

Velodyne VLP32C (UltraPuck)



By : N Sai Pranay Kumar

Version : 1

Velodyne Lidar[®]

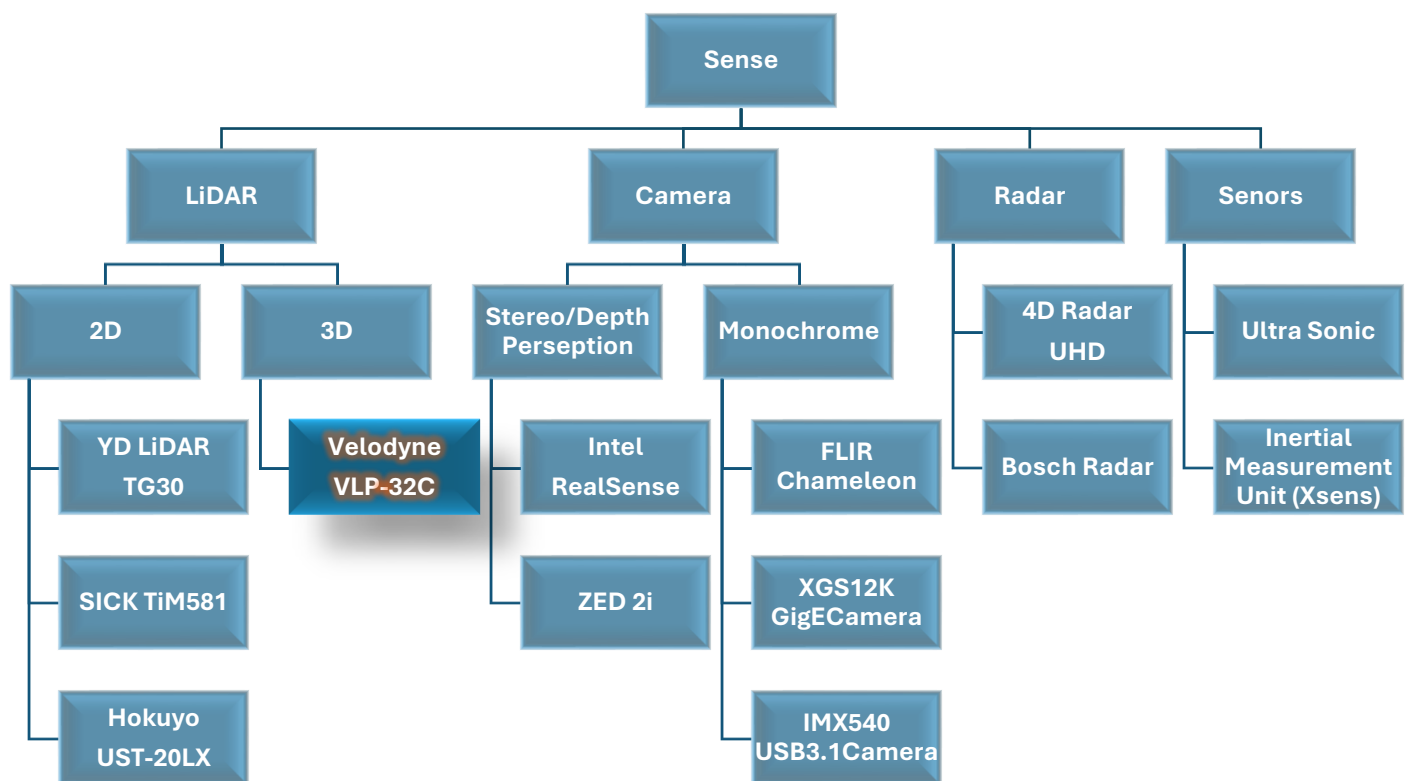
Version Control

Version	Summary	Created by	Reviewed by	Date
1	Starter Guide for Velodyne VLP32C	N Sai Pranay Kumar	G Nithish Chandra Reddy	

Contents

Sense Segment	4
I. Introduction to LiDAR	5
○ History of LiDAR	5
○ Types of LiDARs	5
○ Applications of LiDAR	5
II. Velodyne VLP32c Introduction	6
○ Product Overview :	6
○ Specifications :	6
III. Prerequisites	7
IV. Velodyne VLP32C Interface	8
V. Visualization using SDK	9
VI. Integration with ROS	11
○ Build Velodyne ROS Driver	11
○ Run Velodyne ROS Driver	12
○ Recording the data into a .bag file	14
○ Visualizing .bag file	14

Sense Segment



I. Introduction to LiDAR

LiDAR (Light Detection and Ranging) is a remote sensing technology that uses laser light to measure distances. By emitting laser pulses and measuring the time it takes for the light to return after hitting an object, LiDAR can create precise, high-resolution 3D maps of the environment.

