005 大末建設用データ解析

Data analysis for Osue Construction

overview

Environment

Fast_LIO

The name of the workspace is arbitrary, but `~/catkin_ws` it should be written as . The editing results below `FAST_LIO.7z` are summarized in , so please unzip and use them.
As a premise, `livox_ros_driver2` it is assumed that has already been built. For the environment construction of , please refer to the environment construction of ROS1Noetic for Osue Construction, or unzip and use the `livox_ros_driver2` summarized . If you want to unzip and , please proceed from the Build Fast_LIO item. `livox_ros_driver2.7z`
`FAST_LIO.7z` `livox_ros_driver2.7z` `~/catkin_ws/src`

Clone the Fast_LIO source

```
cd ~/catkin_ws/src
git clone https://github.com/hku-mars/FAST_LIO.git
cd FAST_LIO
git submodule update --init
```

Editing the Fast_LIO source code

Fast_LIO `livox_ros_driver` is set to by default, so `livox_ros_driver2` change it to .
The following five files need to be changed. All the changes are as follows `livox_ros_driver` : `livox_ros_driver2`

1. `CMakeLists.txt` Line 54 of
2. `package.xml` Lines 28 and 39 of
3. `src/preprocess.h` Lines 4, 90, and 106 of
4. `src/preprocess.cpp` Lines 44 and 88 of
5. `src/laserMapping.cpp` Line 59,301 of

Creating a launch file

Create a launch that runs fastlio from a bag file and saves the results to a bag file. Note that since it is set `rosbag play` to `required="true"`, the entire program will terminate when playback ends. The benefit of creating this is that it is a convenient way to run fastlio on multiple rosbag data in succession from a shell script. Of course, this technique can be used for things other than fastlio as well.

`mapping_mid360.launch` Copy fastlio's , `bag_mapping_mid360.launch` create a , and edit it in vscode.

```
cd ~/catkin_ws/src/FAST_LIO/launch/
cp mapping_mid360.launch bag_mapping_mid360.launch
code bag_mapping_mid360.launch
```

bag_mapping_mid360.launch is written as follows.

```
<launch>
<!-- Launch file for Livox MID360 LiDAR -->

        <arg name="rviz" default="true" />

        <rosparam command="load" file="$(find fast_lio)/config/mid360.yaml" />

        <param name="feature_extract_enable" type="bool" value="0"/>
        <param name="point_filter_num" type="int" value="3"/>
        <param name="max_iteration" type="int" value="3" />
        <param name="filter_size_surf" type="double" value="0.5" />
        <param name="filter_size_map" type="double" value="0.5" />
        <param name="cube_side_length" type="double" value="1000" />
        <param name="runtime_pos_log_enable" type="bool" value="0" />
    <node pkg="fast_lio" type="fastlio_mapping" name="laserMapping" output="screen" />

        <group if="$(arg rviz)">
        <node launch-prefix="nice" pkg="rviz" type="rviz" name="rviz" args="-d $(find fast_lio)/rviz_cfg/loam_livox.rviz"
/>
        </group>

        <arg name="src_bag" />
        <arg name="dst_bag" />
        <node pkg="rosbag" type="record" name="record" output="screen" args="-a -O $(arg dst_bag)" />
        <node pkg="rosbag" type="play" name="play" output="screen" args="$(arg src_bag)" required="true" />

</launch>
```

Build Fast_LIO

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

Running the launch file

`src_bag` `dst_bag` Enter the path of the bag file to be played in and the path of the bag file to be saved in, then execute. Please change the file path of bag to suit your PC environment. Please make sure that
the bag file `/livox/lidar[livox_ros_driver2/CustomMsg]` to be `/livox/imu[sensor_msgs/Imu]` includes and.

```
roslaunch fast_lio bag_mapping_mid360.launch src_bag:="/home/user/catkin_ws/rosbag/2024-06-06-07-17-15.bag"
dst_bag:="/home/user/catkin_ws/rosbag/fastlio_2024-06-06-07-17-15.bag"
```

troubleshooting

If you are unable to tab complete or are told that the file does not exist, please try the following and then try again.

```
rospack profile
```

