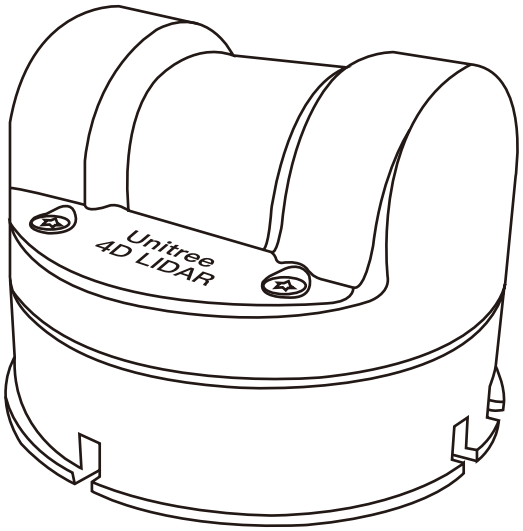
Unilidar SDK

User Manual v 1.0

2023.05



Unitree Lidar

目 录

Preface

This repository is a SDK for [Unitree L1 LiDAR](https://www.unitree.com/LiDAR).

You can use the code interfaces in this repository to obtain point cloud data and IMU data measured in our lidar, as well as configure related parameters.

We provide three commonly used interfaces for our LiDAR:
- if you prefer to use the original **C++ SDK** directly, you can refer to [README.md](./unitree_lidar_sdk/README.md);
- if you want to use **ROS**, you can refer to [README.md](./unitree_lidar_ros/src/unitree_lidar_ros/README.md);
- if you are developing with the latest **ROS2**, you can refer to [README.md](./unitree_lidar_ros2/src/unitree_lidar_ros2/README.md).

Unilidar SDK

·Introduction

Unilidar SDK is a cmake package, which is specially used for running 'Unitree LiDAR L1'.

The functions that this package can provide includes:

- Parse the raw data transmitted from the lidar hardware, and convert it into pointcloud and IMU data
- Get the pointcloud data
- Get the IMU data

The output pointcloud defaultly uses a self-defined data type so that this SDK doesn't rely too much on external dependencies. In other case,

- If you are used to use [Point Cloud Library](https://pointclouds.org/), you can use the header `unitree_lidar_sdk_pcl.h` to transform our pointcloud to PCL format;
- If you wish to directly use a [ROS](https://www.ros.org/) package, you are also able to utilize our ROS pacakge for this lidar.

·Dependency

We have verified that this package can successfully run under this environment:

- Ubuntu 20.04

This SDK hardly depends on any external dependencies.
But if you want to use PCL cloud format, you need to install one.

·Configure

Connect your lidar to your computer with a USB cable, then confirm your serial port name for lidar:

```
$ ls /dev/ttyUSB*
/dev/ttyUSB0
```

The default serial port name is '/dev/ttyUSB0' .
If it is not the default one, you need to modify the configuration parameter in `example_lidar.cpp`.

·Build

You can build this program as a cmake project:

```
cd unitree_lidar_sdk
mkdir build
cd build
cmake .. && make -j2
```

'mkdir build' is used to create a folder named "build" for storing the compiled files;
'cd build' is used to enter the "build" folder;
'cmake ..' is used to generate the Makefile;
'make -j2' is used to compile and generate the executable file.
During the compilation process, -j2 indicates that 2 threads are used for the compilation, which can speed up the compilation process. If your computer has more processor cores, you can adjust this parameter accordingly.

