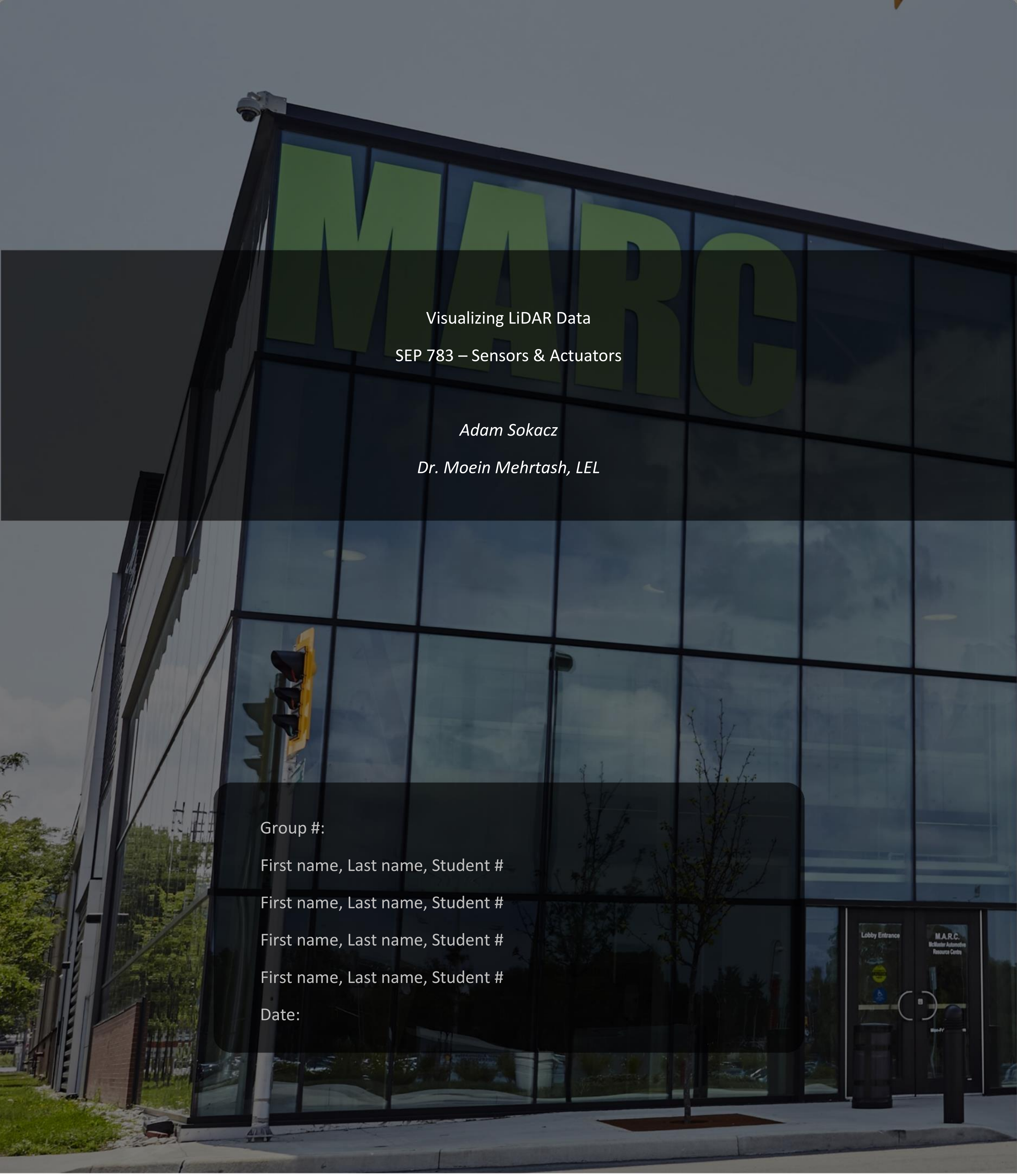


Visualizing LiDAR Data

SEP 783 – Sensors & Actuators

Adam Sokacz

Dr. Moein Mehrtash, LEL

Group #:

First name, Last name, Student #

First name, Last name, Student #

First name, Last name, Student #

First name, Last name, Student #

Date:

W BOOTH SCHOOL OF ENGINEERING PRACTICE AND TECHNOLOGY
McMaster-Mohawk Bachelor of Technology Partnership

McMaster University ENGINEERING

MOHAWK

Objective
To familiarize yourselves with Linux, the basics of ROS middleware, and be comfortable reading and visualizing LiDAR sensor data.

Table of Contents

Pre-Lab Questions

Q1 - What is *Ubuntu*? How is it different than *Windows* or *MacOS*? How is it similar?

(Suggested: 3 sentences)

Q2 - What is the difference between *sudo apt update* and *sudo apt upgrade*?

(Suggested: 1 sentence)

Q3 - What does the *sudo* keyword to when using it in front of a terminal command?

(Suggested: 1 sentence)

Q4 - What is the bash command to navigate into a *directory/folder*? How do we list the *file contents* of that folder?

(Suggested: 2 sentences)

Q5 - How do you create a *new file* from the bash terminal?

(Suggested: 1 sentence)

Post-Lab Questions

Q1 - What is *Robot Operating System (ROS)* in your own words? Search for and list *3 applications* of ROS in industry.

(Suggested: Short paragraph)

Q2 - What does the GREP command do in Linux? Use an example in your explanation.

(Suggested: Short paragraph)

Q3 - What does the WGET command do in Linux? Use an example in your explanation.

(Suggested: Short paragraph)

Q4 - What is an SSH tunnel? Use a Bash command example in your explanation.

(Suggested: Short paragraph)

Q5 - What is a Daemon in Linux? Use an example in your explanation.

(Suggested: Short paragraph)

Q6 - What role does *ROSCore* play in a functioning ROS system?

(Suggested: 1 sentence)

Q7 - What ROS command is used to *view* the current active topics?

(Suggested: 1 sentence)

Q8 - Which command was used to *echo* that topic to the terminal?

(Suggested: 1 sentence)

Q9 - Which command can be used to get *information* on a particular topic?

(Suggested: 1 sentence)

Q10 - What ROS tool can be used to *visualize* a stream of data?

(Suggested: 1 sentence)

Q11 – Write a brief LinkedIn post on 4 key concepts that were learned in this lab.

(Suggested: Brief Paragraph)

Optional Assignment

Follow the tutorial below to program and build a 'Hello World' publish and subscribe node in either Python or C++. Take a screenshot that captures the data being sent and received.

Python: http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28python%29

C++: http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28c%2B%2B%29

Feedback

Q1 - What would you rate the difficulty of this lab?

(1 = easy, 5 = difficult)

1	**2**	**3**	**4**	**5**

Comments about the difficulty of the lab:

Q2 - Did you have enough time to complete the lab within the designated lab time?

YES	**NO**

Q3 - How easy were the lab instructions to understand?

(1 = easy, 5 = unclear)

1	**2**	**3**	**4**	**5**

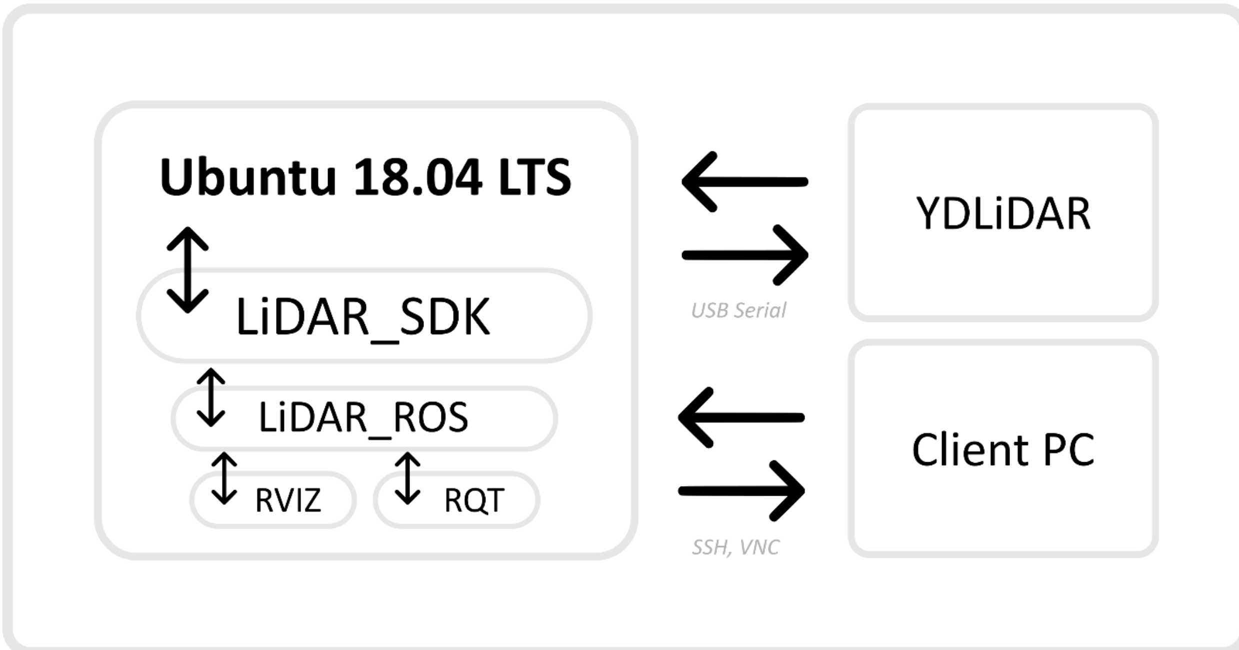
List any unclear steps:

Q4 - Could you see yourself using the skills learned in this lab to tackle future engineering challenges?

(1 = no, 5 = yes)

1	**2**	**3**	**4**	**5**

Overview



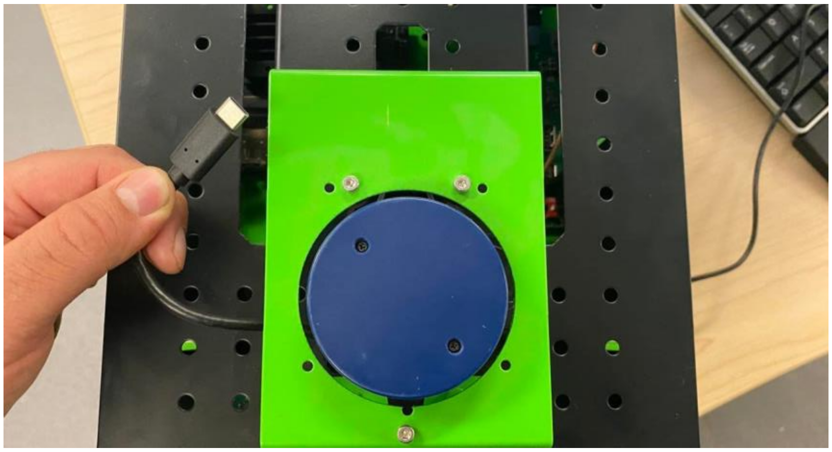
The goal of this lab is to visualize LiDAR data. The Jetson Nano microcontroller in the MacBot runs a distribution of Linux called Ubuntu version 18.04. In this lab, we will be installing the C++ & Python libraries for Linux to talk to the LiDAR. After, we use ROS, which is a middleware for creating robotic applications using Linux, to visualize the LiDAR data and create a chart outlining how data is flowing live in the MacBot. To make the raw libraries compatible with ROS, we build the LiDAR ROS driver package. Its job is to translate the C++ and Python interfaces to a standard node interface that ROS can understand and communicate with.

This lab is a great introduction to ROS, and sensor visualization using ROS. I hope you enjoy it!

