

Navigation2 multi-point navigation avoid

1. Program Functionality

Note: To learn this course, you must have studied [【8.Navigation2 single-point navigation avoid】](#) and have a basic understanding of Navigation2 navigation.

After running the program, a map will load in rviz. In the rviz interface, use the nav2 plugin for multi-point navigation.

2. Running Examples

2.1 Pre-use Instructions

This lesson uses the Raspberry Pi as an example.

For Raspberry Pi and Jetson-Nano boards, you need to open a terminal on the host computer and enter the command to enter the Docker container. Once inside the Docker container, enter the commands mentioned in this lesson in the terminal. For instructions on entering the Docker container from the host computer, refer to [\[01. Robot Configuration and Operation Guide\] -- \[4.Enter Docker \(For JETSON Nano and RPi 5\)\]](#).

For RDKX5 and Orin boards, simply open a terminal and enter the commands mentioned in this lesson.

2.2 Multi-Point Navigation

Note:

- **For Raspberry Pi and Jetson-Nano boards, you need to enter the Docker container first. For RDKX5 and Orin controllers, this is not necessary.**

Enter the Docker container (see [Docker course] --- [4. Docker Startup Script] for steps).

All the following commands must be executed within the same Docker container (**see [Docker course] --- [3. Docker Submission and Multi-Terminal Access] for steps**).

- This section requires at least one existing map. Refer to [\[5.Gmapping-SLAM mapping\]](#), [\[6.Cartographer-SLAM mapping\]](#), [\[7.slam_toolbox mapping\]](#) or any of the SLAM Mapping courses.

