Lab 3 Report: Wall Follower

Team 11

Luis De Anda
Suchitha Channapatna
Amber Lien
Prince Patel

6.4200

March 14, 2025

1 Introduction

Autonomous navigation in complex environments is an essential skill for robots that requires robust perception and control systems. Many autonomous vehicles use LiDAR sensors because they provide a stream of environmental data, enabling the detection of obstacles and the estimation of their distances. This lab explores the challenges of processing raw LiDAR data to enable effective wall-following behavior on a robotic mini racecar.

Building on our simulated wall-following algorithms, this lab focuses on deploying and refining a wall-following controller on the physical racecar. There are two primary goals: first, to accurately parse LiDAR data to extract relevant environmental information, and second, to use this information to guide the racecar along a wall at a desired distance and velocity. Additionally, a safety controller will be implemented to prevent collisions with unexpected obstacles, a new problem not seen in simulation. The motivation behind this lab is to bridge the gap between simulation and real-world robotic systems, addressing the complexities of operating an autonomous vehicle in a dynamic environment. This work contributes to the broader understanding of robust control strategies for autonomous navigation, highlighting the importance of sensor data processing and real-time decision-making in robotics.

2 Technical Approach

As described above, the goal for this lab is two-fold: to follow a wall successfully while having a comprehensive safety mechanism to prevent accidents.

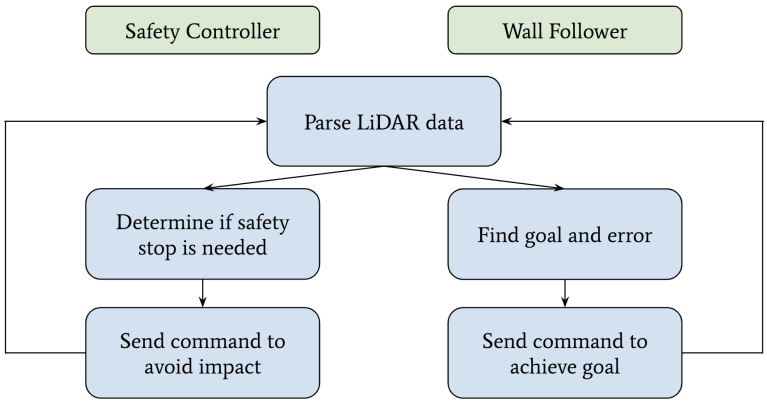


Figure 1: The overall process taken to complete the lab. We parse LiDAR data, using the information to stop or calculate our projected trajectory.

We divided this problem into subproblems, tackling each independently (a choice reflected in our source code) as shown in Figure 1.

To achieve both goals (each implemented as its own module), perceiving the environment is of upmost importance. On the racecar, this involves reading and parsing information from the LiDAR scanner onboard. Once the robot has an understanding of its environment, each module makes a different calculation based on its purpose. The safety controller uses the distance to a front facing obstacle to determine whether stopping is necessary and accordingly sends a "stop driving" command to the racecar. The wall follower calculates its current position from the wall it's following and the orientation relative to that wall, using these errors to send steering controls. Both modules run simultaneously and continuously, limited only by the rate of measurements from the LiDAR. This architecture is supported by the existence of a mux which establishes the hierarchy of commands for our controller. Safety commands are considered higher priority than wall-following drive commands, allowing the safety controller to override the wall follower when needed.

2.1 Safety Controller

The primary goal of the safety controller is to avoid impacts with obstacles. By making the assumption that the racecar will only hit obstacles head-on, we are able to reduce the obstacle search area. This assumption is supported by the restricted steering angle and the lack of LiDAR scan data at the rear of the racecar.

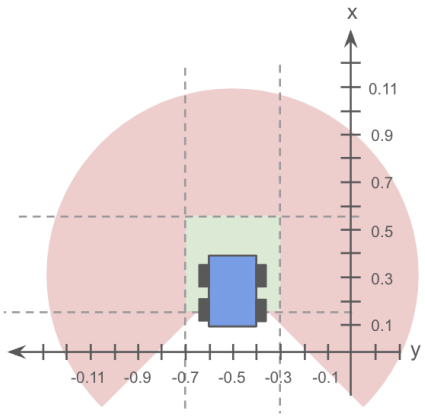


Figure 2: The angle ranges considered in calculations in the safety controller. The red region denotes possible ranges from which LiDAR data can be obtained. The green region denotes the portion used in our calculations for the safety controller.