Setup

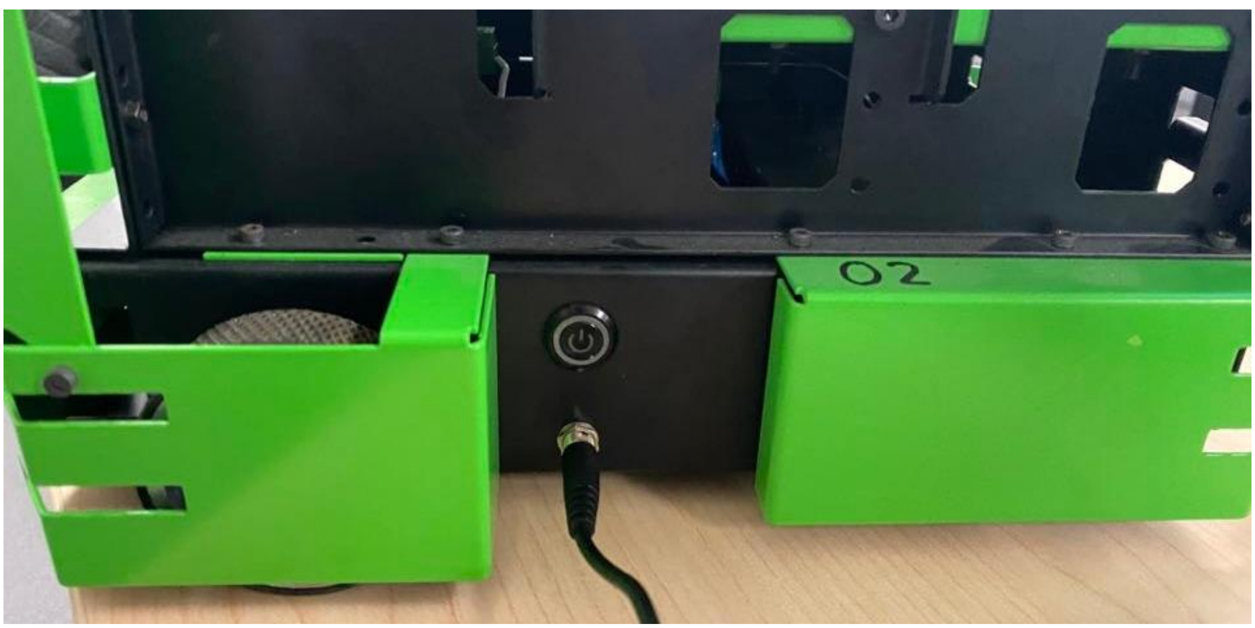
Disconnecting the LiDAR

When the LiDAR is connected, it draws a considerable amount of current. This makes it difficult for the MacBot to complete its BOOT process. Before powering on the MacBot, ensure that you disconnect the LiDAR temporarily.

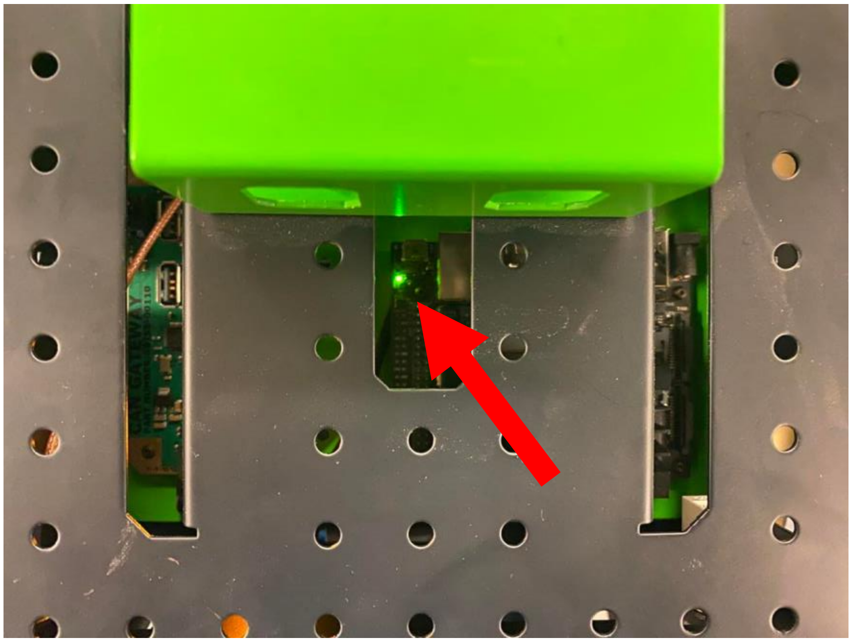


Powering on the MacBot

Notice the power switch and charging port on the right-side of the MacBot chassis. Press the POWER button until it latches in the ON position and begins to glow.



Wait 10 seconds and ensure that the Jetson Nano POWER status LED remains lit. If not, your battery requires charging.



Connecting to the MacBot

Follow the instructions on the **connect/** page of the macbot documentation to connect to your macbot unit:

https://adam-36.gitbook.io/macbot/connect

Ensure that you see a remote desktop connection in MobaXTerm.


Trash
Jetson Developer...
Jetson Support F...
Jetson Zoo
L4T-README
LXTerminal
VPI Demos v1.2
Chromium Web Brow...
Deepin Terminal
MARC
W Booth School of Engineering
Practice and Technology
15:18

Setting Up the LiDAR Drivers

Downloading the YDLiDAR SDK

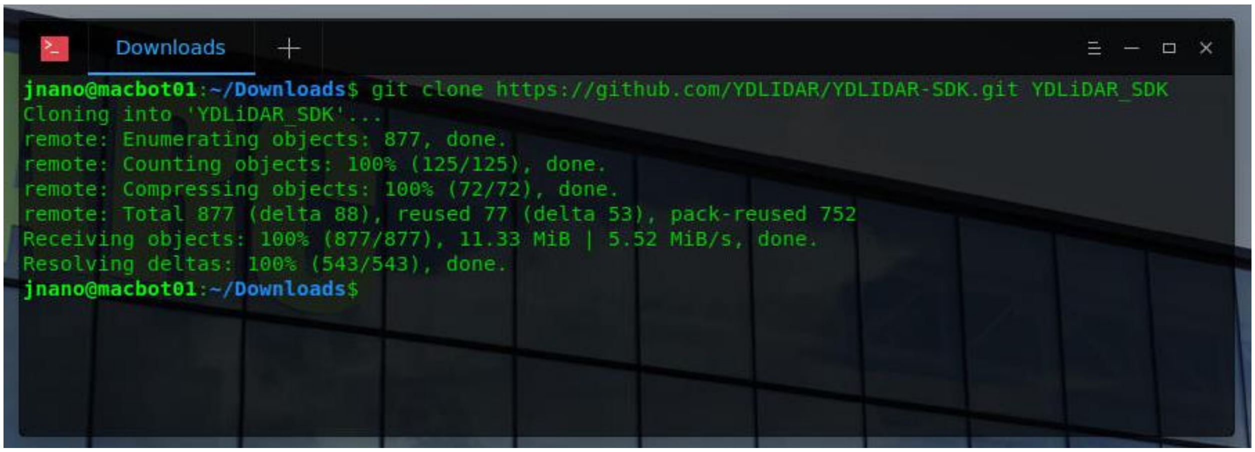
Navigate to your ~/Downloads/ folder.

cd ~/Downloads

Clone the following GitHub repo into your Downloads/ folder:

https://github.com/YDLIDAR/sdk.git

git clone https://github.com/YDLIDAR/YDLidar-SDK YDLiDAR_SDK

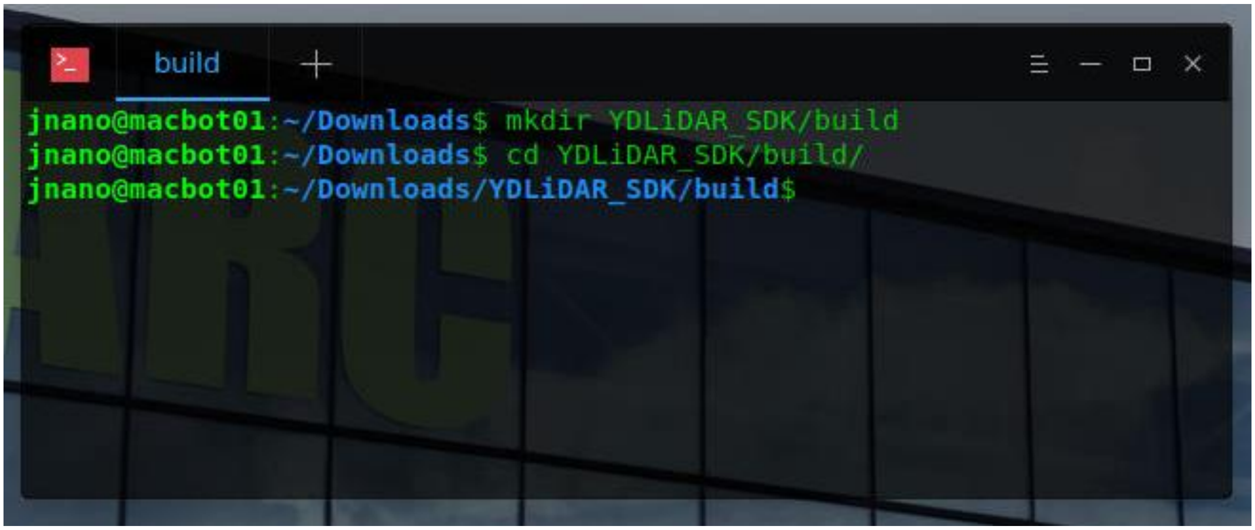


Building the YDLiDAR SDK

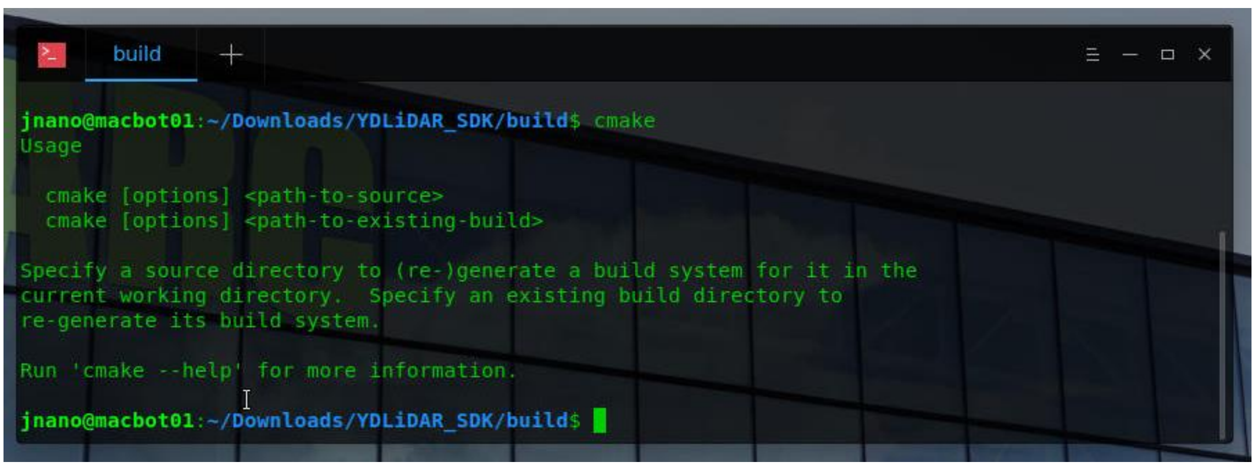
Create a folder inside of YDLiDAR_SDK/ called build/.