Commands for launching the underlying sensors on the robot vehicle terminal:

```
ros2 launch yahboomcar_nav laser_bringup_launch.py
```

```

root@raspberrypi: ~
File Edit Tabs Help
[ydlidar_ros2_driver_node-9] [YDLIDAR] SDK Version: 1.2.3
[static_transform_publisher-11] [INFO] [1755056541.074142877] [static_transform_publisher_hkPl67SIG9QIUxf7]: Spinning until stoppe
d - publishing transform
[static_transform_publisher-11] translation: ('0.000000', '0.000000', '0.000000')
[static_transform_publisher-11] rotation: ('0.000000', '0.000000', '0.000000', '1.000000')
[static_transform_publisher-11] from 'ascamera_hp60c_camera_link_0' to 'ascamera_hp60c_color_0'
[static_transform_publisher-10] [INFO] [1755056541.084073695] [static_transform_publisher_IR46JMCUTLR00sih]: Spinning until stoppe
d - publishing transform
[static_transform_publisher-10] translation: ('0.000608', '0.000000', '0.149120')
[static_transform_publisher-10] rotation: ('0.000046', '0.000046', '-1.000000', '0.000000')
[static_transform_publisher-10] from 'base_link' to 'laser'
[ydlidar_ros2_driver_node-9] [YDLIDAR] Lidar successfully connected [/dev/rplidar:230400]
[ydlidar_ros2_driver_node-9] [YDLIDAR] Lidar running correctly! The health status: good
[joint_state_publisher-1] [INFO] [1755056541.807820230] [joint_state_publisher]: Waiting for robot_description to be published on
the robot_description topic...
[ydlidar_ros2_driver_node-9] [YDLIDAR] Baseplate device info
[ydlidar_ros2_driver_node-9] Firmware version: 1.2
[ydlidar_ros2_driver_node-9] Hardware version: 1
[ydlidar_ros2_driver_node-9] Model: Tmini Plus
[ydlidar_ros2_driver_node-9] Serial: 2025021500090139
[ydlidar_ros2_driver_node-9] [YDLIDAR] Current scan frequency: 10.00Hz
[ydlidar_ros2_driver_node-9] [YDLIDAR] Lidar init success, Elapsed time 1031 ms
[imu_filter_madgwick_node-5] [INFO] [1755056542.843712101] [imu_filter_madgwick]: First IMU message received.
[ydlidar_ros2_driver_node-9] [YDLIDAR] Create thread 0xA68268E0
[ydlidar_ros2_driver_node-9] Successed to start scan mode, Elapsed time 2097 ms
[ydlidar_ros2_driver_node-9] [YDLIDAR] Fixed Size: 404
[ydlidar_ros2_driver_node-9] [YDLIDAR] Sample Rate: 4.00K
[ydlidar_ros2_driver_node-9] [YDLIDAR] Successed to check the lidar, Elapsed time 0 ms
[ydlidar_ros2_driver_node-9] [2025-08-13 11:42:24][info] [YDLIDAR] Now lidar is scanning...

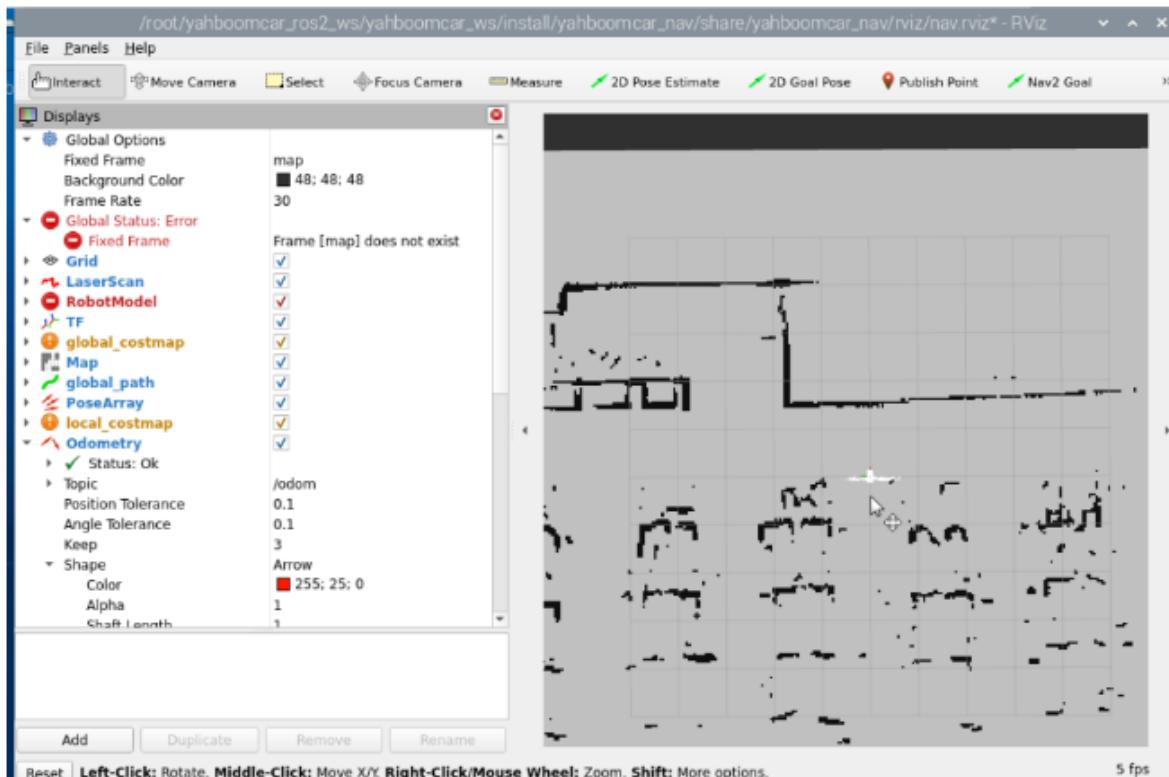
```

At this point, the map is not yet loaded because the navigation program has not started. Next, run the navigation node by entering (**you need to be in the same Docker terminal as above**) in the terminal.

