7. ORB_SLAM2 Octomap mapping

This routine is implemented on Orin NX and cannot be implemented on a virtual machine. It just explains how the process is implemented. If you want to implement this function on your own motherboard, you need to compile the entire function package and connect related peripherals.

octomap official website: http://octomap.github.io/

octomap source code: https://github.com/OctoMap/octomap

octomap wiki: http://wiki.ros.org/octomap

octomap_server: http://wiki.ros.org/octomap_server

The operating environment and software and hardware reference configuration are as follows:

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

7.1. Introduction

Octomap is a three-dimensional map creation tool based on octrees, which can display complete 3D graphics including barrier-free areas and unmapped areas, and sensor data based on occupancy rasters can be fused and updated in multiple measurements; The map can provide multiple resolutions, the data can be compressed, and the storage is compact. In fact, the code of octomap mainly contains two modules: the three-dimensional map creation tool octomap and the visualization tool octovis.
Compared with point cloud, it can save a lot of space. The map created by octomap probably looks like this: (different resolutions from left to right)

7.2, camera-based octomap mapping

The terminal executes the following launch file:

1. Start the camera

```
ros2 launch orbbec_camera gemini2.launch.py
```

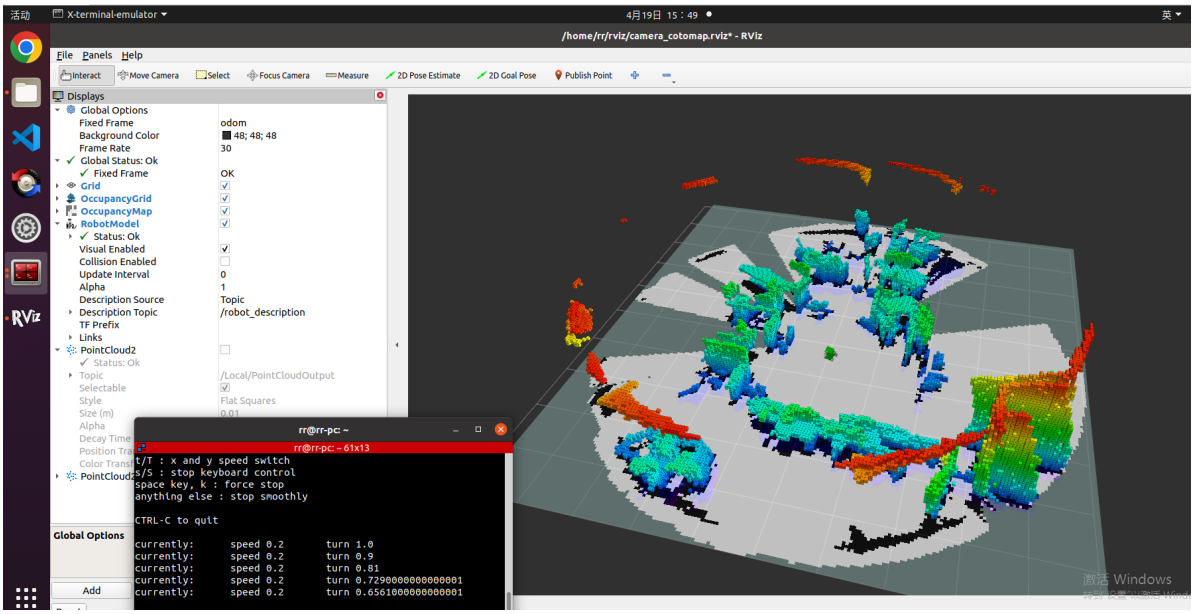
2. Start octomap_server to build maps

```
ros2 launch yahboomcar_slam camera_octomap_launch.py
```

3. Start rviz in docker or on the virtual machine [recommended]:

```
ros2 launch yahboomcar_slam display_octomap_launch.py
```

4. Use the remote control or keyboard to control the nodes to slowly move the robot to build the map. Loss of key frames may cause the map to fail.



