Simulating Mowito Rosbot Documentation

Release 0.0.1

Mowito

Dec 17, 2020

CONTENTS

CHAPTER

ONE

SETTING UP MAXL

1.1 Prerequisites

- **Ubuntu 20** - Currently we have only tested MaxL on Ubuntu 20. Please mail to us, if you want a Ubuntu 18 version of MaxL.
- **ROS2 Foxy** - MaxL has been tested only for Foxy. The instructions for ROS2 can be found on ROS2 page.
- **Nav2** - Setup Nav2. The instructions can be founder on Nav2 documentation. We, during our testings, built it from source.
- **TurtleBot3** - Setup turtleBot3. Our instructions for this setup can be found on TurtleBot3 page.

Note: NOTE: In the instructions it has been assumed that you have cloned and build navigation2 in `~/navigation_ws/src` . If that's not the case, please replace it by the address of ros2 workspace you intend to use

1.2 Install the dependencies and library

1.2.1 1. Setup your sources.list

```
echo "deb [trusted=yes] https://mowito-packages.s3.amazonaws.com stable main" | sudo␣
↪tee -a /etc/apt/sources.list
```

1.2.2 2. Update

```
sudo apt update
```

1.2.3 3. Install MLicense

```
sudo apt install ros-foxy-mlicense
```

1.2.4 4. Install PCL_ros

```
sudo apt install ros-foxy-pcl-ros
```

If you prefer to build it from source then you can clone it by doing

```
cd ~/navigation2_ws/src && git clone https://github.com/ros-perception/perception_pcl.
↪git -b foxy-devel
```

and you can compile/build it while building the MaxL wrapper for ROS2

1.2.5 5. Install MaxL library

```
sudo apt install ros-foxy-mw-maxl-planner
```

1.3 Register

Execute the following

```
source /opt/ros/foxy/setup.bash && ros2 run mlicense robot_reg.py -nr
```

when asked, type a name for your computer/robot

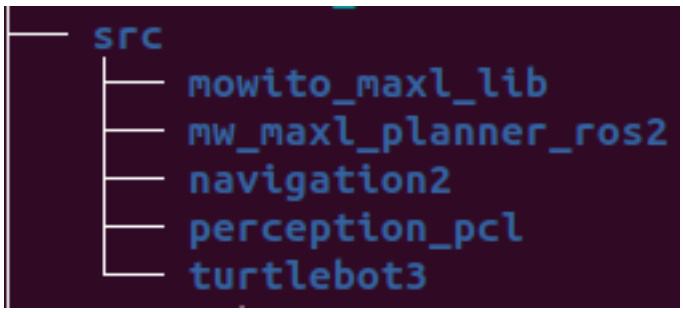
and that's it!

1.4 Build the MaxL Nav2 Plugin

1.4.1 1. Clone the plugin

- `cd ~/navigation2_ws/src`
- `git clone https://github.com/mowito/mw_maxl_planner_ros2.git`

At this point the source directory of your workspace would look something similar to the image below. It may defer if you have installed pcl_ros (perception_pcl) by binaries and not buil by source



1.4.2 2. Update rosdep sources

- Assuming the path to the Maxl Plugin is `~/navigation_ws/src/mw_maxl_planner_ros2`, do the following

```
echo "yaml file://$HOME/navigation2_ws/src/mw_maxl_planner_ros2/maxl_foxy_rosdep.yaml
↪" | sudo tee -a /etc/ros/rosdep/sources.list.d/50-my-packages.list
```

- now do

```
rosdep update
```

Note: NOTE: in the above commands replace `$HOME/navigation2_ws/src/mw_maxl_planner_ros2/maxl_foxy_rosdep.yaml` by the path of **mw_maxl_planner_ros2/maxl_foxy_rosdep.yaml** file in your system, in case it is different.

1.4.3 3) Build

- `source ~/navigation2_ws/install/setup.bash`
- `cd ~/navigation2_ws && colcon build`
- `source ~/navigation2_ws/install/setup.bash`

CHAPTER

TWO

USAGE

Note: NOTE: In the instructions it has been assumed that you have cloned and build navigation2 and turtlebot3 in `~/navigation_ws/src` . If that's not the case, please replace it by the address of ros2 workspace you intend to use

Now that all the required packages are installed correctly, it's time we see the mw_maxl_planner in action. First let's launch the simulation first. In a new terminal.

