[Week 4]
Path Planning
(Practice)

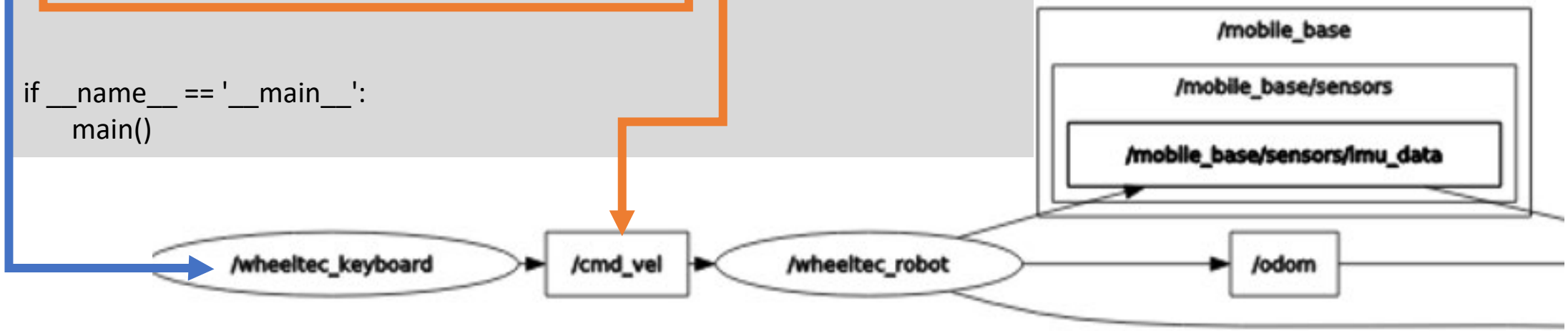
授課教師：郭重顯 教授
助教：Nguyen Thanh Thien Phuc、莊明洋
助教實驗室：工綜 106

Question

❖What does rqt_graph display?

- ❑ Circle : Node
- ❑ Square : Topic

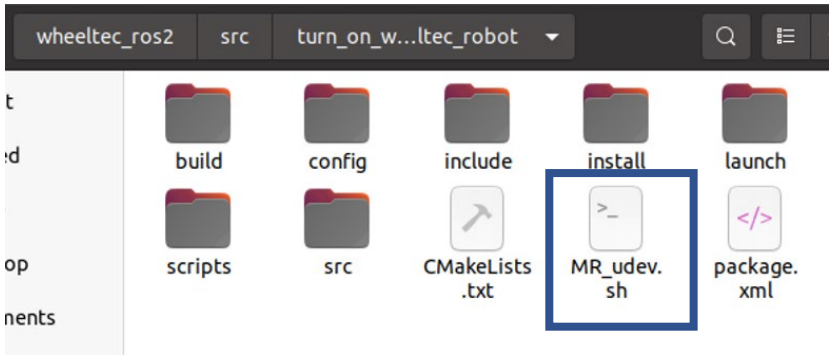
```
def main():
    rclpy.init()
    qos = QoSProfile(depth=10)
    node = rclpy.create_node('wheeltec_keyboard')
    pub = node.create_publisher(Twist, 'cmd_vel', qos)


if __name__ == '__main__':
    main()
```



Last week practice - SLAM

❖Step 1: Setting up the information for the lidar.

- ❑ Downland the code from NTU cool [MR_udev.sh].
- ❑ Put it into the /home/wheeltec/wheeltec_ros2/src/turn_on_wheeltec_robot.
- ❑ Run the following code. (Password :dongguan)
  - cd ~/wheeltec_ros2/src/turn_on_wheeltec_robot/
  - sudo sh MR_udev.sh
- ❑ ***Reconnect the Lidar!!!!!***



Last week practice - SLAM

❖Step 2: Use the following commands to start SLAM.

- ❑ Keyboard shortcut Ctrl+Alt+T: Create a new terminal. [Terminal A]
- ❑Enter "ros2 launch slam_gmapping slam_gmapping.launch.py" to launch the launch file.

