SLAM and Navigation Steps for Any Robot (ROS2 Nav2 Course - Section 7)

Those commands are the general commands to run for SLAM and Navigation, when using any robot that is configured for the Nav2 stack.

Steps - SLAM

You will need to install the slam_toolbox package:

```
$ sudo apt install ros-humble-slam-toolbox
```

1. Start your robot

This will be specific to your own robot.
Example with simulation:
$ ~~ros2 launch turtlebot3_gazebo turtlebot3_world.launch.py~~

2. Start a Navigation launch file

```
$ ros2 launch nav2_bringup navigation_launch.py
```

(~~add *use_sim_time:=True* if using Gazebo~~)

3. Start SLAM with slam_toolbox

```
$ ros2 launch slam_toolbox online_async_launch.py
```

(~~add *use_sim_time:=True* if using Gazebo~~)

4. Start Rviz

```
$ ros2 run rviz2 rviz2
```

(you will need to configure RViz, follow the instructions in the video)

5. Generate and save your map

Make the robot move in the environment (specific to your own robot).
Example with simulation:

```
$ ros2 run turtlebot3_teleop teleop_keyboard
```

Save the map:

```
$ ros2 run nav2_map_server map_saver_cli -f ~/my_map
```

Steps - Navigation

1. Start your robot

This will be specific to your own robot.
Example with simulation:

```
$ ros2 launch turtlebot3_gazebo turtlebot3_world.launch.py
```

2. Start the main Navigation2 launch file

```
$ ros2 launch nav2_bringup bringup_launch.py map:=path/to/map.yaml
```

(add *use_sim_time:=True* if using Gazebo)

3. Start RViz

```
$ ros2 run rviz2 rviz2
```

(you will need to configure RViz, follow the instructions in the video)

4. Send navigation commands

Use the “2D Pose Estimate” button to set the initial pose, and the “Nav2 Goal” button to send navigation goals.

Note: instead of using RViz to send commands, you can directly interact with the Nav2 interfaces in your own code, for example using the Simple Commander API (see Section 8 of the course)