EMBEDDED SYSTEM DESIGN (MCTA 4392)

SECTION 1

SEMESTER 2 24/25

PROJECT REPORT

TITLE:
TURTLE BOT3 GUIDELINE

BIL	NAME	MATRIC NUMBER
1	MUHAMMAD NURHAKIMIE THAQIF BIN ABDULLAH	2213217
2	TASHFIN MUQTASID	2119609

GITHUB REPOS:

- https://github.com/KimieCrafter/Embeded-Design-RasPI
- https://github.com/muqtasid87/embedded-system-design/tree/main/Turtle bot

TABLE OF CONTENTS

Introduction

The TurtleBot3 is a compact, affordable, and versatile robot platform widely recognized as an excellent tool for learning and experimenting with robotics, particularly within the ROS (Robot Operating System) ecosystem. Equipped with sensors and actuators, it supports manual control via keyboard teleoperation and advanced functionalities like SLAM (Simultaneous Localization and Mapping), enabling it to autonomously map its surroundings while tracking its location. This report serves as a comprehensive guide for setting up and operating the TurtleBot3, offering step-by-step instructions on configuring the PC and TurtleBot, updating WiFi credentials, establishing connections, and leveraging its capabilities for both teleoperation and autonomous navigation.

PC Setup

Firstly, the PC needs to be set up with Ubuntu 22.04 LTS and ROS2 Humble, so that it can be used to control the turtlebot. Ubuntu can either be installed on the PC as a dual boot or purely Linux, or as a virtual machine.

1. Ubuntu Virtual Machine: The following YouTube video shows how to set up Ubuntu using VMware. (VMware is better in terms of performance than VirtualBox)
   https://www.youtube.com/watch?v=tkXQAeMYZwA
2. Setup ROS2 Humble
   a. Once Ubuntu has been installed, install ROS2 Humble and the Turtlebot3 packages by following the guide in the following link:
      https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/#pc-setup
   b. The setup is quite straightforward, just copy and paste each command into the Ubuntu CLI sequentially.
3. After step 2, update your environment variable:
   a. Accessing bash file

      ```
      nano ~/.bashrc
      ```

   b. Add the following line to the bottom of the bashrc file. Then press Ctrl + O, then Enter to save, and Ctrl + X to exit.

      ```
      export TURTLEBOT3_MODEL=burger
      ```

   c. then run to save to bash

      ```
      source ~/.bashrc
      ```

Turtlebot Setup

The OpenCR Board and the Raspberry Pi 3 has already been set up with Ubuntu Server 22.04.5 LTS, with all the necessary packages installed.

Login Details: Turtle Bot 1

```
Username: turtlebot3
Password: 1234
```

Login Details: Turtle Bot 2

```
username: KimieCrafter
password: 12345678
```

Updating WiFi Credentials

- To be able to use the Turtlebot with the PC, they need to be on the same WiFi network. To setup the WiFi in the turtlebot, follow the below steps:
    - Connect the Turtlebot3 to a screen and keyboard.
    - Login with the credentials above.
    - Enter the following commands:

```
cd /etc/netplan/
sudo nano 01-netcfg.yaml
```

Note: The turtlebot must be on different wifi between each bot. You can set up such as it on same network but additional method must be done

This is Common ID for both robots. Change the ID based on ros documentation if want to use on same network, ros domain id - 17

- Once the yaml file is open, enter the following details, replacing "YOUR-SSID" and "YOUR_PASSWORD", with the WiFi SSID and Password. Both SSID and Password should be under the double quotes.

```
network:
  version: 2
  renderer: networkd
  wifis:
    wlan0:
      dhcp4: true
    access-points:
      "YOUR_SSID":
        password: "YOUR_PASSWORD"
```

- Save the yaml file (Ctrl + O, Enter, Ctrl + X). Then:

```
sudo netplan apply
sudo reboot
```

Note: This will restart the bot and the bot should connect to the wifi if it is nearby. You can check via hotpots terminal in handphone if using hotspot or just simply use the screen for initial check