mkdir YDLiDAR_SDK/build

cd YDLiDAR_SDK/build/

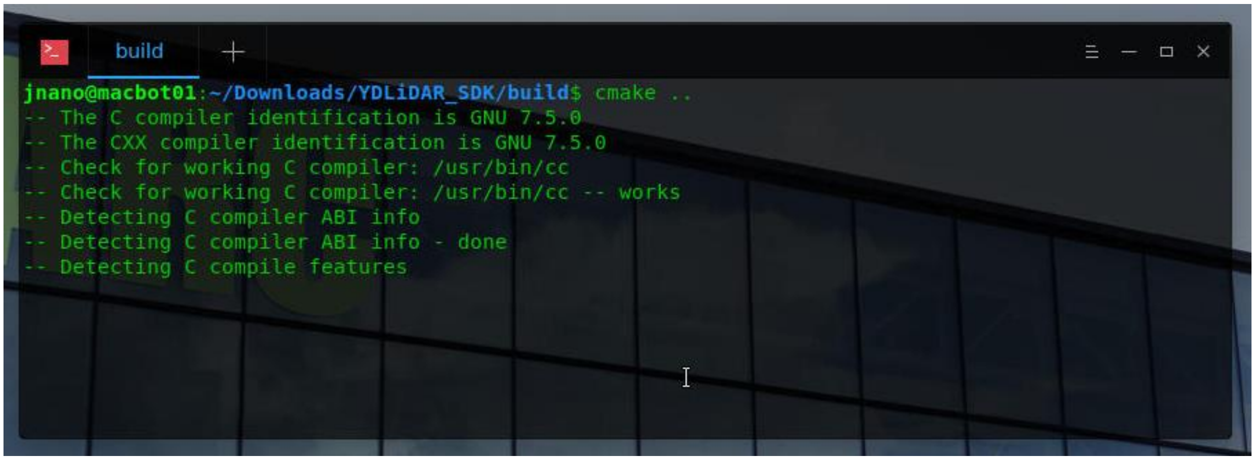


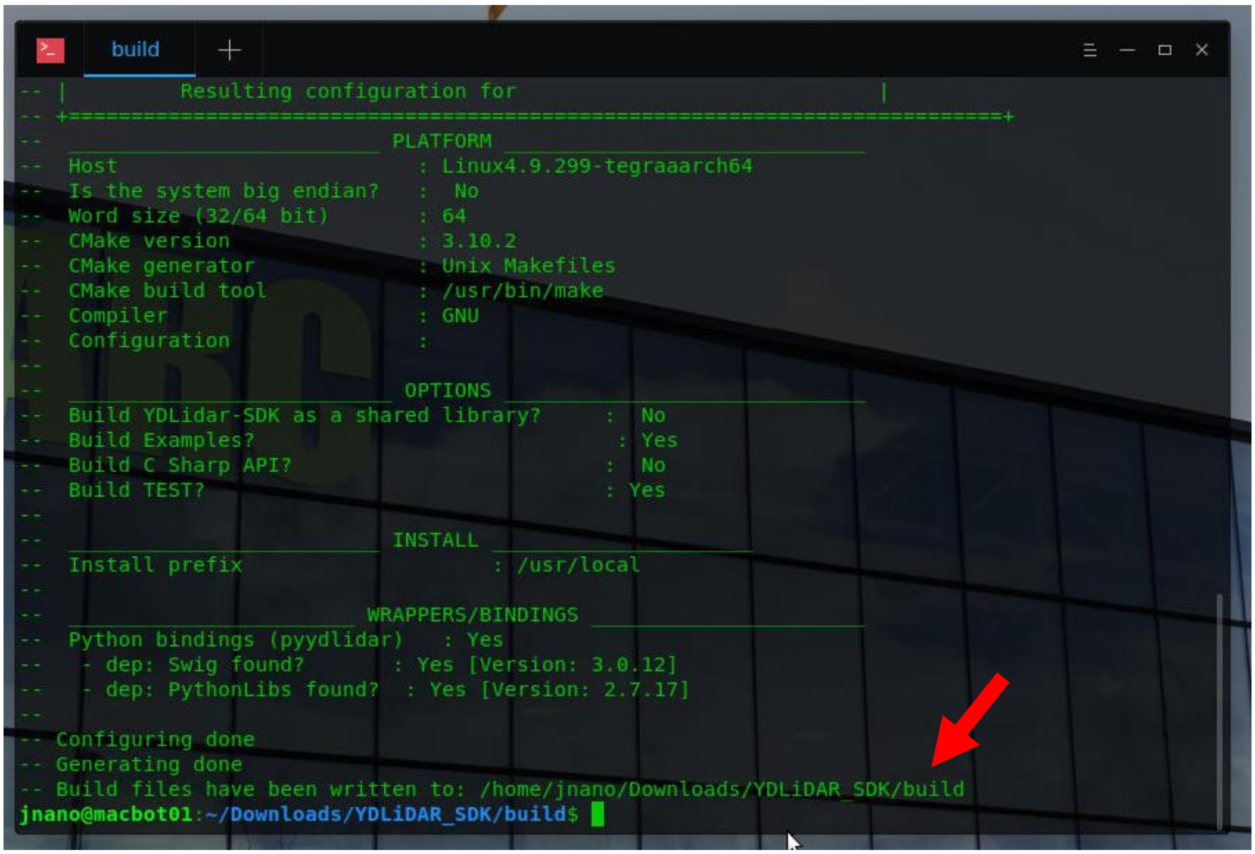
Run cmake and see what paths it requires.



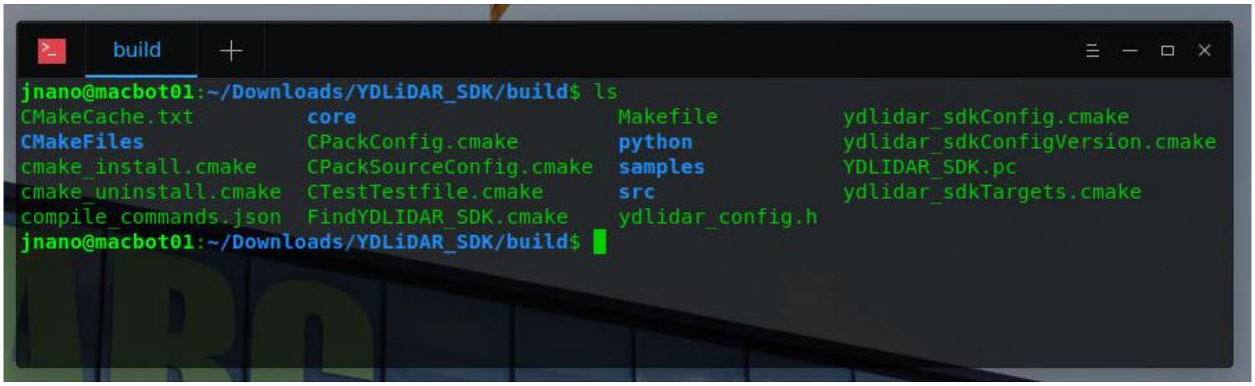
Run cmake on the project.

cmake ..



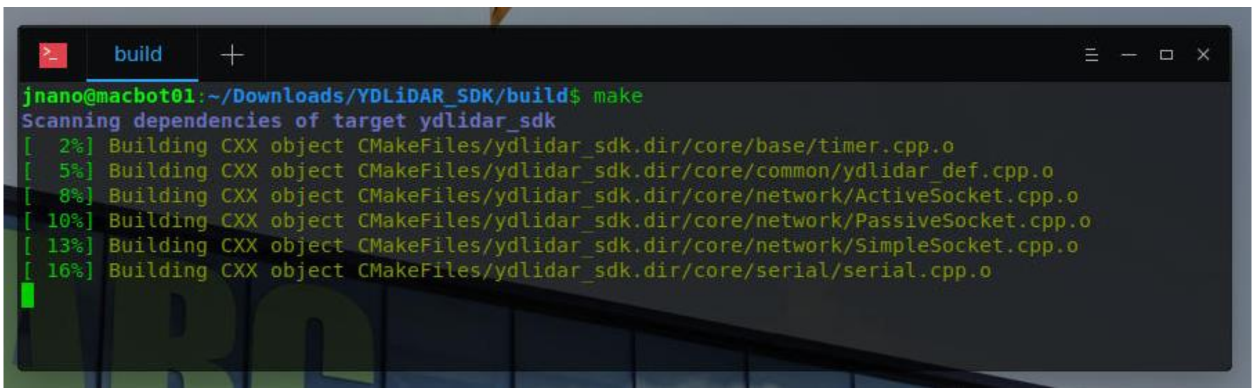


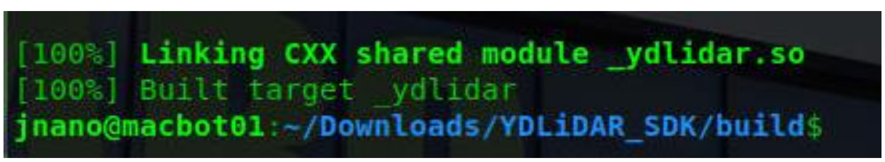
Notice that build files have been created.



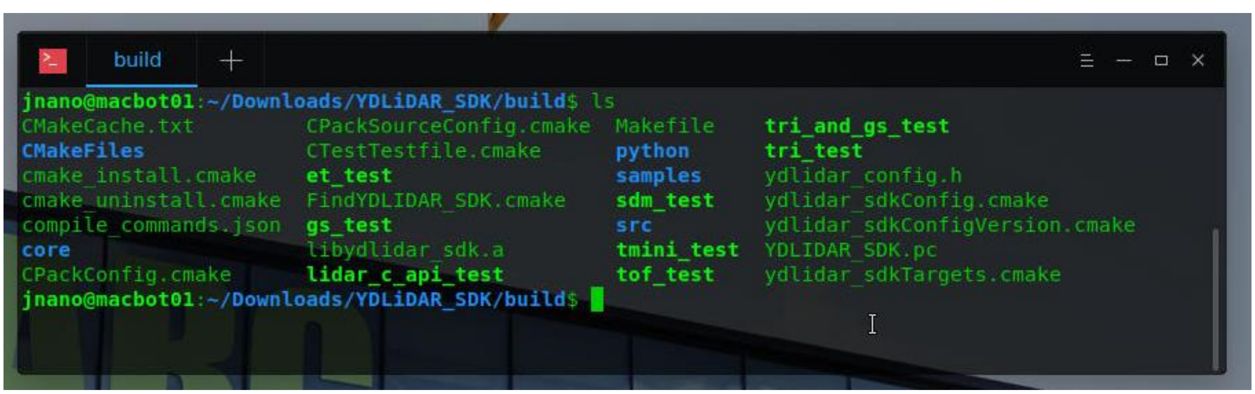
Run make on the project.

make





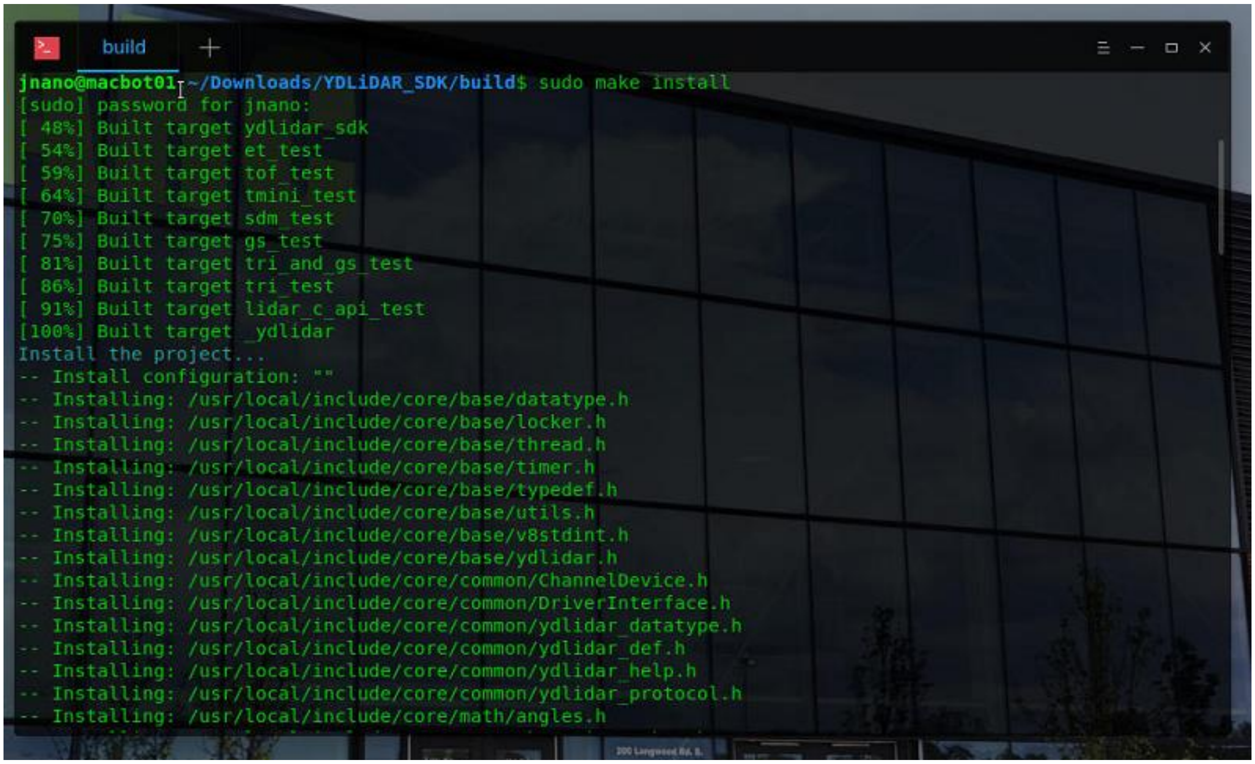
Notice that the project has been built.



