Mobile Robot Maze Performance

Moldir Zabirova, Valeriya Kostyukova, Daryn Kenzhebek

I. Global and Local Paths

In this laboratory, we focused on tuning DWA parameters for the TurtleBot3 mobile robot.

In ROS, mobile robot control and navigation involve global and local path planning. A global planner generates a collision-free path from the robot's current position to its goal and a local planner continuously adjusts the trajectory based on real-time sensor feedback. The global path is represented as waypoints, while the local path is expressed as a dynamic trajectory.

Inverse kinematics for a differential drive robot is often non-trivial, and closed-form solutions may not exist due to the inherent complexities of the differential drive system. As a result, approximate methods are commonly employed to address this challenge. While the global path provides a high-level plan from the robot's current position to its destination, the local path planner is essential for addressing dynamic obstacles, localization inaccuracies, and fine-tuning the robot's trajectory in real-time.

II. DWA Parameters

For the robot to create a local path and follow the global path effectively, we use DWA (Dynamic Window Approach) local planner algorithm, as was stated in the previous report. This DWA planner depends on the local costmap which provides obstacle information. The map was progressively changed in a photoshop as the actual osbactles seen by the robot did not coincide with the costmap information. The final version of the map is depicted on Fig. 1.

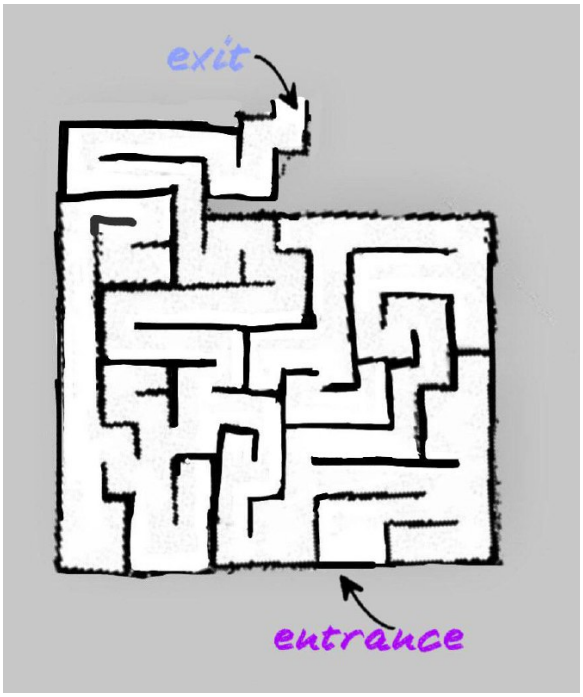

Fig. 1. Maze Costmap for the Robot Navigation

Our costmap_common_params_burger.yaml file contains the following parameters and their values:

```
obstacle_range: 3.0
raytrace_range: 3.5

footprint: [[-0.103, -0.103],
[-0.103, 0.103], [0.041, 0.103],
[0.041, -0.103]]
robot_radius: 0.103

inflation_radius: 0.75
cost_scaling_factor: 5.0

map_type: costmap
observation_sources: scan
scan: {sensor_frame: base_scan, data_type:
LaserScan, topic: scan, marking: true,
clearing: true}
```

In this file, we changed

1) *inflation_radius* takes into account a safety margin around obstacles, preventing the robot from getting too close to potential collisions during navigation. The optimal value was **0.75**.
2) *cost scaling factor* influences the robot's path planning behavior by adjusting the importance of different cost components in the costmap. The optimal value was **5.0**.

We leaved other parameters, such as footprtint and robot_radius parameters, as default.

The dwa_local_planner_params_burger.yaml includes

```
DWAPlannerROS:

# Robot Configuration Parameters
  max_vel_x: 0.22
  min_vel_x: -0.22

  max_vel_y: 0.0
  min_vel_y: 0.0

  max_vel_trans:  0.28
  min_vel_trans:  0.11

  max_vel_theta: 2.5
  min_vel_theta: 1.37

  acc_lim_x: 3.5
  acc_lim_y: 0.0
```

```
  acc_lim_theta: 10

# Goal Tolerance Parametes
  xy_goal_tolerance: 0.2
  yaw_goal_tolerance: 0.2
  latch_xy_goal_tolerance: false

# Forward Simulation Parameters
  sim_time: 3.5
  vx_samples: 50
  vy_samples: 0
  vth_samples: 80
  controller_frequency: 20.0

# Trajectory Scoring Parameters
  path_distance_bias: 20.0
  goal_distance_bias: 1.0
  occdist_scale: 0.02
  forward_point_distance: 0.325
  stop_time_buffer: 0.2
  scaling_speed: 0.25
  max_scaling_factor: 0.2

# Oscillation Prevention Parameters
  oscillation_reset_dist: 0.05

# Debugging
  publish_traj_pc : true
  publish_cost_grid_pc: true
```

