6.ORB_SLAM2 PCL mapping

This routine is implemented on Orin NX and cannot be implemented on a virtual machine. We are only explaining how the process is implemented.
If you want to implement this function on your own motherboard, you need to compile the entire feature pack and connect relevant peripherals.

pcl webiste:: http://pointclouds.org

pcl_ros wiki: https://wiki.ros.org/pcl_ros

pcl_ros github: https://github.com/ros-perception/perception_pcl

The operating environment and reference configurations for software and hardware are as follows:

- Reference model: ROSMASTER X3
- Robot hardware configuration: Arm series main control, Silan A1 LiDAR, Dabai_ Dcw2 depth camera
- Robot system: Ubuntu 20.04
- PC virtual machine: Ubuntu (20.04)+ROS2 (Foxy)
- Usage scenario: Use on a relatively clean 2D plane

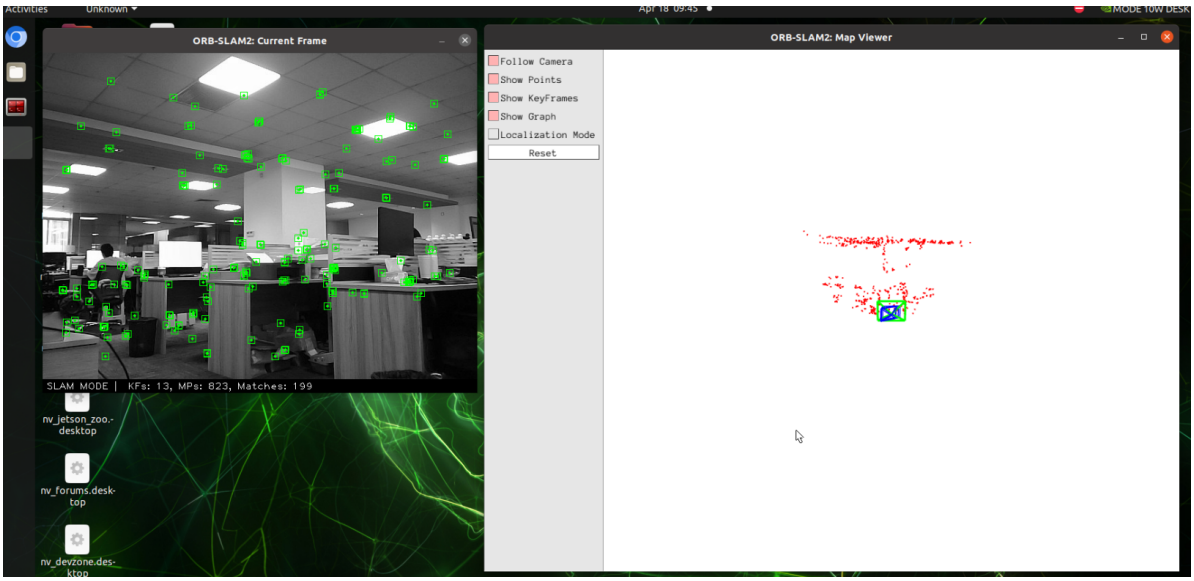
1. Usage

1)Start camera

```
ros2 launch orbbec_camera dabai_dcw2.launch.py
```

2)Launch orbslam to release camera pose, color and depth maps. Depending on the performance of different controllers, the waiting time here is approximately within 10 seconds.

```
ros2 launch yahboomcar_slam orbslam_base_launch.py
```

