1 Lab Scope and Aim

The aim of this lab is to introduce the students to the basic functionalities in motion and control of a mobile robot. The robot that you will use in this lab is the turtlebot3 burger and you will be developing your control and motion algorithms within Simulink framework. This lab assignment involves (a) an intro to the Turtlebot robot (b) an intro to basic commands to use your Turtlebot from the command line and from a remote laptop (c) template code that we give you to make the bringup your robot. The main aim of this lab is to experiment with different control strategies so that the turtlebot follows the desired motion patterns.
Prerequisites: The students are suggested to have a good background in the following areas:
- *Automatic Control Systems*
- *Advanced Mathematics*
- *Complete Lab1 Tasks*
- *Read Turtlebot manual which is uploaded in Canvas*
- *Simulink*

Lab Supplementary material: This lab is accompanied with the supplementary material to support the assignments. The material is included in the lab2.zip file.

Note1: You should always run your algorithms in your local machine following the instructions provided in section 3
Note2: Always locate the robot on the ground while doing experiment.
Note3: Always make sure that the battery is disconnected after finishing your work with turtlebot.
Note4: Never run the robots with low level of the battery, the robot makes warning sounds when the battery has low level.
Note5: Never charge the battery while it is connected to the robot.
Note6: In case of emergency press STOP slider bottom in Simulink.
Note7: For Tasks 1 to 4 use "Lab2_Task_1234.slx"
Note8: For Tasks 5 to 8 use "Lab2_Task_5678.slx"

2 Simulink

The lab implementation is structured around Simulink. Simulink, developed by MathWorks, is a graphical programming environment for modeling, simulating and analyzing dynamical systems. Its primary interface is a graphical block diagramming tool and a customizable set of block libraries. More information regarding simulink can be find in `https://se.mathworks.com/products/simulink/getting-started.html`

3 TurtleBot Burger

TurtleBot3 (Figure 1) is a small, affordable, programmable, mobile robot to use in education, research, hobby, and product prototyping. The TurtleBot3 can be customized into various ways depending on how you reconstruct the mechanical parts and use optional parts such as the computer and sensor. In addition, TurtleBot3 is evolved with cost-effective and small-sized single board computer that is suitable for robust embedded system, 360 degree distance sensor and 3D printing technology.

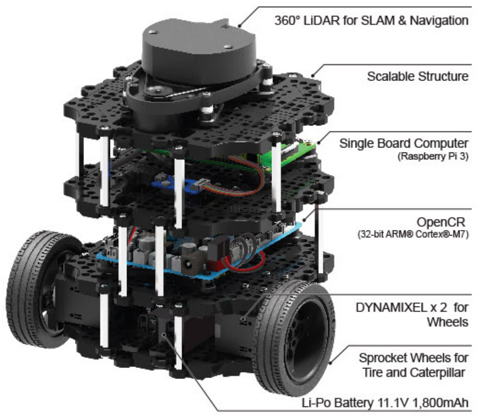

360° LiDAR for SLAM & Navigation
Scalable Structure
Single Board Computer (Raspberry Pi 3)
OpenCR (32-bit ARM® Cortex®-M7)
DYNAMIXEL x 2 for Wheels
Sprocket Wheels for Tire and Caterpillar
Li-Po Battery 11.1V 1,800mAh

Figure 1: TurtleBot3

3.1 Turtlebot Bringup

This section provides instructions on how to connect to Turtlebot from your local machine.

SSHing onto your Turtlebot Before running your code on your local machine you should remotely connect to the turtlebot and start the basic functionalities. Initially, you need to make sure that you desktop-pc is connected to the following WiFi network
SSID:*turtlenet*
Password:*R7010E_robotics*
Then connect the battery and turn on the Turtlebot's Rasberry Pi using the button as shown in Figure 2.