In this file, we changed

1) *sim_time* time allowed for the robot to move with the sampled velocities. The optimal value was **3.5**. It allowed the robot to successfully pass through the narrow doorways and generate optimal lengths for the arc trajectories.
2) We have also increased the number of *velocity samples* that determine how many translational and rotational velocity samples to take in x, y and θ directions, respectively. Because we did not use y-velocities, we set the number of samples to zero. The optimal value for vx_samples was **50** and for vth_samples **80**. vth_samples are set to be higher than translational velocity samples, because turning is generally a more complicated condition than moving straight ahead.
3) *path_distance_bias* influences the local planner's preference for following the global path over deviating from it. We wanted our robot to focus more on the global path, so its value was much higher than *goal_distance_bias* parameter, which determines the local planner's bias towards approaching the goal. The respective optimal values were **20.0** and **1.0**.
4) *occdist_scale* scales the obstacle distance in the cost function used by the local planner. We set that "sensitivity" parameter to a small value that allowed the robot to make more confident movements when being near to the walls and, at the same time, be relatively cautious. The optimal value was **0.02**.
5) *xy_goal_tolerance* and *yaw_goal_tolerance* define the acceptable tolerances for reaching the goal in terms of position and orientation. The optimal values were **0.2** for both parameters.

We leaved other parameters, such as oscillation_reset_dist and scaling_speed parameters, as default. With these parameters, the TurtleBot3 robot went through the maze for a minimum of 2 minutes and 55 seconds (Fig. 2).

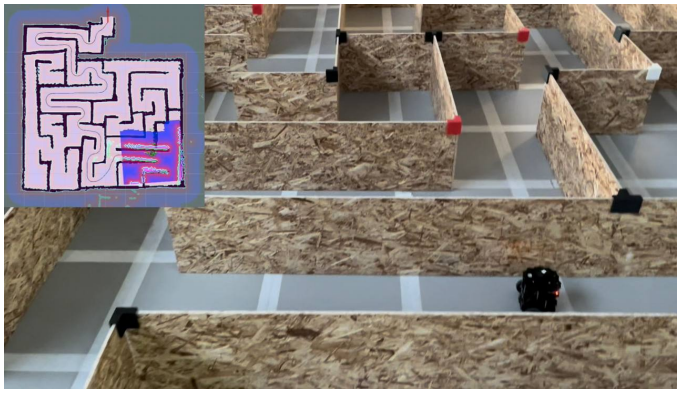

Fig. 2. Mobile Robot Maze Performance with RViz

III. Encountered Problems

The major problem that we encountered was the lighting. LiDAR sensors use laser light to measure distances and create a 3D map of the environment. Sometimes we needed to work in the evening and dim environment. In dark environments, the sensor may face challenges in accurately perceiving the surroundings. Thus, the robot poorly followed the global path.

The gaps between the flooe tiles were also a difficulty for the robot to overcome. Although they were covered by tape, the surface was still insufficiently even and smooth for the robot movements.

IV. Conclusion

The laboratory project focused on tuning the DWA parameters for the TurtleBot3 mobile robot, optimizing its performance in maze navigation. Through the adjustment of key parameters, including inflation radius, cost scaling factor, simulation time, and goal tolerances, the robot successfully navigated the maze with a runtime of 2 minutes and 55 seconds. However, challenges such as varying lighting conditions and uneven surfaces posed difficulties for the LiDAR sensor and the robot's movements. Despite these challenges, our team collaboratively addressed parameter tuning, highlighting the importance of careful parameter selection for effective mobile robot navigation.

V. Team Evaluation

All team members contributed equally in this laboratory session.