Path Planning with Collision Avoidance for Highway Operations with Obstacle Tracking

Abstract:

In this project, a path planning algorithm with collision avoidance for highway operation based on the modified rapidly exploring random trees (RRT*) and Linear Quadratic Regulator (LQR) is developed. The LQR is used to generate an optimum path connecting the proposed vertices from the RRT* algorithm and to evaluate the cost-to-go of these paths. The performance of the proposed LQR-RRT* algorithm is evaluated on two simulation scenarios: a straight segment of the highway and a curved segment of the highway. The LQR-RRT* was able to plan a path to safely overtake the moving vehicle in front by using a lane change maneuver.

Introduction:

According to the motor vehicles traffic crash data released by the National Highway Traffic Safety Administration (NHTSA), up to 94% of the total fatal accident in 2015 is caused by a human error or a bad decision made by a human operator, and motor vehicle crashes were the leading cause of death for age 10 and every age 16 through 23 [1]. Furthermore, human driving is also susceptible to reaction times and delays which can negatively impact the traffic flow. In a bid to reduce the number of fatal accidents and increase traffic flow, various researches have been conducted on transferring mundane driving tasks to be controlled by computer systems, such as adaptive cruise control and lane keeping features. In this work, we will be studying methods to incorporate collision avoidance using path planning for highway operations. This work can be seen as an extension to adaptive cruise control. Instead of automatically adjusting the vehicle speed to maintain a safe distance from the vehicle ahead, the new algorithm allows the system to reroute its path and conduct a lane change to pass the front vehicle if the front vehicle is traveling below the speed limit. To summarize, this research work hopes to design an algorithm for collision avoidance on highway operation under the presence of non-cooperative vehicles as well as stationary obstacles and lane closure.

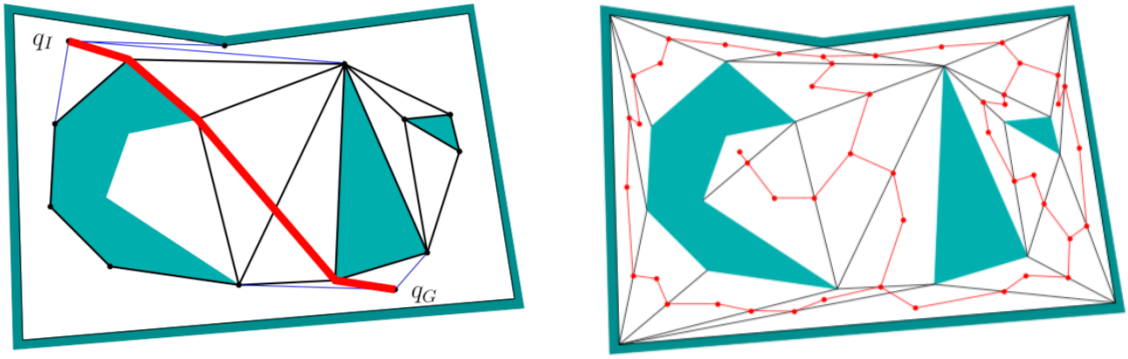
Trajectory generation or path planning plays an important role in autonomous navigation of vehicles in an unknown environment and has been extensively studied in the field of robotics. The basic notion of path planning is to generate a collision free, optimal trajectory that allows the agent (autonomous vehicle) to transverse from a starting pose to a desired final pose while subjected to the agent's dynamic and kinematic constraints. The optimality notion varies based on the application and specific user requirements; an optimal trajectory can be a trajectory that minimizes travel time, minimizes energy consumption and so on.

Configuration spaces are normally used as it allows the transformation of the problem into a simpler problem, where the agent is treated as a point object trying to navigate through a space of enlarged configuration space obstacles. The configuration space consists of all possible configurations (position and orientation) that the agent can take within the region of interest

(map) and this configuration space is made up of “free region” and “obstacle region”. Both the starting pose and desired final pose is a particular point within this configuration space, the path planning problem now becomes the problem of finding an optimum path that connect these two points within the configuration space while only traversing the “free region”. The continuous configuration space is normally first discretized and possible trajectories are then identified. The optimum path can then be found using search algorithm such as Dijkstra’s algorithm, A*, D* and heuristic search algorithm.

