CS 489/689 Autonomous Racing: **slam_toolbox** and **particle_filter** simuator instructions

Section 1001, Fall 2023

---

1 Simulator Installations

This section will inlcude abbreviated instructions for installing `slam_toolbox`, Nav2 binaries and a Turtlebot for the Gazebo simualtor. Being brief, details will be omitted please check the documentation provided in Resources (Section 4) for more information.

1.1 `slam_toolbox` installation

On the system running ROS 2.0:

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

1.2 Nav2 package installations

```
$ sudo apt install ros-foxy-navigation2
$ sudo apt install ros-foxy-nav2-bringup
```

1.3 Turtle Bot package installations

```
$ sudo apt install ros-foxy-turtlebot3-gazebo
```

2 `particle_filter` installation

The abbreviated instructions for installing the `particle_filter` and its dependencies follow. Being brief, details will be omitted please check the documentation for ROS the `particle_filter` and `range_libc` packages for more information.

The following instructions assume there exists a workspace, such as `$HOME/.../f1tenth_ws/`; some of the instructions require execution in specific directories to work.

2.1 `range_libc`

The `particle_filter` package depends on the `range_libc` library. Install `range_libc` first.

```
$ cd /home/.../f1tenth_ws/
f1tenth_ws $ git clone https://github.com/f1tenth/range_libc.git

f1tenth_ws $ cd range_libc/pywrapper
pywrapper $ sudo WITH_CUDA=OFF python setup.py install
```

2.1.1 CUDA Errors

The vehicle has a cuda compiler (`nvcc`) installed on it. A local environment may not have the compiler, libraries, or supported NVIDIA card. When cuda support is not available, run the following commands:

```
$ cd /home/.../f1tenth_ws/range_libc
range_libc $ cmake -D WITH_CUDA=OFF
```

2.2 particle_filter

```
# clone the repository
$ cd ${HOME}/f1tenth_ws/src
src $ git clone https://github.com/f1tenth/particle_filter.git

# update dependencies- especially if
#     you had to change the CUDA cmake flag
$ source /opt/ros/foxy/setup.bash
$ source ${HOME}/f1tenth_ws/install/setup.bash
$ cd ${HOME}/f1tenth_ws
f1tenth_ws $ rosdep install -r --from-paths src --ignore-src --rosdistro foxy -y

# rebuild
f1tenth_ws $ colcon build
f1tenth_ws $ source install/setup.bash
```

3 Running slam_toolbox

Our approach will be to use the `slam_toolbox` pannel in rviz2 to generate a map as a Turtlebot operates around the Gazebo simulator.

The generated map will then be used by the `particle_filter`.

The following instructions will need to be run to begin mapping and simulation. Remember to source the terminals if required.

```
# Terminal One
$ source /opt/ros/foxy/setup.bash
$ export TURTLEBOT3_MODEL=waffle
$ ros2 launch turtlebot3_gazebo turtlebot3_world.launch.py

# Terminal Two
$ ros2 launch slam_toolbox online_async_launch.py

# Terminal Three
$ ros2 launch nav2_bringup navigation_launch.py

#Terminal Four
$ rviz2

#Terminal Five
$ ros2 run teleop_twist_keyboard teleop_twist_keyboard
```

1. Add a the `Map` by topic
2. Add `Path` by topic
3. Add `/LaserScan` by topic
4. Add SlamToolBoxPlugin by panel

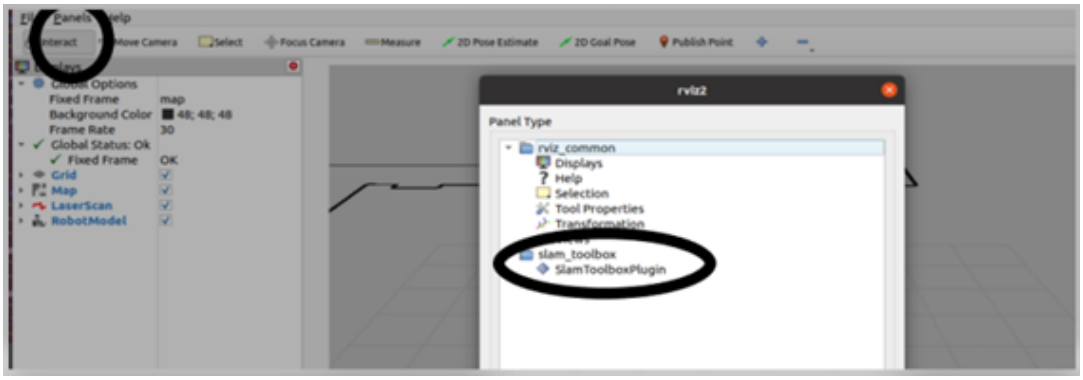

Figure 1: Location of `slam_toolbox` plugin in rviz2

At this stage, there may (or may not) be a relatively sparse map with a few impact points. Drive the bot around as **smoothly** and slowly as possible. It is more important to be smooth than it is slow. Complete at least one "loop" with the bot so that the pose graph can add a closure.