Convert the Fast_LIO execution result to an NPZ file

Contents of the Fast_LIO execution result bag file

`fastlio_2024-06-06-07-17-15.bag` The contents are as follows. The ones used here are . `/cloud_registered` In the example below `/cloud_registered` , there are 38 pieces of data. are the point clouds output based on the coordinates at the start of Fast_LIO. If you want the point cloud in LiDAR coordinates, `/cloud_registered_body` use . `/Odometry` are the position and orientation at each time.

```
rosbag info fastlio_2024-06-06-07-17-15.bag
path:        fastlio_2024-06-06-07-17-15.bag
version:     2.0
duration:    5.4s
start:       Jun 07 2024 20:06:53.20 (1717758413.20)
end:         Jun 07 2024 20:06:58.59 (1717758418.59)
size:        60.1 MB
messages:    2256
compression: none [58/58 chunks]
types:       livox_ros_driver2/CustomMsg [e4d6829bdfe657cb6c21a746c86b21a6]
             nav_msgs/Odometry           [cd5e73d190d741a2f92e81eda573aca7]
             rosgraph_msgs/Clock         [a9c97c1d230cfc112e270351a944ee47]
             rosgraph_msgs/Log           [acffd30cd6b6de30f120938c17c593fb]
             sensor_msgs/Imu             [6a62c6daae103f4ff57a132d6f95cec2]
             sensor_msgs/PointCloud2     [1158d486dd51d683ce2f1be655c3c181]
             tf2_msgs/TFMessage          [94810edda583a504dfda3829e70d7eec]
topics:      /Odometry                 38 msgs    : nav_msgs/Odometry
             /clock                   467 msgs    : rosgraph_msgs/Clock
             /cloud_registered         38 msgs    : sensor_msgs/PointCloud2
             /cloud_registered_body    38 msgs    : sensor_msgs/PointCloud2
             /imu_raw                 762 msgs    : sensor_msgs/Imu
             /livox/imu               762 msgs    : sensor_msgs/Imu
             /livox/lidar              38 msgs    : livox_ros_driver2/CustomMsg
             /points_raw               38 msgs    : sensor_msgs/PointCloud2
             /rosout                   19 msgs    : rosgraph_msgs/Log          (2 connections)
             /rosout_agg               17 msgs    : rosgraph_msgs/Log
             /tf                       38 msgs    : tf2_msgs/TFMessage
             /tf_static                 1 msg     : tf2_msgs/TFMessage
```

`/cloud_registered` Convert to NPZ file

In NumPy, an array ndarray can be saved as a binary file (npy, npz) in NumPy's own format. It can be written and read (output/input) while retaining information such as data type dtype and shape. Here, we `sensor_msgs/PointCloud2` will `npz` convert to a file.

For conversion, `pointcloud2_to_npz.py` use . For how to do this, `pointcloud2_to_npz_README.md` please refer to the following or read .

Check the list of topics in the bag file

```
python3 pointcloud2_to_npz.py fastlio_2024-06-06-07-17-15.bag -l
```

```
python3 pointcloud2_to_npz.py fastlio_2024-06-06-07-17-15.bag -l
Topics in the bag file:
  /Odometry
  /clock
  /cloud_registered
  /cloud_registered_body
  /imu_raw
  /livox/imu
  /livox/lidar
  /points_raw
  /rosout
  /rosout_agg
  /tf
  /tf_static
```

`/cloud_registered` Convert each scan into an NPZ file

`-t` Optionally specify a topic. `-o` Optionally specify a destination folder. `-o` If you do not specify a destination folder, the file `output` will be saved in the folder named.

```
python3 pointcloud2_to_npz.py fastlio_2024-06-06-07-17-15.bag -t /cloud_registered -o test/
```