Motion planning methods can be roughly classified into three main categories; potential field methods, combinatorial planning and sampling-based planning. Potential field methods model the environment as an artificial potential field, U, and the agent is modeled as a particle acting under the influence of this artificial potential field, U [2]. This artificial potential field, U, superimposes repulsive forces from obstacles and attractive force from the goal. These methods have been shown to be applicable to real time path planning, however these methods are susceptible to local minima.

Combinatorial planning characterizes the “free region” of the configuration space explicitly by capturing its connectivity in a graph and then finding the optimum trajectory within the graph using a search algorithm. A node is normally used to represent a possible pose in the configuration space, while an edge between two nodes is used to represent the presence of a viable transition between the two configurations. Combinatorial planning algorithms are complete, in the sense that it will either find a solution or will correctly report that no solution exists. Some of the frequently used combinatorial planning algorithm includes visibility graph [3], Voronoi diagrams [4], exact cell decomposition and approximate cell decomposition. Visibility graph is one of the earliest configuration space discretization method, where possible paths are constructed as lines connecting vertices of obstacles [3]. Under this formulation, any line segments between vertices of obstacles that does not pass through any obstacle is a feasible path and an edge is drawn between them. One of the advantage of this formulation is that it generates the shortest path when the obstacles are polygonal in shape. Under the Voronoi diagrams formulations, paths are constructed by identifying lines having the same maximum clearance from all nearest obstacles [4]. This generates the safest path with the maximum clearance from the obstacles in the operating environment, however the generated paths tend to be suboptimal in the other aspects (time, energy, etc.) and has an unnatural attraction to open spaces. The exact cell decomposition technique and approximate cell decomposition technique segments the free space into non-overlapping cells and a node is assigned to each cell, normally at its centroid location. A connectivity graph is then used to represent adjacencies between neighboring cells. The main difference between these two methods is that the exact cell decomposition technique allows the shape of the cell to change, whereas the approximate cell decomposition technique uses a simple predefined shape and only allows the scale of the shape to change [5,6]. An example of path planning using visibility graph and exact cell decomposition is as shown in Fig. 1 below. However, these algorithms are computationally heavy as it discretizes the whole configuration space and quickly become intractable for large spaces.



(a) Visibility graph, possible trajectories are indicated by the black edges.

(b) Cell decomposition (triangulation), possible trajectories are indicated by the red edges.

Fig. 1: Combinatorial Path Planning Algorithms [7].

In sampling-based planning, instead of discretizing the whole free configuration space, obstacle collision detection is used to probe and incrementally search the configuration space for an optimum trajectory. The advantages of these method are the lower computational cost and the ease of implementation compared to combinatorial planning algorithms. However, these methods have a weaker guarantee on solution completeness. Some examples of the sampling-based algorithm are the probabilistic road map [8], rapidly exploring random trees (RRT) [9] and modified rapidly exploring random trees (RRT*) [10]. In probabilistic road map, random sample is drawn in the configuration space. The random samples that are located in the "free region" of the configuration space will then be used as nodes. Possible trajectories are then considered by joining these nodes. A search algorithm is then used to find the optimum path.

RRT uses a random tree construction method to efficiently search the configuration space. Random samples are drawn from the configuration space and is used to incrementally construct the tree. This method has been shown to be able to handle obstacles as well as nonlinear and nonholonomic vehicle dynamics with state constraints. At each instance, a random sample is drawn from the "free region" of the configuration space, the nearest vertex of the current tree to the new sample is identified. The vertex and the sample are then connected and the feasibility of this connection is examined. If there is no collision between these two points and the random sample is located further than a distance limit, a new node is added at the maximum distance limit along the line to the random sample. If the sample is located within the distance limit and there is no collision, the new sample is set as the new node. If there is an obstacle between the line connecting the nearest vertex and the random sample, the random sample is discarded. A new random sampled is then drawn from the "free region" of the configuration space and the whole process is repeated. A common method to drive the tree towards the goal is to set the goal pose as the random sample at every fixed interval and if the tree reaches the goal pose, then the problem is considered to be solved. An example of the path generation using RRT with random sampling is shown in Fig. 2. It can be observed that the RRT algorithm quickly reaches the unexplored regions of the configuration space. However, it was shown that the cost of the best path from the traditional RRT converge almost surely to a non-optimal value [10].