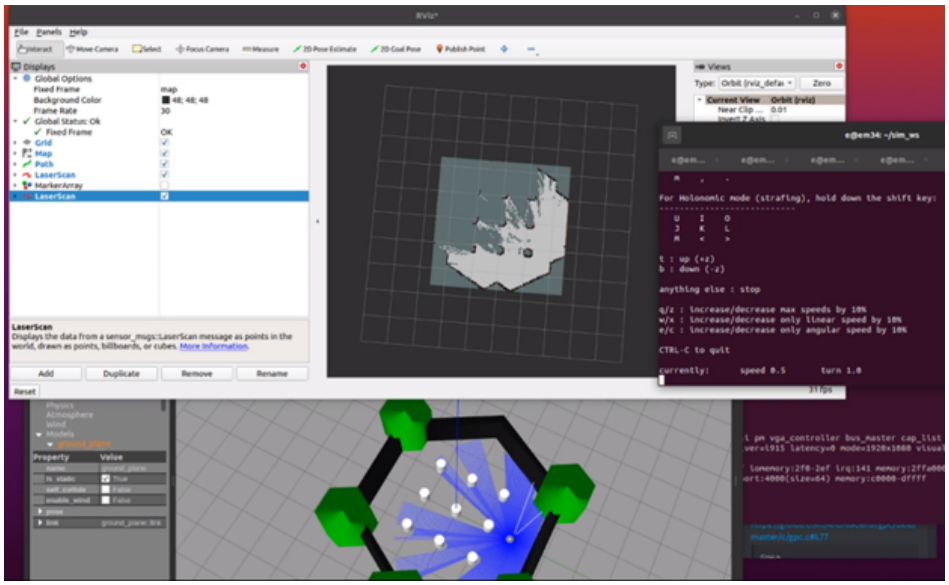

Figure 2: Example of map generating in rviz2

Once you have a sufficiently detailed map, use the panel to enter a name for your map and click `Save Map`. The new map will be saved as two files a `.pgm` and a `.yaml` file in the directory where you ran the `slam_toolbox` from (in our example that is the `$HOME/f1tenth_ws` directory). These instructions assume the map is named `tbe` (`tbe.pgm` and `tbe.yaml`).

3.1 Localization

Ensure none of the ROS packages are running. Add the map's `.pgm` and `.yaml` file to the `/map` directory under the `particle_filter` package.

Next, edit the configuration for the particle filter to reflect the new map.

```
# EDITOR of your choice
$ ${EDITOR} ${HOME}/f1tenth_ws/src/particle_filter/config/localize.yaml

# Modify the line that sets the 'map' to
    map: 'tbe' # from 'levine'
```

WITH_CUDA=OFF: If you installed `particle_filter` without cuda support, modify `localize.yaml` to use a different `range_method`.

```
# EDITOR of your choice
$ ${EDITOR} ${HOME}/f1tenth_ws/src/particle_filter/config/localize.yaml

# Modify the line that sets the ‘range_method‘ to
    range_method: ’rm’ # from ’rmgpu’
```

Rebuild the particle filter.

```
$ source /opt/ros/foxy/setup.bash
$ source ${HOME}/f1tenth_ws/install/setup.bash
$ cd ${HOME}/f1tenth_ws
f1tenth_ws $ colcon build

f1tenth_ws $ source install/setup.bash
```

Use terminals to start the ROS system and launch the particle filter.

```
# Terminal One
$ source /opt/ros/foxy/setup.bash
$ source ${HOME}/f1tenth_ws/install.setup.bash
$ cd ${HOME}/f1tenth_ws
f1tenth_ws $ ros2 launch f1tenth_stack bringup_launch.py

# Terminal Two
f1tenth_ws $ ros2 launch particle_filter localize_launch.py

#Terminal Three
$ rviz2

#Terminal Four
$ ros2 run teleop_twist_keyboard teleop_twist_keyboard
```

1. Add a the `map` by topic
2. Set the `map` topic's durability to `transient local`
3. Add `/pf/viz/inferred_pose` by topic
4. Add `/pf/viz/particles` by topic to view the particles – this is an expensive operation, avoid it if performance is an issue.

4 Resources

ROS2 Installation Guide Helpful information regarding the installation of ROS2.

https://docs.ros.org/en/foxy/Installation/Ubuntu-Install-Debians.html

Initializing rosdep Instuctions Instructions for initializing rosdep, convieniently highlighting necessary instructions.

https://docs.ros.org/en/foxy/How-To-Guides/Building-a-Custom-Debian-Package.html?highlight=rosdep#install-dependencies

Nav2 Installation Guide Instructions for use and installation of Nav2 and its tutorials.

https://navigation.ros.org/getting_started/index.html