Homework 1 - Obstacle Avoidance

Arnav Iyer, Hayley LeBlanc, and Josh Hoffman
The Vroom Vroomers
CS 393 - Autonomous Robots

September 20, 2021

1 Math questions

Question 1

The point on the car that traces an arc of maximum radius is the outer front corner. Let $e = \frac{l-b}{2}$ be the distance from the front axel to the front edge of the car. This radius is $\sqrt{(r + \frac{w}{2}) + (e+b)^2}$.

It is possible that if the back of the car extends out far beyond the rear axel, the end of the car could trace a larger radius. However, for the purposes of this question and in our implementation, we assume this is not the case and that the distance from the rear axel to the back end of the car is small enough that this does not happen.

Question 2

The point on the car that traces an arc of minimum radius is the point on the inner side of the car where the turning radius intersects with the car. The radius to this point is $r - \frac{w}{2}$.

Question 3

Point p will hit the car on the inner side of the car as the car drives forward if the distance between p and the center of turning is between the radius traced by the point described in Question 2, given by $r_s = \frac{w}{2}$, and the radius traced by the inner front corner of the car, given by $r_m = \sqrt{(l+w)^2 + (\frac{w}{2})^2}$. The distance from p to the center of the turn is $\sqrt{(r-y)^2 + x^2}$, so p will hit the inner side of the car when $r_s < \sqrt{(r-y)^2 + x^2} < r_m$.

Question 4

Point p will hit the front of the car as the car drive forward if p sits between the radius traced by the inner front corner and the radius traced by the outer front corner. The radius traced by the inner front corner is $r_m = \sqrt{(l+w)^2 + (\frac{w}{2})^2}$ and the radius traced by the outer front corner is $r_\ell = \sqrt{(r + \frac{w}{2}) + (e+b)^2}$. p will hit the front of the car when $r_m < \sqrt{(r-y)^2 + x^2} < r_\ell$. Here, x, y is the center of turning.

Question 5

Under the assumptions stated above, p will never hit the car on the outer side as the car drives forward. The largest radius traced by any point on the car is traced by the outer front corner, which means that any point that is going to hit the car would hit the front of the car at that corner before it could hit the outer side.

Question 6

The maximum free path that the car can drive before hitting p depends on whether p will hit the car's inner or front side.
If p will hit the front of the car: When p hits the car, there is a right triangle with one leg starting at the center of the turn and ending at the base_link of the car with length r, and the other leg parallel to the wheelbase ending at p with length $\frac{l+b}{2}$. Let r' be the distance from p to the center of the turn. Let $\beta = \arcsin(\frac{\frac{l+b}{2}}{r'})$, i.e., the angle at the vertex of the triangle that is on the center of the turn. This triangle is similar to a triangle constructed in the same way at the initial position of the car.

Call θ the angle between the initial position of base_link and the position of base_link when the car hit p. We'd like to find $\alpha = \theta - \beta$, as this α angle will give us the angle of the actual turn the car will make before hitting p, from which we can obtain the free path length. θ is known - we can calculate it based on the relative position of p to the center of the turn.

Then, $\alpha = \theta - \beta$ gives the angle of the turn that the car actually completes before hitting p. The length of this arc (the free path length) is $r\alpha$.
If p will hit the side of the car: This approach is roughly the same as if p hits the front of the car, except that β is calculated differently. This time, we consider a right triangle constructed with a hypotenuse starting at the center of the turn and ending at p with length r', and one leg starting at the center of the turn and ending at the point described in question 2 with length $r - \frac{w}{2}$. Then, $\beta = \arccos(\frac{r-\frac{w}{2}}{r'})$. We find θ and α in the same way as in the other case, and the free path length is again $r\alpha$.

Question 7

We derive the minimum stopping distance from one of the four fundamental kinematic equations: $v_f^2 = v_i^2 = 2ad$ where v_f is the final velocity, v_i is the current velocity, a is acceleration (which is negative here, since we are decelerating), and d is distance. We know that we are decelerating to 0, so we set $v_f = 0$. We simplify to $d = -\frac{v^2}{2a}$ to obtain the minimum stopping distance d.