Installing YDLiDAR_SDK onto the MacBot

Run the following command to install the SDK onto your system.

sudo make install



Verifying that ROS Melodic is Installed

Open the Terminal and run the following command to verify that ROS Melodic has been correctly installed:

rosversion -d

Creating and Sourcing a ROS Workspace

Navigate to the home/ directory using the following command:

cd ~

Create a new folder called **lidar_ws/** with a subdirectory within it called **src/**.

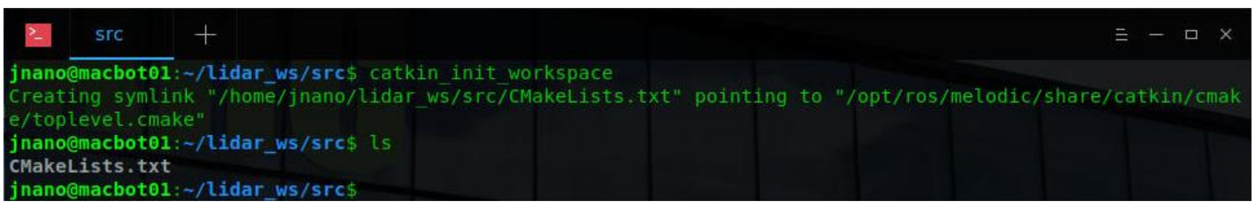
mkdir -p lidar_ws/src

cd lidar_ws/src

Initialize the workspace using the following command from the **~/lidar_ws/src** directory:

catkin_init_workspace

A new file called **CMakeLists.txt** should have been generated.

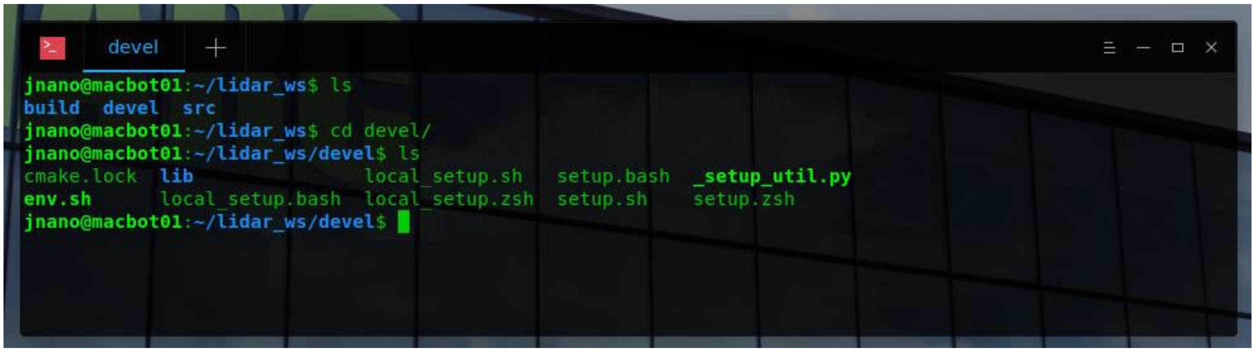


Navigate to the **~/lidar_ws** directory and build the empty workspace using the **catkin_make** command.

cd ~/lidar_ws

catkin_make

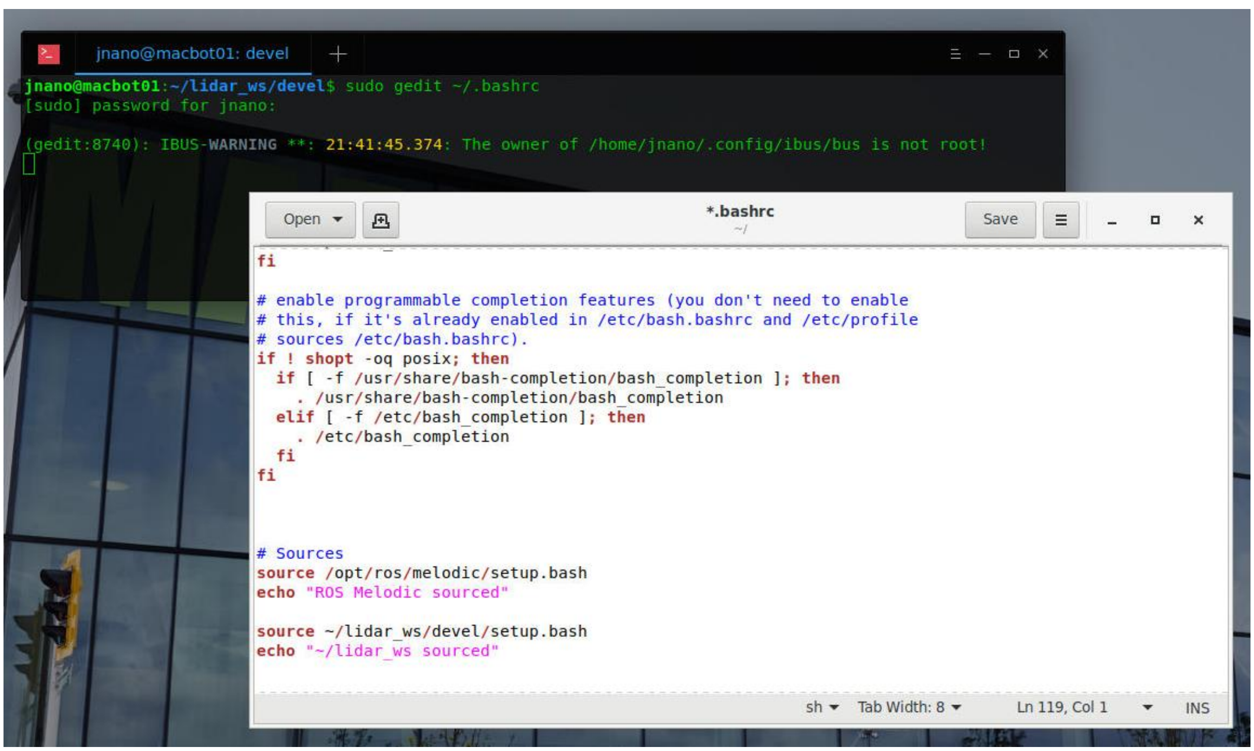
Notice that after the build is successful, some new directories have been generated. An important file is the **devel/setup.bash** script. We will need to load this script every time we open a new terminal emulator window in order to access workspace files when running ROS commands.



Use **gedit** as **superuser** to open the ~/.**bashrc** configuration script. This script runs each time a new terminal window is open. We will be appending commands to the **end** of bashrc to automatically source our ROS installation and workspace.

sudo gedit ~/.bashrc

<type_password>



Notice the #Sources section that was previously added. Ensure that you have these two paths sourced and echoed to the terminal window.

 # Sources

source /opt/ros/melodic/setup.bash

echo "ROS Melodic sourced"

source ~/lidar_ws/devel/setup.bash

echo "~/lidar_ws sourced"

Save and close the file.

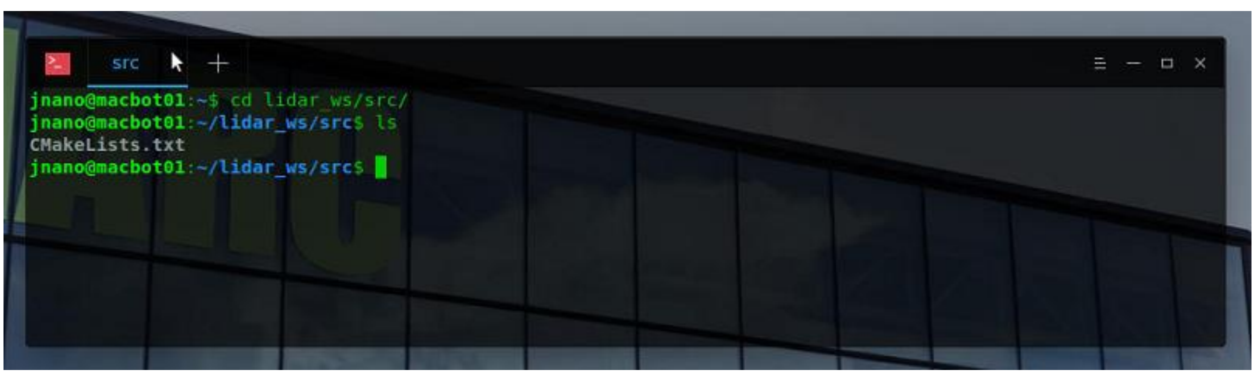
Open a new teminal window.

You should notice the statements printing at the top of the window. **Close** the old window.