·Run

Directly run the example:

```
../bin/example_lidar
```

The output is like this:

```
$ ../bin/example_lidar
lidar firmware version = 0.3.2+230511
lidar sdk version = 1.0.3


Dirty Percentage = 5.145833 %
Dirty Percentage = 4.166667 %
Dirty Percentage = 4.166667 %
Turn on all the LED lights ...
Turn off all the LED lights ...
Set LED mode to: FORWARD_SLOW ...
Set LED mode to: REVERSE_SLOW ...
Set LED mode to: SIXSTAGE_BREATHING ...
Set Lidar working mode to: NORMAL_MODE ...
```

```
An IMU msg is parsed!
 stamp = 1683874160.559222, id = 729
 quaternion (x, y, z, w) = [0.0131, -0.0091, 0.6888, -0.7225]
An IMU msg is parsed!
 stamp = 1683874160.564979, id = 121
 quaternion (x, y, z, w) = [0.0102, -0.0093, 0.7099, -0.7018]
An IMU msg is parsed!
 stamp = 1683874160.568425, id = 122
 quaternion (x, y, z, w) = [0.0118, -0.0096, 0.7099, -0.7018]
An IMU msg is parsed!
 stamp = 1683874160.573472, id = 123
 quaternion (x, y, z, w) = [0.0126, -0.0093, 0.7098, -0.7018]
An IMU msg is parsed!
 stamp = 1683874160.577348, id = 124
 quaternion (x, y, z, w) = [0.0128, -0.0093, 0.7099, -0.7018]

A Cloud msg is parsed!
 stamp = 1683874145.535888, id = 1
 cloud size  = 278
 first 10 points (x,y,z,intensity,time,ring) =
   (-0.029885, -0.136897, 0.000448, 88.000000, 0.000000, 0)
   (-0.035384, -0.171399, 0.005140, 91.000000, 0.000023, 0)
   (-0.043000, -0.219542, 0.012437, 127.000000, 0.000046, 0)
   (-0.054879, -0.294965, 0.024572, 132.000000, 0.000069, 0)
   (-0.055288, -0.301204, 0.033170, 106.000000, 0.000093, 0)
   (-0.054542, -0.300348, 0.041173, 101.000000, 0.000116, 0)
   (-0.053764, -0.299285, 0.049148, 99.000000, 0.000139, 0)
   (-0.056123, -0.318406, 0.060981, 92.000000, 0.000162, 0)
   (-0.055230, -0.316827, 0.069422, 91.000000, 0.000185, 0)
   (-0.051257, -0.294854, 0.072849, 128.000000, 0.000208, 0)
   ...
```

Here, we print the first 10 points of the pointcloud message and the quaternion of the IMU message.
The meanings of each variable are as follows:

- Dirty Percentage: The dirt percentage of the lidar. If the dirt percentage is high, clean the radar optical window.
- stamp: the timestamp of the data, in seconds.
- id: the identifier of the data.

- cloud size: the number of points in the point cloud.
- x,y,z: the coordinates of the point in the x, y, and z directions.
- intensity: the intensity of the reflected light of the point.
- time: the time difference of the point relative to the point cloud timestamp (stamp).
- ring: the number of the laser ring to which the point belongs.
- quaternion (x, y, z, w): the quaternion vector [x, y, z, w] of the IMU.

Notice:

- In Ubuntu, accessing a serial port device requires the appropriate permissions. If your C++ program does not have sufficient permissions to access the serial port device, you will get a **"Permission denied"** error.
- To solve this error, you can use the following command to add the current user to the dialout group:

  ```
  sudo usermod -a -G dialout $USER
  ```

- After adding the user to the dialout group, you need to log out and log back in for the changes to take effect.

·Version History

v1.0.0(2023.05.04)

- Support firmware version of 0.3.1

v1.0.1 (2023.05.05)

- Support firmware version of 0.3.1
- Add support of setting lidar working mode, e.g. `NORMAL_MODE` and `STANDBY_MODE`
- Add support of LED lights

v1.0.2(2023.05.11)

- Support firmware version of 0.3.2

v1.0.3(2023.05.12)

- Support firmware version of 0.3.2
- Add support of getting the percentage of removed dirty points

v1.0.4(2023.05.31)

- Support firmware version of 1.0.1

v1.0.5(2023.06.05)

- Support firmware version: 1.0.1
- Update default `rotate_yaw_bias` to calibrated value `-38.5` degree.
- Update `README.md` with notice to solve the "Permission denied" error while opening serial port.

v1.0.6(2023.06.19)

- Support firmware version: 1.0.1
- Modify the `initialize()` function to check whether the specified serial port exist.
- If the serial port name does not exist, initialization fails and `return -1` rather than throw out an error unexpectedly。
- Add z bias to lidar basis plane

Unilidar ROS

·Introduction

Unilidar ROS is a ROS package, which is specially used for running Unitree LiDAR products.

The functions that this package can provide includes:

- Parse the raw data transmitted from the lidar hardware, and convert it into pointcloud and IMU data
- Publish the pointcloud data to a ROS topic
- Publish the IMU data to a ROS topic
- In addition, we provide a yaml file to configure the relevant parameters of the lidar.