Not all LiDAR data is relevant for the safety controller. We only aim to stop if there is an obstacle directly in front of the racecar. Figure 2 depicts the range of LiDAR data considered. Notably, this should not impact the accuracy of wall following.

The lookahead distance is defined as the distance d_l between the LiDAR frame's origin and the furthest point (in cartesian absolute distance) we will consider in our search for potential obstacles. We recognize that a faster moving racecar should attempt to break sooner than a slow moving racecar when in danger of colliding with an obstacle. For that reason, within our safety controller module, d_l is defined as $max(d_{l\_base}, d_{l\_base} * v)$ where d_{l_base} is a user defined baseline lookahead distance and v is the current speed of the racecar. d_{l_base} is a tunable parameter, and we have chosen 0.25 meters as a reasonable baseline. By taking the maximum, we ensure a large enough lookahead distance for velocities less than 1 and for greater velocities more typical of racing conditions.

We follow the below steps to determine conditions in which we must safely stop.

1. Convert the LiDAR point cloud data from polar to Cartesian coordinates in (x, y).
2. Filter out all data that is more than 0.2 meters to the left or right of the racecar (in the y-axis).

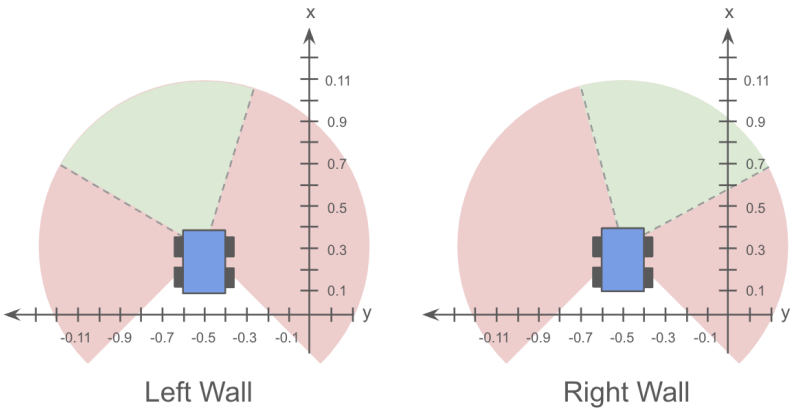


Figure 3: The laser ranges considered when estimating the wall for the wall follower. The red section depicts the range from which LiDAR data can be obtained and the green section depicts the range of information we use to estimate where the wall is located. Note the difference in the ranges used when following the left wall versus the right.

3. Filter out all data that is farther than the lookahead distance or behind the robot (in the x-axis).
4. Check if the minimum point is closer than the set buffer distance. If so, send a stop command.

2.2 Wall Follower

The primary goal of the wall follower is to allow the racecar to keep at some fixed distance from the wall (either on its left or right side) while traveling at a desired velocity.

We define the following terms used in our wall follower:

x = actual distance from wall
D = desired distance from wall
e = error between desired distance and actual distance
θ = current heading

Because we always aim to keep a fixed distance from the wall, a PD controller was the most intuitive option for reducing error. The main features of a PD controller include proportional and derivative gains, used to adjust how strongly

and how quickly the racecar responds to errors from a set point.

We employ two separate PD controllers. One minimizes distance error between the current racecar position and desired distance from the wall. The second minimizes angle error between the heading of the racecar and the position of the wall, ensuring that the robot always attempts to drive parallel to the wall.

The wall follower has seven parameters to tune. The first three are used to determine the robot's trajectory by estimating the wall's location: lookahead distance, front distance, and the view angle for parsing LiDAR data. The last four are involved in controlling the robot: K_p, K_d, K_{pa}, and K_{da}.

The view angle θ_v defines the range in radians to the left and right of the LiDAR frame's origin considered in wall calculations. θ_v is asymmetrical, dependent on which wall the racecar is following. As defined from the front of the robot (x-axis), angles less than 60 degrees on the side of the wall and angles less than 15 degrees on the opposite side are considered. This reduces the influence of noise and other objects on the opposite wall.