Building the YDLiDAR ROS Driver

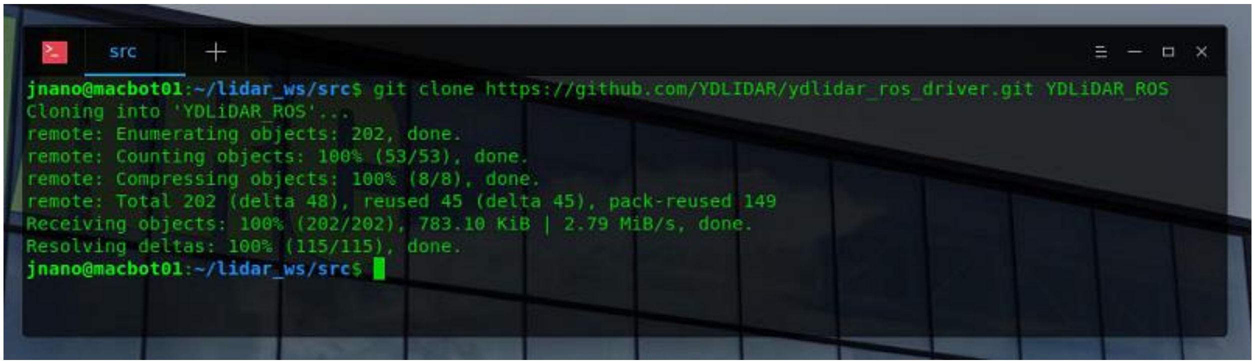
Navigate to your **~/lidar_ws/src** directory.

cd ~/ydlidar_ws/src



Clone the following YDLiDAR ROS driver into your workspace:

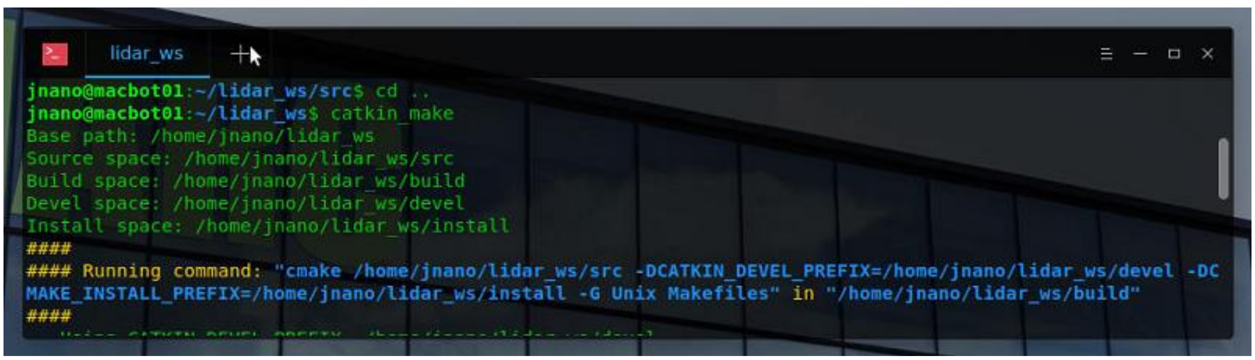
https://github.com/YDLIDAR/ydlidar_ros_driver *git clone https://github.com/YDLIDAR/ydlidar_ros_driver.git YDLiDAR_ROS*



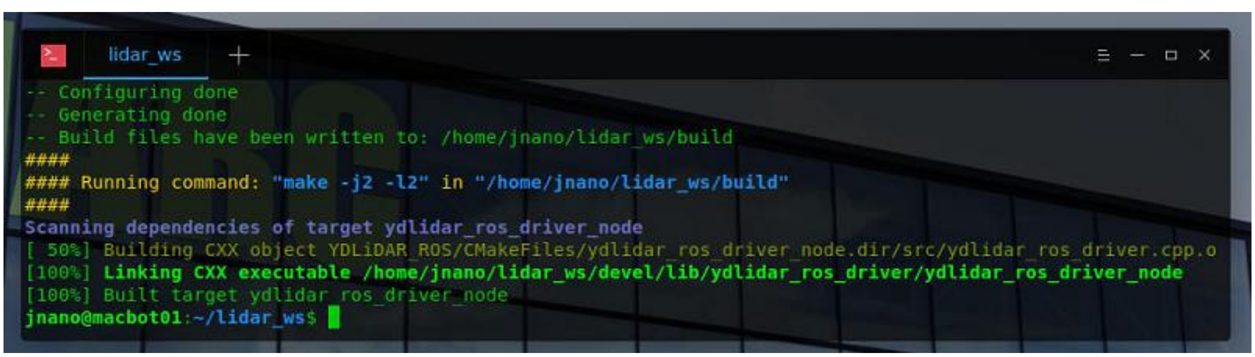
Build the workspace using **catkin_make**.

cd ..

catkin_make



Ensure that the build is successful.



Setting Permissions for the ROS LiDAR Driver

Navigate to your built LiDAR ROS package.

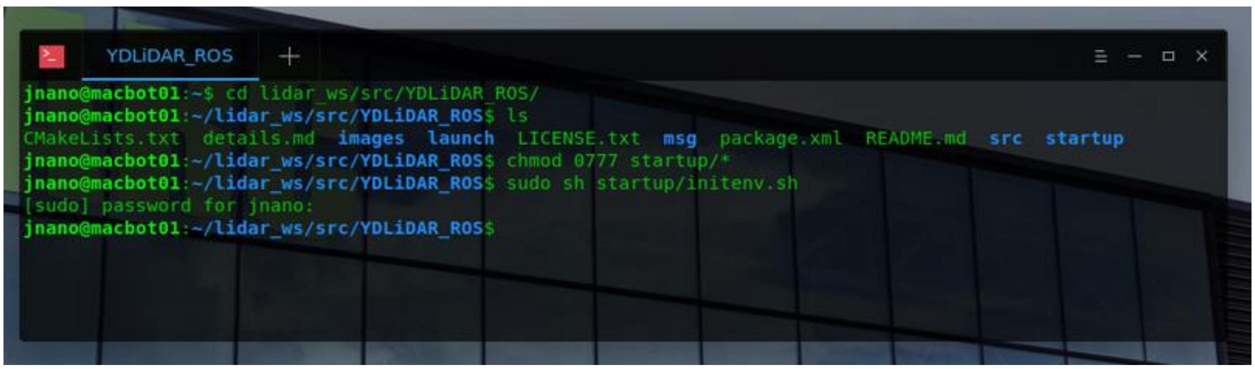
cd lidar_ws/src/YDLiDAR_ROS/

Give all files in the startup/ directory read, write, and executable permissions for all users.

*chmod 0777 startup/**

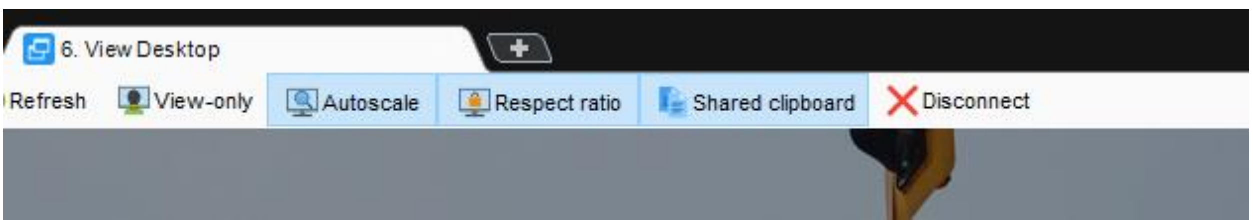
Run the environment initialization script inside of the startup/ directory. This script modifies the USB kernel device module to be able to communicate with the LiDAR. It then restarts the Linux device daemon/service.

sudo sh startup/initenv.sh



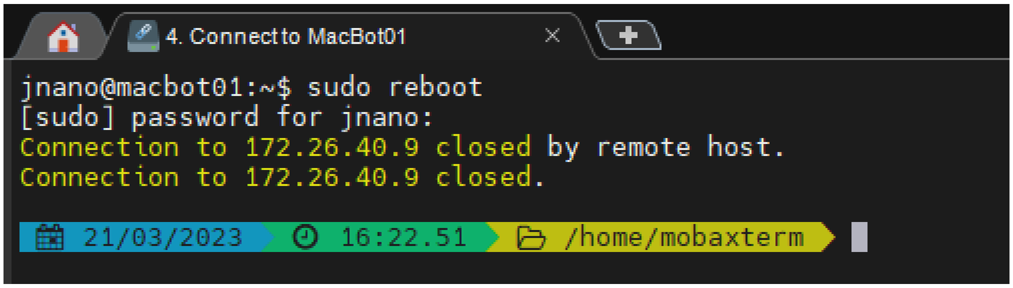
Now, we need to reboot the MacBot to ensure that the changes take effect.

Close the remote desktop connection by pressing **Disconnect**.

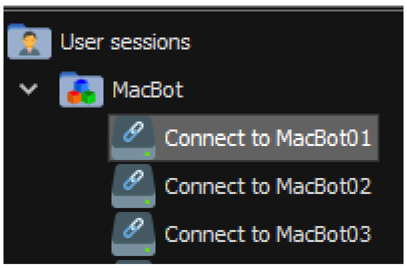


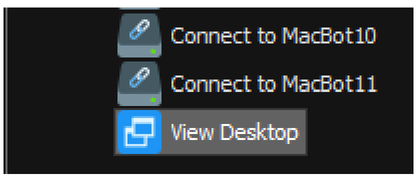
In the connection bash terminal, type the following command:

sudo reboot



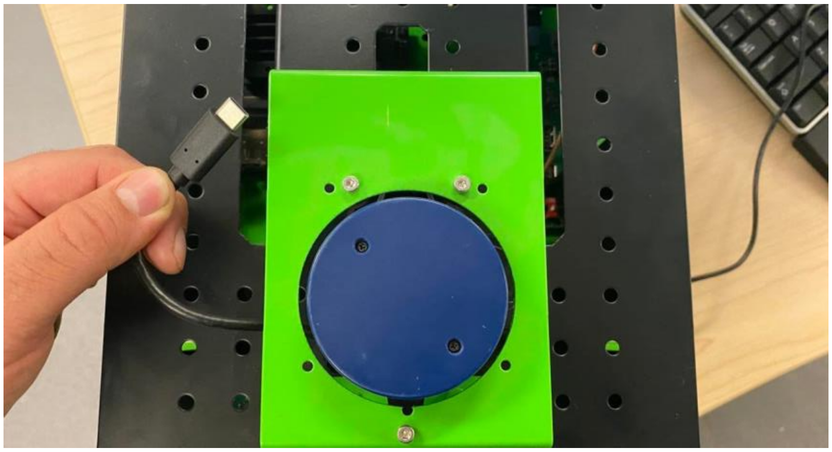
Wait 2 minutes before reconnecting to the MacBot.





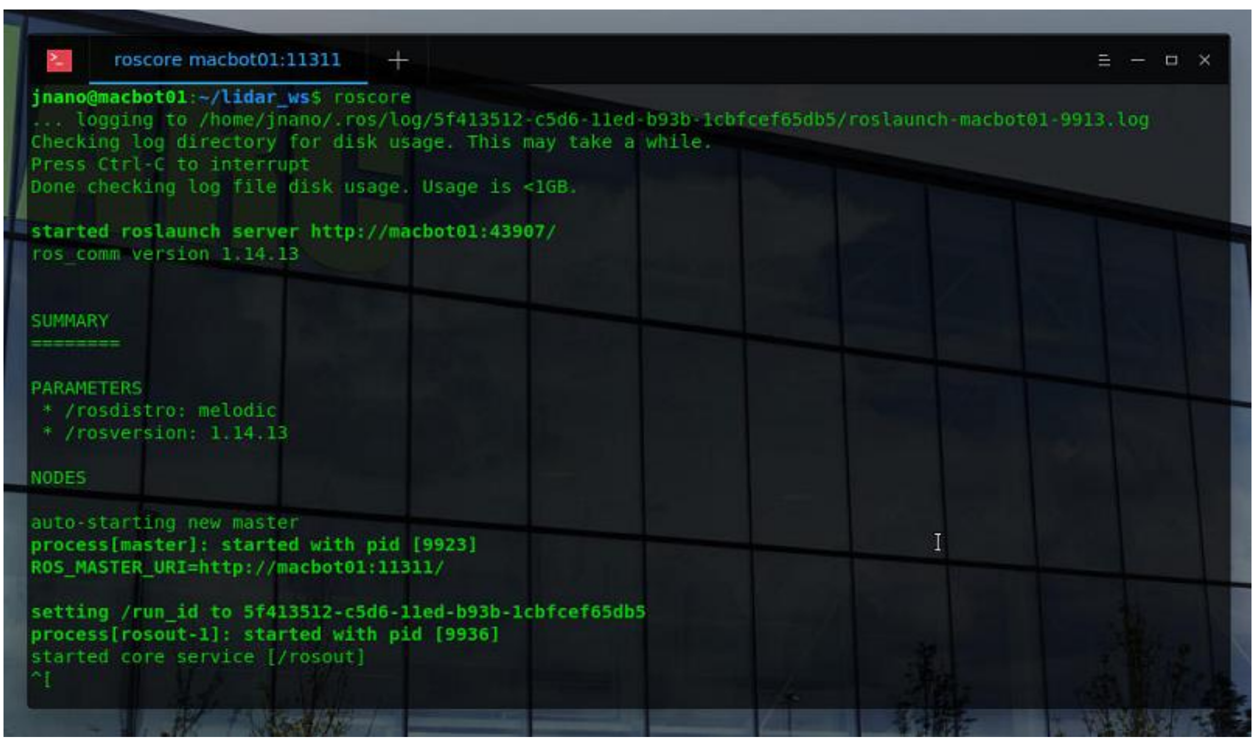
Visualizing LiDAR Point Cloud Data

Connect the LiDAR to your MacBot using the USB-C cable.