·Dependency

The dependencies include 'PCL' and 'ROS'.

We have verified that this package can successfully run under this environment:

- Ubuntu 20.04
- ROS noetic
- PCL-1.10
- unitree_lidar_sdk

You are suggested to configure an environment like this to run this package.

·Configuration

Connect your lidar to your computer with a USB cable, then confirm your serial port name for lidar:

```
$ ls /dev/ttyUSB*
/dev/ttyUSB0
```

The default serial port name is '/dev/ttyUSB0' .

If it is not the default one, you need to modify the configuration in 'unitree_lidar_ros/config/config.yaml' and change the 'port'name to yours. For example:

```
# Serial Port
port: "/dev/ttyYourUSBPortName"
```


You can leave other parameters in the configuration file with their default value.

If you have special needs such as changing the cloud topic name or IMU topic name, you are allowed to configure them in the configuration file as well.

The defalut cloud topic and its frame name is:

- topic: "unilidar/cloud"
- frame: "unilidar_lidar"

The defalut IMU topic and its frame name is:

- topic: "unilidar/imu"
- frame: "unilidar_imu"

·Build

You can just build this ROS package as follows:

- Clone the repository:

```
git clone https://github.com/unitreerobotics/unilidar_sdk.git
```

- Compile:

```
cd unilidar_sdk/unitree_lidar_ros
catkin_make
```

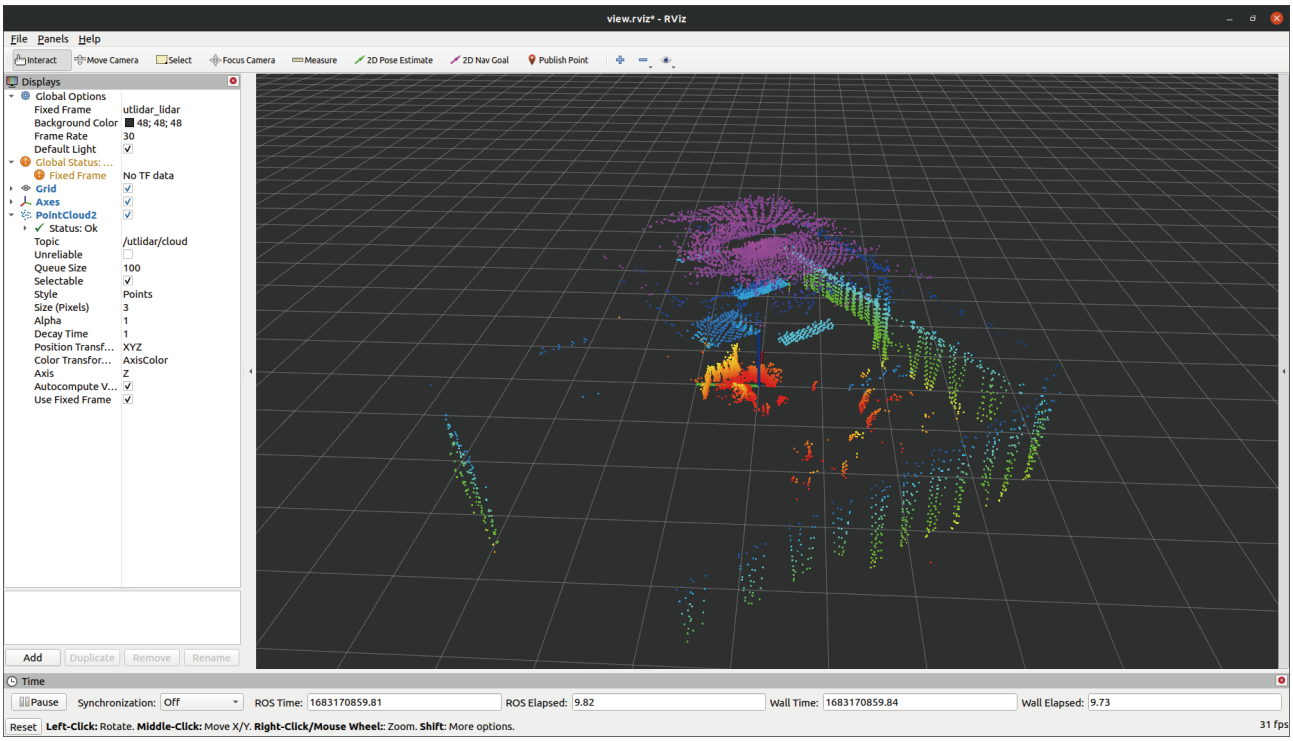
The compiled content will be saved in the unilidar_sdk/unitree_lidar_ros/devel.

