance, pivots 90 *deg* counterclockwise and resumes its forward motion
- "move forward, touch, reverse and turn" cycle repeats itself four times (four walls)
- at the end of the final (4th) turn, the robot reverses to the home position, and stops with rear bumper between the lines and touching the wall with the rear bumper in two places
- required LED sequence: a) program ready, flashing green, b) moving forward, solid green, c) wall hit, solid red (0.5 sec pause before reversing), d) moving backwards, solid yellow (0.5 sec pause before pivoting) and e) program end, all LED's flashing

Rules for the Task (robot testing)

1) It is the student's responsibility to ask questions if they are unsure of the rules.
2) There is no touching of the robot once the robot starts to move. If anything falls off the robot, don't touch.
3) Trial ends if robot gets stuck (i.e. switch fails to switch, robot hangs on a wall).

4) Between trials, there can be no reprogramming or mechanical alterations, except for mechanical and electrical repairs (i.e. repairs to broken wires or disconnected sensors).
5) Repositioning of batteries, removal of rear bumper, rewiring of front switches, not allowed.
6) If a robot stalls due to a poor battery connection, trial can be repeated, but an "air" test is required to confirm that the connection was the source of the problem.
7) A trial doesn't start until the robot starts moving forward (in the event of a hung start due to a sleeping motor controller).

Class Testing Session

This lab will run over two lab sessions, with a class test is the last hour of the second lab session. In the test, students will be required to demonstrate their robot's ability to complete the navigation task of **Part Three**. During the test, students will be asked to report: a) Unloaded 12 V battery voltage, b) value(s) for ***delta***(s) and c) anything noteworthy about their lab experience, or anything novel about their coding. Each test consists of three trials, with a press of **PBB** to start the program. A perfect score on a trial means subsequent trials aren't required.

1.0 – correct LED sequence
2.0 – front bumper contacts each wall in sequence, and only once (i.e. 0.5 per wall touch)
1.0 – when stopped, robot has two points of contact (or 0.5 if only one point of contact) where "contact" means rear bumper touching the wall
1.0 – finishes inside (or on) inner home lines (or 0.5 if inside (or on) outer home lines), but
__ only if also stopped with two points of contact (i.e. 0 marks if only one point of contact)
5.0 – individual trial subtotal

Overall marking scheme for the lab is
5.0 – best trial mark
0.5 – basic flowchart (submitted separately, replicate ***Fig. 3*** with software of your choice)
4.5 – report
10.0 – marks total

Report

Please refer to **onQ** for the due date of the report**.** Submit your report as an Adobe (***.pdf***) file by uploading to **onQ**. Use ***Group#Lab2Report*** as the filename. The report must include:

a) **Summary** section.

b) **Design** section with description of design and two photos (1st = close-up of robot showing bumper placement and 2nd = the testing setup, as in **Fig. 1**).

c) **Program** section with flowchart for ***Group#Lab2BumpTurn*** and text that explains the flowchart and the sequence of operations. Flowchart should provide more detail than seen in **Fig. 3,** for example what "ReversePivotPause" and "GoHome" are about. But not too much detail.

d) **Results** section, which must include:

i) Calibration results (**Table 1** and **Table 2** and plot using the template) with supporting text.

ii) Answer to the question: "What is the biggest source of the deadzone seen in **Fig. 2** and why does it vary so much between the LynxBots? Cite a URL to support your answer.

iii) Discussion on what didn't work and what did work, as you developed your code.

e) **Appendix** section with listing of ***Group#Lab2BumpTurn*** (no need to highlight changes).

As always, before submitting your report, please review the marking rubric for **Lab #2** as posted to **onQ**

