Lab 3 Report - LIDAR Mapping with Wheel Odometry

Team: Aditya Jain, Geet Patel, Sugumar Prabhakaran

1 Introduction

The first part of this lab is designing the calibration experiment for determining the robot's wheel radius and baseline. The second part focuses on building an occupancy grid map using lidar measurements and pose estimates from wheel odometry. The latter is done in a ROS simulation of Turtlebot 3 Waffle Pi.

2 Vehicle Calibration

1. **What path will the robot drive for each experiment?**
   - **Wheel Radius Experiment:** In order to calibrate the wheel radius through experiment, we will use the first environment that accurately measures how far the robot has travelled in a straight line. In the experiment:
     (a) we will drive the robot straight for 10m, as measured in the environment and the robot will not rotate at all (set angular velocity $(\omega) = 0$);
     (b) Given that we have traveled a distance of 10m in a straight line, we can then calculate the radius using the following equation:
     $r = 2\frac{x_{true}}{\sum_k(\Delta\psi_r(kh)+\Delta\psi_l(kh))}$
     (c) in the above equation, $x_{true} = 10$ is the total distance travelled in the x direction. The terms $\Delta\psi(kh)$ are the odometer changes (in radians) for the left and right wheels over that distance.
   - **Wheel Separation Experiment:** In order to experimentally determine the wheel separation, we will use the second environment that accurately measures the number of rotations performed by the robot. In the experiment:
     (a) we will rotate the robot $N = 10$ times, as measured in the environment and the robot will not move forward at all (set velocity $(v) = 0$);
     (b) Given that we have traveled an angular distance of $N \cdot 2\pi$, we can then calculate the wheel separation using the following equation (for a counter-clockwise rotation):
     $b = \frac{r}{2}\frac{\sum_k(\Delta\psi_r(kh)-\Delta\psi_l(kh))}{N\cdot 2\pi}$
     (c) Similar to the wheel radius equation, the $\Delta\psi(kh)$ are the odometer changes (in radians) for the left and right wheels over that distance. The radius r was calculated in the previous experiment.
2. **How many encoder ticks do you expect from each wheel in your experiments?**
   - **Wheel Radius Experiment:** From specs, the wheel radius should be $r = 33mm$ and since both wheels rotate the same as we are travelling in a straight line, $\Delta\psi_l(kh) = \Delta\psi_r(kh)$. From this, we can estimate the number of ticks for the encoder:
     $0.033m = 2\frac{10m}{2\cdot\Delta\psi(kh)} \longrightarrow \Delta\psi(kh) = \frac{10m}{0.033m}rads \cdot \frac{4096}{2\pi}\frac{ticks}{rads}$ = **approx. 197545 ticks from each wheel**

- **Wheel Separation Experiment:** From the specs again, the wheel separation should be $b = 143.5mm$ and radius remains $r = 33mm$. Also, since the robot is in pure counter-clockwise rotation, the left wheel and right wheel should be moving the same distance but in opposite directions, so $\Delta\psi_l(kh) = -\Delta\psi_r(kh)$. Also, we defined in the experiment that the number of full rotations, $N = 10$. Using this, we can calculate the ticks of the wheel encoder:
  $0.1435m = \frac{0.033}{2}\frac{2 \cdot \Delta\psi(kh)}{(10) \cdot 2\pi} \longrightarrow \Delta\psi(kh) = \frac{0.1435m \cdot 20\pi}{0.033m} rads \cdot \frac{4096}{2\pi}\frac{ticks}{rads}$ = **approx. 178114 ticks from each wheel**
- Note, that the right wheel should record 178114 ticks and the left wheel should record -178114 ticks.

3. **Identify at least two sources of uncertainty or bias that could make your answer differ from the factory calibration:**

   - **Source 1 - Wheel slip:** If the wheels slip in either experiment, more ticks will be recorded than what should have been recorded, resulting in a smaller calculated radius r and a larger wheel separation b. This could be mitigated by conducting these experiments at low speeds and a surface with a sufficiently high coefficient of friction both of which reduce the possibility of wheel slippage.
   - **Source 2 - Errors in our kinematic vehicle model:** Our kinematic model as outlined in the notes of Lecture 14, was obtained assuming a 0 mean Gaussian distributed noise that was then linearized about the mean. First, it is possible that the uncertainty is not distributed as a Gaussian. Further, during linearization, some simplifications were made to eliminate the product of small angle terms. These error terms are eliminated since we have an external environment that accurately measures the position of the robot (ie. performing loop closure).