As with the safety controller, the lookahead distance is the distance d_l between the LiDAR frame's origin and the furthest point we'll consider in estimating our wall. The front distance d_f defines the distance at which the robot will begin to more heavily weight the points in the front, preparing the vehicle to take a turn at a closed corner. We make use of the fact that the safety controller already determines the distance to the closest object in front of the racecar, subscribing to this published information. The point at which the racecar prepares to turn is dependent on the desired distance from the wall. For this reason, we consider the racecar to be in "turning preparation" when $d_f \leq 2 * D$.

We prepare for a turn by manipulating the racecar's perception of the wall, creating a sloped line that encourages our robot to turn. We first parse LiDAR data to only consider points less than or equal to 5 degrees from the side of the wall we are following and less than or equal to 15 degrees on the opposite side. We then weight these points more heavily in forming our line of best fit by adding them again to our initial points of consideration. This skews the line towards the wall in front of the robot, encouraging a steeper turn.

In a normal, non-turn scenario, these extra points are weighted equally and the line of best fit is formed using points gathered from the LiDAR data in the range of θ_v. From this line, we compute the current distance from the wall x and error e using $|b|/\sqrt{m^2+1}$ where b is the internet of the line of best flip and m its slope.

The angle to the wall is defined as the arctangent of the slope of the line of best fit. We compute derivative terms as the difference between the current and previous state, assuming small time steps between each call of the callback function.

The overall control input angle is a combination of both PD controllers defined by the following equation:

$$\text{steering} = K_p * e + K_d * \frac{\mathrm{d}e}{\mathrm{d}t} + K_{pa} * \theta * K_{da} * \frac{\mathrm{d}\theta}{\mathrm{d}t}$$

We followed a standardized procedure for tuning these parameters. We tuned the distance and angle PD controllers independently for the first iteration. We kept the derivative term constant, $K_d = 0$, and increased the proportional constant K_p until the racecar seemed to exhibit fast, oscillatory motion. Then, we increased K_d incrementally until the oscillations were damped and the racecar could travel in a straight line. We repeated the process for K_{pa} and K_{da}, observing how parallel the racecar was to the wall. We then combined both controllers and continued to adjust the parameters based on the observed behavior (e.g. too aggressive, too slow, etc.).

3 Experimental Evaluation

We evaluated the performance of our safety controller and wall follower with a combination of qualitative and quantitative metrics in a variety of situations. We tested the safety controller by placing objects in front of the racecar and by having individuals walk in front of it to block its path. Additionally, we set up trials such that we could analyze performance at a speed of 1.0 m/s, 1.5 m/s, and 2.0 m/s while following the left wall and right wall.

For quantitative analysis, we made use of rosbags. We recorded all messages across topics of interest while running the safety controller and wall follower. The data passed along these topics was then extracted to form plots and statistics.

3.1 Safety Controller

We ran repeated trials to determine how many times the safety stop would successfully override drive commands in a dangerous scenario. In the first set of trials, the obstacle obstructing the racecar was static (e.g. a wall), and in the second, the object was dynamic (e.g. a person walking). In both trials, we drove the racecar between two fixed points along a straight wall.

In the fixed obstacle setting, we placed a box large enough to prevent the vehicle from moving around it before we started driving. Then, we ran the wall follower, observing if the safety stop engaged near the wall. Over 10 trials, we obtained the following results. Note that in the case of collisions, we considered the distance to the wall to be 0.0 meters.

In the dynamic obstacle setting, we began to walk in front of the robot as it reached the halfway mark while we were positioned a quarter away from the end.

Speed	Times Stopped	Average Distance
1.0 m/s	10	0.24 m
1.5 m/s	9	0.22 m
2.0 m/s	8	0.20 m

Table 1: Number of times the racecar successfully stopped during 10 trials and the average distance at which it stopped is recorded in the above table. A successful stop is defined as not touching the box.

We obtained the following results over 10 trials. Note again that the distance in collisions was considered to be 0.0 meters.

Speed	Times Stopped	Average Distance
1.0 m/s	8	0.20 m
1.5 m/s	6	0.18 m
2.0 m/s	7	0.13 m

Table 2: Number of times the racecar successfully stopped during 10 trials and the average distance at which it stopped is recorded in the above table. A successful stop is defined as not touching the person walking in front of the racecar.