❖IncludeLaunchDescription is an action in ROS 2 Launch used to include the contents of another launch file within the current launch file. This allows for organizing and combining the startup of multiple subsystems or nodes within a single launch file. By using IncludeLaunchDescription, code can be modularized and made more reusable, while also making the overall startup configuration clearer.

```
11 def generate_launch_description():
12     use_sim_time = launch.substitutions.LaunchConfiguration('use_sim_time', default='false')
13 
14     bringup_dir = get_package_share_directory('turn_on_wheeltec_robot')
15     launch_dir = os.path.join(bringup_dir, 'launch')
16 
17     wheeltec_lidar = IncludeLaunchDescription(
18             PythonLaunchDescriptionSource(os.path.join(launch_dir, 'wheeltec_lidar.launch.py')),
19     )
20     wheeltec_robot = IncludeLaunchDescription(
21             PythonLaunchDescriptionSource(os.path.join(launch_dir, 'turn_on_wheeltec_robot.launch.py')),
22     )
23     return LaunchDescription([
24         wheeltec_robot,wheeltec_lidar,
25         SetEnvironmentVariable('RCUTILS_LOGGING_BUFFERED_STREAM', '1'),
26         launch_ros.actions.Node(
27             package='slam_gmapping', executable='slam_gmapping', output='screen', parameters=[{'use_sim_time':use_sim_time}]),
28     ])
```

Last week practice - SLAM

❖**Step 3: Open RViz to visualize the SLAM process.**

- ❑Keyboard shortcut Ctrl+Shift+T: Create a new terminal. [Terminal B]
- ❑Enter “rviz2” to open rviz.
- ❑Using keyboard to control the robot.

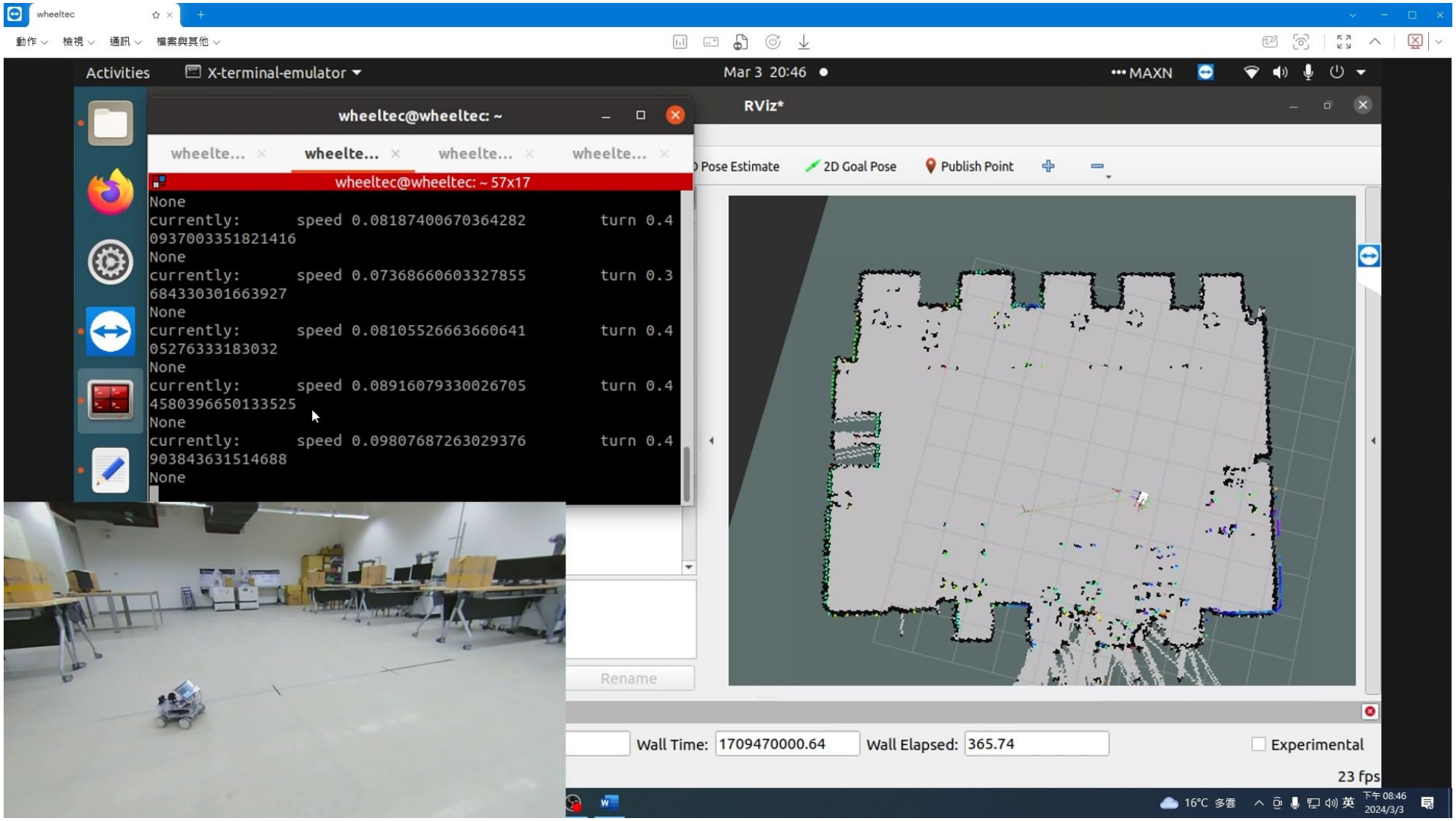
❖When building the map, it is necessary to slow down the AMR speed to achieve optimal results.