3 Construction of Occupancy Grid Map

1. **Mapping Algorithm**: The below sequential process is implemented for building the occupancy grid map in figure 1:

   - For every range measurement r_i for laser scans from 0-360° at a given robot pose, the obstacle position (x_{obs}, y_{obs}) in the inertial frame is given by:
     $x_{obs} = x_{robot} + r_i * \cos(\psi_{robot} + r_{angle})$
     $y_{obs} = y_{robot} + r_i * \sin(\psi_{robot} + r_{angle})$
     where (x_{robot}, y_{robot}) is the robot's inertial position, ψ_{robot} is the robot's yaw and r_{angle} is the angle r_i makes from the robot's forward axis
   - We map (x_{robot}, y_{robot}) and (x_{obs}, y_{obs}) to grid positions (x_{robot_map}, y_{robot_map}), (x_{obs_map}, y_{obs_map}) for a map of size (600, 600)
   - Grid locations that lie in a straight line from (x_{robot_map}, y_{robot_map}) to (x_{obs_map}, y_{obs_map}) are decremented with $\beta = -1$ (obstacle-free) and last three locations are incremented with $\alpha = +4$ (obstacle). This forms the Log_Odds_Map
   - Probabilities are recovered Log_Odds_Map and scaled from 0-100 before publishing it to the topic /map

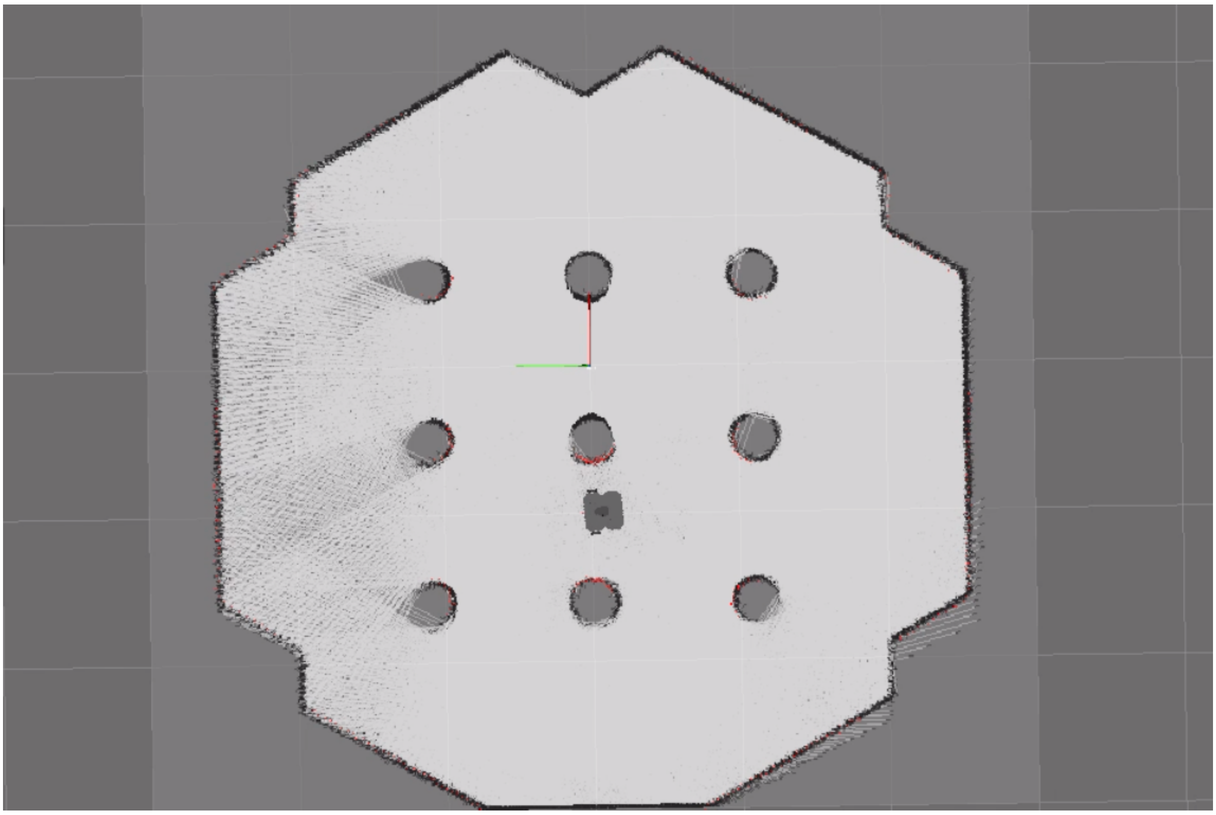

Figure 1: Final Occupancy Grid Map

2. **Source of Error**: The pose of the robot is estimated using wheel odometry, which is a *dead reckoning* technique. Over a period of item, the error in the pose estimates accumulates and drifts away significantly from the ground truth pose. Hence, the grid mapping also worsens with time.

3. **Error Correction**: We can use SLAM to correct for pose estimates at regular intervals with respect to the map already built. Pose estimates from wheel odometry will be used for short horizon and periodically corrected with the current map.