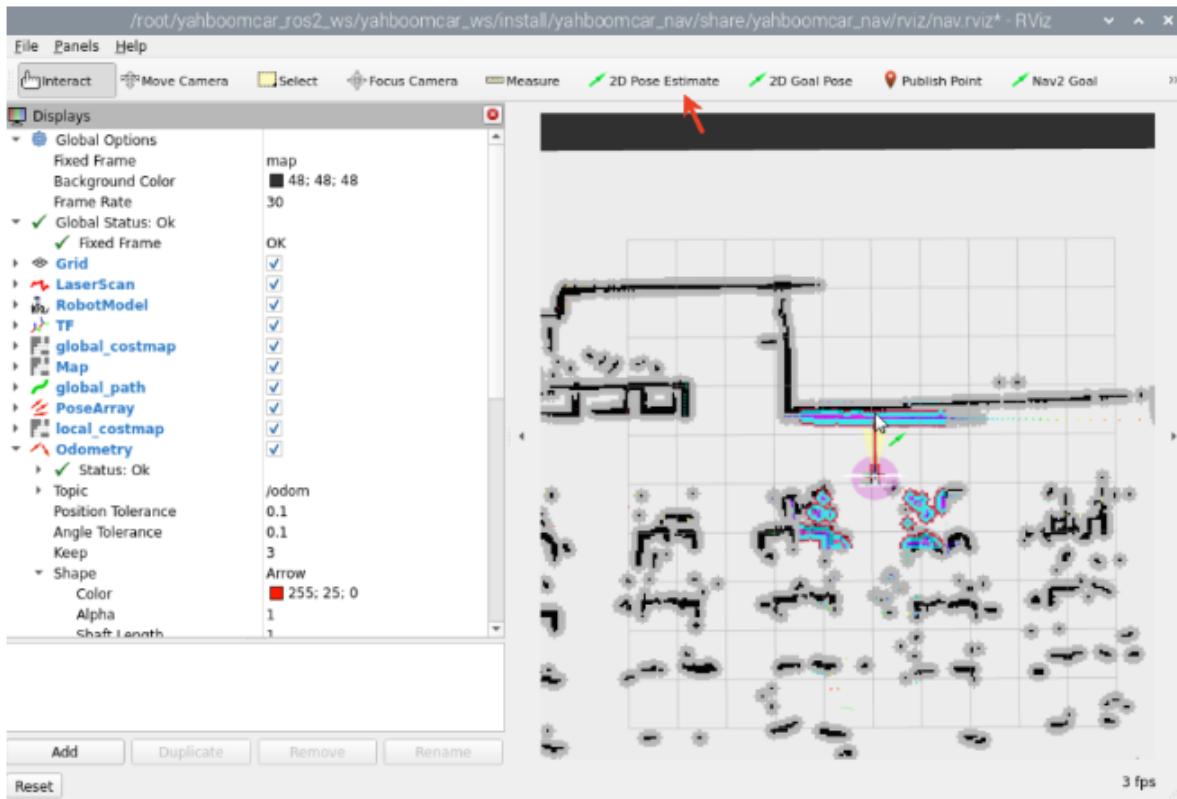
```

# Choose one of two navigation algorithms
# teb
ros2 launch yahboomcar_nav navigation_teb_launch.py
# dwa
ros2 launch yahboomcar_nav navigation_dwa_launch.py

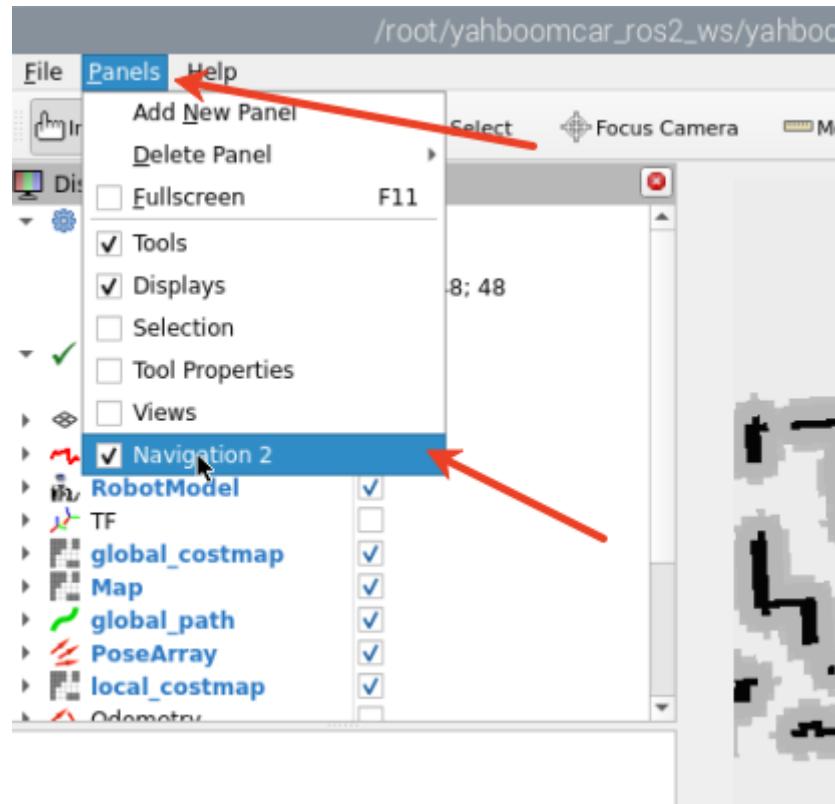
```