- `source /opt/ros/foxy/setup.bash`
- `source ~/navigation2_ws/install/setup.bash`
- `export TURTLEBOT3_MODEL=waffle`
- ```
  export GAZEBO_MODEL_PATH=$GAZEBO_MODEL_PATH:~/navigation2_ws/src/turtlebot3/
  ↪turtlebot3_simulations/turtlebot3_gazebo/models
  ```
- `ros2 launch turtlebot3_gazebo turtlebot3_world.launch.py`

TIP: Use *killall gzserver* if your gazebo does not restart

Now, in a new terminal

- `source /opt/ros/foxy/setup.bash`
- `source ~/navigation2_ws/install/setup.bash`
- `export TURTLEBOT3_MODEL=waffle`
- ```
  export GAZEBO_MODEL_PATH=$GAZEBO_MODEL_PATH:~/navigation2_ws/src/turtlebot3/
  ↪turtlebot3_simulations/turtlebot3_gazebo/models
  ```
- `ros2 launch mw_maxl_planner_ros2 mw_maxl_planner_ros2_launch.py`

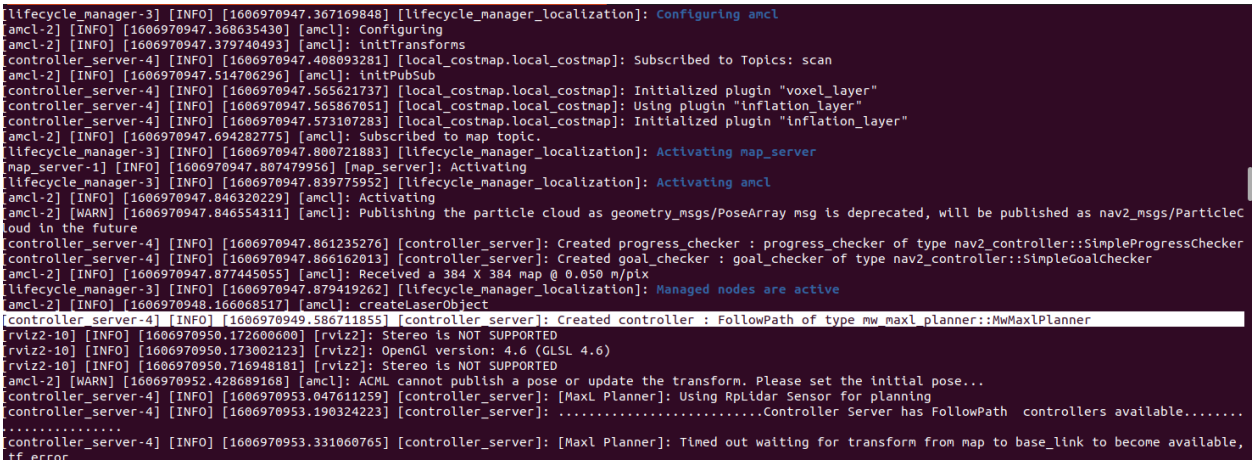
2.1 What to expect

First, give the initial pose of the robot using the 2D Pose Estimate on your Rviz GUI. After this, as soon as you give a navigation goal, the robot would start its navigation in a similar manner as shown in the GIF below.

2.2 Troubleshooting

If you are having issues, make sure you have done the following

1) Make sure you have the `mw_maxl_planner` library installed
2) **Make sure you have sourced your workspace using** `source ~/navigation2_ws/install/setup.bash`
3) Make sure your controller plugin gets loaded during runtime. If your plugin loads successfully, you will see something similar to the picture below.



4) If your plugin is not getting loaded, we suggest you delete the build, install and log folder and rebuild the workspace. Make sure you source your workspace.
5) Make sure that the perception_pcl (if build from source) and each of the turtlebot3 repos you cloned are on the `foxy-devel` branch.