○ History of LiDAR

LiDAR technology has its roots in the 1960s, developed shortly after the invention of the laser. Initially, it was used for meteorological purposes, such as measuring atmospheric particles. Over the decades, advances in technology and computing power have significantly expanded LiDAR's applications, making it a crucial tool in geology, forestry, and, more recently, autonomous vehicles and robotics.

○ Types of LiDARs

2D LiDAR:

This type of LiDAR scans in a single plane, creating a two-dimensional map of the surroundings. It's commonly used in applications where a flat representation is sufficient, such as in industrial automation settings or simple obstacle detection systems.

3D LiDAR:

By rotating or using multiple beams, 3D LiDAR captures data in multiple planes, resulting in a three-dimensional map of the environment. This type of LiDAR is essential for more complex applications requiring a comprehensive spatial understanding, such as autonomous driving, advanced robotics, and detailed topographical mapping.

○ Applications of LiDAR

1. **Autonomous Vehicles:** LiDAR is critical in self-driving cars, providing detailed 3D maps that help the vehicle navigate and avoid obstacles.
2. **Geospatial Mapping:** Used for creating high-resolution topographic maps, LiDAR helps in urban planning, flood modeling, and forestry management.
3. **Archaeology:** LiDAR can penetrate forest canopies to reveal hidden structures and landscapes, aiding in archaeological discoveries.
4. **Environmental Monitoring:** It helps in tracking changes in vegetation, coastline erosion, and other environmental changes over time.
5. **Agriculture:** Precision farming uses LiDAR for crop assessment, field mapping, and soil analysis.
6. **Infrastructure Inspection:** LiDAR is used to inspect power lines, bridges, and other infrastructure for maintenance and safety assessments.

LiDAR's ability to produce accurate and high-resolution data quickly makes it invaluable across various industries, driving innovation and efficiency in multiple fields.

II. Velodyne VLP32c Introduction

○ Product Overview :

Velodyne LiDAR's ULTRA Puck VLP-32C is an advanced sensor that combines long-range performance with 0.33-degree vertical resolution in a compact form factor. It is a high-resolution sensor developed with automotive applications in mind and it retains the innovative breakthroughs in 3D LiDAR, such as 360° surround view with real-time 3D data. The VLP-32C includes distance and calibrated reflectivity measurements at all rotational angles.

○ Specifications :

- Range : Up to 200 meters.
- Field of View (Vertical) : 40° (-25° to +15°).
- Field of View (Horizontal) : 360°.
- Accuracy : ± 3 cm.
- Number of Channels : 32.
- Minimum Angular Resolution (Vertical) : 0.33° (Non-Linear Distribution)
- Angular Resolution (Horizontal/Azimuth) : 0.1° to 0.4°
- Data Rate : 1.2 million points per second.
- Rotation Rate : 5 - 20 Hz.
- Power Consumption : 8 Watts (Typical).
- IP Rating : IP67 (Dust and waterproof).
- Power Consumption: 10 W (Typical)
- Operating Voltage: 10.5 V – 18 V (with interface box and regulated power supply)

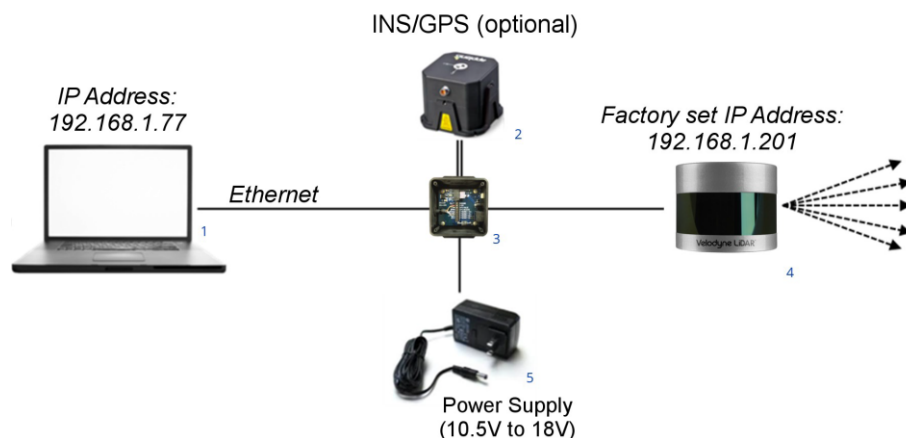


Figure 1 General SetUp of Velodyne with System

Official Website : <https://icave2.cse.buffalo.edu/resources/sensor-modeling/VLP32CManual.pdf>

III. Prerequisites

Linux Version : Ubuntu 20.04 and before versions

Ubuntu Installation : <https://robocademy.com/2020/05/17/best-4-ways-to-install-ubuntu-for-ros/>

YouTube Link : <https://youtu.be/mXyN1aJYefc?si=XJbUZmC5jgrDRQQF> (Dual Boot)

ROS : Noetic and before versions

ROS Installation : <https://wiki.ros.org/noetic/Installation/Ubuntu> (noetic version)

Complete [ROS tutorials](#) to get familiar with ROS.