·Run

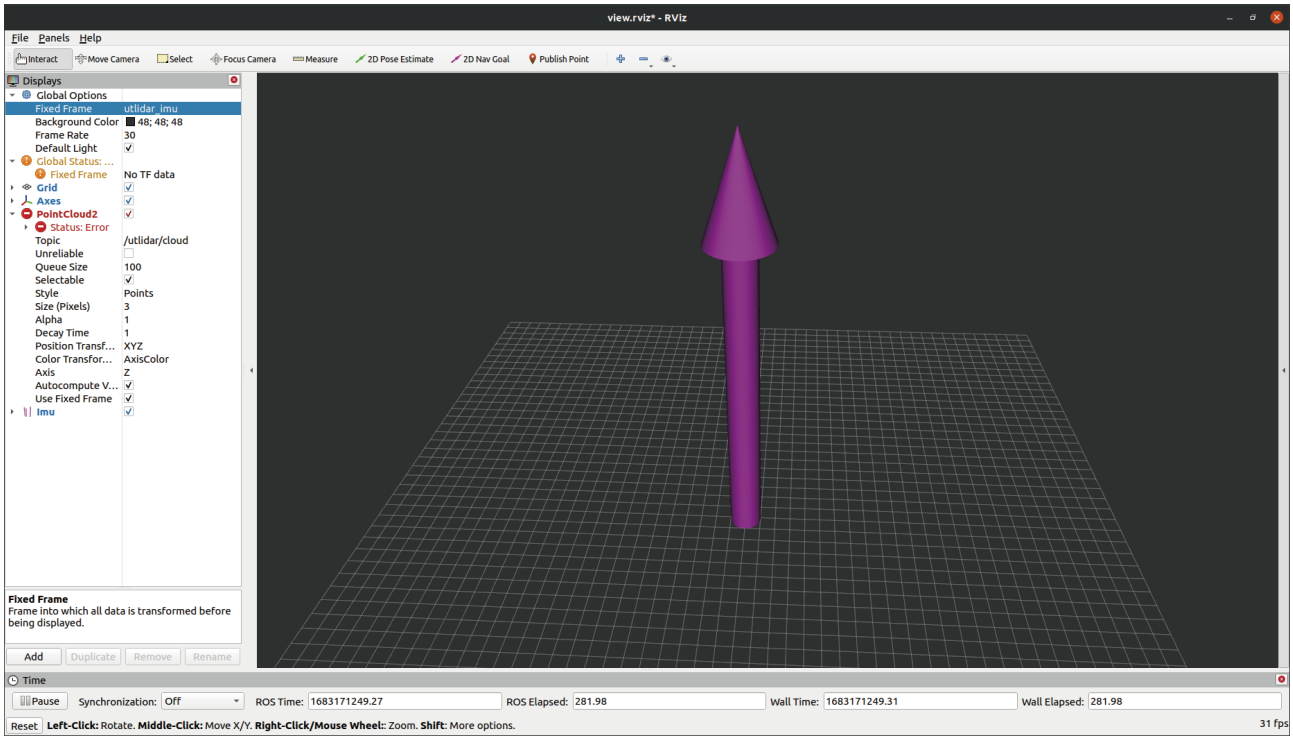
Then you need to source this ROS packege environment and then directly run the launch file:

```
source devel/setup.bash
ros launch unitree_lidar_ros run.launch
```

In the Rviz window, you will see our lidar pointcloud like this:



You can change the 'Fixed Frame' to the imu frame 'unilidar_imu' ,so that you can view the IMU quaternion vector:



·YAML configuration file description

The file is located at unitree_lidar_ros\config\config.yaml.