❖**Step 4: Save the map**

- ❑Enter “ ros2 launch wheeltec_nav2 save_map.launch.py” :After mapping is completed, use this command to save the map for future use.
- ❑You can find your map at following path:

  /home/wheeltec/wheeltec_ros2/src/wheeltec_robot_nav2/map

Last week practice - SLAM



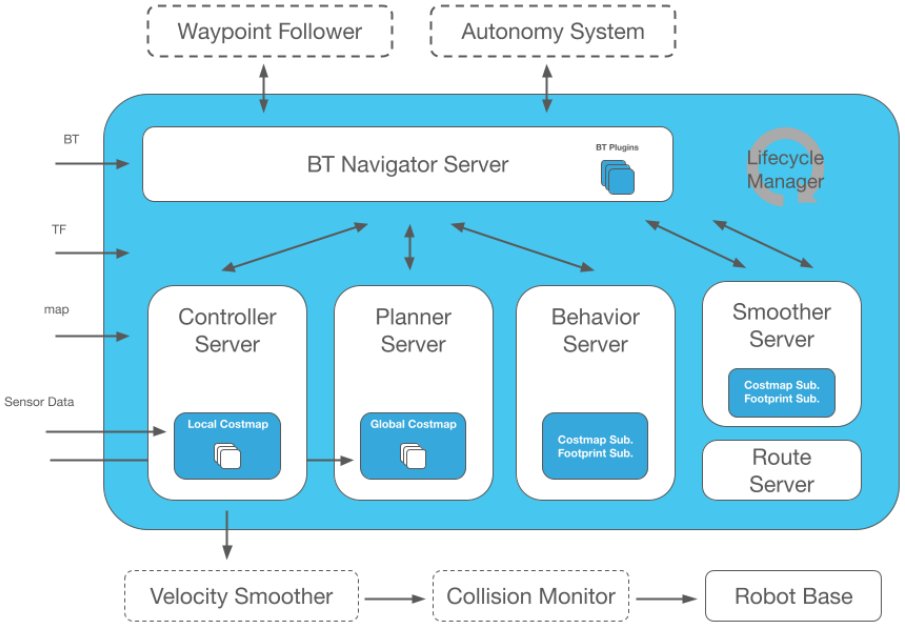
Practice 1 – Navigation2

❖ Introducing Navigation2

❖It provides perception, planning, control, localization, visualization, and much more to build highly reliable autonomous systems. This will compute an environmental model from sensor and semantic data, dynamically path plan, compute velocities for motors, avoid obstacles, and structure higher-level robot behaviors.

❖It has tools to:

- ❑Localize the robot on a provided map
- ❑Control the robot to follows the path and dynamically adjust to avoid collision
- ❑Convert sensor data into an environmental model of the world
- ❑Follow sequential waypoints comprising a mission