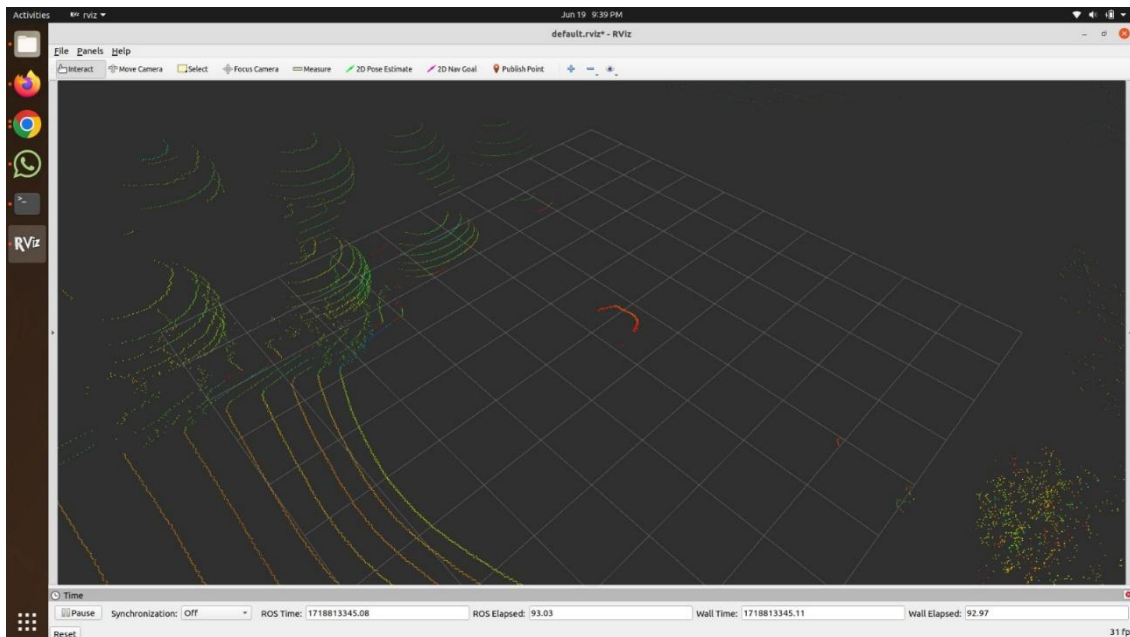
<https://robodev.blog/series/ros101> (ROS Basics)

Before moving to the Hardware interface execute the .bag file and visualize the data... from the following link

Dataset : <https://drive.google.com/drive/folders/1mP5sizC77ARcK9fO5BmCwU82d5yzVs6Y?usp=sharing>

- Instead of recording a .bag file. First, visualize the data with the bag file by following the steps.

Sample Output :



Note

- To uninstall Ubuntu (Dual Boot) - <https://youtu.be/mQyxtWrUNIE?si=3eXOnEqIGjViTH0D>
- If you face a Grub Screen Issue - <https://youtu.be/ih2NjlhLLic?si=TEpkIDzkKJVkWzZB>

IV. Velodyne VLP32C Interface

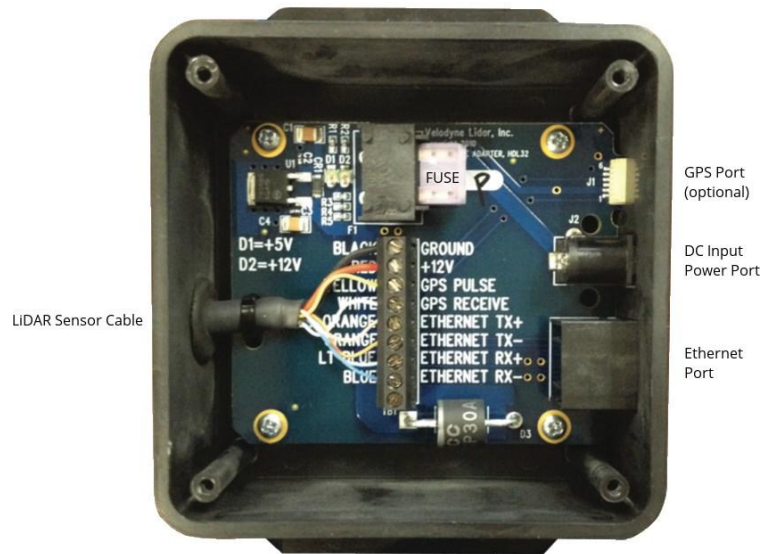


Figure 2 Interface Box (Power and Data Connections)

Signal Specifications	Signal	Input/Output
Black	Ground	Input
Red	Power	Input
Yellow	GPS Sync Pulse	Input
White	GPS Serial Receive	Input
Light Orange	Ethernet TX+	Output
Orange	Ethernet TX-	Output
Light Blue	Ethernet RX+	Input
Blue	Ethernet RX-	Input