Our wall follower successfully stopped in most situations. Notably, in all tests, we observed the vehicle attempt to turn before stopping. This is expected behavior: our vehicle attempts to avoid hitting obstacles, only stopping once it's realized it's too close to an object. Additionally, the safety controller exhibited fewer successes in the dynamic obstacle setting compared to the static. Due to the narrow range in which we considered LiDAR data and the wall following algorithm attempting to turn the racecar away from the obstacle to avoid it, this is expected as fewer points were viewed as closer, causing the safety controller to fail.

3.2 Wall Follower

In order to maintain consistency between tests while testing the wall follower, all of the following trials involved driving the racecar between two fixed points as it followed a wall and turned a corner. We varied the speed and wall followed during these tests.

We use the following metrics to quantitatively determine the wall follower's performance:

x = actual distance from wall
D = desired distance from wall

e = error = $|D - x|$
a = accuracy = $\max(0, 1 - \frac{e}{D}) \cdot 100$

A portion of our analysis also includes a comparison between using a single PD controller to reduce distance error and using our double PD controller to reduce both distance and angle error, results for which are depicted in Figure 4.

Note the increased variance in the top graph compared to the bottom, indicating that using a double PD controller resulted in better performance by reducing oscillatory behavior. The reduced total distance error and increased accuracy in the double PD case validates our choice to control both distance and angle.

Figures 4 and 5 on pages 10 and 11 contain plots displaying the error computed over time for each of our scenarios. These trials had the same location, start point, and end point.

Overall, the deviation from our desired path remains near zero. However, we notice that towards the end of each trial, the error increases drastically. This may be due to the fact that our testing environment's end position was near tables within the lookahead distance that we could not move. In the future, we aim to resolve this testing inaccuracy by finding a more isolated environment.

Additionally, the trials in which the racecar was following the right wall display some consistent offsets in the beginning. Though we attempted to start the racecar at exactly the desired distance from the wall, human error and protrusions in the wall may have contributed to the offset. To better understand this result, we would run additional tests in which the car follows the right wall.

Finally, measures of total and average error overall seem quite pessimistic. This may be because we compute error based on the location of the wall, which is a line of best fit between LiDAR points. As is the case when using any sensor, noise may cause the line of best fit to not perfectly represent the true wall. Our measure of error does not capture this issue, and we would likely find an additional metric to analyze error in the future.