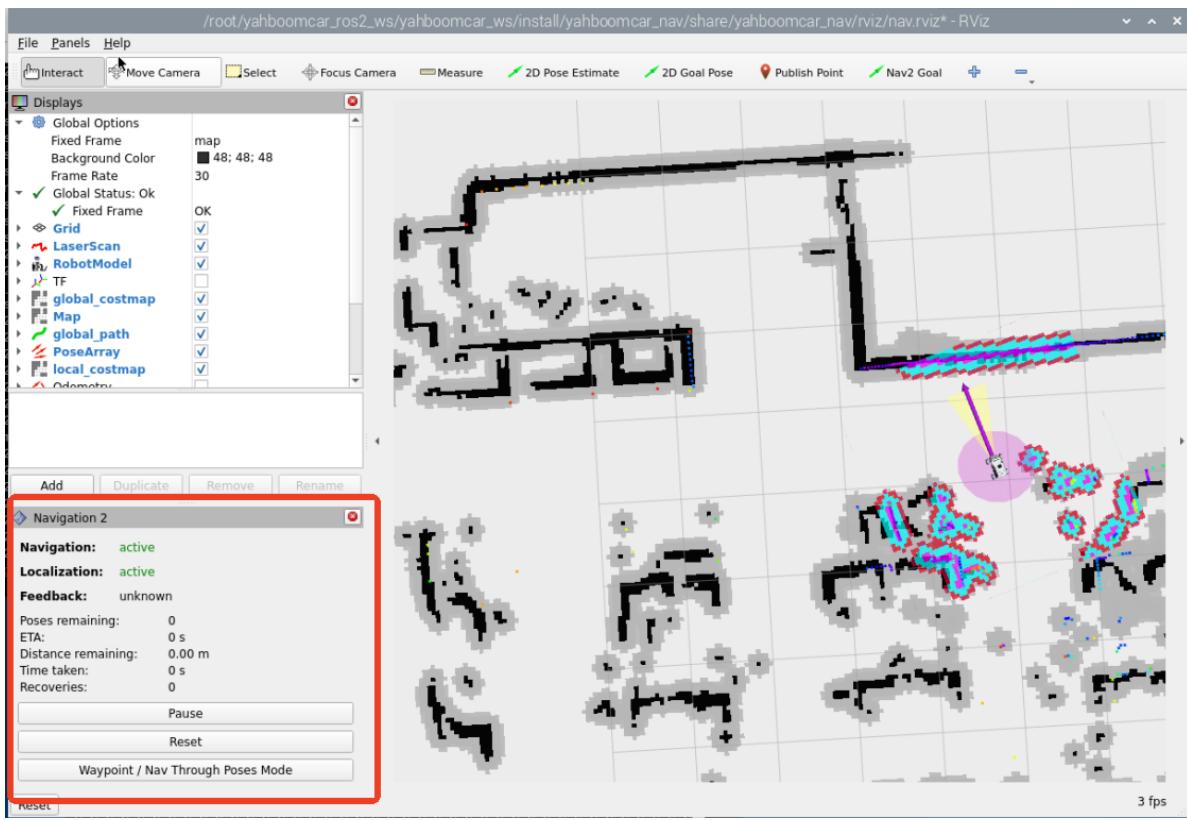
You'll see the map loaded. Click **【2D Pose Estimate】** to set the initial pose for the car. Based on the car's actual position in the environment, click and drag the mouse in rviz to move the car model to the set position. As shown in the figure below, if the lidar scan area roughly overlaps with the actual obstacle, the pose is accurate. After pose initialization is complete, the robot model and inflation area will appear in the rviz interface.