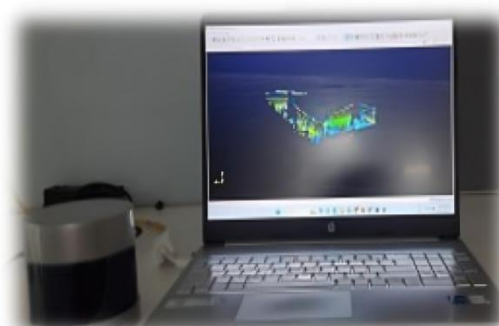


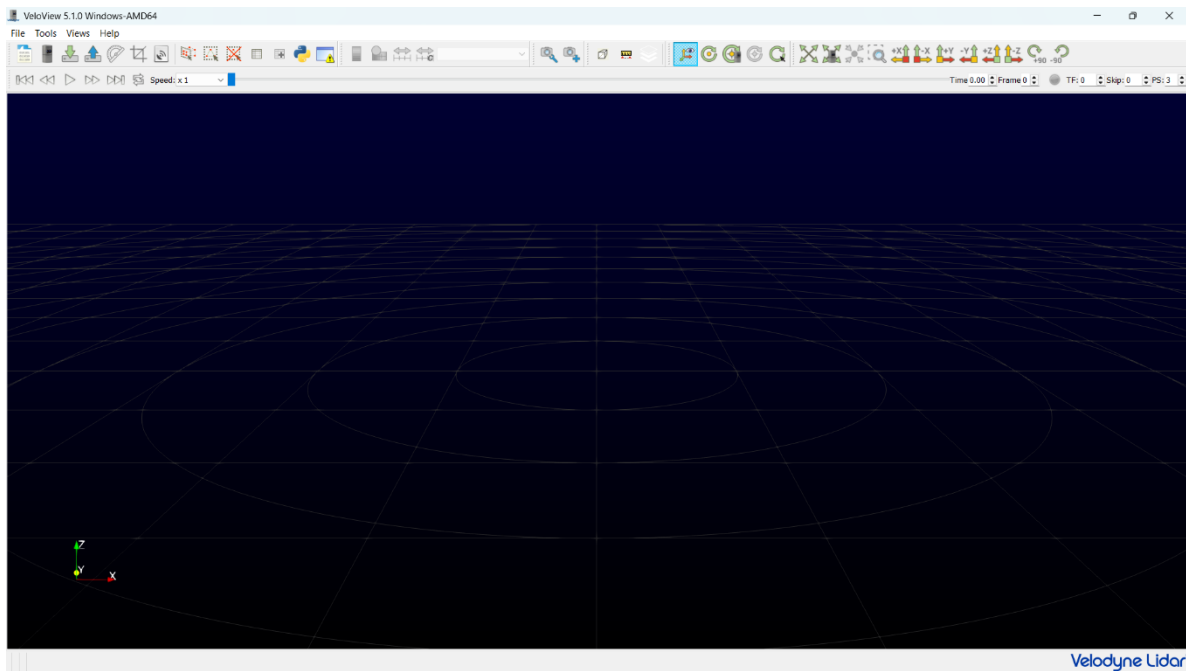
Figure 3 Velodyne LiDAR SetUp

V. Visualization using SDK

Download the SDK from <https://www.paraview.org/veloview/>

Download & Install the latest version of the SDK.

Open the application you will get the screen as shown below...

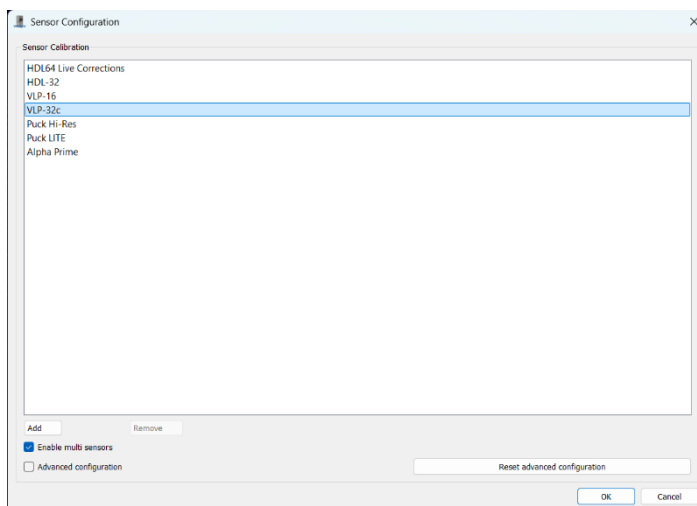


Connect your Velodyne VLP-32c with your System.