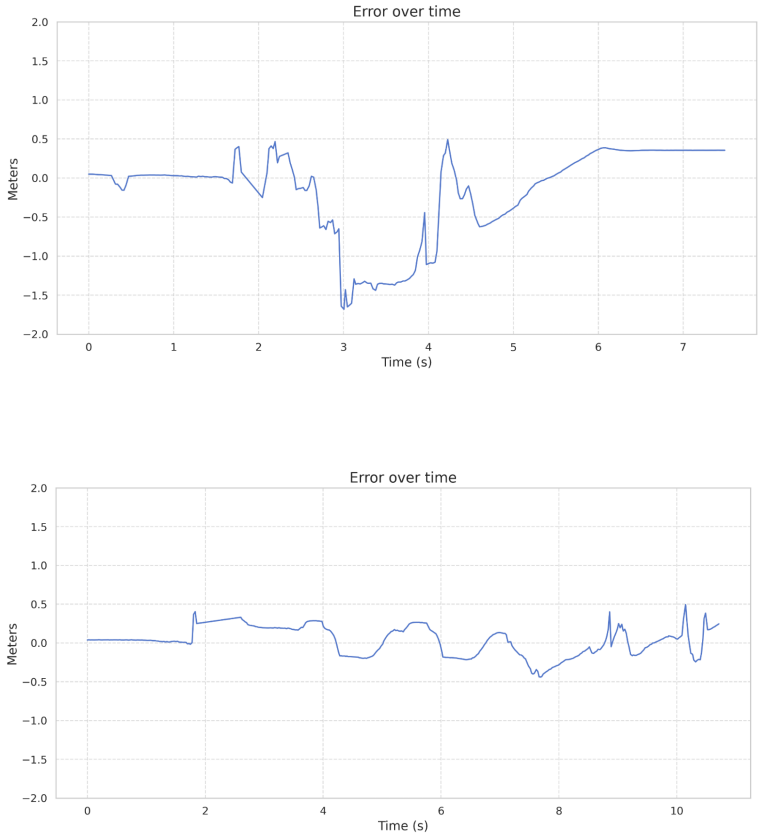
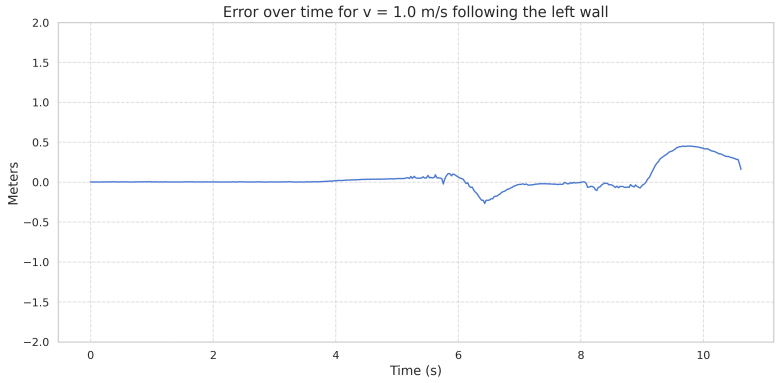
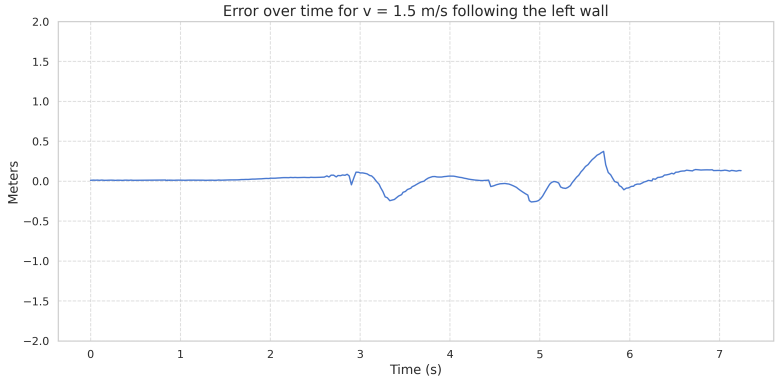


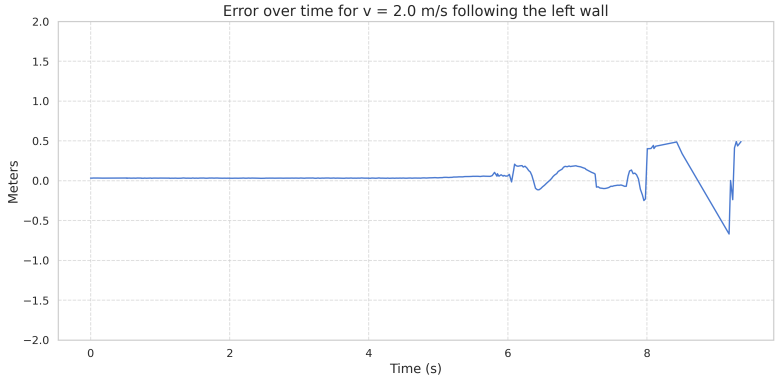
Figure 4: A comparison between the accumulated error of the racecar's deviation from the desired path using a PD controller with distance error and a PD controller with distance error and angle error. The top graph depicts the performance of the single PD controller. The bottom graph shows the performance of the double PD controller.



(a)



(b)



(c)

Figure 5: The figures depict the robot's error from the desired path while moving at three different velocities, 1.0 m/s, 1.5 m/s, and 2.0 m/s, following a wall on its left side. Overall, the error remained close to 0.0 meters, with deviations occurring during turns in which our metric for determining error did not reflect the ground truth.

