Lab 4 - Physical Turtlebot Introduction

In the last lab, you were introduced to the Turtlebot simulation ecosystem. Now, it's time to use the real Turtlebots. Working with real-world systems is the most robust way to ensure a robotic system functions, but can have added complexity.

In this lab, you will configure your ROS setup to connect remotely to the Turtlebots and repeat the simple driving and RViz visualization from last week, but now in real life. We will be using the physical Turtlebots for most of the remaining labs, so make sure you are confident with these steps.

There are a limited amount of Turtlebots, so you will need to work in pairs with your lab mates.

Requirements

- Ubuntu 24.04 Bootable USB or installation from Lab 0
- Installation of ROS2 Jazzy
- Internet connection to your Ubuntu device

Contents

1. Package Installation

First, set up a new ROS workspace and install the Turtlebot packages. These are the same Turtlebot packages installed from the last lab, but since we use a new workspace for each lab to stay organized, you need to install them in the new workspace as well.

```
$ mkdir -p ~/ee3280/lab4_ws/src
$ cd ~/ee3280/lab4_ws/src
$ git clone -b jazzy https://github.com/ROBOTIS-GIT/DynamixelSDK.git
$ git clone -b jazzy https://github.com/ROBOTIS-GIT/turtlebot3_msgs.git
$ git clone -b jazzy https://github.com/ROBOTIS-GIT/turtlebot3.git
$ git clone -b jazzy https://github.com/ROBOTIS-GIT/turtlebot3_simulations.git
$ cd ~/ee3280/lab4_ws
$ colcon build --symlink-install
```

2. Remote ROS Connection

ROS2 is implemented as a Data Distributed Service (DDS), which is a network-based protocol. The ROS topics you've been working with are communicated over your local network between nodes. Since DDS operates over a network, it's very easy to extend its operation across devices, as long as they are the same network and configured properly.

ROS allows easy cross-device connection through DDS by setting the `ROS_DOMAIN_ID` variable. Devices on the same network with the same `ROS_DOMAIN_ID` can access topics, even if that device is not the one where the topic originated. In this course, the Turtlebots will publish and advertise their topics, and your laptops will access and publish to those topics.

To make the remote configuration simple, create a new bash script that sets all the remote variables. When you need to use the Turtlebot, source this bash script, and you should then have access to the robot's topics. Create the bash script with the following commands. `nano` will open a command-line text editor where you need to add the variables outlined below.

```
$ mkdir -p ~/ee3280/scripts
$ cd ~/ee3280/scripts
$ touch turtlebot_connect.sh      # creates empty file with given name
$ chmod +x turtlebot_connect.sh   # make file executable
$ nano turtlebot_connect.sh       # command-line text editor
```

Add the following code to your `turtlebot_connect.sh` file, and be sure to adjust your `ROS_DOMAIN_ID` to the Turtlebot you are using. Each Turtlebot has a label indicating its specific domain ID. **Be sure to change this domain ID with every new Turtlebot.**

```
export TURTLEBOT3_MODEL=burger
export RMW_IMPLEMENTATION=rmw_fastrtps_cpp
export ROS_DOMAIN_ID=0  # change this number for each turtlebot
```

Save via `Ctrl-S` and close nano via `Ctrl-X`. To configure a terminal to access the Turtlebot's ROS topics, run the following line to configure the DDS connection.

```
$ source ~/ee3280/scripts/turtlebot_connect.sh
```

3. Powering on the Turtlebot

Before proceeding, grab a Turtlebot and a wooden block from the shelf and a battery from the charging area near the printer. Be careful when disconnecting the battery from the charger, as the charging connectors are small and fragile.

Next, connect the battery to the robot as shown in Figure 1 and be sure to secure the battery using the removable plastic battery clip.