After, Click on the sensor configuration icon as shown

You will get a screen like this

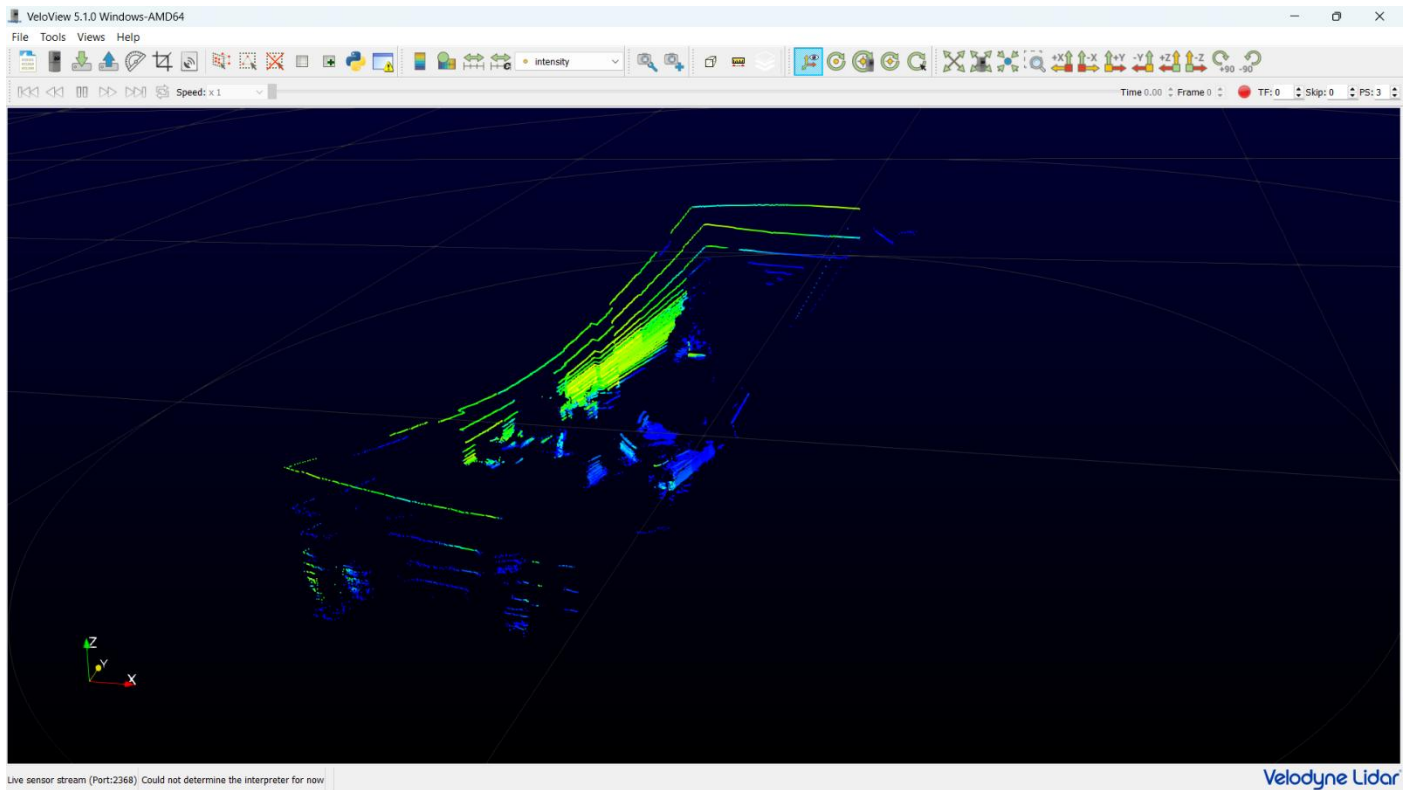
Select VLP-32c



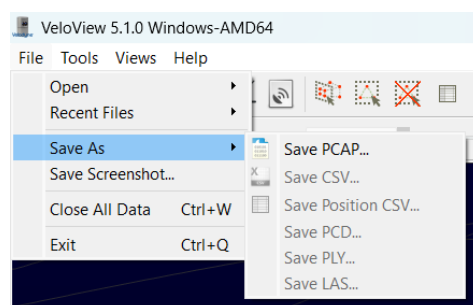
Click on the play button.



You will get LiDAR data as shown...



You can save the data in many formats as shown...



VI.Integration with ROS

The Installation procedure has been done in Ubuntu 20.04 and ROS Noetic Version.

Open the ROS terminal and execute the following commands.

```
$ cd ~/catkin_ws/
$ catkin_make
$ source ./devel/setup.bash
```

○ Build Velodyne ROS Driver

1. Clone ROS driver package from GitHub :

```
$ cd ~/catkin_ws/src
$ git clone https://github.com/ros-drivers/velodyne.git
```

2. Installing dependencies :

```
$ cd ..
$ rosdep install --from-paths src --ignore-src -r -y
```

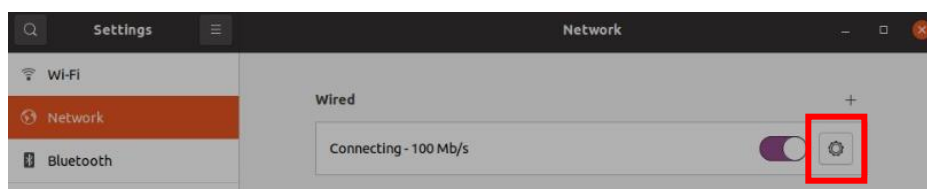
3. Build & Source the workspace

```
$ catkin_make
$ source ./devel/setup.bash
```

Connect your Velodyne LiDAR with the System.