CHAPTER

THREE

CONFIGURING MOWITO MAXL CONTROLLER PARAMETERS

3.1 Overview

Mowito MaxL Controller is a package that is used to drive the robot. It issues the linear and angular velocity commands that are needed to reach the goal.

3.2 Robot Parameter Description

3.2.1 1. Robot Configuration Parameters

Parameter	Units	Description
use_laser	true/false	If true, the robot uses the rpLidar Sensor otherwise uses velodyn Sensor for planning
pathFolder	File path	The relative path to the path folder
pathFile	String	The name of the path
autonomy-Mode	true/false	If true, calculates the relative goal for the robot to follow

3.2.2 2. Linear speed and acceleration

Parameter	Units	Description
maxSpeed	S.I (m/s)	Maximum possible linear velocity
maxAccel	S.I (m/s^2)	Maximum possible linear acceleration

3.2.3 3. Turning Parameters

Parameter	Units	Description
yaw_gain	(Numeric) eg.2.5	Yaw gain used when robot is in motion
stop_yaw_gain	(Numeric) eg. 0.6	Yaw gain used when robot is stopped/almost stopped
max_yaw_rate	(Numeric) eg. 0.5	Maximum angular velocity for the robot

3.2.4 4. Inflation

Parameter	Units	Description
x_inflate	S.I (m)	Obstacle Inflation in the x direction
y_inflate	S.I (m)	Obstacle Inflation in the y direction

3.2.5 5. Frame Names

Parameter	Units	Description
map_frame	String	Name of the map frame
robot_frame	String	Name of the robot base frame
velodyne_frame	String	Name of the velodyn Sensor frame
laser_frame	String	Name of the rpLidar Sensor frame

3.2.6 6. Topic Names

Parameter	Units	Description
odomTopic	String	The topic name which publishes the odometry
velodyneTopic	String	The topic name which publishes the velodyn sensor data
scanTopic	String	The topic name which publishes the rpLidar sensor data

3.2.7 7. Robot Footprint

Parameter	Units	Description
vehicleLength	S.I (m)	Length of the vehicle
vehicleWidth	S.I (m)	Width of the vehicle

3.2.8 8. Obstacle Ranges

Parameter	Units	Description
obstacle_horizon	S.I (m)	Parameter used for cropping the pointcloud
min_path_range	S.I (m)	Minimum path range for finding the path
initial_path_scale	(Numeric) eg. 1.0	Initial path scale value. Path Scales scale the paths and distances. Low pathScale means path elongation and vice-versa.
min_path_scale	(Numeric) eg. 0.75	Minimum path scale value. For particular local goal, pathScale starts with initial value, finds a path, then value of path scale is decreased to find a longer solution path, till it hits the minPathScale.
path_scale_step	(Numeric) eg. 0.25	Path Scale step value

3.2.9 9. Lookahead Parameters

Parameter	Units	Description
min_lookahead	S.I (m)	The minimum lookahead on the global path for the robot
max_lookahead	S.I (m)	The minimum lookahead on the global path for the robot
closest_point_index_search	(Numeric) eg. 10	Search for closest point index within this range of previous closest point
min_radius	S.I (m)	Minimum radius the robot can take from current to goal pose
max_radius	S.I (m)	Maximum radius the robot can take from current to goal pose
max_omega_radius	S.I (m)	Radius set when condition for straight line is satisfied
max_y_deviation	S.I (m)	Maximum deviation in the lateral direction
lookahead_point_distance	S.I (m)	Used to find the point in the global path to follow

3.2.10 10. MaxL Parameters

Parameter	Units	Description
direction_threshold	(Numeric) eg. 120	Direction threshold(in degrees) on either side of robot wrt relative goal
high_accuracy_multiplier	(Numeric) eg. 0.4	High accuracy multiplier for reaching the goal (0,1]
in_place_rotation_penalty	(Numeric) eg. 0.05	Scoring parameter. Higher value penalises in place rotation more
goal_direction_preference	(Numeric) eg. 0.2	Scoring parameter. Higher value means controller prefers paths oriented towards the goal.
vis_pointcloud	true/false	Parameter to enable visualisation of detailed data (pointcloud data)
use_odom_velocity	true/false	Parameter to take velocity from odom messages