`/cloud_registered` Convert all of them into one NPZ file

`-s` Optionally, convert all point clouds into one NPZ file. Note that the topic you specify must be a point cloud that is output based on the coordinates at the start of Fast_LIO (coordinate-converted point cloud).

```
python3 pointcloud2_to_npz.py fastlio_2024-06-06-07-17-15.bag -t /cloud_registered -o test1.npz -s
```

Visualizing NPZ files with Open3D

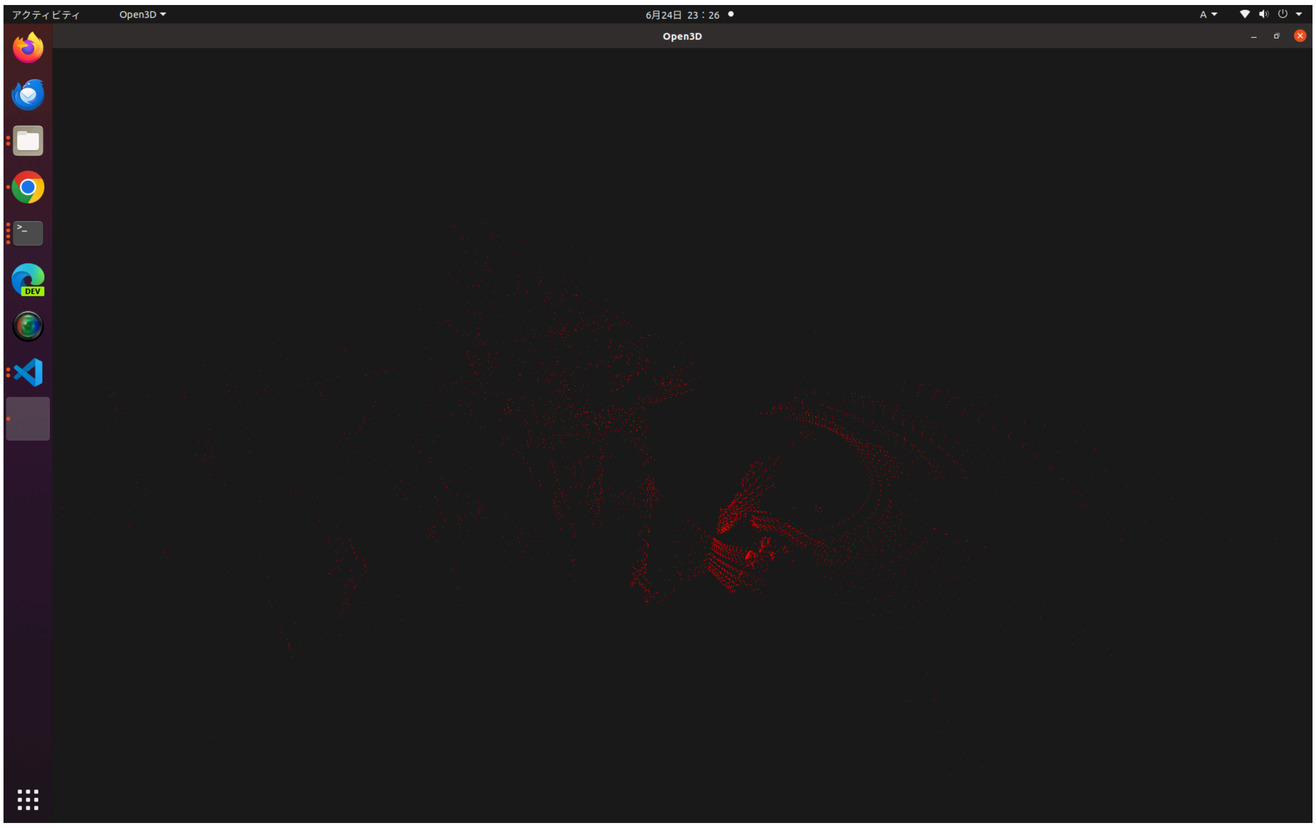
For 3D visualization, ploty, which is used inside Open3D, is used to visualize large-scale point clouds. However, since rendering becomes difficult for point clouds with more than 1 million points, when visualizing data compiled into a single NPZ file, some downsampling is performed (please check the code or log output for details).