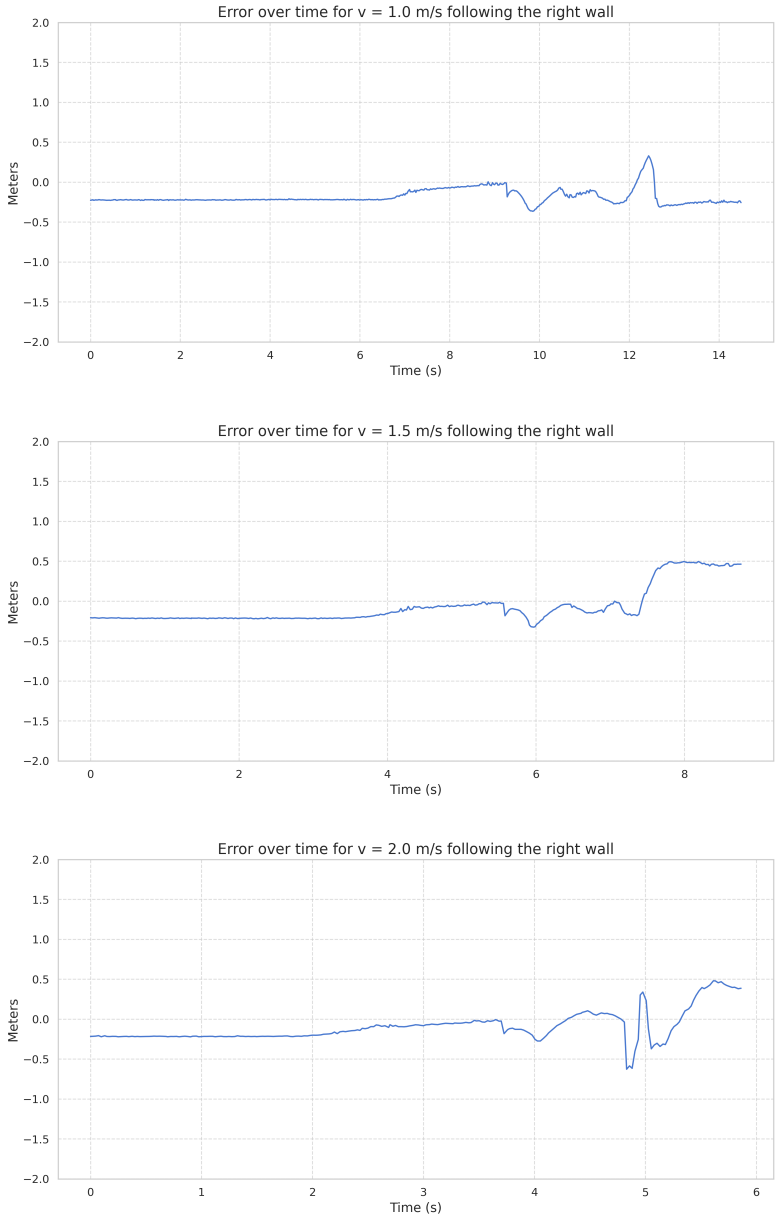


Figure 6: The figures depict the robot's error from the desired path while moving at three different velocities, 1.0 m/s, 1.5 m/s, and 2.0 m/s, following a wall on its right side. Note that in all three graphs, there is negative deviation at the beginning of our trials, indicating that our starting position is too close to our wall. Otherwise, deviations remain relatively constant with larger changes occurring during turns in which our metric for determining error did not reflect the ground truth.

4 Conclusion

We were able to meet the main objectives of this lab. We demonstrated using quantitative and qualitative methods that the racecar can follow a wall at a fixed distance, complete left and right turns, and avoid obstacles while moving in various environments at different speeds. Through the synchronization of the wall follower and safety controller, the racecar will always attempt to navigate around obstacles before stopping if the distance to obstacles gets too close.

Initial testing with the safety controller found that the vehicle continued to roll after the motor turned off, causing it to bump into walls and people moving in front. This issue scaled with the velocity of the robot prior to stopping. This issue has since been resolved by updating the vesc settings, enabling reverse braking which successfully stopped the racecar in a shorter time frame.

In the immediate future, we will focus on reducing distance error during sharp turns. To better evaluate how well our control performs, we will also develop a test metric that uses LiDAR points as ground truth rather than the interpolated line representing a wall.

The progress made in this lab will be the foundation for more complex driving and navigation scenarios. Throughout future labs, we will make use of the safety controller to protect hardware when testing. We also expect to modify our wall follower to fit other control-specific tasks, such as line following.

5 Lessons Learned

TODO: Insert our individual reflections here!