- port: The serial port number for connecting the LiDAR to the computer is /dev/ttyUSB0 by default.
- rotate_yaw_bias: The fixed offset angle in degrees that may exist during the installation of the LiDAR. This parameter is used to calibrate the orientation of the point cloud data, and its default value is 0.
- range_scale: The range scaling factor of the LiDAR, which is used to calibrate the ranging accuracy of the LiDAR. Its default value is 0.001.
- range_bias: The range offset of the LiDAR, which is used to calibrate the ranging accuracy of the LiDAR. Its default value is 0.
- range_max: The maximum range of the LiDAR, measured in meters, with a default value of 50.
- range_min: The minimum range of the LiDAR, measured in meters, with a default value of 0.
- cloud_frame: The name of the coordinate system for the point cloud data, with a default value of 'unilidar_lidar'.
- cloud_topic: The ROS topic name for the point cloud data, with a default value of 'unilidar/cloud'.
- cloud_scan_num: The number of laser ring IDs for the point cloud data, with a default value of 18.
- imu_frame: The name of the coordinate system for the IMU data, with a default value of 'unilidar_imu'.
- imu_topic: The ROS topic name for the IMU data, with a default value of 'unilidar/imu'.

·Version History

v1.0.0(2023.05.04)

- Support 'Unilidar_sdk' : v1.0.0

v1.0.1(2023.05.05)

- Support 'Unilidar_sdk' : v1.0.1

v1.0.2(2023.05.12)

- Support 'Unilidar_sdk' : v1.0.3

v1.0.3(2023.05.30)

- Support 'Unilidar_sdk' : v1.0.4

Unilidar ROS2

·Introduction

Unilidar ROS2 is a ROS2 package, which is specially used for running Unitree LiDAR products.

The functions that this package can provide includes:

- Parse the raw data transmitted from the lidar hardware, and convert it into pointcloud and IMU data
- Publish the pointcloud data to a ROS topic
- Publish the IMU data to a ROS topic
- In addition, you can configure the relevant parameters of the lidar in the `launch.py`.

·Dependency

The dependencies include 'PCL' and 'ROS2'.

We have verified that this package can successfully run under this environment:

- Ubuntu 20.04
- ROS2 foxy
- PCL-1.10
- unitree_lidar_sdk

You are suggested to configure an environment like this to run this package.

·Configuration

Connect your lidar to your computer with a USB cable, then confirm your serial port name for lidar:

```
$ ls /dev/ttyUSB*
/dev/ttyUSB0
```

The default serial port name is '/dev/ttyUSB0' .

If it is not the default one, you need to modify the configuration in `unitree_lidar_ros2/launch/launch.py` and change the `port` parameter to yours. For example:

```
{'port': '/dev/ttyUSB0'},
```

You can leave other parameters in the configuration file with their default value.

If you have special needs such as changing the cloud topic name or IMU topic name, you are allowed to configure them in the configuration file as well.

The defalut cloud topic and its frame name is:

- topic: "unilidar/cloud"
- frame: "unilidar_lidar"

The defalut IMU topic and its frame name is:

- topic: "unilidar/imu"
- frame: "unilidar_imu"

·Build

You can just build this ROS2 package as follows:

- Clone the repository:

```
git clone https://github.com/unitreerobotics/unilidar_sdk.git
```

- Compile:

```
cd unilidar_sdk/unitree_lidar_ros2
catkin_make
```

The compiled content will be saved in the unilidar_sdk/unitree_lidar_ros2/devel.