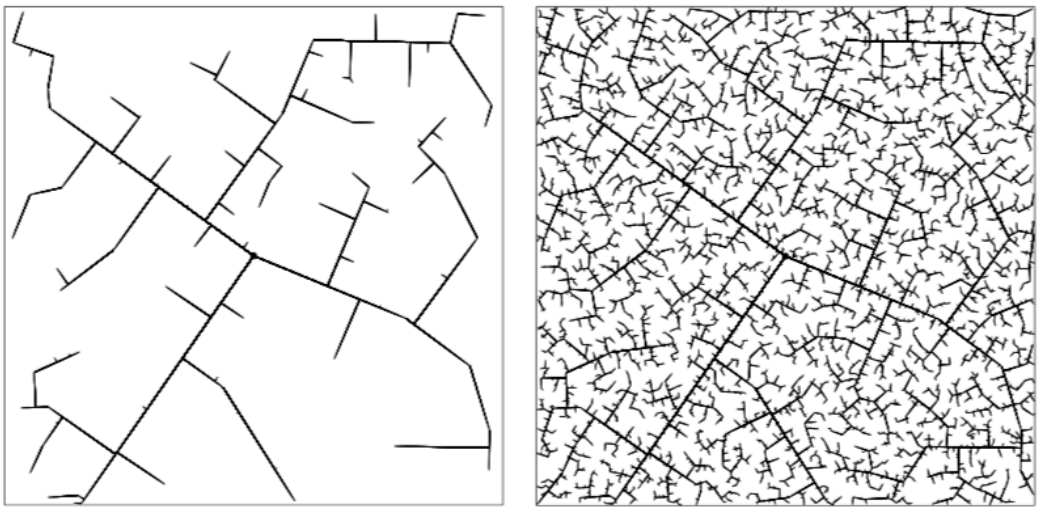
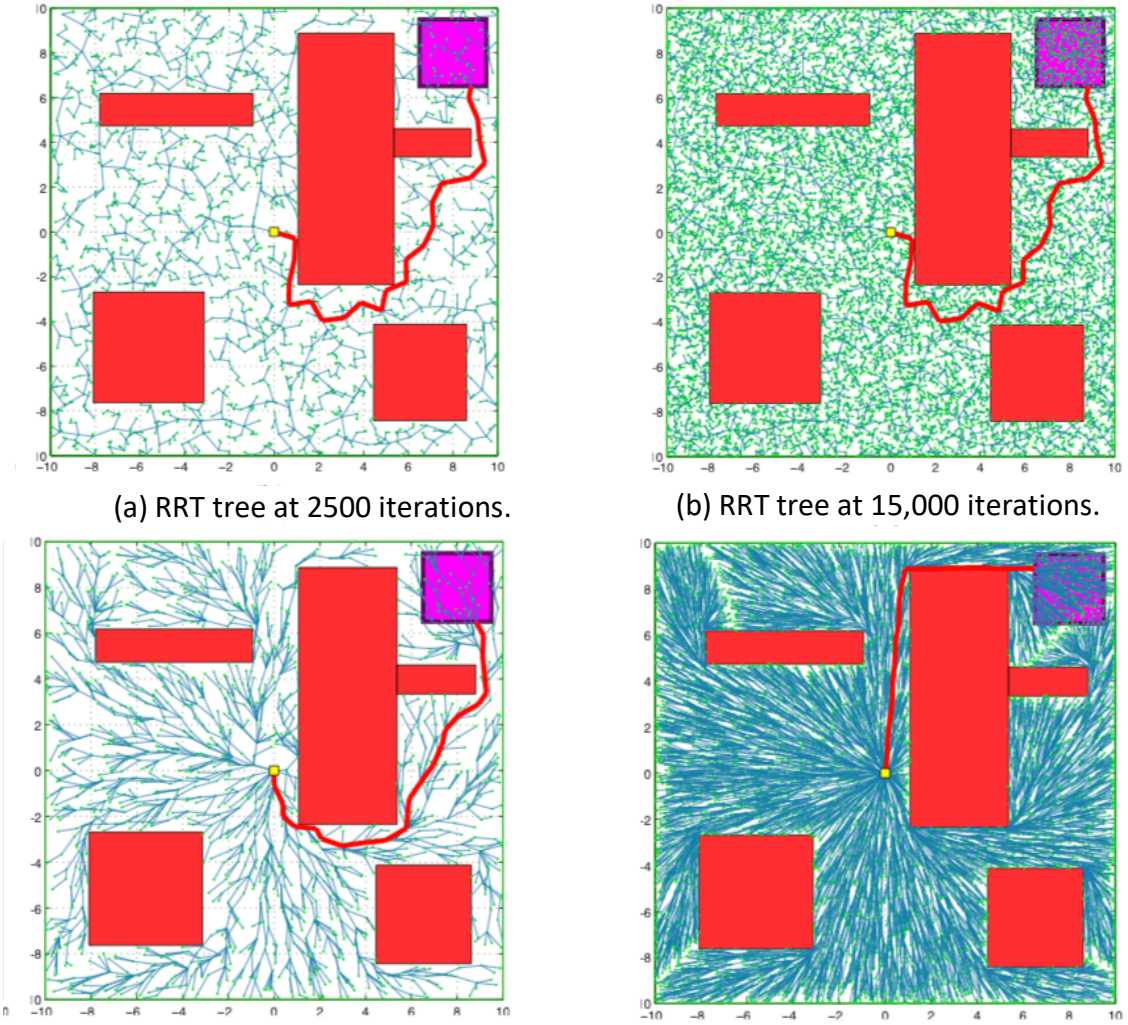