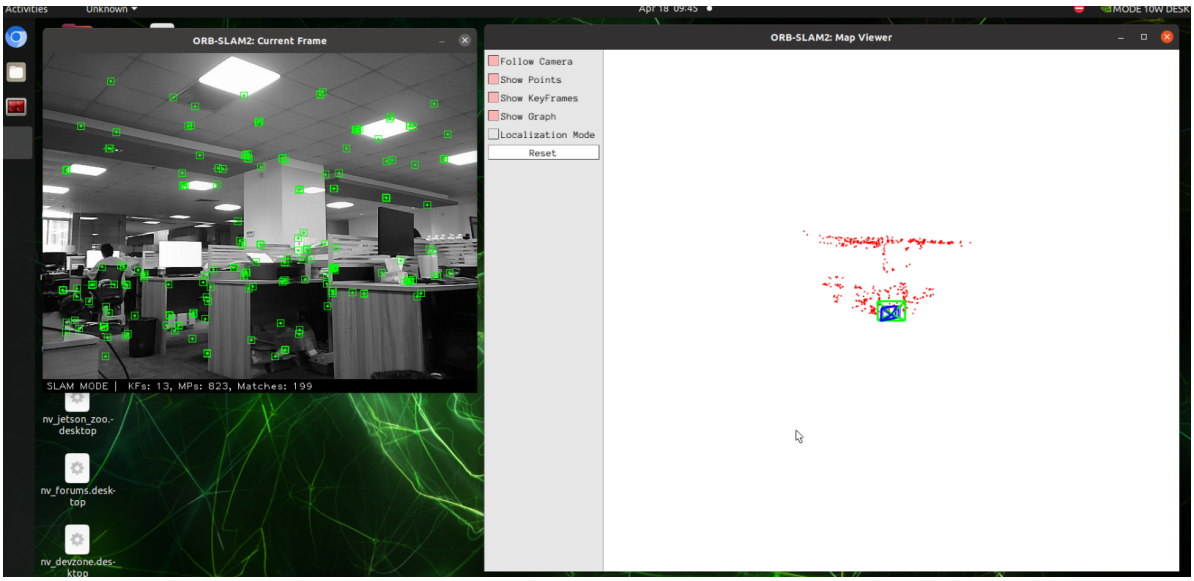
7.3. Octomap mapping based on orbslam and pointcloud_mapping

1. Start the camera

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

2. Start orbslam to publish camera poses, color images, and depth images. Depending on the performance of different main controllers, the waiting time here is about 10 seconds.

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



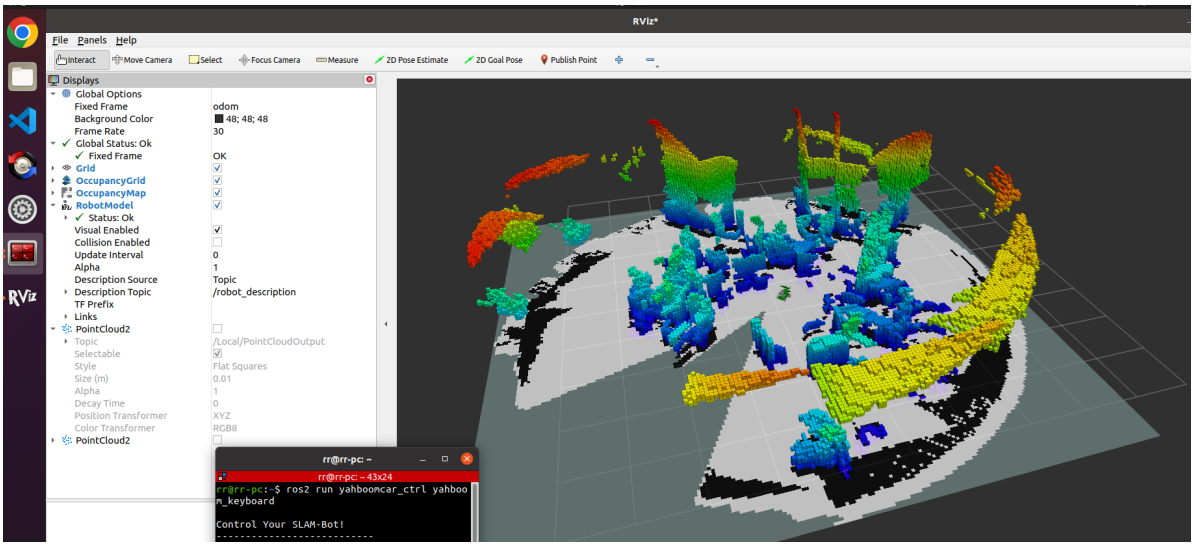
3. Start octomap_server for mapping

```
ros2 launch yahboomcar_slam orbslam_pcl_octomap_launch.py
```