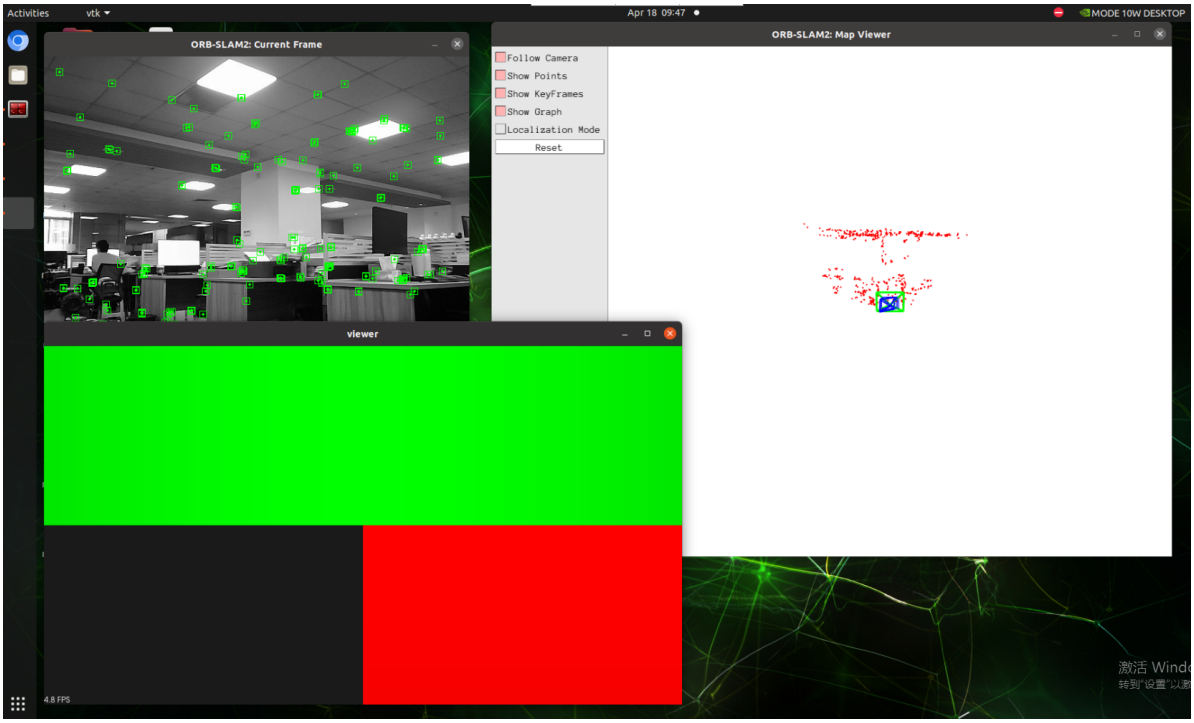


3)Start point cloud mapping

```
ros2 launch yahboomcar_slam orbslam_pcl_map_launch.py
```

After opening, a 【 viewer 】 window will pop up, and slowly move the camera.

When an image appears, as shown in the following figure:



Need to scale and rotate the coordinate system, as shown below.