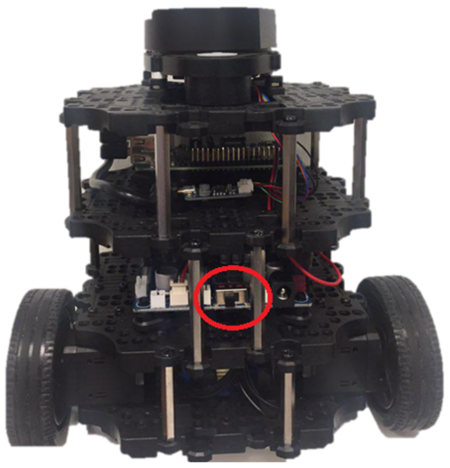
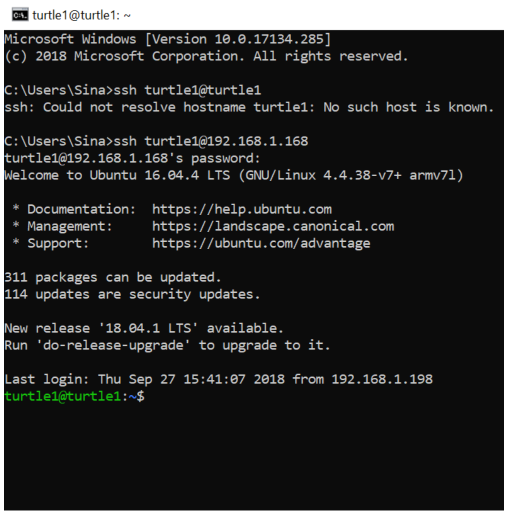


Figure 2: On-Off button of the Turtlebot.

Each turtlebot has unique name and IP address, which are written on the platform. To get remote access to the turtlebot open a command window (cmd) on your local machine and run the command:

ssh turtle@192.168.1.X0
where you should replace X with your turtlebot id. For example for turtlebot 2 use the password
ssh turtle@192.168.1.20

You'll be prompted to enter turtlebot's password. Each turtlebot has unique password.
Password:*turtleX*
where you should replace X with your turtlebot id. For example for turtlebot 5 the password is *turtle5*
Now you are connected to Turtlebot. The screenshot of example is depicted in Figure 3



```
turtle1@turtle1: ~
Microsoft Windows [Version 10.0.17134.285]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Sina>ssh turtle1@turtle1
ssh: Could not resolve hostname turtle1: No such host is known.

C:\Users\Sina>ssh turtle1@192.168.1.168
turtle1@192.168.1.168's password:
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.38-v7+ armv7l)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

311 packages can be updated.
114 updates are security updates.

New release '18.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Thu Sep 27 15:41:07 2018 from 192.168.1.198
turtle1@turtle1:~$
```

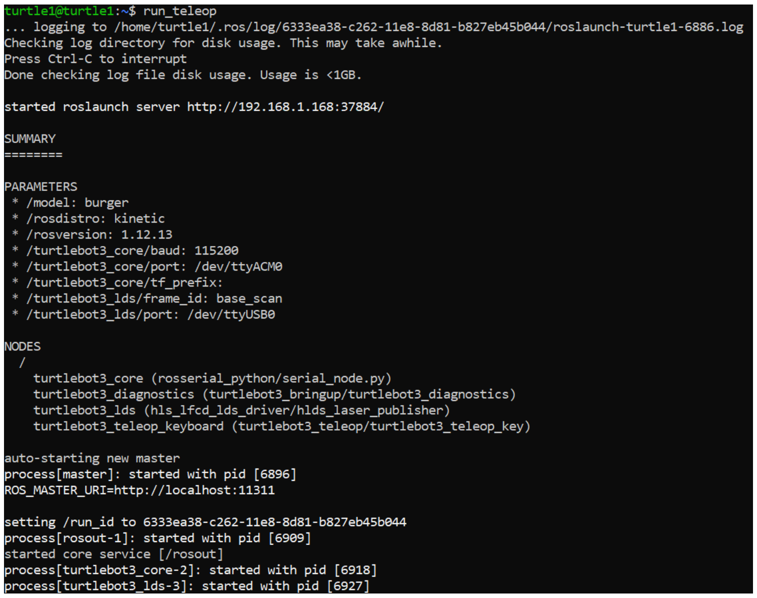
Figure 3: Example for connecting to turtlebot from command windows.

Example to tele-operate turtlebot Now that you have remote access to the turtlebot you can run an example program to tele-operate it using keyboard keys and get familiar with the robot. In the command window with the remote access type the command (Figure 4):
run_teleop
This command initializes the required software onboard Turtlebot. After the robot initialization, the commands for tele-operation are depicted in the following table.

Table 1: Keyboard keys for tele-operating turtlebot.