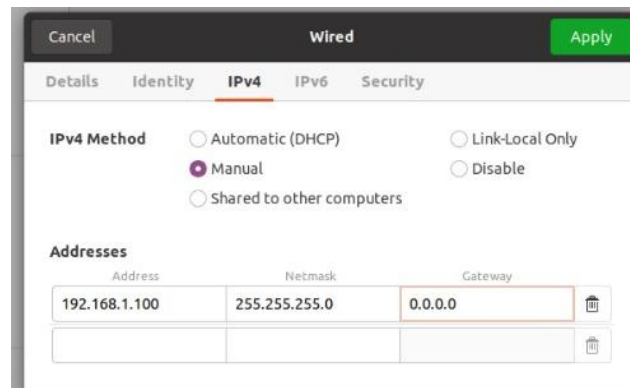
After connecting the LiDAR, open settings in your Linux and Click on Network.

Click the settings icon under Wired as shown...



You will have a screen as shown, click on IPV4, change the option from Automatic (DHCP) to Manual, and enter the values as shown...

Then click on the apply button.



Note :

We are changing the IP Address because the default IP Address of Velodyne VLP-32c LiDAR is 192.168.1.201. The communication will happen in a similar series. So, we changing the IP Address of our system to 192.168.1.100

Now, your LiDAR is successfully connected to your system.

○ Run Velodyne ROS Driver

You can run the driver using the launch file.

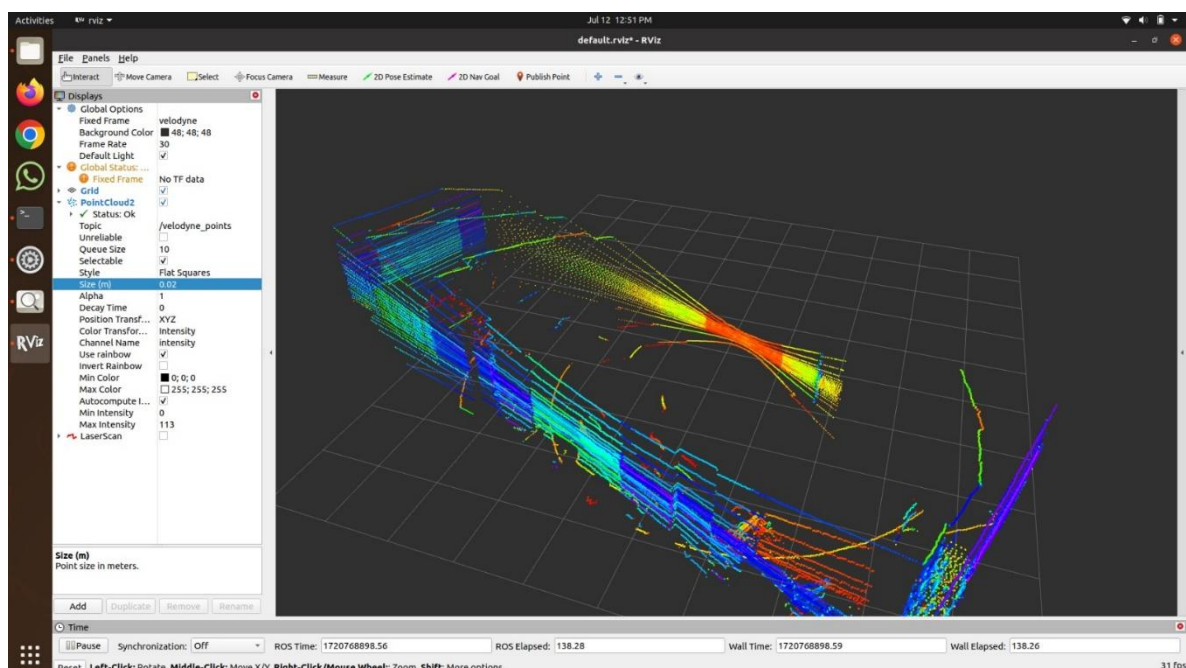
`roslaunch velodyne_pointcloud VLP-32C_points.launch`

This opens an RViz terminal. (If not RViz is not opened, move to another terminal and give **rviz** click enter)