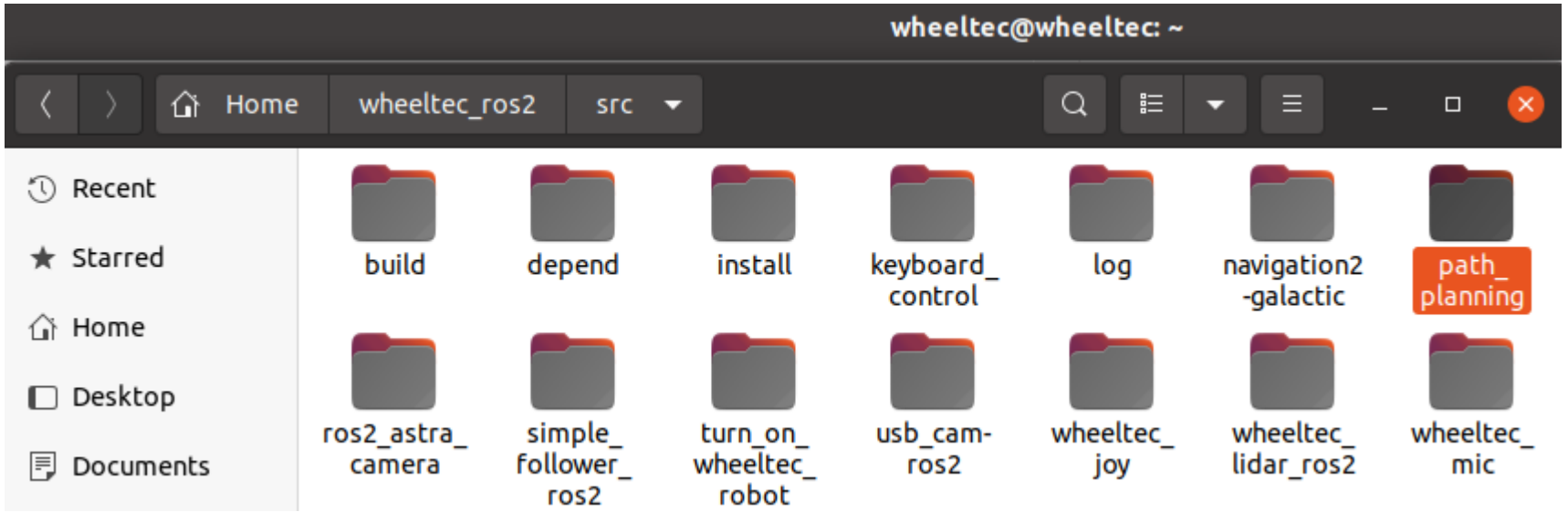
https://navigation.ros.org/behavior_trees/overview/nav2_specific_nodes.html

Practice 1 – Navigation2

❖ **Step 1: Create a new node to send the goal position to robot.**

❑ How to create the new node?

- Usage Ctrl+Alt+T: Create a new terminal. [Terminal A]
- Enter "cd ~/wheeltec_ros2/src" :This command changes the current directory to the 'wheeltec_ros2/src' directory located in the user's home directory .
- Enter "ros2 pkg create --build-type ament_python --node-name path_generater path_planning" to create a new node.



Practice 1 – Navigation2

❖ Step 2: Coding the node program.

- ❑You can find code in "wheeltec_ros2/src/path_planning/path_planning/path_generater.py"
- ❑The example code is upload to "NTU COOL"

❖Coding the ***main*** program.

- ❑ rclpy.init(): Initialize the ROS 2 client library.
- ❑ qos =QoSProfile(depth=10): Define a Quality of Service profile with a queue depth of 10.
  - This sets how messages are queued in the system.
- ❑ node=rclpy.create_node('AMR_path'):
  - Create a ROS 2 node with the name 'wheeltec_keyboard'.
- ❑ pub = node.create_publisher(Twist, /goal_pose', qos): Create a publisher that publishes messages of type Twist to the topic 'cmd_vel'. The publisher will use the specified QoS profile.

Practice 1 – Navigation2

❖ PoseStamped: This is a message type in ROS used to represent the pose of a robot or object in three-dimensional space. The message contains both the position and orientation of the pose. PoseStamped is commonly used in robot navigation and motion control to determine the position and orientation of the robot in space.

- ❑goal_msg.header.frame_id='map'
- ❑goal_msg.pose.position.x=[The x-coordinate of the robot's destination position.]
- ❑goal_msg.pose.position.y=[The y-coordinate of the robot's destination position.]
- ❑goal_msg.pose.position.z=0.0
- ❑goal_msg.pose.orientation.x=0.0
- ❑goal_msg.pose.orientation.y=0.0
- ❑goal_msg.pose.orientation.z=[The orientation of the robot's destination position.]
- ❑goal_msg.pose.orientation.w=[The orientation of the robot's destination position.]

