EE3EY4: Electrical Systems Integration Project
Lab 9: Autonomous Driving Using Virtual Separating Barriers

Instructor: Dr. Sirouspour

Aithihya Kompella - kompella - 400310794
Benji Richler - richlerb - 400296988
Shathurshika Chandrakumar - chands39 - 400315379
W. Tisuka Perera - pererw2 - 400318373

Date of Submission: April 12th, 2023

Note: navigation.py code is attached with this submission. This code has the completed equations referenced in the lab tasks.

Task: **Study the files "navigation_incomplete.py" and "params.yaml" and identify parts of the code that accomplish this step.**
The variable initialization "self.ls_fov=self.ls_len_mod*self.ls_ang_inc" in "navigation_incomplete.py" and the gap following parameters variables in params.yaml (safe_distance, right_beam_angle, left_beam_angle) help to define a field of view in front of the vehicle. These lines establish the 180 degrees in the front hemisphere of the vehicle.

Task: Explain the role function "preprocess_lidar" and it helps accomplish the objective of Step 2.
The function "preprocess_lidar" takes the data from a full 180 degree scan and adds the information to a new data array which can be processed to find the obstacles closer than the safe distance. The first index value of the new array will be a zero if the distance was closer than the maximum safe distance, i.e. it is part of an obstacle. The second index value of the new array indicates what angle that distance scan was at. If the distance is larger than the set maximum safe distance then the first index value will be that distance. If the distance is further than the lidar can detect, then the distance is simply set to the maximum lidar range. This process helps to indicate where the obstacles are, by marking their scans with a distance of zero.

Task: **Complete the code for the function "find_best_point" to compute the desired direction of movement according to the about formulation. Explain why this might be a better choice than the original furthest point in the largest gap.**
Choosing the furthest point in the largest gap might cause the heading direction to change very unpredictably and uncontrollably as the AEV moves forward. This may happen when the wall pattern that is within the AEV's field of view is not uniform, or if the measurements are noisy.

Furthermore, it is possible that a singular furthest point might be an outlier compared to other, closer points that are detected. However, calculating a weighted average of all the points within a largest gap means that it is less likely that the car will take as erratic of a path as it would if the furthest point were chosen.

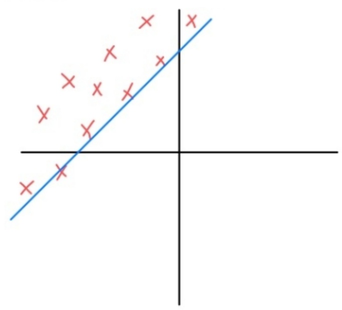
Task: **Derive this optimization formulation.**

To achieve greatest distance to the origin:

$$min_W \frac{1}{d_o} \rightarrow min_W \sqrt{W^T W} \rightarrow min_W W^T W$$

In order to make sure all the points in point set (P_i) are behind or on the line in reference to the base link frame, consider these two cases:

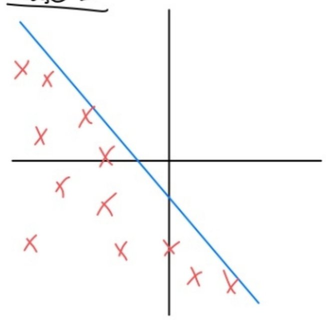
Case 1



↳ inequality: $y \geq mx + b$

↳ inequality constrains region of rational points, behind or on the selected line

Case 2



↳ inequality: $y \leq mx + b$

↳ m & b are -ve numbers

↳ inequality constrains region of rational points, behind or on the selected line

Case 1

$b \neq 0$

$$y \geq mx + b$$

$$\frac{y}{b} \geq \frac{mx}{b} + 1$$

$$0 \geq \frac{mx}{b} - \frac{y}{b} + 1$$

$$0 \geq \begin{pmatrix} \frac{m}{b} & -\frac{1}{b} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + 1$$

$$w = \begin{bmatrix} \frac{m}{b} \\ -\frac{1}{b} \end{bmatrix} \quad P_i = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$0 \geq w^T P_i + 1$$

Case 2

$$y \leq mx + b$$

$$\frac{y}{b} \geq \frac{mx}{b} + 1$$

inequality flips because b is -ve

$$0 \geq \frac{mx}{b} - \frac{y}{b} + 1$$

$$0 \geq \begin{pmatrix} \frac{m}{b} & -\frac{1}{b} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + 1$$

$$w = \begin{bmatrix} \frac{m}{b} \\ -\frac{1}{b} \end{bmatrix} \quad P_i = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$0 \geq w^T P_i + 1$$

same conditions!