Fig. 2: Path generation using RRT with random sampling [7].

The modified rapidly exploring random trees (RRT*) was able to achieve asymptotic optimality while maintaining the same random tree structure of the traditional RRT. RRT* introduces an additional operation where the previously established tree located in the near vicinity of the newest node can be rewired if the cost function to travel to the nearby node from the newest node is lower than the current cost function. A comparison between the tree generated by the RRT and RRT* algorithms are shown in Fig. 3 below. It can be observed that the RRT tree converges to a sub-optimal solution with unnatural (jaggy) trajectory, whereas RRT* converges to the smoother, optimal solution. However, the traditional RRT* is unsuitable for real time applications for on road driving due to its unnatural trajectory, useless sampling and slow exploration. In [11], instead of randomly sampling from the configuration space, a guided sampling strategy is used. The sampling strategy is formulated based on a typical human operator's visual search behavior and a continuous smooth curvature method. Real time variant of RRT* was introduced in [12] and [13]. In [12], a variant of RRT* called anytime RRT* is introduced. An initial feasible trajectory is quickly identified using RRT*, this initial trajectory is then optimized during execution to improve the solution. A branch-and-bound algorithm is also introduced to more efficiently build the tree, where inefficient nodes are periodically deleted from the tree. The real time RRT* (RT-RRT*) was introduced in [13], where an online tree rewiring strategy is introduced. This strategy allows the tree root to move along with the agent without discarding previously sampled path and thus reducing redundant operations. Most of the early work on RRT deals with static cases involving stationary obstacles. In 2017, a new variant of the RRT* called RRT*Finite Node Dynamic (RRT*FND) was proposed in [14] to allow RRT* to be extended for dynamic environment. Under this new formulation, the solution path is reevaluated for obstacle after each execution of a small section of the path. If the original optimal solution path has been obstructed by a dynamic obstacle, the obstructed parts of the tree are removed while the useful parts of the tree are retained. Two greedy heuristics algorithms are then used to repair the solution in an attempt to reconnect the separated segments of the tree, instead of running the whole motion planning process from scratch. However, the kinematic state of the

dynamic obstacles is not leverage in the tree generation algorithm but the dynamic obstacles are treated as stationary obstacles that has shifted to a new position when generating the tree at the different time step.



(a) RRT tree at 2500 iterations.

(b) RRT tree at 15,000 iterations.

(c) RRT* tree at 2500 iterations.

(d) RRT* tree at 15,000 iterations.

Fig. 3: A comparison of the RRT* and RRT algorithm. The goal regions are shown in magenta and the best path is highlighted in red [10].

In [15], a cost function that takes into consideration the trajectory efficiency, comfort, similarity to typical human driving behavior and safety was proposed. The proposed cost function is shown in Table 1 below. The cost function is used to evaluate candidate trajectory generated using parametric optimal control. The optimum trajectory is then selected and optimized iteratively. However, the cost function is sufficiently general and can be used to evaluate trajectory generated using any path generation algorithm, including the RRT* algorithm. For vehicle operating in a multi lane highway, an additional cost function can be introduced to weight the cost toward a particular lane of the highway (for example the left most lane). This cost function will then result in the optimum trajectory being shifted toward that particular lane.