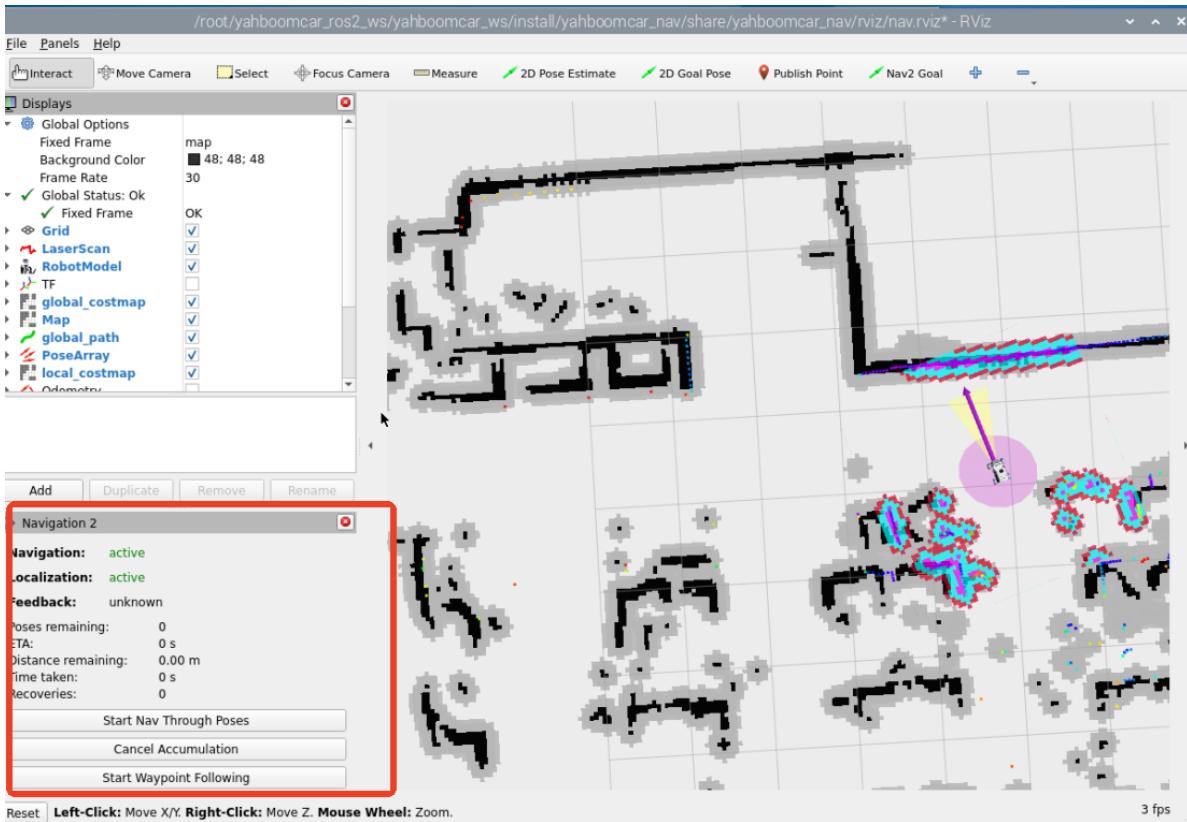
Click 【Panels】 in the upper left corner of rviz to add the Navigation2 plugin.



