**HW4**

**Chenghao Lin & Naian Tao**

Task A:

1. Explain algorithm briefly with help of flowchart or pseudo-code. [4]

Initialize ROS

Set initial robot position

Define control parameters

Create robot control publisher and LIDAR Odom data subscriber

while time\_elapsed < 120 seconds:

Record robot position

Receive and Filter LIDAR data to ignore specific angles and distance

Adjust forward velocity based on obstacle presence in a 1-meter radius

Calculate the left and right boundaries of obstacles

If an obstacle is detected on the right:

Adjust angular velocities based on the angle of the boundary to turn left

If an obstacle is detected on the left:

Adjust angular velocities based on the angle of the boundary to turn right

Modify angular velocity based on distance between the obstacles and robot

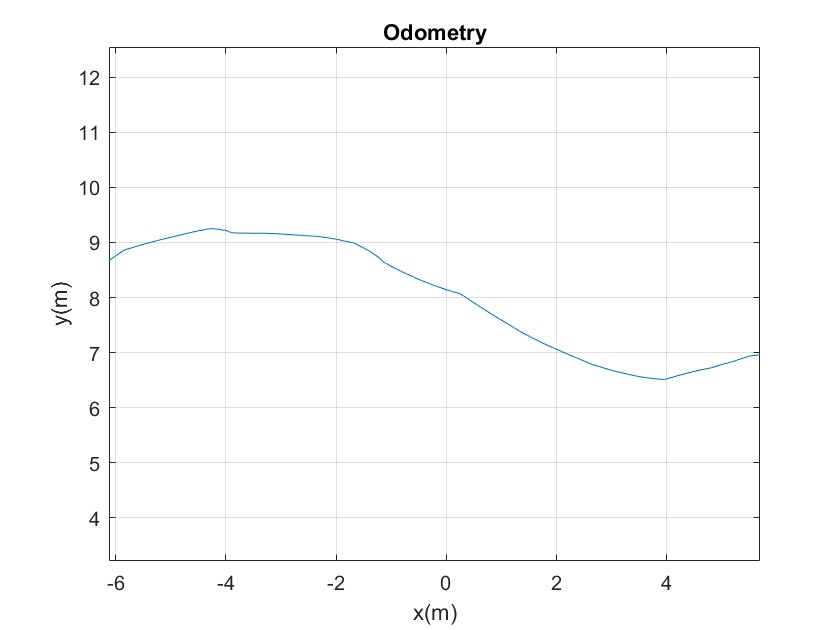
Publish robot control commands (linear and angular velocities)

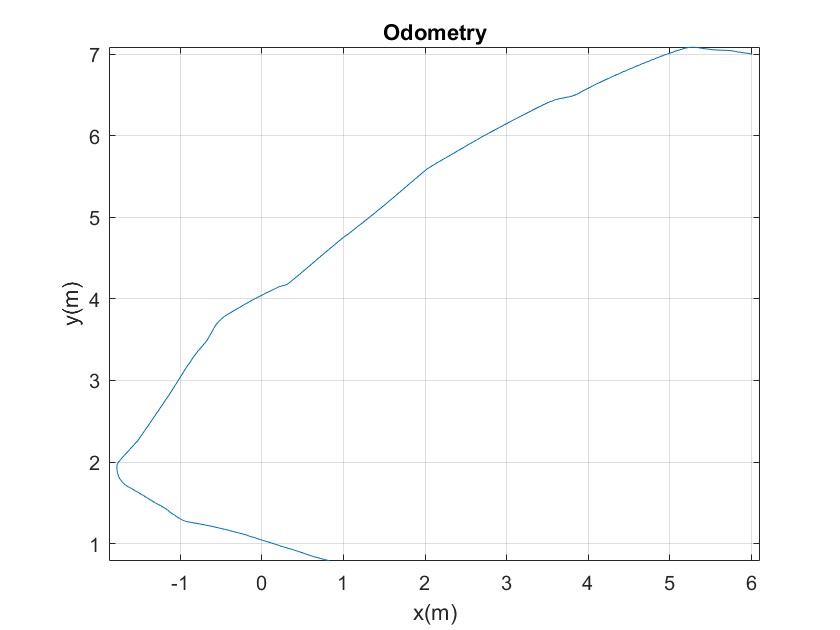
Shutdown ROS

Plot robot trajectory

1. Path plot [2]

1.75m:



3.5m:

3. Path length [2]

LIDAR 1.75m: 12.5878m

LIDAR 3.5m: 13.6464m

4. Video [2]



5. Focused observations (or comments) on the two experiments and comparisons between robot behavior. [4]

In our algorithm, if an obstacle is detected within one meter of the robot, the Turtlebot will execute a command to reduce its speed by half to ensure it can avoid the obstacle safely in critical situations. So, we can conclude that, within the same amount of time, the longer the distance the robot travels during obstacle avoidance, the safer the robot is, as it means it is farther from the obstacle. Calculations show that when the lidar has a range of 3.5 meters, the distance traveled is longer compared to a range of 1.75 meters. This suggests that a 3.5-meter lidar can detect obstacles earlier, allowing the robot to make avoidance decisions sooner.