Table 1: Cost Function Proposed in [15]

Cost	*Formula*	*Physical Interpretation*	*Impact*
c_l	l	l is path length	efficiency
c_k	$\sum_{i=0}^{n} \lvert k_i \rvert$	k_i is curvature	comfort
c_{dk}	$\sum_{i=0}^{n} \lvert \dot{k}_i \rvert$	$\dot{k}_i$ is rate of change of curvature	comfort
c_o	$\sum_{i=0}^{n} \lvert o_i \rvert$	o_i is lateral offset with the closest center line	behavior
c_{obs}^{s}	$\sum_{i=0}^{n} s_i$	s_i is the transformed distance to static obstacles	safety
c_t	t	t is time duration of a trajectory	efficiency
c_e	$\sum_{i=0}^{n} v_i^2$	v_i is speed	energy
c_a	$\sum_{i=0}^{n} a_i^2$	a_i is acceleration	comfort
c_j	$\sum_{i=0}^{n} j_i^2$	j_i is rate of change of acceleration	comfort
c_{ca}	$\sum_{i=0}^{n} v_i^2 k_i$	c_{ca} is centripetal acceleration	comfort
c_{obs}^{d}	$\sum_{i=0}^{n} d_i$	d_i is the transformed distance to dynamic obstacles	safety

The report is organized as follows: The modelling of the 10 degree of freedom vehicle dynamics and modelling of the environment are presented in section I and section II respectively. Section III covers the system specifications used for the modelling and simulation setup, whereas the control system design (LQR-RRT* algorithm) is presented in Section IV. The simulation results and discussion are presented in Section V, while the conclusion and future works are discussed in Section VI.

Section I: Vehicle Dynamic Modeling

The standard vehicle model with seven degrees of freedom as discussed in [16] is modified to the global navigation frame and is used for this project. The seven degrees of freedom consists of the lateral velocity in the global frame $\dot{X}$, longitudinal velocity in the global frame $\dot{Y}$, yaw rate in the body frame $\dot{\psi}$, and the wheel velocities of the four wheels (ω_{fl} , ω_{fr} , ω_{rl} and ω_{rr}) as shown in Fig. 4 below.

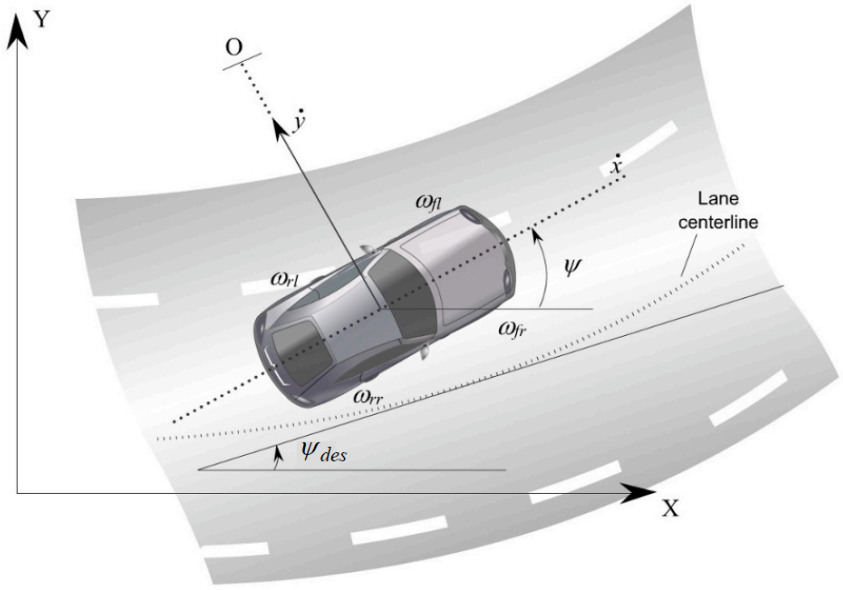


Fig. 4: Seven DOF Vehicle Model [16].

The equation of motion of the vehicle in the longitudinal, lateral and yaw axis is then given by the following equations:

$$m\ddot{X} = \left(F_{xfl} + F_{xfr}\right)\cos(\psi + \delta) - \left(F_{yfl} + F_{yfr}\right)\sin(\psi + \delta) + \left(F_{xrl} + F_{xrr}\right)\cos\psi - \left(F_{yrl} + F_{yrr}\right)\sin\psi \quad (1)$$

$$m\ddot{Y} = \left(F_{xfl} + F_{xfr}\right)\sin(\psi + \delta) + \left(F_{yfl} + F_{yfr}\right)\cos(\psi + \delta) + \left(F_{yrl} + F_{yrr}\right)\cos\psi + \left(F_{xrl} + F_{xrr}\right)\sin\psi \quad (2)$$

$$I_z\ddot{\psi} = l_f\left(F_{xfl} + F_{xfr}\right)\sin\delta + l_f\left(F_{yfl} + F_{yfr}\right)\cos\delta - l_r\left(F_{yrl} + F_{yrr}\right) + \frac{l_w}{2}\left(F_{xfr} - F_{xfl}\right)\cos\delta + \frac{l_w}{2}\left(F_{xrr} - F_{xrl}\right) + \frac{l_w}{2}\left(F_{yfl} - F_{yfr}\right)\sin\delta \quad (3)$$