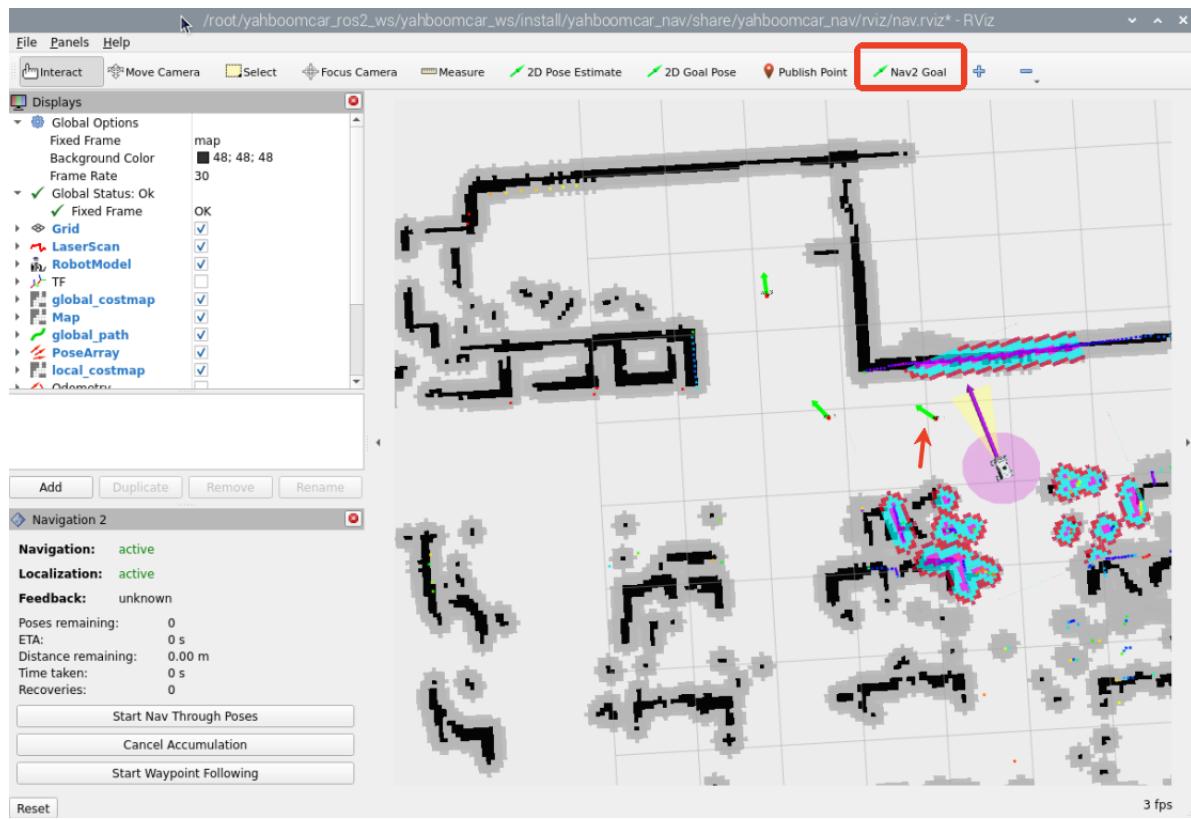
After adding, rviz will display as follows:



Then click **【Waypoint/Nav Through Poses Mode】**,



Use the **【Nav2 Goal】** button in the rviz toolbar to specify a target point, then click **【Start Waypoint Following】** to begin route navigation. The car will automatically proceed to the next point after reaching the target point, following the order of the selected points. No further action is required. After reaching the last point, the car will stop and await the next command.



If navigation to all three points is successful, the terminal will display 【Completed all 3 waypoints requested】.

```
root@raspberrypi:/ 
File Edit Tabs Help
[component_container_isolated-1] [INFO] [1755057180.508864646] [waypoint_follower]: Processing waypoint 2...
[component_container_isolated-1] [INFO] [1755057180.560254108] [waypoint_follower]: Processing waypoint 2...
[component_container_isolated-1] [INFO] [1755057180.668988191] [waypoint_follower]: Processing waypoint 2...
[component_container_isolated-1] [INFO] [1755057180.658931881] [waypoint_follower]: Processing waypoint 2...
[component_container_isolated-1] [INFO] [1755057180.709479921] [waypoint_follower]: Processing waypoint 2...
[component_container_isolated-1] [INFO] [1755057180.759018724] [waypoint_follower]: Processing waypoint 2...
[component_container_isolated-1] [INFO] [1755057180.808977506] [waypoint_follower]: Processing waypoint 2...
[component_container_isolated-1] [WARN] [1755057180.832270700] [BehaviorTreeEngine]: Behavior Tree tick rate 100.00 was exceeded!
[component_container_isolated-1] [INFO] [1755057180.866733260] [waypoint_follower]: Processing waypoint 2...
[component_container_isolated-1] [INFO] [1755057180.912875408] [waypoint_follower]: Processing waypoint 2...
[component_container_isolated-1] [INFO] [1755057180.958986834] [waypoint_follower]: Processing waypoint 2...
[component_container_isolated-1] [WARN] [1755057181.136988414] [BehaviorTreeEngine]: Behavior Tree tick rate 100.00 was exceeded!
[component_container_isolated-1] [INFO] [1755057181.564823382] [controller_server]: Passing new path to controller.
[component_container_isolated-1] [INFO] [1755057181.557384914] [controller_server]: Reached the goal!
[component_container_isolated-1] [INFO] [1755057181.597266881] [bt_navigator]: Goal succeeded
[component_container_isolated-1] [INFO] [1755057182.210766129] [waypoint_follower]: Succeeded processing waypoint 2, processing waypoint task execution
[component_container_isolated-1] [INFO] [1755057182.210841110] [waypoint_follower]: Arrived at 2'th waypoint, sleeping for 200 milliseconds
[component_container_isolated-1] [INFO] [1755057182.411099091] [waypoint_follower]: Task execution at waypoint 2 succeeded
[component_container_isolated-1] [INFO] [1755057182.411332312] [waypoint_follower]: Handled task execution on waypoint 2, moving to next.
[component_container_isolated-1] [INFO] [1755057182.411353127] [waypoint_follower]: Completed all 3 waypoints requested.
```