4. Start rviz:

```
ros2 launch yahboomcar_slam display_octomap_launch.py
```

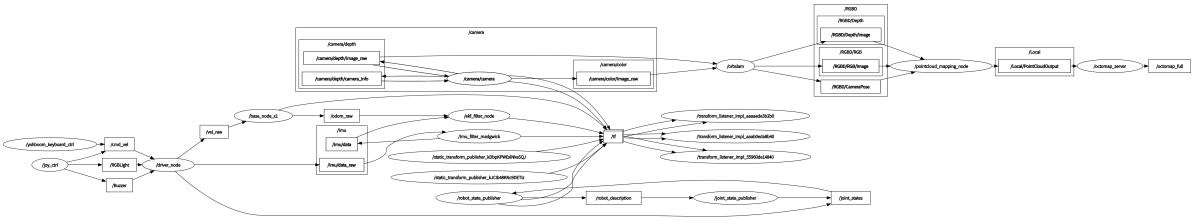
5. Use the remote control or keyboard to control the nodes to slowly move the robot to build the map. Loss of key frames may cause the map to fail.



7.4. Node analysis

7.3.1. Display calculation graph

```
rqt_graph
```



7.3.2. Details of each node

```
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:
```

```
rr@rr-pc:~/rviz$ ros2 node info /octomap_server
/octomap_server
  Subscribers:
    /Local/PointCloudOutput: sensor_msgs/msg/PointCloud2
    /parameter_events: rcl_interfaces/msg/ParameterEvent
  Publishers:
    /free_cells_vis_array: visualization_msgs/msg/MarkerArray
    /occupied_cells_vis_array: visualization_msgs/msg/MarkerArray
    /octomap_binary: octomap_msgs/msg/Octomap
    /octomap_full: octomap_msgs/msg/Octomap
    /octomap_point_cloud_centers: sensor_msgs/msg/PointCloud2
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /projected_map: nav_msgs/msg/OccupancyGrid
    /rosout: rcl_interfaces/msg/Log
  Service Servers:
    /octomap_binary: octomap_msgs/srv/GetOctomap
    /octomap_full: octomap_msgs/srv/GetOctomap
    /octomap_server/clear_bbox: octomap_msgs/srv/BoundingBoxQuery
    /octomap_server/describe_parameters: rcl_interfaces/srv/DescribeParameters
    /octomap_server/get_parameter_types: rcl_interfaces/srv/GetParameterTypes
    /octomap_server/get_parameters: rcl_interfaces/srv/GetParameters
    /octomap_server/list_parameters: rcl_interfaces/srv/ListParameters
    /octomap_server/reset: std_srvs/srv/Empty
    /octomap_server/set_parameters: rcl_interfaces/srv/SetParameters
    /octomap_server/set_parameters_atomically: rcl_interfaces/srv/SetParametersAtomically
  Service Clients:

  Action Servers:

  Action Clients:
```

7.3.3, TF transformation

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