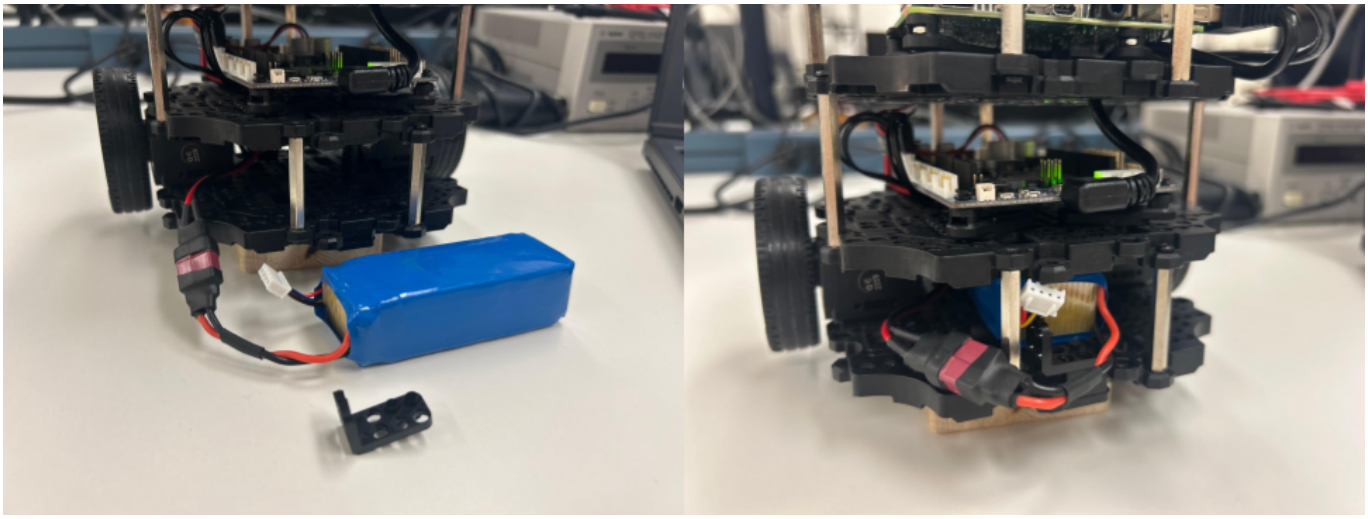

Figure 1: Connect the battery by plugging in the large, polarized connector to the Turtlebot, removing the plastic battery clip, sliding the battery in the robot, and snapping the battery clip back in place.

Turn on the robot by turning on the power switch on the front (camera side) of the blue microcontroller board, highlighted in Figure 2.

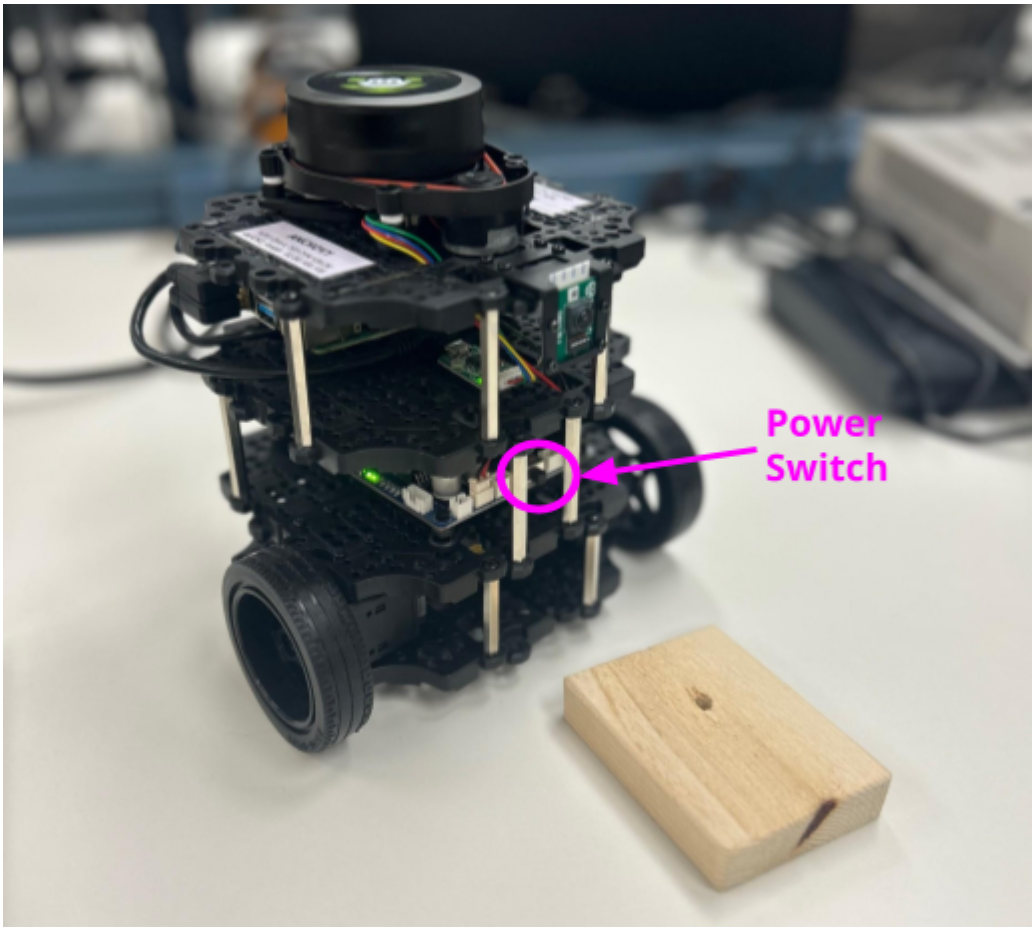

Figure 2: Turtlebot power switch and wooden support block.