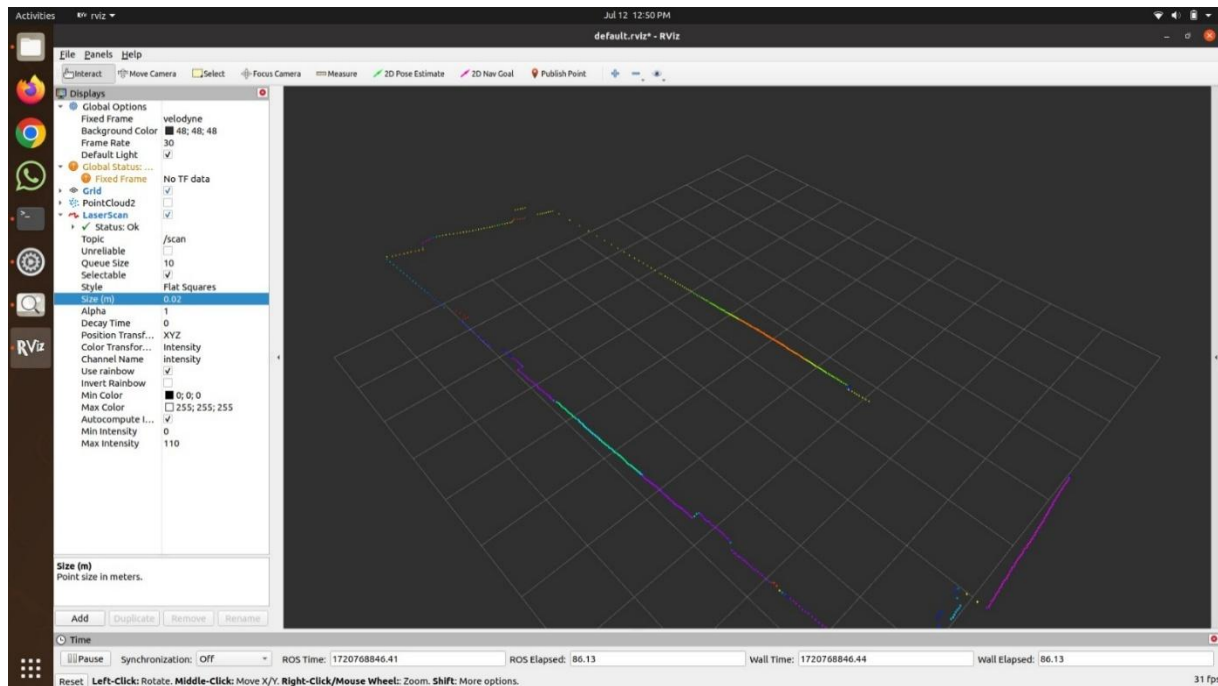
In the RViz terminal select the global frame as *Velodyne* and click ADD to add the topic.

Go to By topic select */Velodyne_points* and click OK.

Now you can visualize the real-time data in the RViz terminal.



It can also publish `/scan` data to visualize it, Add the topic in RViz.



Note :

If you get the error any error similar to this while launching the file...

```

RLEException: [TG.launch] is neither a launch file in package [ydlidar_ros_driver]
nor is [ydlidar_ros_driver] a launch file name
The traceback for the exception was written to the log file
  
```

source your environment using the following command

```
$ source ./devel/setup.sh
```

Now again launch the file to power up the lidar and visualize it in RViz...

If you haven't installed RViz in your system, use the following command.

```
sudo apt-get install ros-noetic-rviz
```

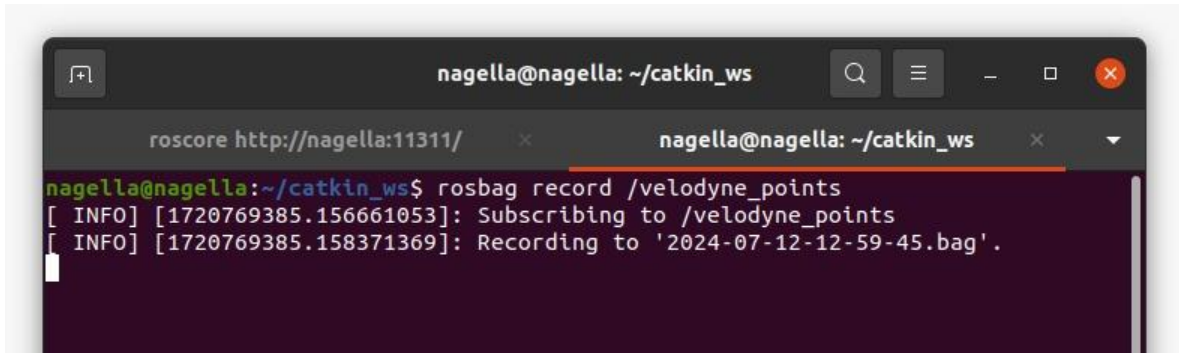
Till Now, we have visualized the data in real time. Now let us record the data and use it for various applications.

○ Recording the data into a .bag file

To record the data firstly you have to connect the LiDAR to your system and launch it in the terminal...