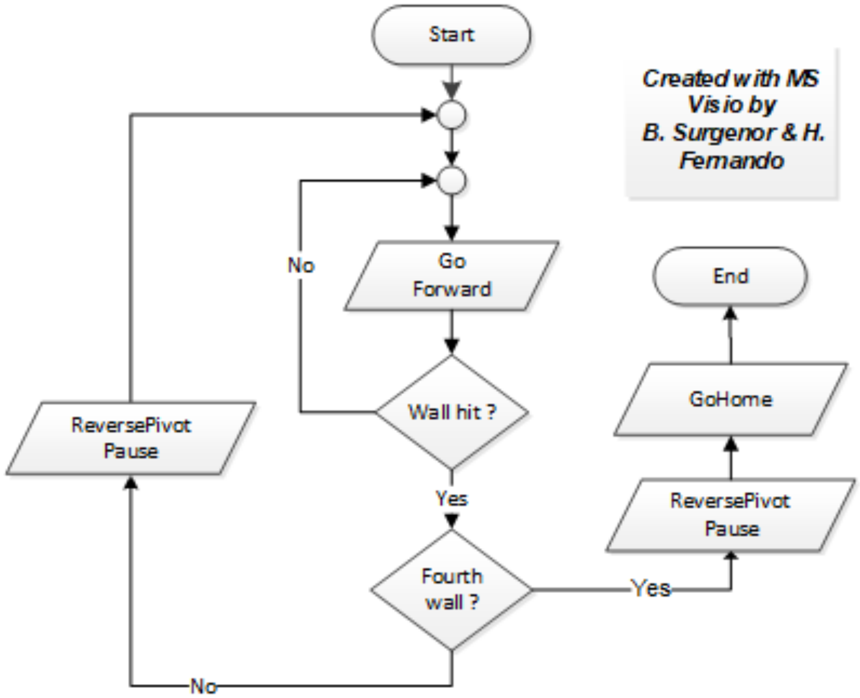
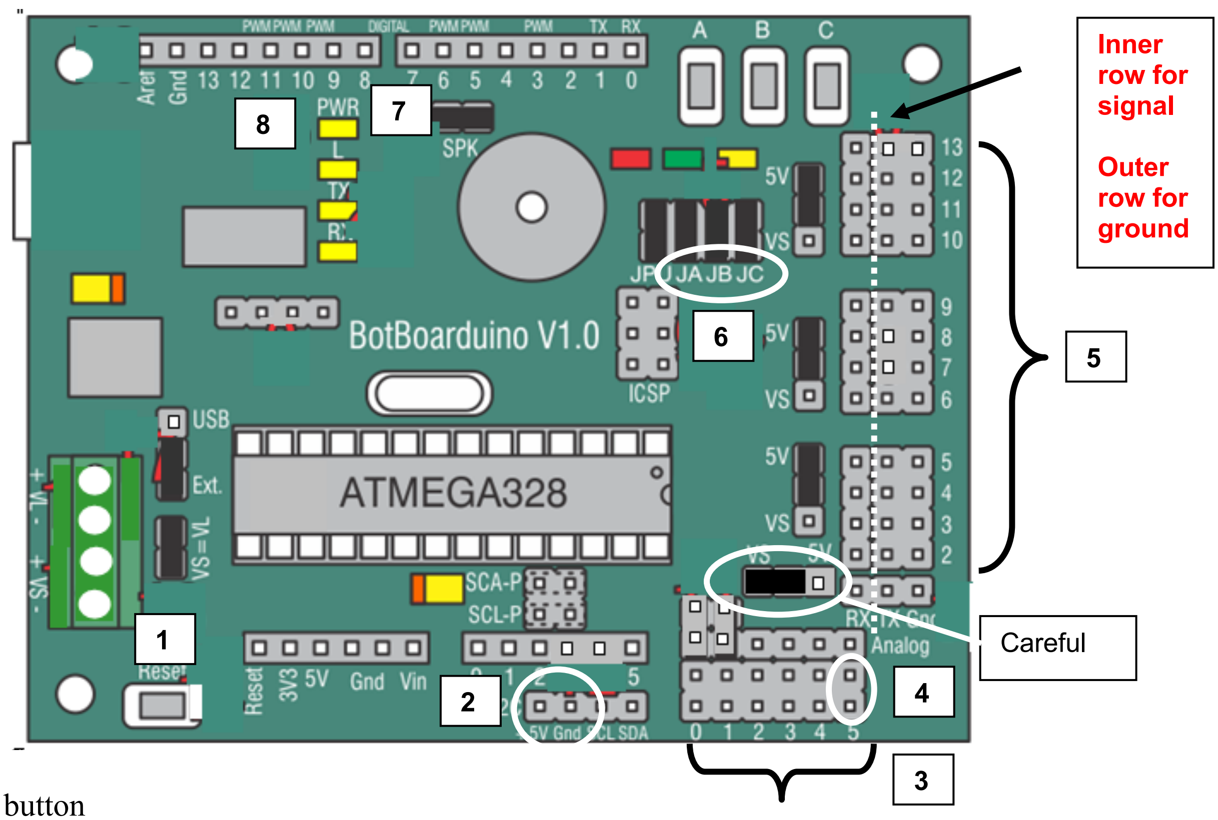


Fig. 3. Basic flowchart fo**r Lab #2** navigation by contact.



1 = reset button
2 = ground and 5v supply
3 = analog I/Os (a0 to a5)
4 = connection for LED battery indicator
5 = digital I/Os (2 to 13)

6 = jumpers to enable pushbuttons A, B, C (as connected to ***pins d7, d8*** and ***d9***)
7 = jumper to enable speaker (on ***pin d5***)
8 = power indicator

Fig. 4. BotBoarduino pin layout and key features.