Start ROSCore in a **new terminal window**.

roscore

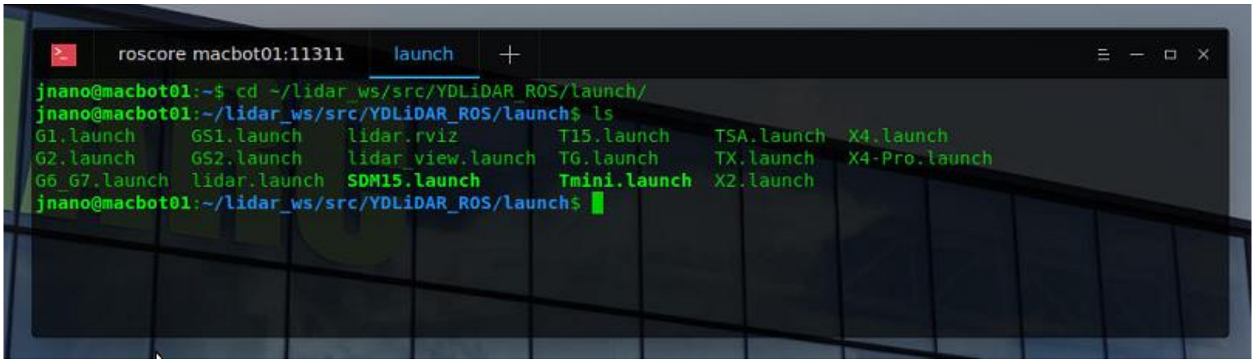


Open another terminal window (or tab, or split screen) and navigate to ~/lidar_ws/src/YDLiDAR/launch.

List out the different launch files available to run.

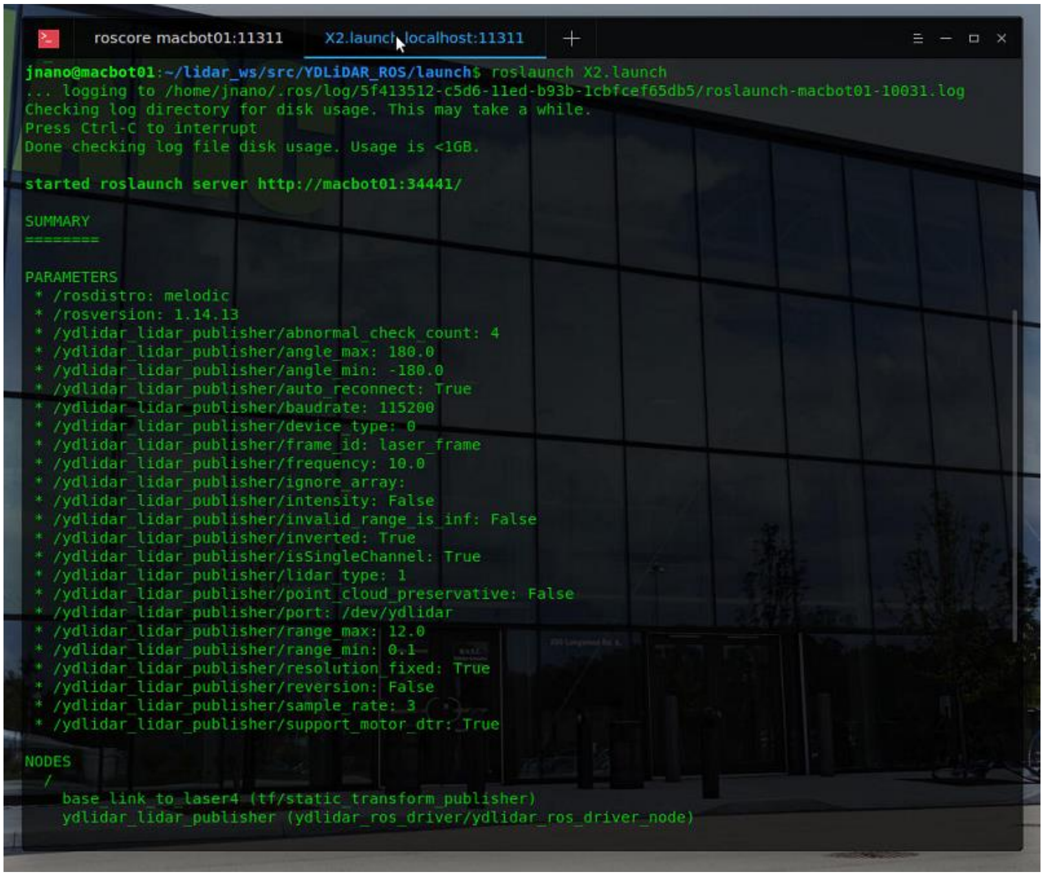
cd ~/lidar_ws/src/YDLiDAR/launch

ls



Using the roslaunch command, launch the X2.launch file. Ensure that communication is established with the LiDAR. You will audibly notice a change in spin rotation speed.

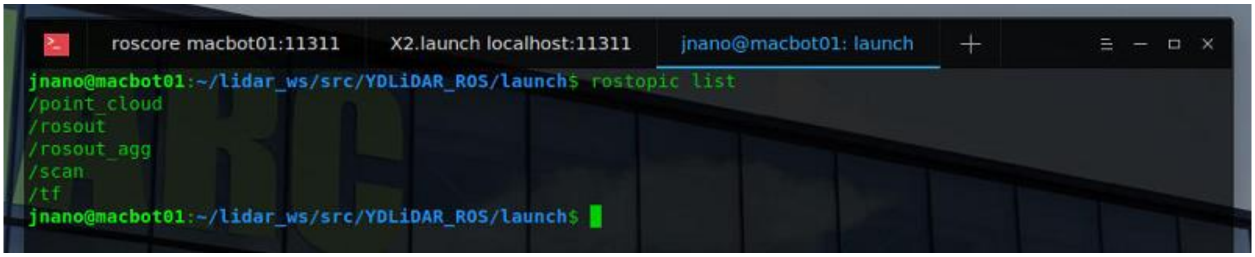
roslaunch X2.launch



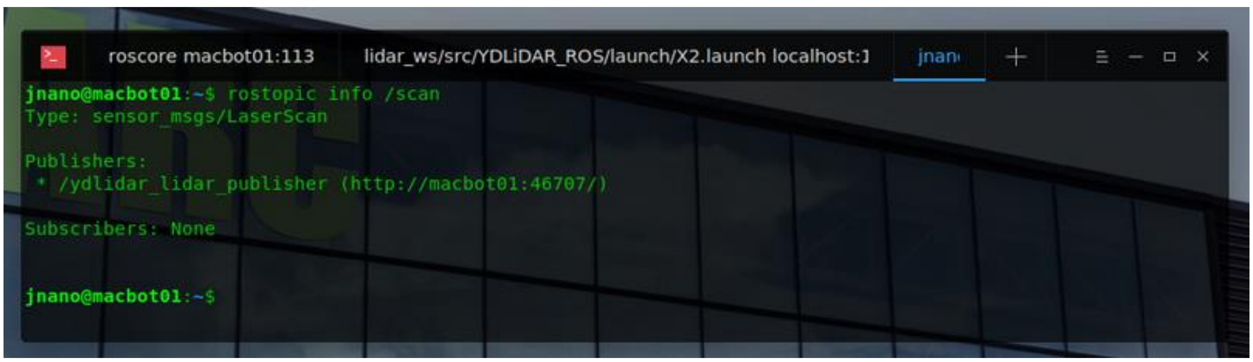
Next, open a new terminal window (or tab, or split-screen).

Use the rostopic list command to view all available streaming topics.

rostopic list

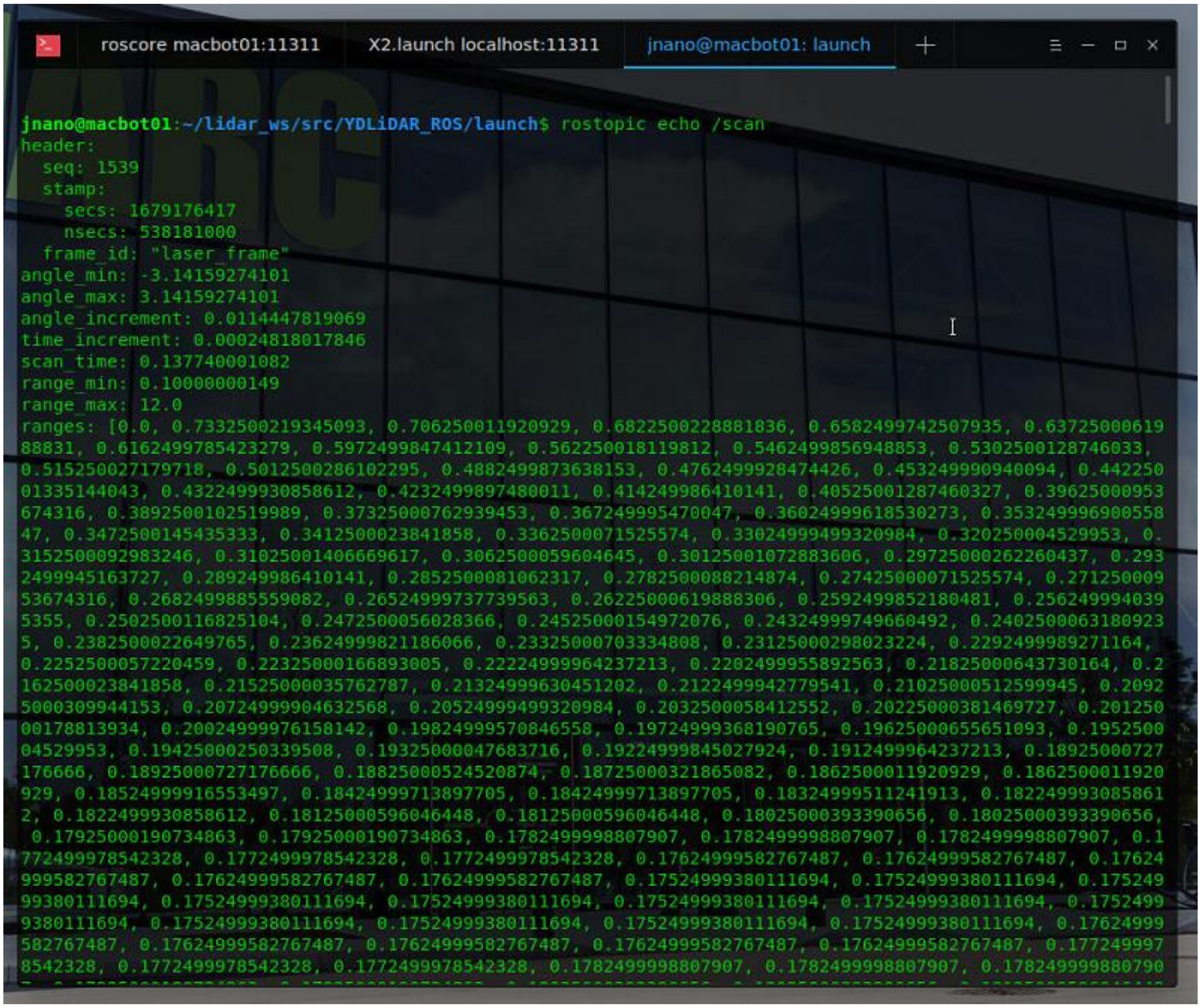


Find more information about the /scan topic including the datatype and port its hosted on.