Finally, please place the turtlebot on the small wooden block so that it's wheels are not making contact with the table while working. This serves to prevent robots from accidentally driving off the table during testing.

4. Connecting to the Turtlebot

You are now ready to connect to the actual Turtlebot and start up it's ROS nodes. But first, you need to connect to the same network as the Turtlebots. A separate wireless network has been configured for this lab to make it easy to remotely connect to the Turtlebots.

This network has internet access, but it can be slower than the standard Michigan Tech networks, so if you are downloading large packages or not using the Turtlebots, it's best to stay on Michigan Tech or eduroam networks.

Connect to the lab Wi-Fi:

```
SSID: ee3280_roslab
Password: not_pizza
```

This network has specific IP settings so you don't need to change any network properties. Double check that your IP address is now on the `32.80.100.XXX` subnet, where XXX is any number:

```
$ hostname -I
```

If your address is not on the `32.80.100.XXX` range, consult your TA for assistance. Once connected, proceed to the next section.

SSH

Secure Shell (SSH) allows remote command-line access to other computers. This is a common way to remote into servers (including Michigan Tech servers) and is how we will access the Turtlebot's Raspberry Pi to start the ROS nodes.

The Turtlebots all have the same username and password:

```
Username: ubuntu
Password: turtlebot
```

In a new terminal, remote in to the robot via SSH by indicating the user account (ubuntu) and where to access it (the robot's IP address, which is printed on the robot's label)

```
$ ssh ubuntu@32.80.100.XXX  # change this to your turtlebot's address!
```

You may be prompted if you wish to connect because the authenticity of the robot can't be established. Type **'yes'** and it will be added to a list of "known hosts" where you won't be prompted again for that specific robot (other robots have different address and will have the same prompt).

Enter the robot's password, and if successful, you'll now be remotely connected to the Turtlebot's Raspberry Pi.

Starting the Turtlebot ROS Nodes

Once remoted into the Turtlebot, start the Turtlebot's sensors and drivers by launching the following file:

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

This is the only line needed to execute onboard the Turtlebot. Once started, all Turtlebot topics are accessible via connected ROS computers.

5. Using the Turtlebot

Now that the Turtlebot driver is running, connect your device to the same `ROS_DOMAIN_ID` by sourcing the `turtlebot_connect.sh` script you made earlier in a new terminal. Then, check for the Turtlebot topics:

```
$ source ~/ee3280/scripts/turtlebot_connect.sh
$ ros2 topic list
```

If you cannot see the topics from Turtlebot, consult your TA for assistance.

Otherwise, proceed to test drive the robot by launching the teleop node used in Lab 3. In the same terminal you sourced, start the teleop node:

```
$ cd ~/ee3280/lab4_ws
$ source install/setup.bash
$ ros2 run turtlebot3_teleop teleop_keyboard
```

Now, you should be able to drive the Turtlebot on the wooden block. When you are ready, set the Turtlebot on the floor in an open area in the lab and drive it around the room. Be careful to keep an eye on your Turtlebot and have your partner ready to grab it if it gets into trouble.

While driving it around, take a short video of the robot as it zooms across the lab for your Lab 4 submission.

Then, while the teleop node is running, set up a new terminal by sourcing the `turtlebot_connect.sh` script and the local workspace `setup.bash` and launch RViz:

```
$ source ~/ee3280/scripts/turtlebot_connect.sh
$ cd ~/ee3280/lab4_ws
$ source install/setup.bash
$ ros2 launch turtlebot3_bringup rviz2.launch.py
```

You should now see the model turtlebot in the world with faint point cloud returns. Be sure to change the point cloud type to 'points' of size 3 to see the cloud better. Also, change the Fixed Frame near the topic to the base_link of the robot, as shown in Figure 3.

Drive around the lab while looking at the RViz point cloud and match the room's features to those you see in RViz. Take a screenshot of an interesting feature you see for your Lab 4 submission.