Key	Direction
w	$\uparrow$
a	$\leftarrow$
d	$\rightarrow$
x	$\downarrow$
s	STOP
crtl+c	TERMINATE

Stop the robot before terminating the cmd windows, using the command shown in Table 1.



```
turtle1@turtle1:~$ run_teleop
... logging to /home/turtle1/.ros/log/6333ea38-c262-11e8-8d81-b827eb45b044/roslaunch-turtle1-6886.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.1.168:37884/

SUMMARY
========

PARAMETERS
 * /model: burger
 * /rosdistro: kinetic
 * /rosversion: 1.12.13
 * /turtlebot3_core/baud: 115200
 * /turtlebot3_core/port: /dev/ttyACM0
 * /turtlebot3_core/tf_prefix:
 * /turtlebot3_lds/frame_id: base_scan
 * /turtlebot3_lds/port: /dev/ttyUSB0

NODES
  /
    turtlebot3_core (rosserial_python/serial_node.py)
    turtlebot3_diagnostics (turtlebot3_bringup/turtlebot3_diagnostics)
    turtlebot3_lds (hls_lfcd_lds_driver/hlds_laser_publisher)
    turtlebot3_teleop_keyboard (turtlebot3_teleop/turtlebot3_teleop_key)

auto-starting new master
process[master]: started with pid [6896]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 6333ea38-c262-11e8-8d81-b827eb45b044
process[rosout-1]: started with pid [6909]
started core service [/rosout]
process[turtlebot3_core-2]: started with pid [6918]
process[turtlebot3_lds-3]: started with pid [6927]
```

Figure 4: Example to tele operation of the Turtlebot with keyboard.

3.2 Basic operation

In order to use the Turtlebot's sensors and motors, you'll need to terminate any running program in the turtlebot using "crtl+c" (tele-operation command). Then type the command (Figure 5):
run
This command will initialize the robot functionalities and should be always running in the background when you want to execute your algorithms. In case you need to reset the main functionalities, terminate the running program in the turtlebot using "crtl+c" and start it again. Remember that you only need one command window to get the remote access and start the onboard functions of the robot.

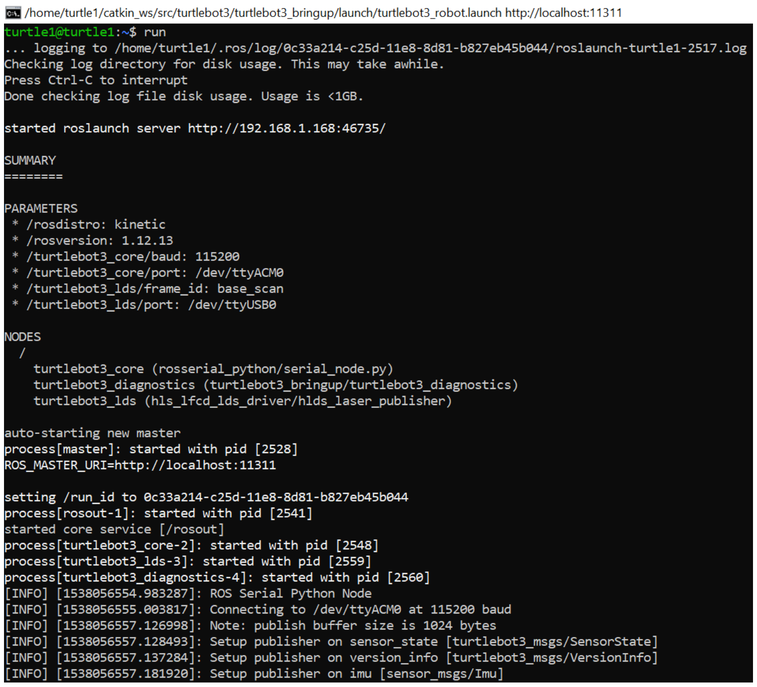
Running your code on your laptop

3.3 MATLAB

Now that the robot is operating you should properly configure your local machine to communicate with it. Open MATLAB in your computers. Then open the Simulink file that corresponds to each task for example "Lab2_Task_1234.slx". Follow Figure 6 steps to open Configure Network Addresses. Now, you must type the IP of your device in the window which is depicted in Figure 7. Press test to ensure your connection to the robot. Now you are ready to move to Assignment section.