`roslaunch velodyne_pointcloud VLP-32C_points.launch`

open another terminal and use the following command to record the data

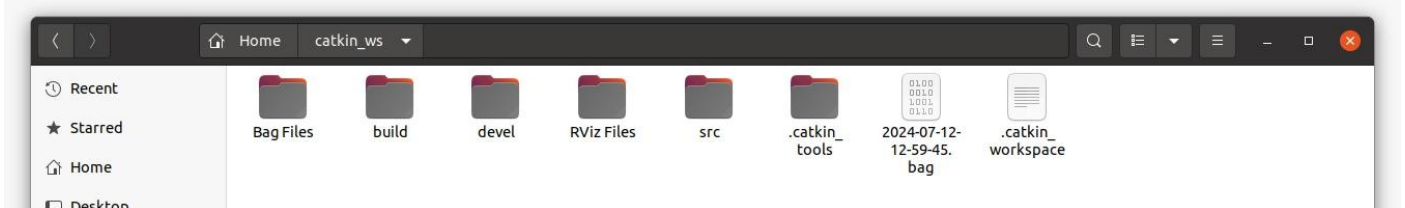


```
nagella@nagella: ~/catkin_ws
roscore http://nagella:11311/
nagella@nagella: ~/catkin_ws$ rosbag record /velodyne_points
[ INFO] [1720769385.156661053]: Subscribing to /velodyne_points
[ INFO] [1720769385.158371369]: Recording to '2024-07-12-12-59-45.bag'.
```

/velodyne_points is used because it is the topic that VLP32C publishes.

To stop the recording use Ctrl+C in the terminal.

This stores the .bag file in your workspace as shown



○ Visualizing .bag file :

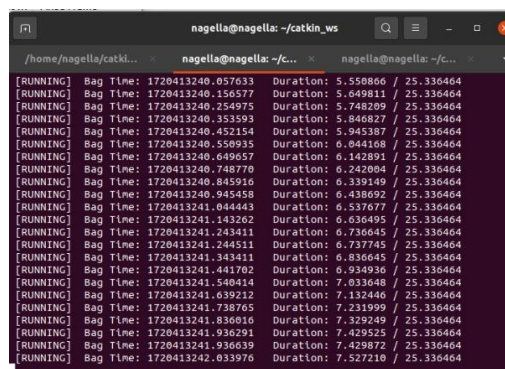
Disconnect the LiDAR if you are connected to the system.

To play the .bag file that you recorded. Open the terminal and run `roscore`.

Open one more terminal, use the following command

`$ rosbag play 'Path_to_your_bag_file'`

This will play the bag file as shown



```
nagella@nagella: ~/catkin_ws
/home/nagella/catkin_ws$ rosbag play
[RUNNING] Bag Time: 1720413240.057633 Duration: 5.550866 / 25.336464
[RUNNING] Bag Time: 1720413240.156577 Duration: 5.049811 / 25.336464
[RUNNING] Bag Time: 1720413240.254975 Duration: 5.748209 / 25.336464
[RUNNING] Bag Time: 1720413240.353593 Duration: 5.846827 / 25.336464
[RUNNING] Bag Time: 1720413240.452154 Duration: 5.945387 / 25.336464
[RUNNING] Bag Time: 1720413240.550935 Duration: 6.044168 / 25.336464
[RUNNING] Bag Time: 1720413240.649657 Duration: 6.142891 / 25.336464
[RUNNING] Bag Time: 1720413240.748770 Duration: 6.242804 / 25.336464
[RUNNING] Bag Time: 1720413240.845916 Duration: 6.339149 / 25.336464
[RUNNING] Bag Time: 1720413240.945458 Duration: 6.438692 / 25.336464
[RUNNING] Bag Time: 1720413241.044443 Duration: 6.537677 / 25.336464
[RUNNING] Bag Time: 1720413241.143262 Duration: 6.636495 / 25.336464
[RUNNING] Bag Time: 1720413241.243411 Duration: 6.736645 / 25.336464
[RUNNING] Bag Time: 1720413241.344511 Duration: 6.837745 / 25.336464
[RUNNING] Bag Time: 1720413241.444102 Duration: 6.934936 / 25.336464
[RUNNING] Bag Time: 1720413241.540414 Duration: 7.033648 / 25.336464
[RUNNING] Bag Time: 1720413241.639212 Duration: 7.132446 / 25.336464
[RUNNING] Bag Time: 1720413241.738765 Duration: 7.231999 / 25.336464
[RUNNING] Bag Time: 1720413241.836010 Duration: 7.329249 / 25.336464
[RUNNING] Bag Time: 1720413241.936291 Duration: 7.429525 / 25.336464
[RUNNING] Bag Time: 1720413241.936639 Duration: 7.429872 / 25.336464
[RUNNING] Bag Time: 1720413242.033976 Duration: 7.527218 / 25.336464
```


To visualize it open RViz using another terminal.

Set the global frame as *velodyne* and add the topic */velodyne_points*.

Now you can visualize the recorded data as shown below...