We believe that directly performing a fixed-angle rotation after detecting an obstacle is not advisable because it may result in the robot still having obstacles in its path after the rotation, rendering the avoidance ineffective. To address this, we have developed a function to detect the boundary points of obstacles and use a coefficient (k) to control the smoothness and speed of rotation. We can deduce that when larger obstacles are in front of the robot, the rotation speed is faster until there are no obstacles directly in front of the car. However, it's important to note that this algorithm is not perfect, and we have implemented special handling when no boundary points are detected. And by comparing the plots, it is clear that, except in the case of emergency turns with obstacles on both sides and a wall in front in the 3.5m lidar scenario, the obstacle avoidance trajectory is smoother with a 3.5m lidar compared to a 1.75m lidar.

Task B:

1. Explain algorithm briefly with help of flowchart or pseudo-code. [4]

Initialize ROS

Set initial robot position

Define control parameters

Create robot control publisher and LIDAR Odom data subscriber

while time\_elapsed < 1200 seconds:

Record robot position

Receive and Filter LIDAR data to ignore specific angles and distance

Adjust forward velocity based on obstacle presence in a 1-meter radius

Calculate the left and right boundaries of obstacles

If an obstacle is detected on the right:

Adjust angular velocities based on the angle of the boundary to turn left

If an obstacle is detected on the left:

Adjust angular velocities based on the angle of the boundary to turn right

Modify angular velocity based on distance between the obstacles and robot

Publish robot control commands (linear and angular velocities)

If robot exists the U-shape obstacle:

Using a proportional controller to control the angular velocity to return to the initial point

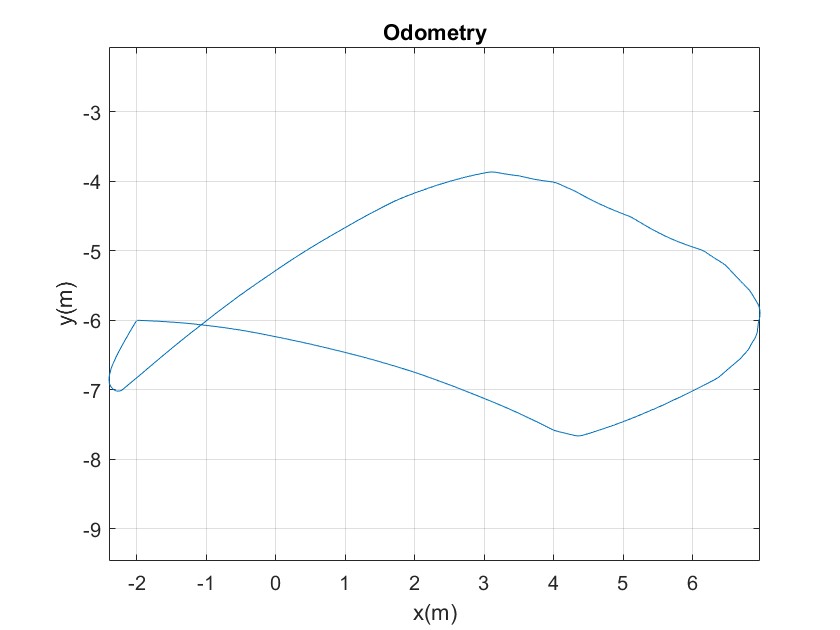
Break the loop

Shutdown ROS

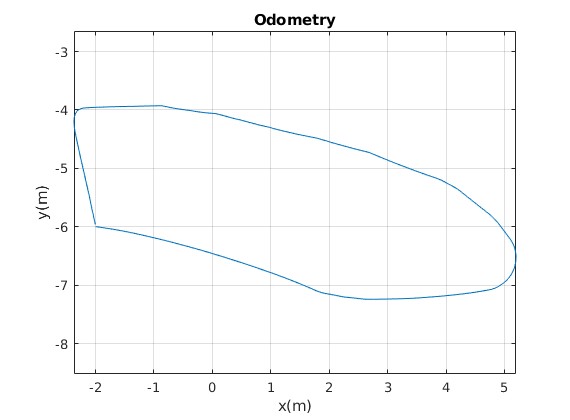
Plot robot trajectory

1. Path plot [2]

1.75m:



3.5m:



1. Video [2]



4. Focused observations (or comments) on the two experiments and comparisons between robot behavior. [4]Since our code has a directional preference, after discussions with the professor, we are allowed to return to the starting point after exiting a U-shaped obstacle.

In the video, you can observe that with a lidar range of 1.75 meters, in the early stages of navigating the U-shaped trap, the robot is almost tightly attached to the obstacle. However, with a lidar range of 3.5 meters, the robot can detect obstacles earlier, allowing it to make turns sooner. In other words, the 3.5m lidar provides a greater distance from obstacles, making it safer, while the 1.75m lidar is the opposite.

Simultaneously, we also recorded the distance traveled by the car. We assumed that getting closer to obstacles and making slower turns would result in a longer travel distance. The data has also confirmed this (1.75m:21.95m, 3.5m:18.04m ).