2 Question 1: Parameters

Our algorithm required the wheel base distance, the width of the car, the length of the car, the current curvature, the current velocity, the latency, the number of proposed curvatures to consider at any given time, the max curvature value, and the front safety margin, and a side safety margin. Noticeably missing, we did not use the track distance of the car.

Additionally, to execute the J-Turn maneuver, we have a variable titled $FLRR\_DURATION$ used to describe what state we are in for the J-Turn, we have the J-Turn velocity parameter, and the J-Turn curvature value which are the present values to turn along and the speed to drive at when executing the maneuver.

All of the length and width measurements are used in calculating the free-path for a specific arc. They are also used when checking if the car will collide with any point along a proposed curvature, and also used when deciding if a point will collide with the front or the side of the car. The current velocity is used to determine what the next velocity should be based on the calculated free path length in our 1D TOC method. The front of side safety margins while implicitly included in the length and width of our car, have an impact in calculating the clearance and also it impacts the free path length calculations which in turn impacts our 1D TOC calculations thus ensuring that we do not collide with a wall upon breaking. We also used the current velocity, the latency parameter, and the current curvature to forward predict the motion of the car during latency prediction and when updating the point cloud. Finally, the curvature number and the max curvature value represent the number of curves to consider in between the highest and lowest allowed curvature values, and the tightest curve that we will allow the car to take either left or right respectively.

3 Question 2: Parameter Tuning

We tuned parameters by utilizing the simulator and by add points to the visualization. For example, we visualized the point cloud latency forward prediction to help with tuning. We also used printouts while running the car to make sure that certain calculate values were aligned with expectations. Finally, we simply ran the car to check for performance and tuned it accordingly.

4 Question 3: Challenges

It was a challenge to come together as a time and coordinate our schedules. Another challenge was trying to distill the information shared in class and parse out what was truly necessary in order to get the car to run in the way we desired versus what was auxiliary information that was intended for our broader understanding of autonomous robots.

Furthermore, it was a challenge to figure out the branches and quadrants that we needed to reason over in order to properly properly compute the angle (in radians) needed to compute the free path length. Finally, it was a challenge to properly tune the reward function.

We overcame the scheduling issues by being comfortable with the fact that there may be times when only two of us could be there, and we always trusted the third to put in work another time. We overcame the knowledge distillation through open and candid conversations amongst ourselves. The mathematical challenges were overcome by stepping away from our computers and working it out on a whiteboard. Finally, the challenges of tuning were solved by just running it over and over whether in the real world, in simulation, or by joysticking the robot to best intuit the solution and hyperparameters.

5 Question 4: Future Improvements

There are two future improvements. First, we do not explicitly throw out curves where the free path for that curve is less than the distance needed to decelerate to zero. We do this implicitly through rewarding curves with the longest free path, but we could update the code to explicitly throw out curves for whom we cannot decelerate in time if we took that free path.

The second improvement is that we could include some metric of the foreground being too cluttered or if the robot thinks that it would be too hard for the robot to navigate. Under this condition, the robot would automatically do a U-Turn if feasible.

Otherwise, we are rather proud of the work, and we believe that we have delivered everything requested in the brief.

6 Question 5: Github

See section 9.1 to find the link to the public repo.

7 Question 6

All group members contributed equally. We regularly met as a group and all discussed design, worked on code, and tested our implementation in simulation and on the robot together. Josh and Hayley did most of the work on the writeup, and Arnav implemented the J-turn functionality.

8 Extra Credit

We successfully completed both extra credit assignments. Please see section section 9.2 to see the video of the car navigating at a higher speed. Please see section section 9.2 to see the video of the car performing a J-turn.

9 Results and Code

Here we share the links to the video showing our final results as well as a link to our final public repo.

1. Final code base:
2. Video of obstacle avoidance, high speed avoidance, and both extra credit assignments: THE VIDEO

10 Conclusions

We worked hard, and achieved? You can be the judge of that, but I think our videos look pretty cool.