For visualization, `npz_pointcloud_viewer_open3d.py` use . For how to run it, `npz_pointcloud_viewer_open3d_README.md` please refer to the following or read .

To operate the screen, left click to rotate, use the mouse wheel to zoom in and out, press the mouse wheel or Ctrl + left click to move in translation, and use `q` the key to exit.

Visualize a single scan of an NPZ file

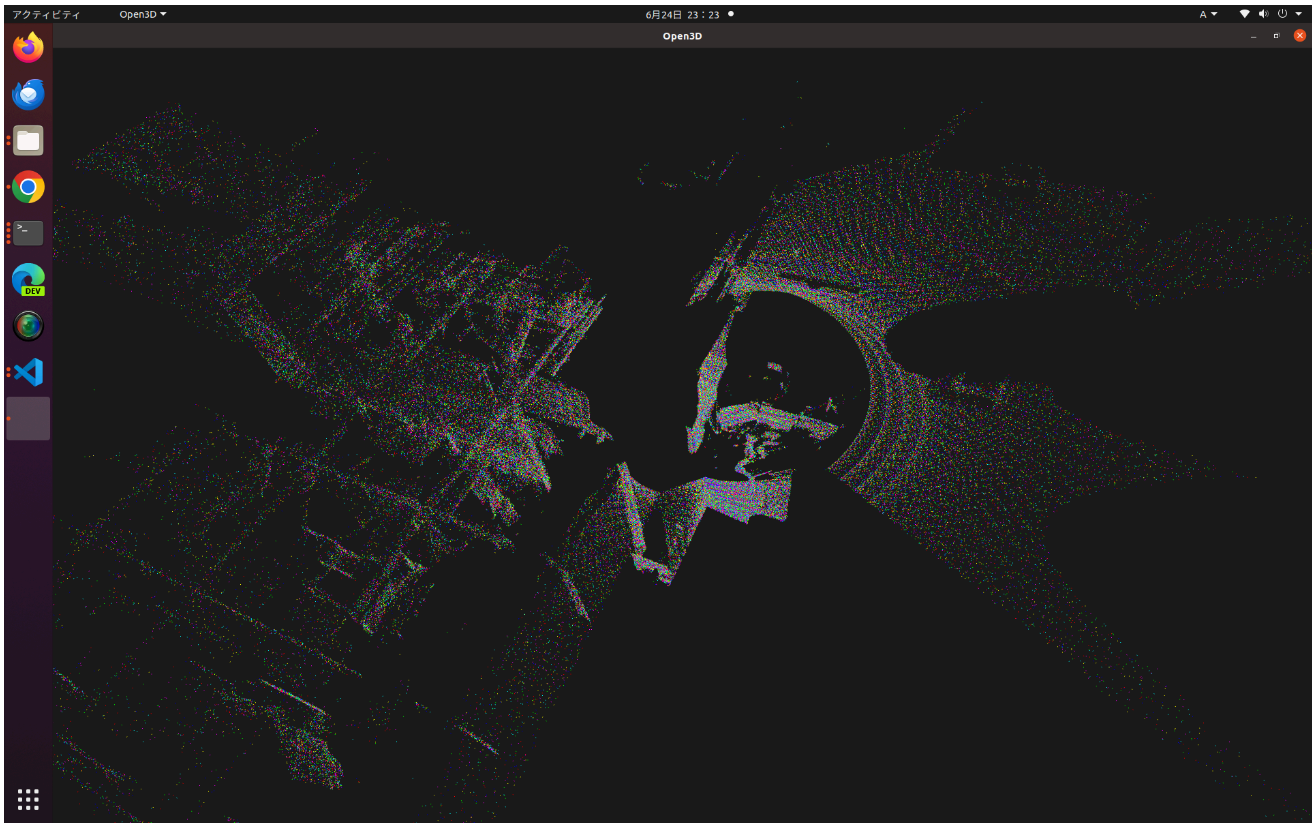
```
python3 npz_pointcloud_viewer_open3d.py test/pointcloud_1717758414467514023.npz
```