Practice 1 – Navigation2

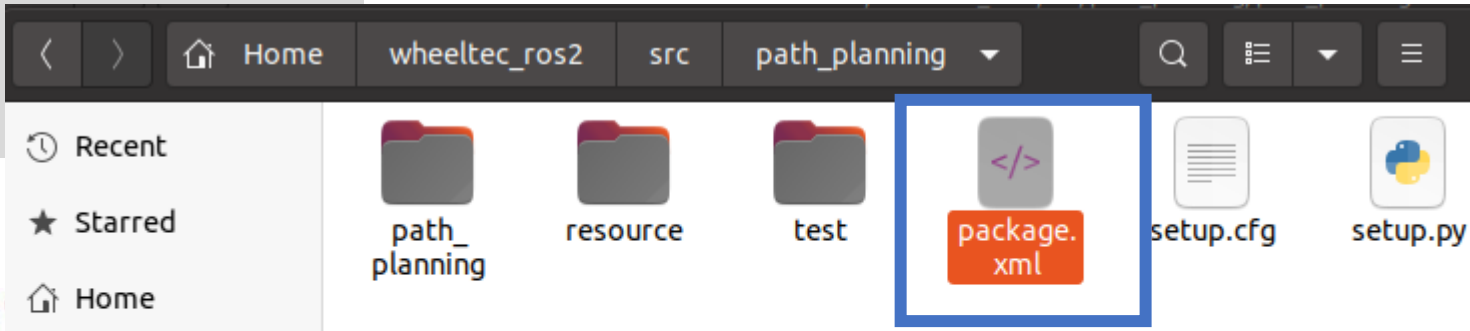
```
def main():
    rclpy.init()
    qos = QoSProfile(depth=10)
    node = rclpy.create_node('AMR_path')
    pub = node.create_publisher(PoseStamped, '/goal_pose', qos)

    try:
        goal_msg=PoseStamped()
        goal_msg.header.frame_id='map'
        goal_msg.pose.position.x=0.0
        goal_msg.pose.position.y=0.0
        goal_msg.pose.position.z=0.0
        goal_msg.pose.orientation.x=0.0
        goal_msg.pose.orientation.y=0.0
        goal_msg.pose.orientation.z=0.0
        goal_msg.pose.orientation.w=1.0
        pub.publish(goal_msg)
    except Exception as e:
        print('error!!')
    rclpy.spin(node)
    node.destroy_node()
    rclpy.shutdown()
if __name__ == '__main__':
    main()
```

You can modify the program to perform more robot functionalities.

Practice 1 – Navigation2

❖ Step 3: Edit the package.xml file to add the dependency on rclpy.

```
...
<exec_depend>rclpy</exec_depend>
...
```




```
 7   <maintainer email="wheeltec@todo.todo
 8   <license>TODO: License declaration</l
 9
10   <test_depend>ament_copyright</test_de
11   <test_depend>ament_flake8</test_depen
12   <test_depend>ament_pep257</test_depen
13   <test_depend>python3-pytest</test_depend>
14
15   <exec_depend>rclpy</exec_depend>
16
17   <export>
18     <build_type>ament_python</build_type>
19   </export>
20 </package>
```

Practice 1 - Navigation2

❖ **Step 4: Go back to the Workspace directory and compile the program.**

- ❑Enter “cd ../” :Change directory to the previous directory.
- ❑Enter “colcon build --packages-select path_planning” to compile the program.

❖**Step 5: Source**

- ❑Enter “ source install/setup.bash “
- ❑The “source” command is used in ROS 2 to load ROS 2 environment variables into the current shell session. When you run this command, it executes the necessary scripts to set up the environment for working with ROS 2, including configuring paths and variables required by ROS 2 tools and packages. This ensures that ROS 2 commands and tools are available and properly configured for use within the current shell session.

❖**Step 6: Run the nav2 launch file**

- ❑Enter “ros2 launch wheeltec_nav2 wheeltec_nav2.launch.py”

Practice 1 - Navigation2

❖Step 7: Open RViz to visualize the navigation process.

- ❑Keyboard shortcut Ctrl+Shift+T: Create a new terminal. [Terminal B]
- ❑Enter “rviz2” to open rviz.