·Run

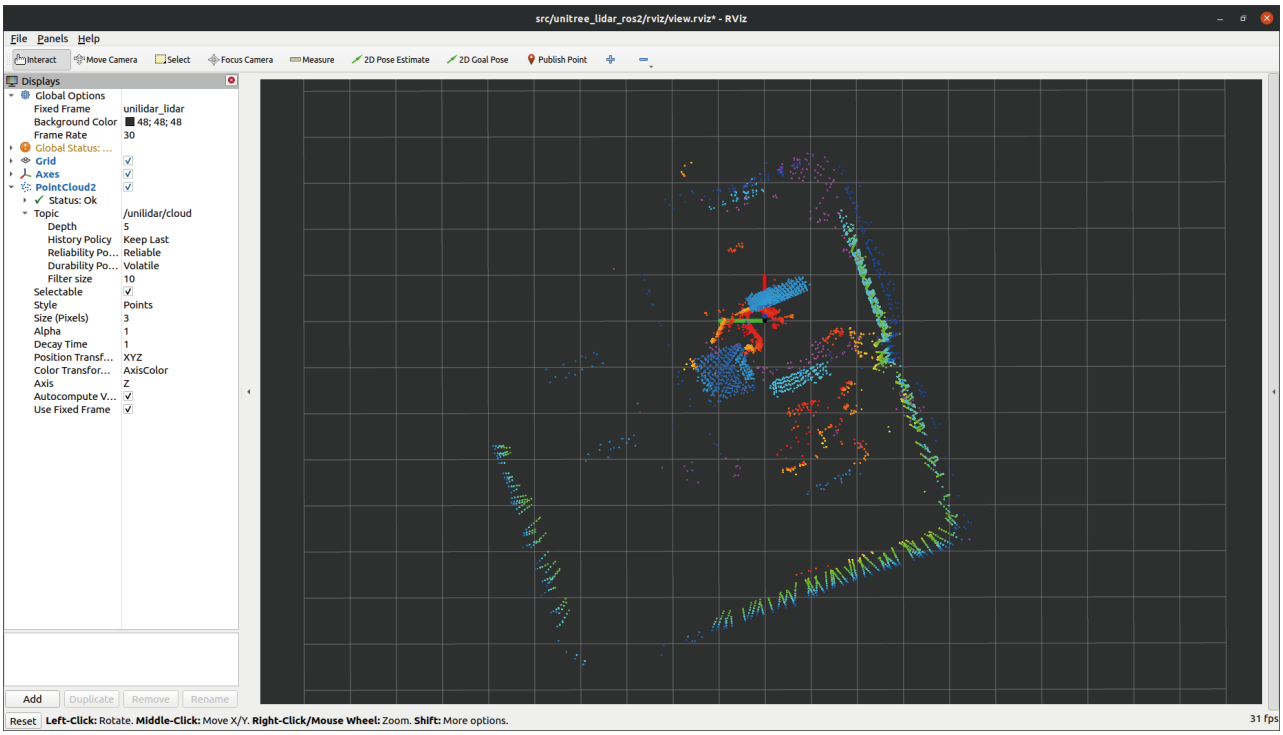
Then you need to source this ROS2 packege environment and then directly run the launch file:

```
source install/setup.bash
ros2 launch unitree_lidar_ros2 launch.py
```

To visualize our cloud in Rviz2, you need to run:

```
rviz2 -d src/unitree_lidar_ros2/rviz/view.rviz
```

In the Rviz window, you will see our lidar pointcloud like this:



You can change the 'Fixed Frame' to the imu frame 'unilidar_imu' ,so that you can view the IMU quaternion vector.

·launch.py configuration file description

The file is located at unitree_lidar_ros2\launch\launch.py

- port: The serial port number for connecting the LiDAR to the computer is /dev/ttyUSB0 by default.
- rotate_yaw_bias: The fixed offset angle in degrees that may exist during the installation of the LiDAR. This parameter is used to calibrate the orientation of the point cloud data, and its default value is 0.
- range_scale: The range scaling factor of the LiDAR, which is used to calibrate the ranging accuracy of the LiDAR. Its default value is 0.001.
- range_bias: The range offset of the LiDAR, which is used to calibrate the ranging accuracy of the LiDAR. Its default value is 0.
- range_max: The maximum range of the LiDAR, measured in meters, with a default value of 50.
- range_min: The minimum range of the LiDAR, measured in meters, with a default value of 0.
- cloud_frame: The name of the coordinate system for the point cloud data, with a default value of 'unilidar_lidar'.
- cloud_topic: The ROS topic name for the point cloud data, with a default value of 'unilidar/cloud'.

- cloud_scan_num：The number of laser ring IDs for the point cloud data, with a default value of 18.
- imu_frame：The name of the coordinate system for the IMU data, with a default value of 'unilidar_imu'.
- imu_topic：The ROS topic name for the IMU data, with a default value of 'unilidar/imu'.

·Version History

v1.0.0(2023.05.30)

- Support 'Unilidar_sdk' : v1.0.1

This manual will not be notified separately if updated.

You can check the latest version of the "User Manual" on the official website of Unitree.

https://www.unitree.com/en/download