4 Assignments

Task 1 (Moving to a point) In this task you should develop your controller to move turtlebot from Point A $[x_A, y_A]$ to Point B $[x_B, y_B]$. Repeat the experiment with different points B.


```
/home/turtle1/catkin_ws/src/turtlebot3/turtlebot3_bringup/launch/turtlebot3_robot.launch http://localhost:11311
turtle1@turtle1:~$ run
... logging to /home/turtle1/.ros/log/0c33a214-c25d-11e8-8d81-b827eb45b044/roslaunch-turtle1-2517.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.1.168:46735/

SUMMARY
========

PARAMETERS
 * /rosdistro: kinetic
 * /rosversion: 1.12.13
 * /turtlebot3_core/baud: 115200
 * /turtlebot3_core/port: /dev/ttyACM0
 * /turtlebot3_lds/frame_id: base_scan
 * /turtlebot3_lds/port: /dev/ttyUSB0

NODES
  /
    turtlebot3_core (rosserial_python/serial_node.py)
    turtlebot3_diagnostics (turtlebot3_bringup/turtlebot3_diagnostics)
    turtlebot3_lds (hls_lfcd_lds_driver/hlds_laser_publisher)

auto-starting new master
process[master]: started with pid [2528]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 0c33a214-c25d-11e8-8d81-b827eb45b044
process[rosout-1]: started with pid [2541]
started core service [/rosout]
process[turtlebot3_core-2]: started with pid [2548]
process[turtlebot3_lds-3]: started with pid [2559]
process[turtlebot3_diagnostics-4]: started with pid [2560]
[INFO] [1538056554.983287]: ROS Serial Python Node
[INFO] [1538056555.003817]: Connecting to /dev/ttyACM0 at 115200 baud
[INFO] [1538056557.126998]: Note: publish buffer size is 1024 bytes
[INFO] [1538056557.128493]: Setup publisher on sensor_state [turtlebot3_msgs/SensorState]
[INFO] [1538056557.137284]: Setup publisher on version_info [turtlebot3_msgs/VersionInfo]
[INFO] [1538056557.181920]: Setup publisher on imu [sensor_msgs/Imu]
```

Figure 5: Example to bringup turtlebot.

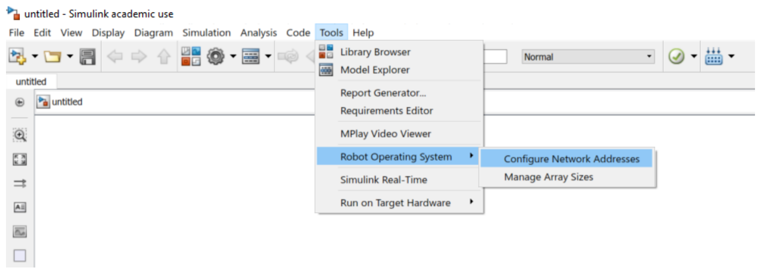
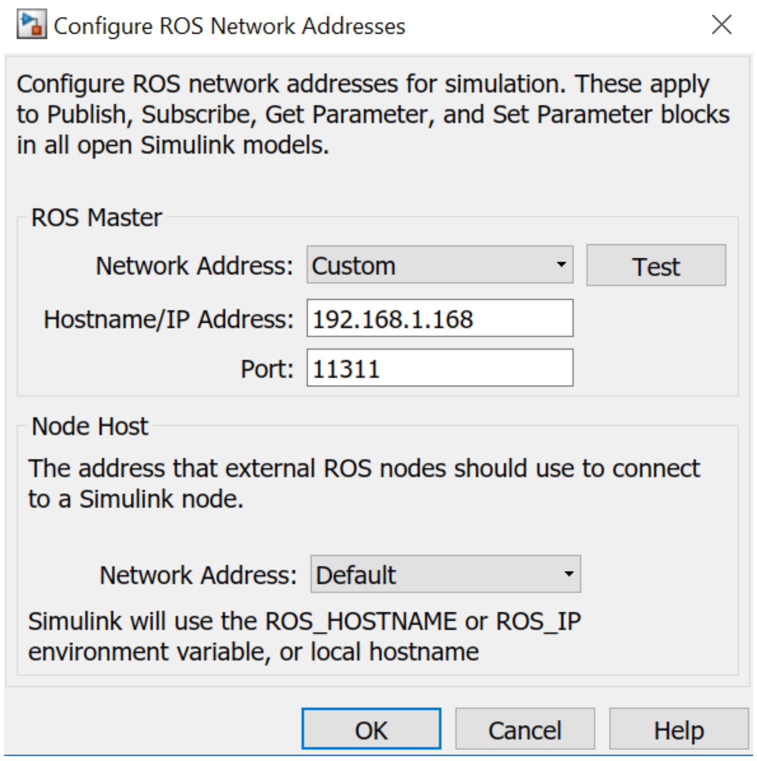


Figure 6: Bridge between turtlebot and Simulink.