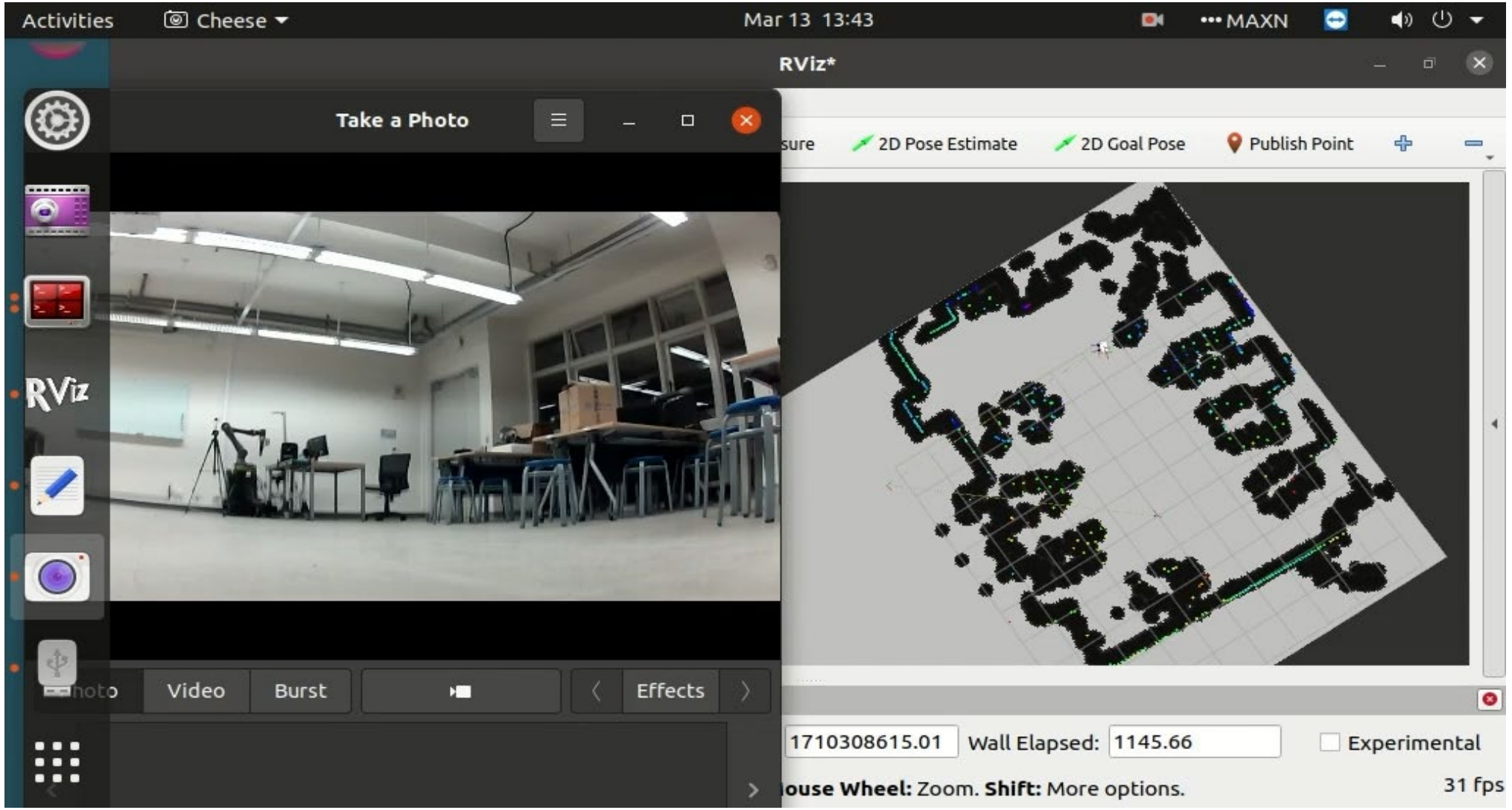
❖Step 8: Run the program

- ❑Keyboard shortcut Ctrl+Shift+T: Create a new terminal. [Terminal C]
- ❑Enter “ros2 run path_planning path_generater”

❖How to check the message?

- ❑Keyboard shortcut Ctrl+Shift+T: Create a new terminal. [Terminal D]
- ❑Enter “ros2 topic echo /goal_pose”

Practice 1 - Navigation2



Practice 2 – Rapidly-Exploring Random Trees

❖Step 1: Run the 2D SLAM launch file

- ❑Enter "ros2 launch wheeltec_slam_toolbox online_sync.launch.py"

❖Step 2: Run the RRT launch file

- ❑Enter "ros2 launch wheeltec_robot_rrt wheeltec_rrt_slam.launch.py"
- ❑Wait for about 1 minute.