Connecting PC to Turtlebot

- Ensure both PC and Bot are connected to the same WiFi. Check the IP address of the Turtlebot using the WiFi admin interface. If the interface is unavailable, you can check the IP address by connecting the turtlebot to a screen and typing:

```
Ip a
```

Example:

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
     valid_lft forever preferred_lft forever
   inet6 ::1/128 scope host
     valid_lft forever preferred_lft forever

2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
group default qlen 1000
   link/ether 00:1e:67:df:32:7a brd ff:ff:ff:ff:ff:ff
   inet 192.168.1.42/24 brd 192.168.1.255 scope global dynamic eth0
     valid_lft 86344sec preferred_lft 86344sec
   inet6 fe80::21e:67ff:fedf:327a/64 scope link
     valid_lft forever preferred_lft forever
```

Here ip address is **192.168.1.42**

- Once you have the IP, ssh into the bot using code below, remove <>.

```
ssh turtlebot3@<ip_address>   # for bot 1
ssh kimiecrafter@<ip_address> # for bot 2
```

- Then login to the bot. After logging in, execute the following command so that the bot can be controlled by the PC from now onwards:

```
ros2 launch turtlebot3_bringup robot.launch.py
```

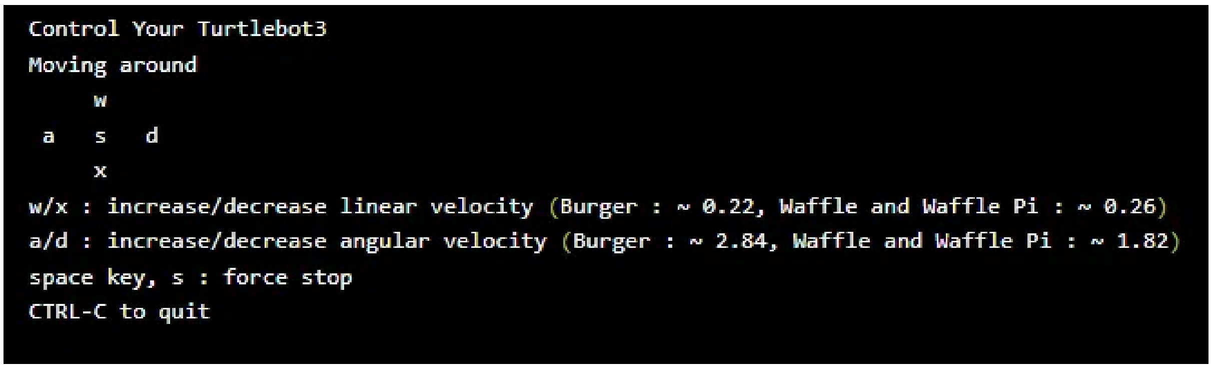
- The terminal in which this command is run needs to stay up. Open new terminal windows for all further work.

Controlling Turtlebot using Keyboard (Teleoperation)

- After the bringup has been executed, open a new terminal window on your PC and run the command:

```
ros2 run turtlebot3_teleop teleop_keyboard
```

The terminal should show this:



```
Control Your Turtlebot3
Moving around
     w
 a   s   d
     x
w/x : increase/decrease linear velocity (Burger : ~ 0.22, Waffle and Waffle Pi : ~ 0.26)
a/d : increase/decrease angular velocity (Burger : ~ 2.84, Waffle and Waffle Pi : ~ 1.82)
space key, s : force stop
CTRL-C to quit
```

While this terminal is active, the bot can be controlled with the WASD keys.

SLAM Node

What is SLAM?

Simultaneous Localization and Mapping (SLAM) is a key robotics technique that enables a robot to build a map of an unknown environment while simultaneously tracking its own location within that map. It's a critical capability for autonomous robots operating in spaces where pre-existing maps are unavailable.

How SLAM Works on TurtleBot3 with LDS-01

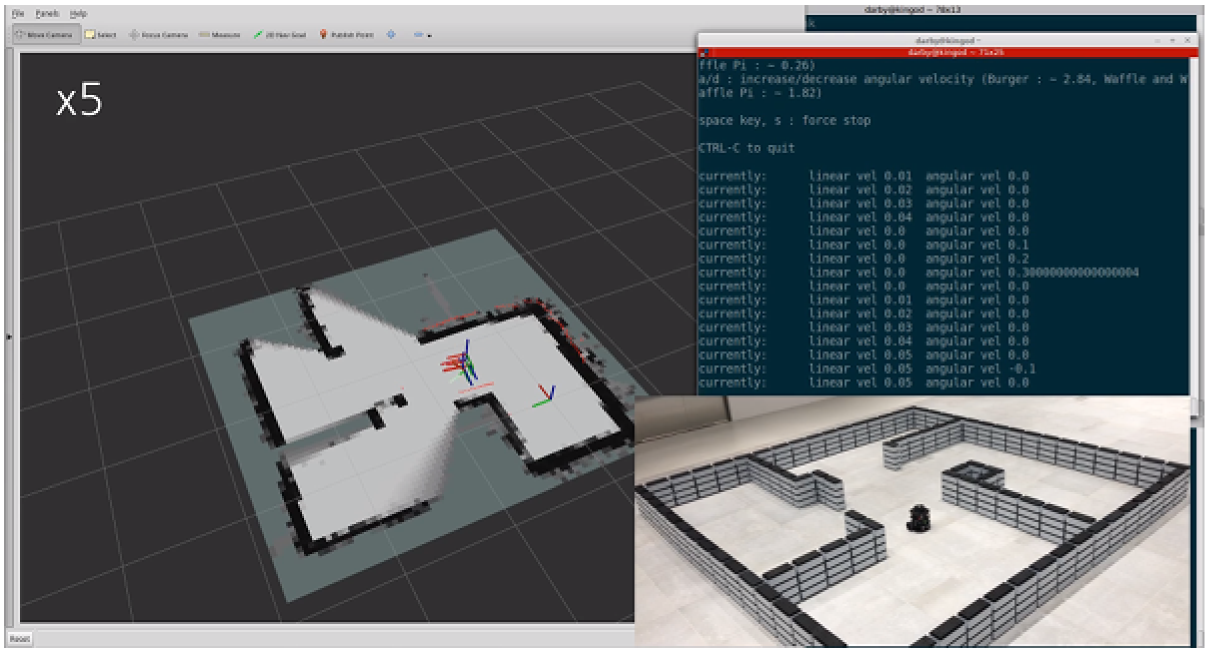
On the TurtleBot3, SLAM is powered by the LDS-01 (Laser Distance Sensor), a 360-degree laser scanner that measures distances to objects around the robot. As the TurtleBot3 moves, the LDS-01 collects real-time data about its surroundings. This data is processed by SLAM algorithms—specifically the Cartographer algorithm integrated with ROS2—to create a 2D occupancy grid map. This map shows free and occupied spaces, allowing the robot to detect obstacles and plan its navigation. By continuously updating the map and refining its position using the laser scan data, the TurtleBot3 achieves accurate, real-time mapping and localization, making it ideal for autonomous operation in dynamic environments.

To run the SLAM node, first the turtlebot needs to be connected to the PC as detailed in the steps above. Next, the SLAM mapping can be done by launching the cartographer node on the PC. https://emanual.robotis.com/docs/en/platform/turtlebot3/slam/#run-slam-node

NOTE

- Make sure to launch Bringup on the TurtleBot3 before executing any operations.