Sliding roller: retraction and retraction

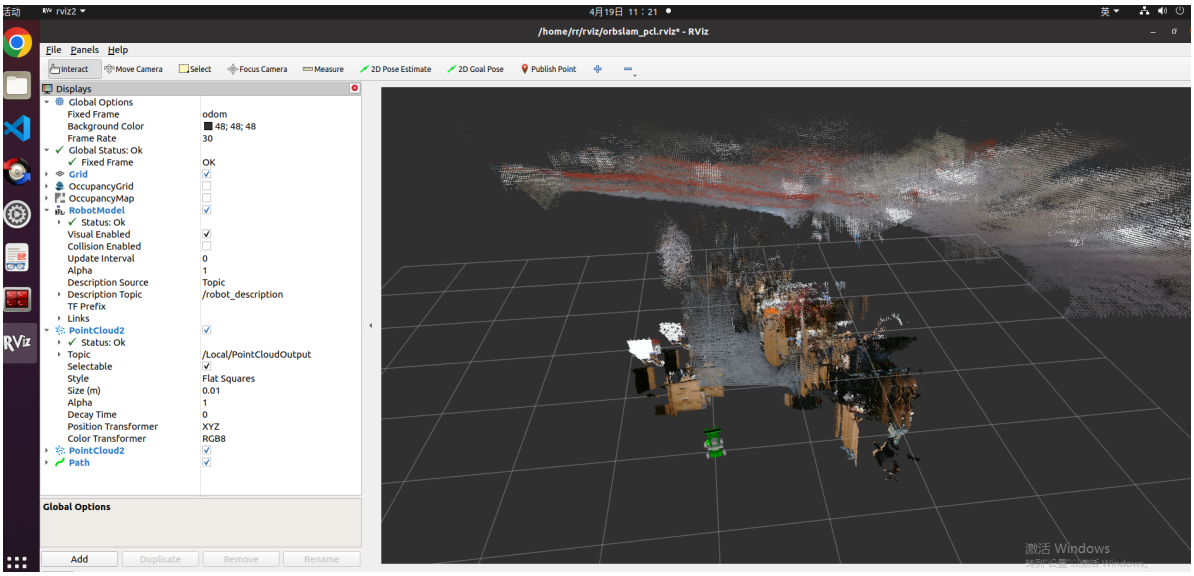
Hold down the scroll wheel: pan

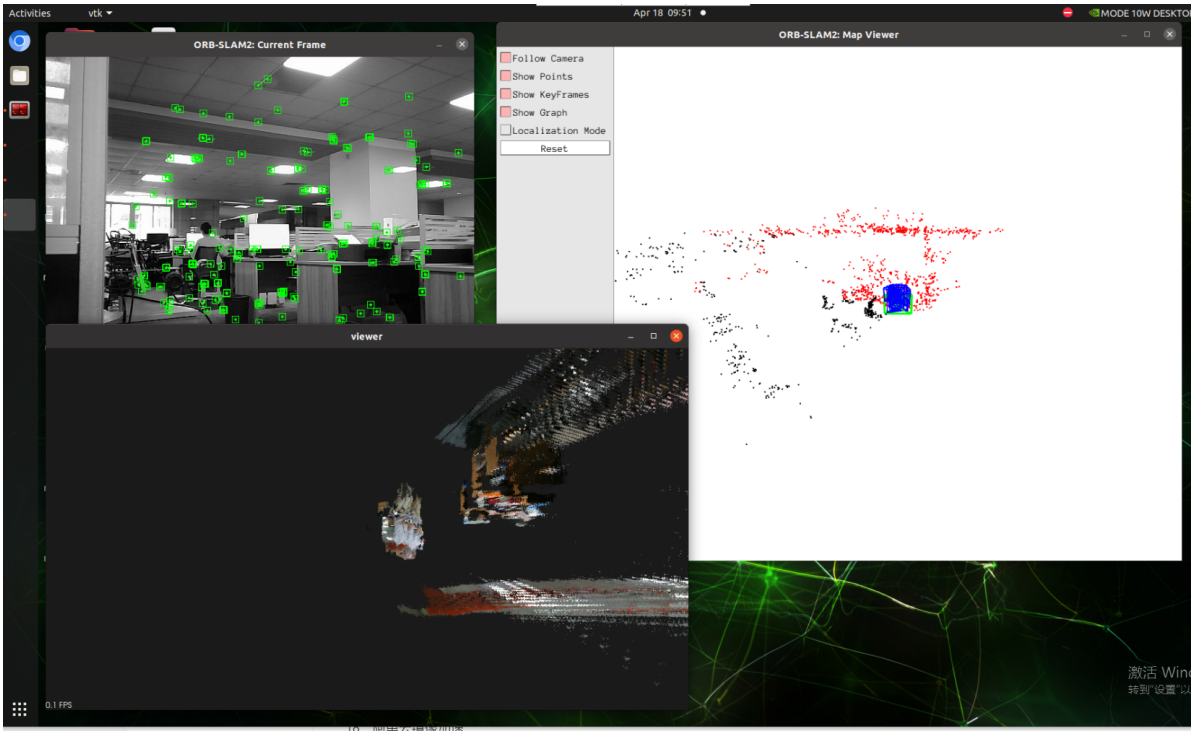
Left click on mouse: Rotate

Right click on mouse: Zoom in/out

You can simultaneously open rviz to view real-time point cloud mapping effects. It is recommended to use the virtual machine to open rviz to view.

```
ros2 launch yahboomcar_slam display_pcl_launch.py
```





4)Slowly move the camera to create the image as shown below. After creating, press Ctrl+c to close and save the PCD point cloud file resultPointCloudFile.pcd.

The path is as follows:

```
~/orbbec_ws/src/yahboomcar_slam/pcl/resultPointCloudFile.pcd
```