3. Principle Analysis

3.1 Waypoint Data

When the user turns on **[Waypoint/Nav Through PoseMode]** and enters multi-point navigation mode, the user's marked point information will be published to the **/waypoints** topic (the rviz waypoint navigation plugin adds additional waypoints between target waypoints to smooth the path). We can view this **waypoint data** through the RQT interface.

Terminal startup command:

```
ros2 run rqt_topic rqt_topic
```

In the rqt interface, we can see the topic **/waypoints**. After checking it, we can observe the data on the topic (you need to check the topic first, then publish the waypoints in the rviz interface). The waypoints we manually mark in rviz will be published to this topic.

Topic	Type	Bandwidth	Hz	Value
/initialpose	geometry_msgs/msg/PoseWithCovarianceStamped			not monitored
/intel_realsense_r200_depth/points	sensor_msgs/msg/PointCloud2			not monitored
/joint_states	sensor_msgs/msg/JointState			not monitored
/joy	sensor_msgs/msg/Joy			not monitored
/joy/set_feedback	sensor_msgs/msg/JoyFeedback			not monitored
/local_costmap/clearing_endpoints	nav_msgs/msg/OccupancyGrid2D			not monitored
/local_costmap/costmap	nav_msgs/msg/OccupancyGrid			not monitored
/local_costmap/costmap_raw	nav2_msgs/msg/Costmap			not monitored
/local_costmap/costmap_updates	map_msgs/msg/OccupancyGridUpdate			not monitored
/local_costmap/footprint	geometry_msgs/msg/Polygon			not monitored
/local_costmap/local_costmap/transition_event	lifecycle_msgs/msg/TransitionEvent			not monitored
/local_costmap/published_footprint	geometry_msgs/msg/PolygonStamped			not monitored
/local_costmap/voxel_grid	nav2_msgs/msg/VoxelGrid			not monitored
/local_plan	nav_msgs/msg/Path			not monitored
/map	nav_msgs/msg/OccupancyGrid			not monitored
/map_server/transition_event	lifecycle_msgs/msg/TransitionEvent			not monitored
/map_updates	map_msgs/msg/OccupancyGridUpdate			not monitored
/move_base/cancel	actionlib_msgs/msg/GoalID			not monitored
/navigate_through_poses/_action/feedback	nav2_msgs/action/NavigateThroughPoses_FeedbackMessage			can not get message class for type "nav2_msgs/action/NavigateThroughPoses_F
/navigate_through_poses/_action/status	action_msgs/msg/GoalStatusArray			not monitored
/navigate_to_pose/_action/feedback	nav2_msgs/action/NavigateToPose_FeedbackMessage			can not get message class for type "nav2_msgs/action/NavigateToPose_Feedbac
/navigate_to_pose/_action/status	action_msgs/msg/GoalStatusArray			not monitored
/obstacles	costmap_converter_msgs/ObstacleArrayMsg			not monitored
/odom	nav_msgs/msg/Odometry			not monitored
/odom_raw	nav_msgs/msg/Odometry			not monitored
/parameter_events	rcl_interfaces/msg/ParameterEvent			not monitored
/particle_cloud	nav2_msgs/msg/ParticleCloud			not monitored
/particlecloud	geometry_msgs/msg/PoseArray			not monitored
/plan	nav_msgs/msg/Path			not monitored
/plan_smoothed	nav_msgs/msg/Path			not monitored
/planner_server/transition_event	lifecycle_msgs/msg/TransitionEvent			not monitored
/point_cloud	sensor_msgs/msg/PointCloud			not monitored
/robot_description	std_msgs/msg/String			not monitored
/rosout	rcl_interfaces/msg/Log			not monitored
/scan	sensor_msgs/msg/LaserScan			not monitored
/set_pose	geometry_msgs/msg/PoseWithCovarianceStamped			not monitored
/smooth_path/_action/feedback	nav2_msgs/action/SmoothPath_FeedbackMessage			can not get message class for type "nav2_msgs/action/SmoothPath_FeedbackMe
/smooth_path/_action/status	action_msgs/msg/GoalStatusArray			not monitored
/smoother_server/transition_event	lifecycle_msgs/msg/TransitionEvent			not monitored
/speed_limit	nav2_msgs/msg/SpeedLimit			not monitored
/teb_feedback	teb_msgs/msg/FeedbackMsg			not monitored
/teb_markers	visualization_msgs/msg/Marker			not monitored
/teb_poses	geometry_msgs/msg/PoseArray			not monitored
/tf	tf2_msgs/msg/TFMessage			not monitored
/tf_static	tf2_msgs/msg/TFMessage			not monitored
/vel_raw	geometry_msgs/msg/Twist			not monitored
/velocity_smoothen/transition_event	lifecycle_msgs/msg/TransitionEvent			not monitored
/voltage	nav_msgs/msg/Path			not monitored
/voltage	std_msgs/msg/Float32			not monitored
/waypoint_follower/transition_event	lifecycle_msgs/msg/TransitionEvent			not monitored
✓ /waypoints	visualization_msgs/msg/MarkerArray	unknown	unknown	
└ markers	sequence<visualization_msgs/Marker>			
└ [0]	visualization_msgs/Marker			
└ action	int32		0	
└ color	std_msgs/ColorRGBA		[]	False
└ colors	sequence<std_msgs/ColorRGBA>			
└ frame_locked	boolean			
└ header	std_msgs/Header			
└ id	int32		0	
└ lifetime	builtin_interfaces/Duration			
└ mesh_file	visualization_msgs/MeshFile			
└ mesh_resource	string		"	
└ mesh_use_embedded_materials	boolean		False	
└ ns	string		"	
└ points	sequence<geometry_msgs/Point>			
└ pose	geometry_msgs/Pose			
└ orientation	geometry_msgs/Quaternion			
└ w	double		0.9897298469191461	
└ x	double		0.0	
└ y	double		0.0	
└ z	double		0.14295044637007442	
└ position	geometry_msgs/Point			
└ x	double		1.0541921854019165	
└ y	double		2.462282657623291	
└ z	double		0.0	
└ scale	geometry_msgs/Vector3			
└ text	string		"	
└ texture	sensor_msgs/CompressedImage			
└ texture_resource	string		"	
└ type	int32		0	