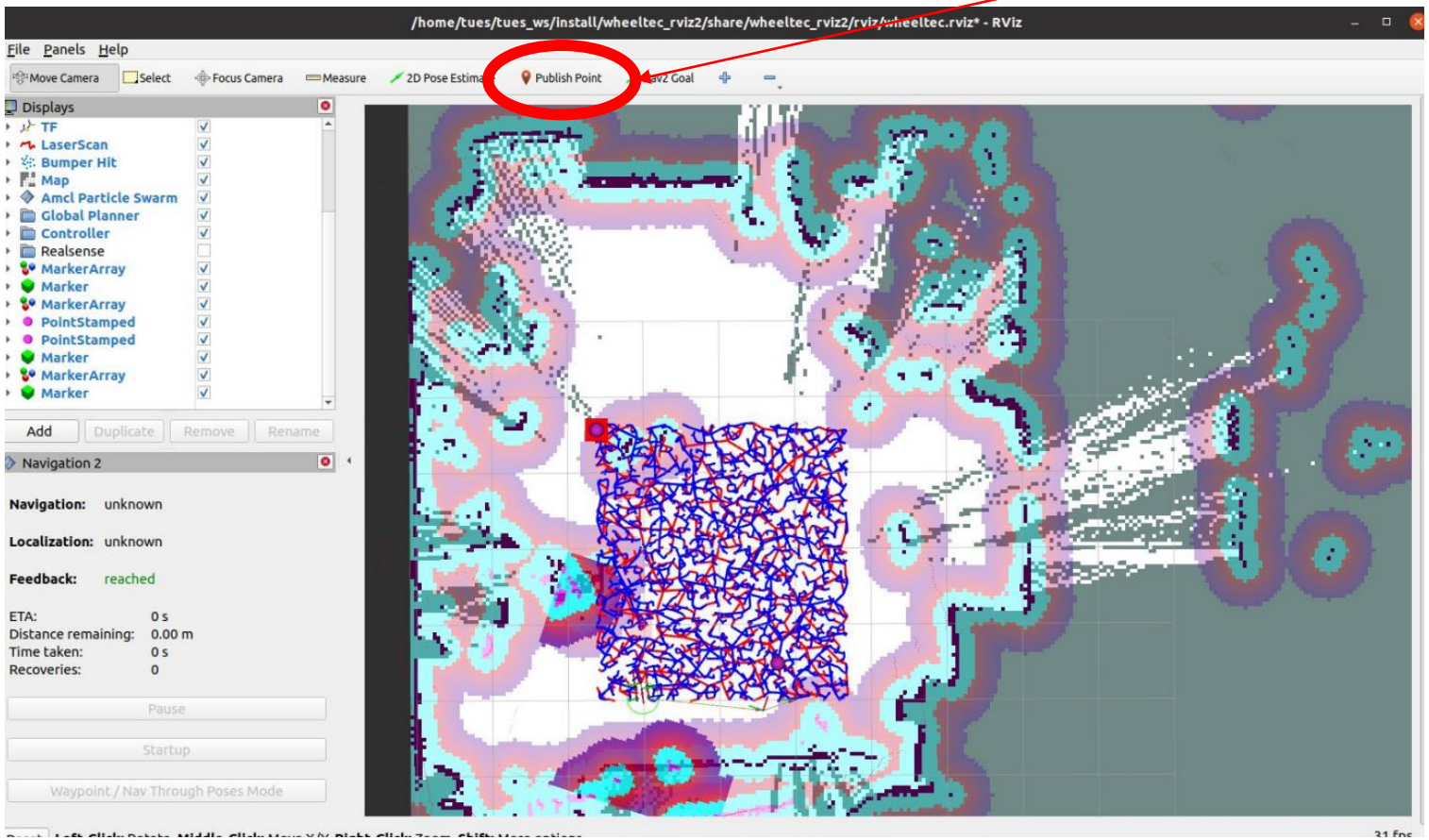
❖Step 3: Open RViz to visualize the process.

- ❑Enter "ros2 launch wheeltec_rviz2 wheeltec_rviz.launch.py"

Practice 2 – Rapidly-Exploring Random Trees

❖Step 4: Select five points.

❑The last point needs to be at the robot's position.



Practice 2 – Rapidly-Exploring Random Trees