```
ros2 launch turtlebot3_cartographer cartographer.launch.py
```



```
ffle Pi : ~ 0.26)
a/d : increase/decrease angular velocity (Burger : ~ 2.84, Waffle and W
affle Pi : ~ 1.82)

space key, s : force stop

CTRL-C to quit

currently:      linear vel 0.01  angular vel 0.0
currently:      linear vel 0.02  angular vel 0.0
currently:      linear vel 0.03  angular vel 0.0
currently:      linear vel 0.04  angular vel 0.0
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel 0.0   angular vel 0.1
currently:      linear vel 0.0   angular vel 0.2
currently:      linear vel 0.0   angular vel 0.30000000000000004
currently:      linear vel 0.0   angular vel 0.0
currently:      linear vel 0.01  angular vel 0.0
currently:      linear vel 0.02  angular vel 0.0
currently:      linear vel 0.03  angular vel 0.0
currently:      linear vel 0.04  angular vel 0.0
currently:      linear vel 0.05  angular vel 0.0
currently:      linear vel 0.05  angular vel -0.1
currently:      linear vel 0.05  angular vel 0.0
```

Figure 1: Cartographer GUI for SLAM

Open another terminal to run the teleoperation node. While the teleoperation terminal is active, the robot can be moved around the area and the mapping from the SLAM will be visualised on the GUI running on the PC.

Once the mapping has been done, save the map by entering the following command:

```
ros2 run nav2_map_server map_saver_cli -f ~/map
```

Note: "map" can be changed to whatever name of the map you wish to save it as. This name will be later required for navigation and all other future tasks you wish to carry out in that specific map.

The data from the LiDAR and other sensors can be quite noisy and may require extensive tuning. The number of variables that can be tuned are vast. For a complete guide on tuning, explore the official documentation: https://google-cartographer-ros.readthedocs.io/en/latest/algo_walkthrough.html

Navigation

Once mapping has been done, the navigation node can be launched to make the robot autonomously navigate around the area by executing the following command:

```
ros2 launch turtlebot3_navigation2 navigation2.launch.py map:=$HOME/map.yaml
```

Here “map” in map.yaml should correspond to the filename the map was saved after SLAM.
Initial Pose Estimation must be performed before running Navigation as this process initializes the AMCL parameters that are critical for Navigation. The TurtleBot3 has to be correctly located on the map with LDS sensor data that overlaps the displayed map.

1. Click the 2D Pose Estimate button in the RViz2 menu.
2. Click on the map where the actual robot is located and drag the large green arrow toward the direction where the robot is facing.
3. Repeat step 1 and 2 until the LDS sensor data is overlayed on the saved map.

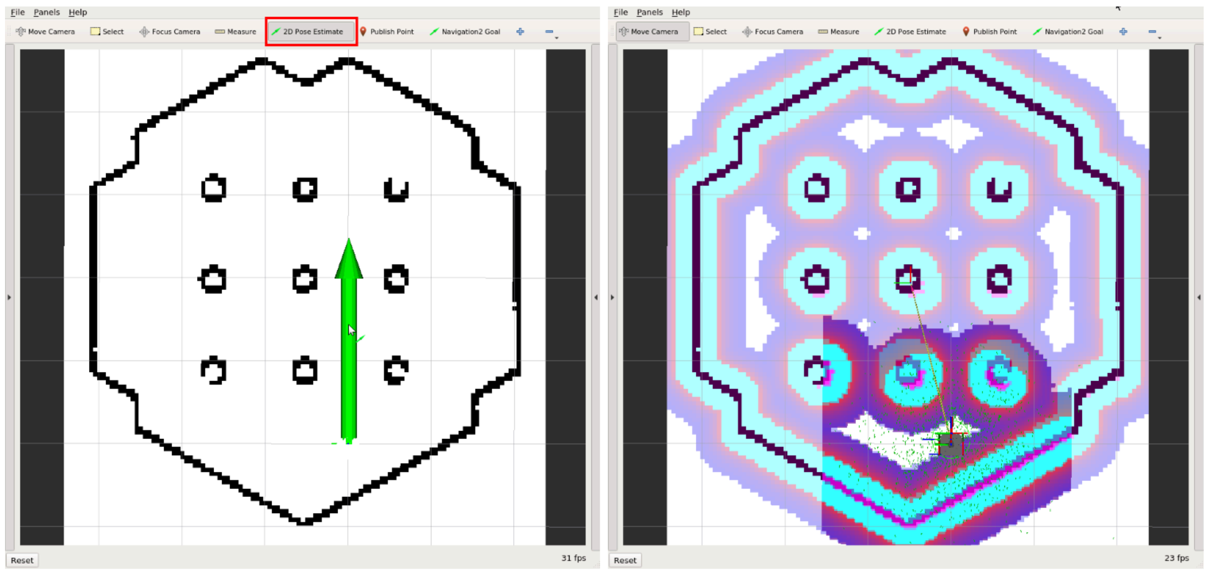


Figure 2: Estimating Initial Position

After estimating the initial position, autonomous navigation can be done:

1. Click the Navigation2 Goal button in the RViz2 menu.
2. Click on the map to set the destination of the robot and drag the green arrow toward the direction where the robot will be facing.
3. This green arrow is a marker to can specify the destination of the robot.
4. The root of the arrow is the x, y coordinate of the destination, and the angle θ is determined by the orientation of the arrow.
5. As soon as x, y, θ are set, the TurtleBot3 will start moving to the destination immediately.

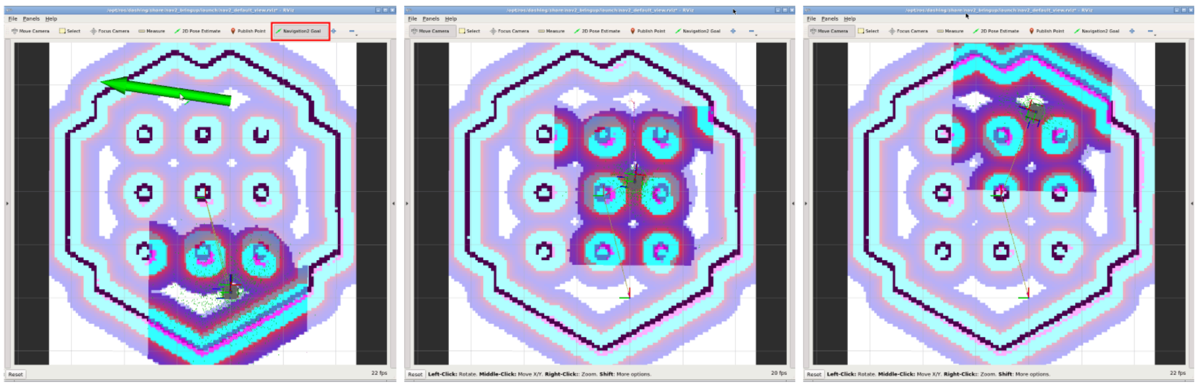


Figure 3: Running Navigation

Conclusion

The TurtleBot3 platform, characterized by its modular architecture and robust integration with the Robot Operating System (ROS2), provides a versatile and powerful framework for advancing robotics research and education. Beyond its foundational capabilities in teleoperation and Simultaneous Localization and Mapping (SLAM), the platform supports sophisticated applications, including autonomous navigation, sensor fusion, and multi-agent coordination. Its inherent flexibility enables the integration of supplementary sensors, enhanced computational resources, and bespoke algorithms, positioning TurtleBot3 as an ideal tool for investigating emergent domains such as computer vision, reinforcement learning, and human-robot interaction.

Bolstered by comprehensive documentation and a vibrant open-source community, TurtleBot3 serves as both an accessible entry point for novices and a dynamic testbed for seasoned researchers. This dual role empowers users to explore innovative methodologies, contribute meaningfully to the robotics discipline, and propel the evolution of intelligent, autonomous systems. Consequently, TurtleBot3 emerges as a pivotal resource, facilitating profound exploration and discovery within the dynamic and rapidly advancing field of robotics. This report contains only a basic guide on getting started with the TurtleBot3. For further exploration and debugging, use the official documentation.
https://emanual.robotis.com/docs/en/platform/turtlebot3/overview/