5)View the resultPointCloudFile.pcd file

Method-1: Use PCL_ Viewer tool

Input following command to install PCL_ Viewer

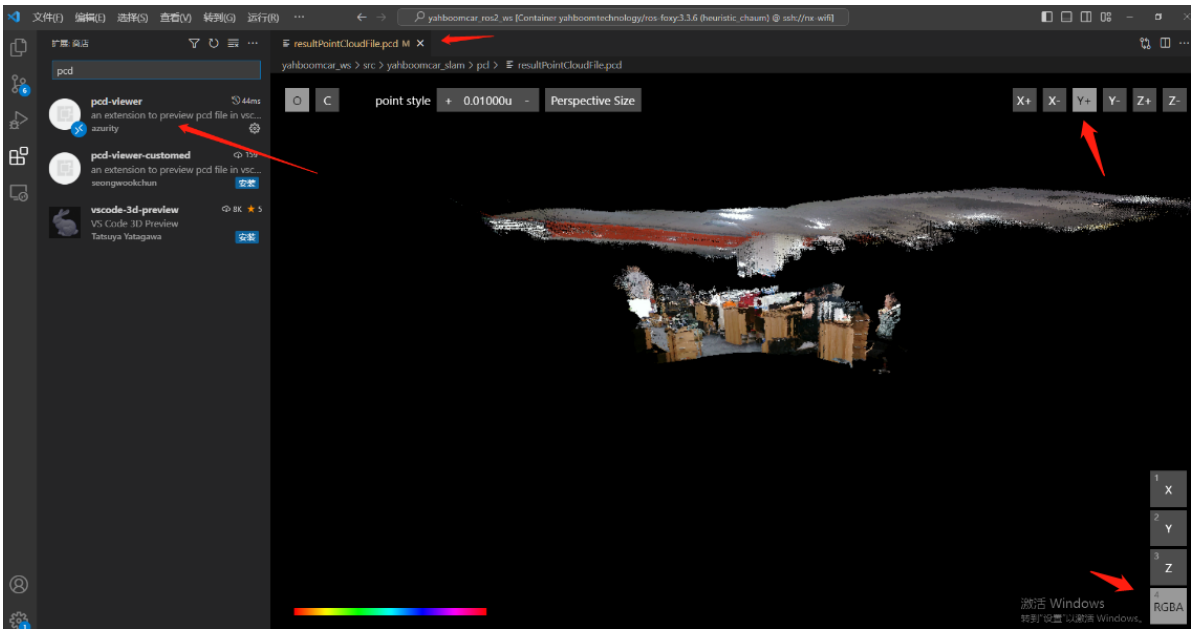
```
sudo apt-get install pcl-tools
```

Enter the directory where the PCD component is located and enter the following command to view the PCD point cloud map

```
pcl_viewer resultPointCloudFile.pcd
```

Method-2: Use vscode's PCD viewer plugin [recommended]

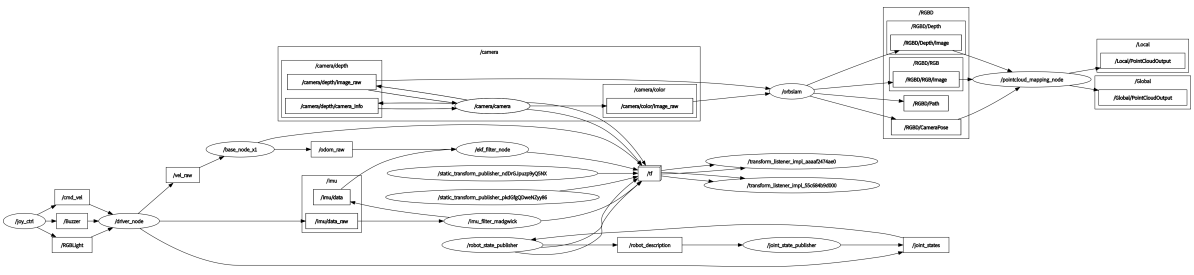
First install the pcd viewer plugin in vscode, and then open resultPointCloudFile.pcd to view the pcd point cloud image. Directly selecting the Y+direction and RGBA mode at the bottom here can quickly rotate the image to the main view.



2. Node analysis

1.Display calculation chart

```
rqt_graph
```



2. pointcloud_mapping node details

```
rr@rr-pc:~/rviz$ ros2 node info /pointcloud_mapping_node
/pointcloud_mapping_node
  Subscribers:
    /RGBD/CameraPose: geometry_msgs/msg/PoseStamped
    /RGBD/Depth/Image: sensor_msgs/msg/Image
    /RGBD/RGB/Image: sensor_msgs/msg/Image
    /parameter_events: rcl_interfaces/msg/ParameterEvent
  Publishers:
    /Global/PointCloudOutput: sensor_msgs/msg/PointCloud2
    /Local/PointCloudOutput: sensor_msgs/msg/PointCloud2
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /rosout: rcl_interfaces/msg/Log
  Service Servers:
    /pointcloud_mapping_node/describe_parameters: rcl_interfaces/srv/DescribeParameters
    /pointcloud_mapping_node/get_parameter_types: rcl_interfaces/srv/GetParameterTypes
    /pointcloud_mapping_node/get_parameters: rcl_interfaces/srv/GetParameters
    /pointcloud_mapping_node/list_parameters: rcl_interfaces/srv/ListParameters
    /pointcloud_mapping_node/set_parameters: rcl_interfaces/srv/SetParameters
    /pointcloud_mapping_node/set_parameters_atomically: rcl_interfaces/srv/SetParametersAtomically
  Service Clients:

  Action Servers:

  Action Clients:
```

3. TF transformation

```
ros2 run tf2_tools view_frames.py
```