Use the rostopic echo command to view the data being streamed on the /scan topic.

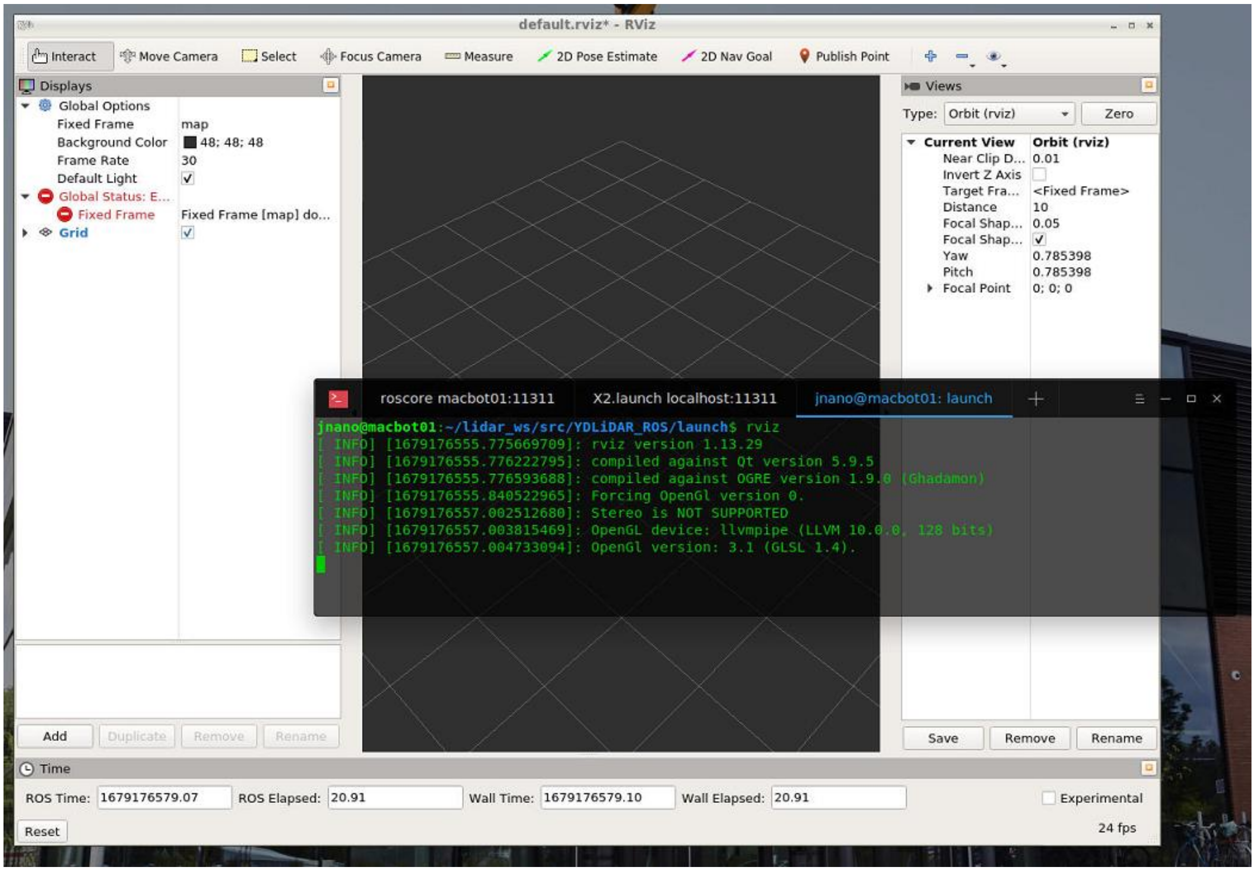
rostopic echo /scan



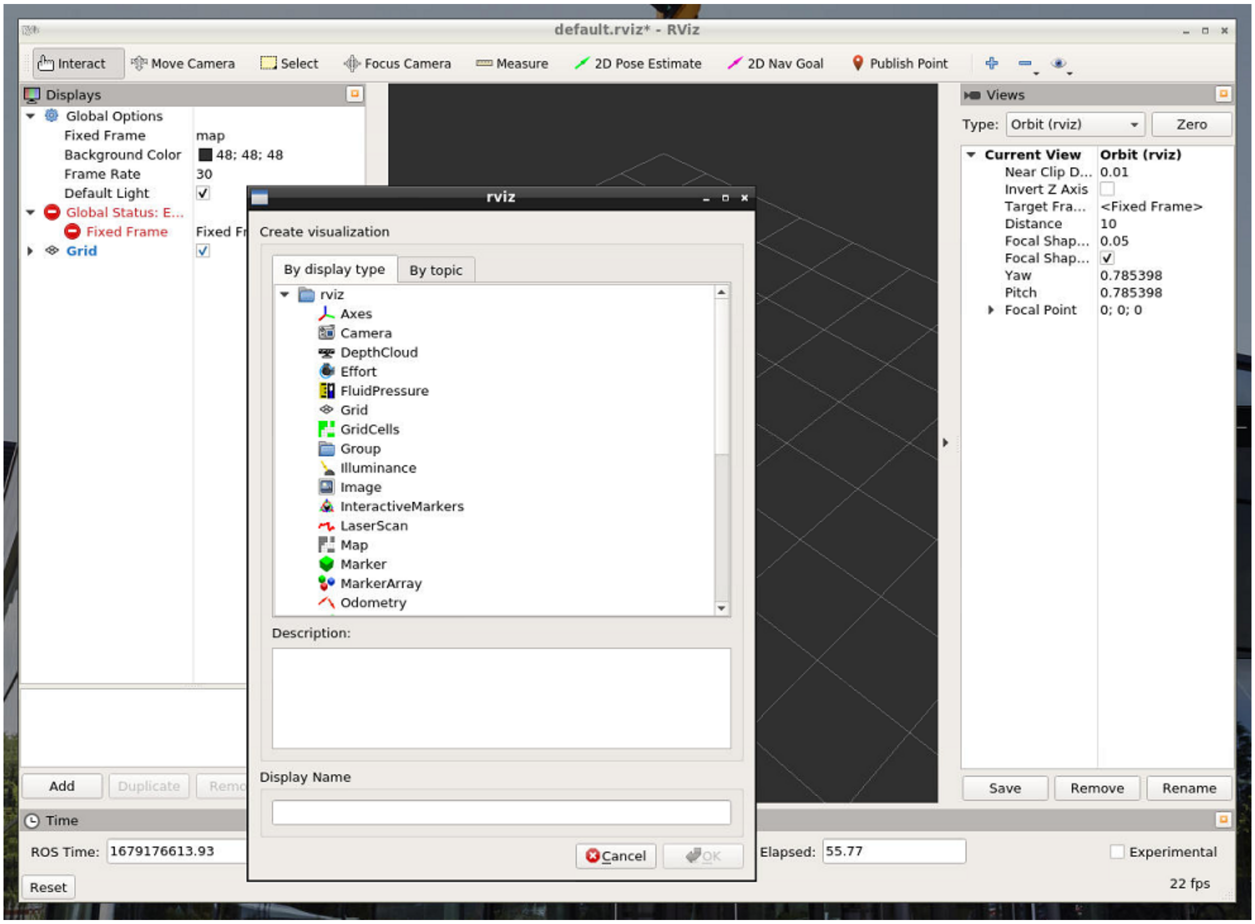
Press **CTRL + C** to stop viewing the data stream.

Next, open the ROS Visualization tool. A graphical utility will launch.

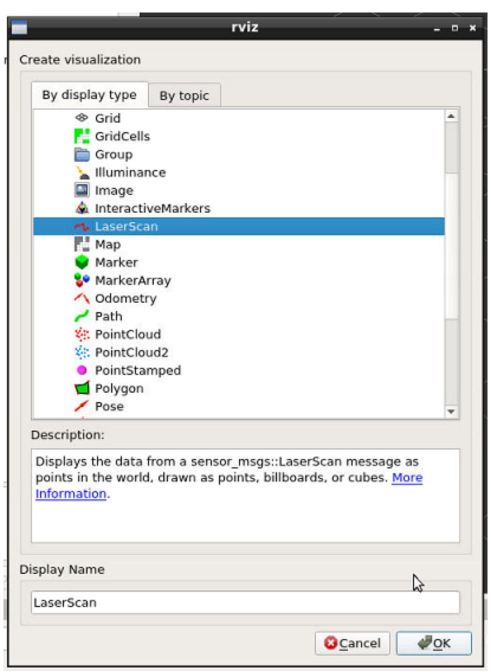
rviz



In the left pane, click **Add**. A window will appear to select the display type to add.



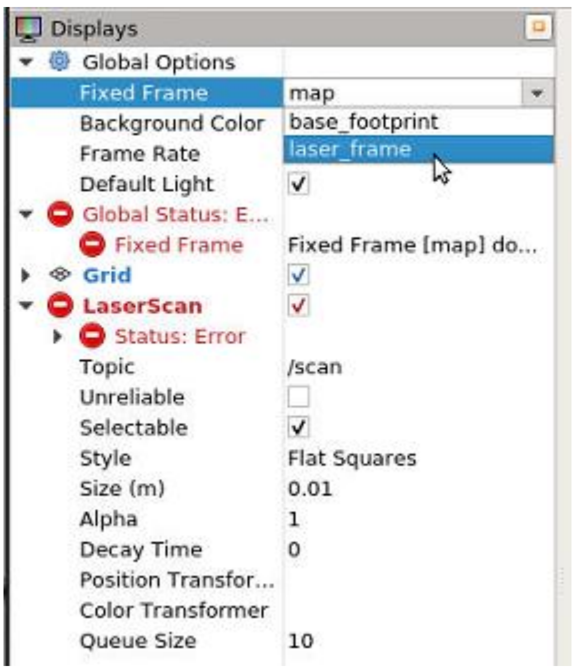
Choose laserscan and press OK.



Back in the left pane, set the laserscan topic to /scan.

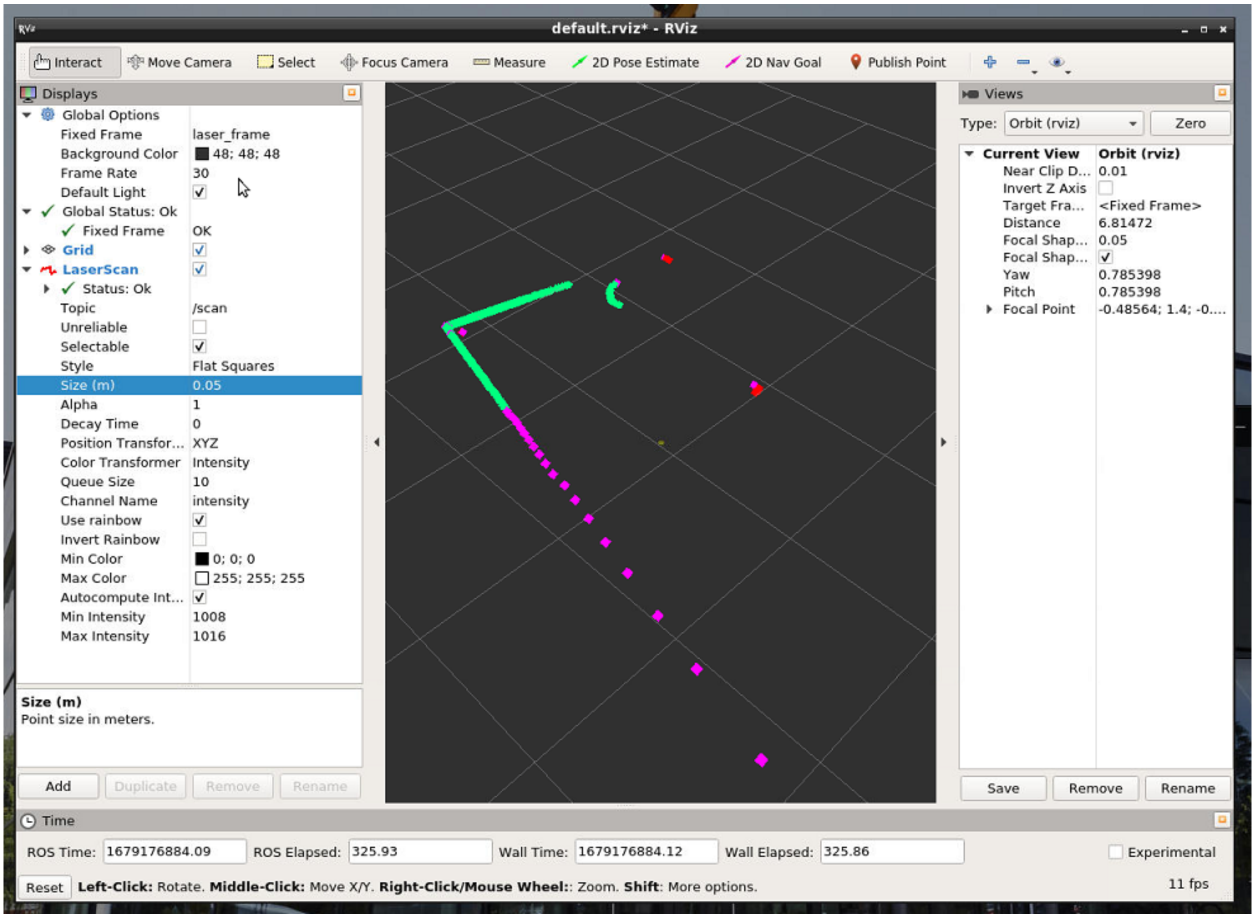
You will notice that RVIZ is in error state. This is because it does not have a reference point to plot the pointcloud data against.