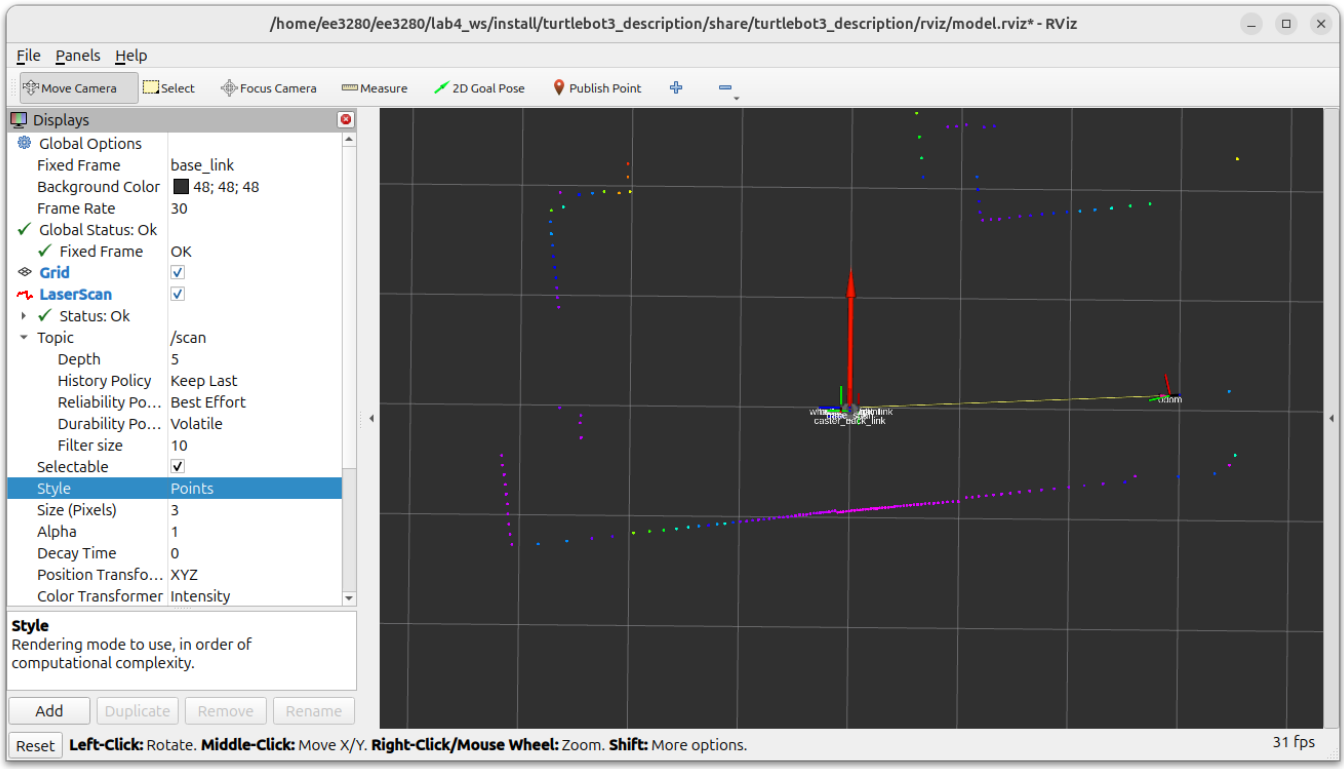

Figure 3: Sample RViz setup of the Turtlebot scanning the room.

Once you get all your submission artifacts, make sure everyone in your group has an opportunity to remotely connect to the robot and drive it around.

6. TA Check

Demonstrate to your TA that you can drive the robot in the lab and visualize the point cloud in RViz, and make sure that everyone in your group has had a chance to connect to and drive the robot from their laptops.

7. Turtlebot Shutdown

When complete, shutdown the robot by `Ctrl-C` the SSH terminal that started the robot's nodes and type `exit` to return to your local terminal. Then, power off the robot by flipping the power switch, put the robot and wooden block back on the shelf, and connect the battery to the charger.

Note that while you will connect the battery to the charger, it will **not** light up. The chargers are set on a timer to only charge on weekdays when staff are present to comply with University battery safety policies.

8. Canvas Submission

Under the Lab 4 Submission, submit:

- A short video of the Turtlebot driving around the lab (25 pts)
- A screenshot of the RViz point cloud scanning an interesting feature in the room (25 pts)

Everyone in your group should submit the deliverables, but your group members can share the video and screenshot.

Designed for Michigan Technological University's EE3280 Robot Operating Systems (ROS) course

Michigan Technological University
Electrical and Computer Engineering