Task: **Explain the role of each of these parameters in the self-driving algorithm:**

CenterOffset: determines how far away from the center of the two walls the vehicle should drive along

angle_bl/al & **angle_br/ar**: angles that define the left and right virtual barriers from LIDAR scans

n_pts_l: number of times to scan from the left side of the FOV to the centre

n_pts_r: number of times to scan from the right side of the FOV to the centre

k_p & **k_d**: gains within the proportional control system with velocity feedback (k_p is proportional gain, k_d is derivative gain). These gains control the transient characteristics of the system response

velocity_high: fastest velocity to go when comfortable with going fast and straight

velocity_medium: for when the steering angle is set between “angle_threshold_low” and “angle_threshold_high”

velocity_low: slowest velocity to go when traversing turns

angle_threshold_low: angle threshold where if angle is less than this value, speed is set to velocity_high

angle_threshold_high: angle threshold where if angle is greater than this value, speed is set to velocity_low

safe_distance: minimum distance to obstacle from vehicle for vehicle to not try to avoid

right_beam_angle: rightmost angle at which to begin scan of FOV

left_beam_angle: leftmost angle at which to begin scan of FOV

Task: **Add a Display of Marker type with the topic "wall_markers" to the "rviz" visualization environment. Study the code in "navigation.py" and explain how these markers are computed and what they represent.**

```
self.marker.points.append(Point(dl*(-wl_h[0]-line_len*wl_h[1]), dl*(-wl_h[1]+line_len*wl_h[0]) , 0))
self.marker.points.append(Point(dl*(-wl_h[0]+line_len*wl_h[1]), dl*(-wl_h[1]-line_len*wl_h[0]) , 0))
self.marker.points.append(Point(dr*(-wr_h[0]-line_len*wr_h[1]), dr*(-wr_h[1]+line_len*wr_h[0]) , 0))
self.marker.points.append(Point(dr*(-wr_h[0]+line_len*wr_h[1]), dr*(-wr_h[1]-line_len*wr_h[0]) , 0))
self.marker.points.append(Point(0, 0 , 0))
self.marker.points.append(Point(line_len*math.cos(heading_angle), line_len*math.sin(heading_angle), 0))
```

Task: Briefly report your observations on the impact of the parameters "k_p", "k_d" and "n_pts_l", "n_pts_r", and "safe_distance" on vehicle response

The markers are computed by using filtered LIDAR sensor data (filtered so as to remove noise) to obtain scanned points of walls. These points are then organized into clusters representing each wall. With this information, endpoints of the wall segments are calculated, and stored in a 3D points list, as seen in the figure above. In the rviz visualization, the wall segments calculated by the code are visualized as a series of lines. These line segments symbolize the walls sensed by the AEV.

Task: Briefly report your observations on the impact of the parameters "k_p", "k_d" and "n_pts_l", "n_pts_r", and "safe_distance" on vehicle response

k_p is a gain parameter relating to the natural frequency (w_n) of the system transfer function. This value impacts the transient characteristics of the system. An increase in k_p results in a lower settling time with the tradeoff of a larger percentage overshoot, i.e. faster but more unstable response. Conversely, a decrease in k_p results in a larger settling time, with a lower percentage overshoot, i.e. a slower but more stable response.

k_d is another gain parameter that relates to the natural frequency and is proportional to the damping ratio of the system's transfer function. An increase in k_d results in an increase in the damping ratio, resulting in a more overdamped response. This means system response is slower, has a lower settling time, and is also more stable (less overshoot). Conversely, a decrease in k_d results in a decrease in the damping ratio, resulting in a more under-damped response. This means the system response is faster, but with a higher settling time and less stability (more overshoot).

n_pts_l and n_pts_r control the number of LIDAR scanned points to the left and right of the AEV that the algorithm uses to create the virtual barriers. Increasing these points increases the accuracy of the vehicle's obstacle detection. Conversely, decreasing, reduces the accuracy of the algorithm.

The safe_distance parameter controls the minimum measured distance of an object scanned that allows it to be considered a gap. An increase in safe_distance parameter, means the objects at a lesser distance from the vehicle are considered obstacles. This means greater ability to avoid obstacles as foresight is enhanced, but with less options for the car to maneuver to. Conversely, a decrease in safe_distance means that objects at a larger distance are considered part of the gap. This means more number of direction options for the car, while also increasing the probability of collisions with sustained driving (decreased foresight).

Task: **Make further adjustments to the controller parameters as you see necessary to achieve a satisfactory response. Briefly report on your observations in experiment.**
The controller parameters adjusted were safe_distance, k_p, k_d n_pts_l, n_pts_. The parameter safe_distance was increased from 2 to 5, in order to increase the gap maintained from the virtual walls. This allowed greater maneuvering of the AEV while driving, and helped avoid collisions. The k_p and k_d parameters, which controlled the transient characteristics of

the control system (percentage overshoot, settling time, etc.), were left as is. This is because hanging these parameters gave undesirable instability in the system response. n_pts_l and n_pts_r were also left as their default values, as they were large enough so that enough scanned points were considered for the algorithm, while also being small enough not to increase computation time greatly.

Task: **Reflect on the operation of the self-driving control algorithm and suggest potential ways for improving it.**
While the self-driving control algorithm works well, it does not come without its flaws. For example, there are times where the AEV collides with a wall, but is unable to reverse and resume its course. It may be beneficial to implement an algorithm that would allow the AEV to reverse if there is a very long obstacle blocking its course entirely.
Another thing with the AEV is that when it is driven, it swerves left and right repeatedly while scanning. While it is much less noticeable when the AEV follows the algorithm from this lab as opposed to the last, it might be beneficial to make it so that the vehicle does not swerve back and forth as much. It was observed that there were times when when the car was zig-zagging it would crash into obstacles that were too close to this. This may perhaps be resolved by expanding the range that the LiDAR can scan. Also, it may be beneficial to have the AEV follow one wall in particular instead of both, as this will help stabilize the part of the car.

-END-