Under **Display**, change the Fixed Frame to laser_frame.

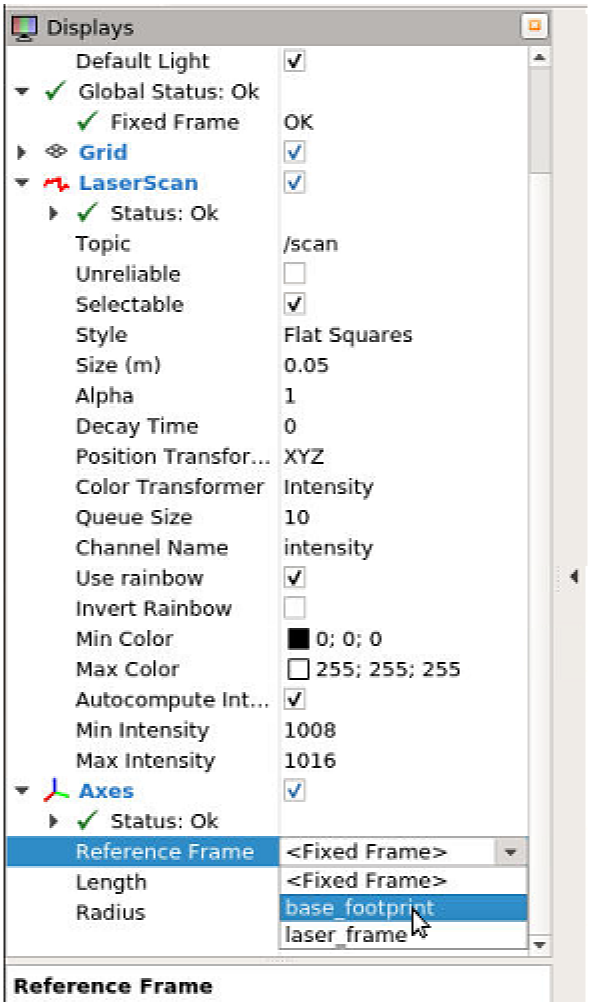


You should now observe data being visualized in RVIZ.

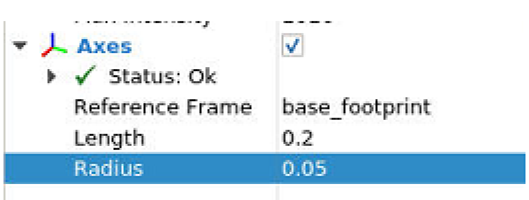
Modify Laser Scan > Size (m) to 0.05 to make the points a bit larger.

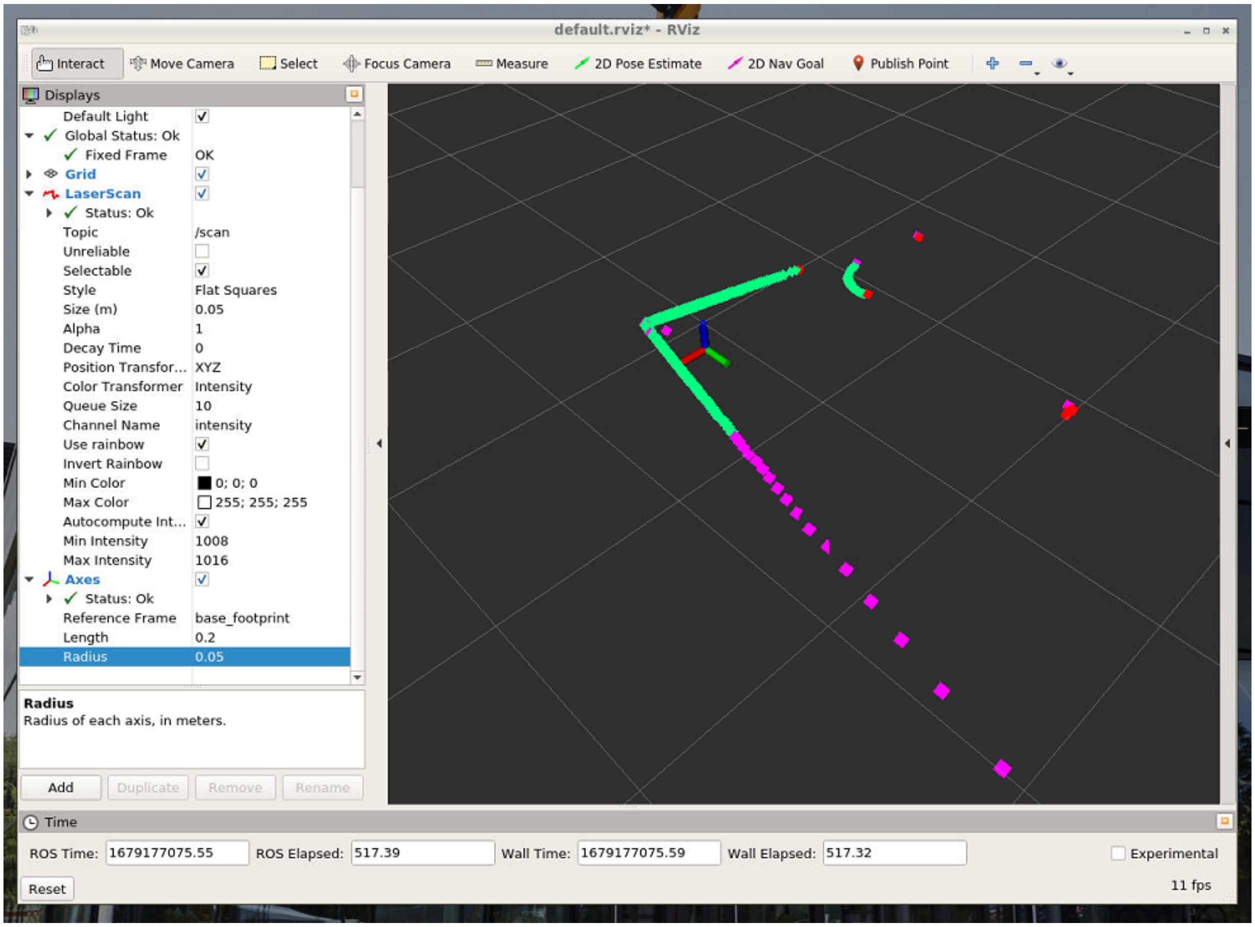


Lastly, lets input a marker for where the LiDAR is located. Insert an Axis display and set it to the base_footprint frame.



Modify the Length and Radius values.





Exercise A

You have successfully visualized LiDAR data. Take a screenshot of RVIZ and include it with your report.

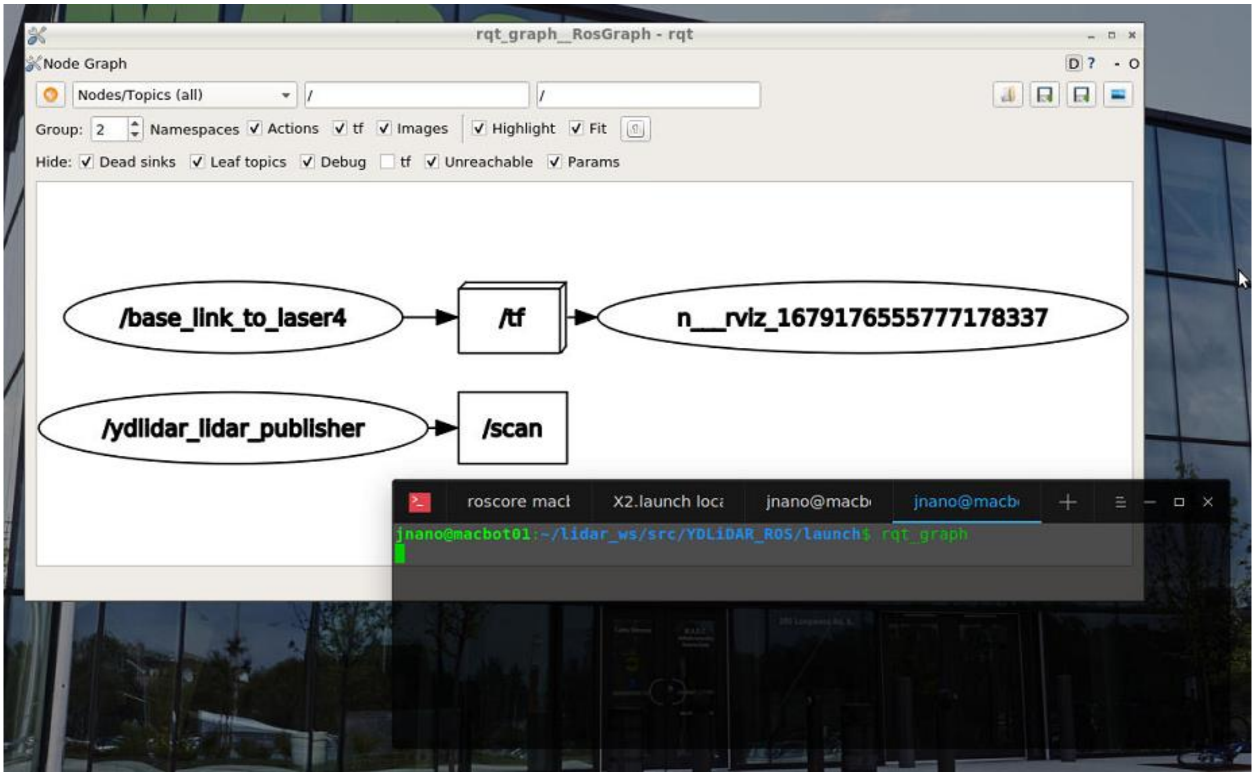
Generating a Diagram

Lastly, lets use the rqt_graph tool to generate a live-updated diagram of our ROS system.

In a new terminal window (or tab, or splt-screen), run the rqt_graph command.

rqt_graph

Set the graphical tool to display **Nodes/Topics (all)**



Exercise B

You have successfully used the RQT_Graph tool to generate a live diagram of your ROS project. Take a screenshot and include it with your report.