Configure ROS Network Addresses

Configure ROS network addresses for simulation. These apply to Publish, Subscribe, Get Parameter, and Set Parameter blocks in all open Simulink models.

ROS Master

Network Address: Custom

Hostname/IP Address: 192.168.1.168

Port: 11311

Node Host

The address that external ROS nodes should use to connect to a Simulink node.

Network Address: Default

Simulink will use the ROS_HOSTNAME or ROS_IP environment variable, or local hostname

Figure 7: Example of configuring IP.

Task 2 (Moving to a pose) In this task you should develop your controller to move turtlebot from initial pose A $[x_A, y_A, \theta_A]$ to pose B $[x_B, y_B, \theta_B]$. Repeat the experiment with different poses B.

Task 3 (Following a trajectory) In this task you should develop your controller to move turtlebot along a trajectory. For this task your turtlebot should be able to follow the trajectories defined below:

- square $\rightarrow$ square pattern of your choice. Observe your robot repeating this square motion
- letter $\rightarrow$ the first letter of your last name
- curved path $\rightarrow$ e.g. sinusoid of your choice

Task 4 (Following a line) In this task you should develop your controller to follow a line on the plane defined by $ax + by + cx = 0$. For this task your turtlebot should be able to follow the lines with the coefficients defined below:

- $a = 1, b = -1, c = 1$
- $a = 1, b = -2, c = 4$

Task 5 (Lidar data plotting) In this task, you should plot the surroundings of your robot using ranges from lidar. Move your turtlebot close and far from objects to evaluate the min-max ranges of the lidar and report the limitations.

Task 6 (Keep distance from wall surface) In this task you should develop your controller to move turtlebot keeping a distance from an wall surface. In this case you should use measurements from lidar sensor. The schematic of the task is shown in Figure 8. In this task you can select how many and which beams you want to use from the lidar. Try your algorithms with different beam configurations and select the one that performs better, providing an overview in your report.

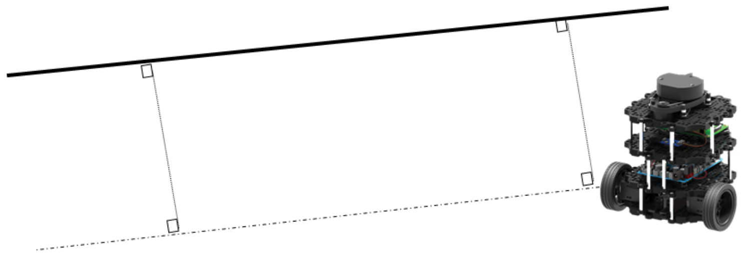


Figure 8: Example of path with constant distant to the wall.

Task 7 (Keep distance from uneven wall surface) In this task you should develop your controller to move turtlebot keeping a distance from an uneven wall surface. In this case you should use measurements from lidar sensor. The schematic of the task is shown in Figure 9.

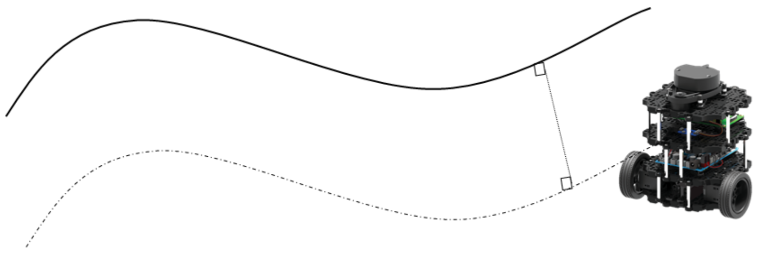


Figure 9: Example of path with constant distant to the uneven wall.

Task 8 (Lidar data registration) In this task, you should plot a map of your robot surroundings using ranges from lidar, while following a straight path. More specifically you should provide a plot that contains all lidar measurements registered during the motion of turtlebot. Hint: Convert ranges to x, y points and associate them with the robot pose in the local coordinate frame.