Click on a waypoint to view the waypoint data. Here, we take [0] as an example, where pose is the coordinate data.

/waypoints	visualization_msgs/msg/MarkerArray	unknown	unknown
└ markers	sequence<visualization_msgs/Marker>		
└ [0]	visualization_msgs/Marker		
└ action	int32		0
└ color	std_msgs/ColorRGBA		[]
└ colors	sequence<std_msgs/ColorRGBA>		False
└ frame_locked	boolean		
└ header	std_msgs/Header		
└ id	int32		0
└ lifetime	builtin_interfaces/Duration		
└ mesh_file	visualization_msgs/MeshFile		"
└ mesh_resource	string		False
└ mesh_use_embedded_materials	boolean		
└ ns	string		"
└ points	sequence<geometry_msgs/Point>		
└ pose	geometry_msgs/Pose		
└ orientation	geometry_msgs/Quaternion		
└ w	double		0.9897298469191461
└ x	double		0.0
└ y	double		0.0
└ z	double		0.14295044637007442
└ position	geometry_msgs/Point		
└ x	double		1.0541921854019165
└ y	double		2.462282657623291
└ z	double		0.0
└ scale	geometry_msgs/Vector3		
└ text	string		"
└ texture	sensor_msgs/CompressedImage		
└ texture_resource	string		"
└ type	int32		0

3.2 Data Sending Execution

After setting the waypoint coordinates, click **Start Waypoint When following a waypoint, the rviz plugin packages the waypoint coordinate sequence into a `Follow waypoints` action request and sends it to the `/follow_waypoint` action server, which executes all waypoints in sequence.

Enter the following command in the terminal:

```
ros2 run rqt_graph rqt_graph
```

In the node relationship graph, you can see the `/follow_waypoint` action server. This action server receives the waypoint sequence and then navigates sequentially.