Visualize all NPZ files for each scan together

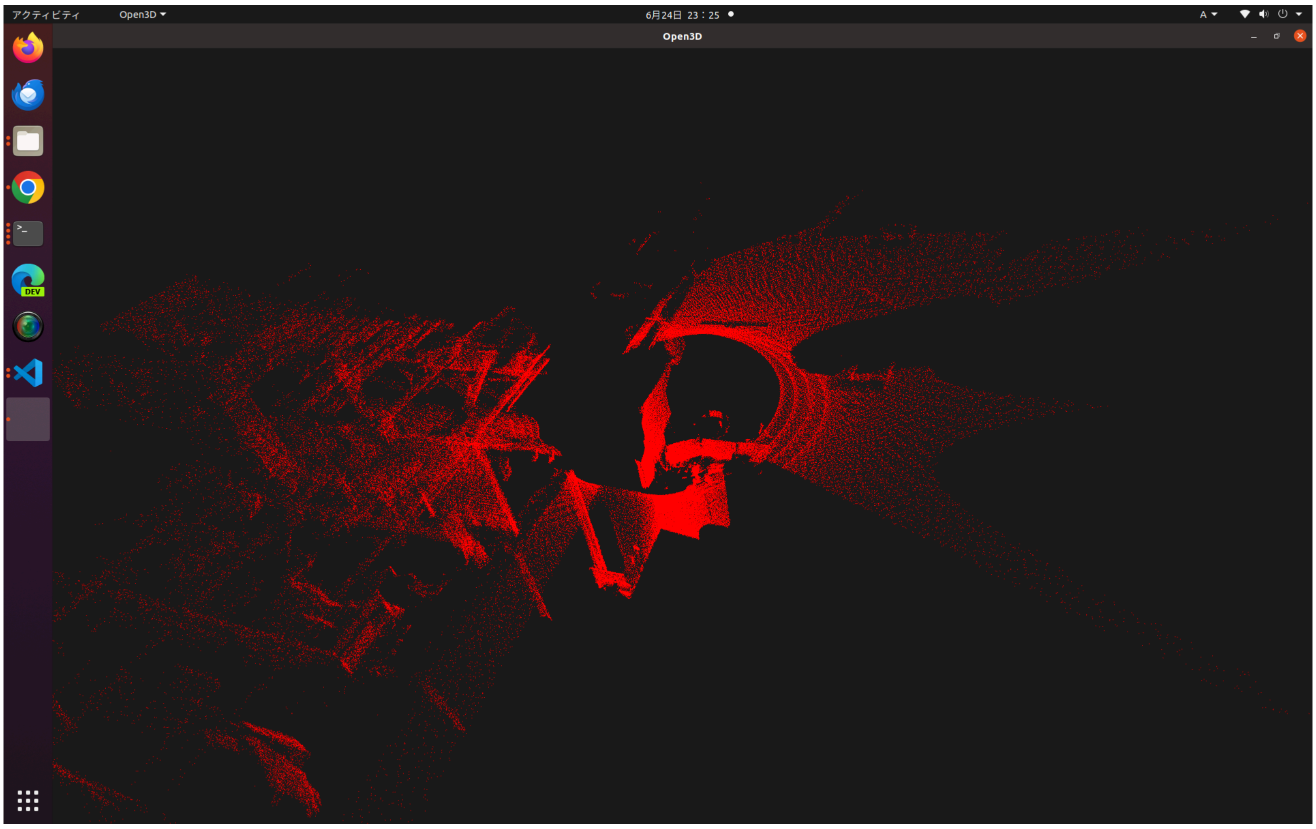
`test` Visualize all NPZ files in a folder at once.

```
python3 npz_pointcloud_viewer_open3d.py test/
```



Visualization of what is compiled into one NPZ file

```
python3 npz_pointcloud_viewer_open3d.py test1.npz
```



Execution result