δ is the front wheel steering angle, F_{xfl} , F_{xfr} , F_{xrl} , F_{xrr} are the front left, front right, rear left and rear right longitudinal tire forces respectively, F_{yfl} , F_{yfr} , F_{yrl} , F_{yrr} are the front left, front right, rear left and rear right lateral tire forces respectively, and l_f , l_r , l_w are the longitudinal distance from the center of gravity of the vehicle (c.g.) to the front wheels, longitudinal distance from the c.g. to the rear wheels and the lateral distance between the left and right wheels (track width) respectively.

The slip angles at the front and rear tires are given by:

$$\alpha_f = \delta - \frac{\dot{y} + l_f\dot{\psi}}{\dot{x}} \quad (4)$$

$$\alpha_r = -\frac{\dot{y} + l_r\dot{\psi}}{\dot{x}} \quad (5)$$

While the longitudinal slip ratio is given by:

$$\sigma_z = \frac{r_{eff}\omega_w - \dot{x}}{\dot{x}}, \quad \text{during braking} \quad (6)$$

$$\sigma_z = \frac{r_{eff}\omega_w - \dot{x}}{r_{eff}\omega_w}, \quad \text{during acceleration} \quad (7)$$

where ‘z’ takes ‘fl’, ‘fr’, ‘rl’, ‘rr’ to represents the front left, front right, rear left and rear right slip ratios respectively, whereas $\dot{x}$ and $\dot{y}$ are the longitudinal and lateral velocities of the wheels and is given by:

$$V = \sqrt{\dot{X}^2 + \dot{Y}^2}\,, \beta = \arctan\frac{\dot{Y}}{\dot{X}} - \psi\,, \gamma_f = \arctan\frac{l_w}{2l_f}\,, \gamma_r = \arctan\frac{l_w}{2l_r} \quad (8)$$

$$\dot{x}_{fl} = V\cos(\beta - \delta) + \dot{\psi}\sqrt{{l_f}^2 + \left(\frac{l_w}{2}\right)^2}\,\sin\left(\delta - \gamma_f\right) \quad (9)$$

$$\dot{y}_{fl} = V\sin(\beta-\delta) + \dot{\psi}\sqrt{{l_f}^2 + \left(\frac{l_w}{2}\right)^2}\cos(\delta-\gamma_f) \quad (10)$$

$$\dot{x}_{fr} = V\cos(\beta-\delta) + \dot{\psi}\sqrt{{l_f}^2 + \left(\frac{l_w}{2}\right)^2}\cos\left(\delta-\left(90°-\gamma_f\right)\right) \quad (11)$$

$$\dot{y}_{fr} = V\sin(\beta-\delta) - \dot{\psi}\sqrt{{l_f}^2 + \left(\frac{l_w}{2}\right)^2}\sin\left(\delta-\left(90°-\gamma_f\right)\right) \quad (12)$$

$$\dot{x}_{rl} = V\cos\beta - \dot{\psi}\sqrt{{l_r}^2 + \left(\frac{l_w}{2}\right)^2}\sin\gamma_r \quad (13)$$

$$\dot{y}_{rl} = V\sin\beta - \dot{\psi}\sqrt{{l_r}^2 + \left(\frac{l_w}{2}\right)^2}\cos\gamma_r \quad (14)$$

$$\dot{x}_{rr} = V\cos\beta + \dot{\psi}\sqrt{{l_r}^2 + \left(\frac{l_w}{2}\right)^2}\sin\gamma_r \quad (15)$$

$$\dot{y}_{rr} = V\sin\beta - \dot{\psi}\sqrt{{l_r}^2 + \left(\frac{l_w}{2}\right)^2}\cos\gamma_r \quad (16)$$

Under the Dugoff tire model, the longitudinal tire force and lateral tire force are given by:

$$F_x = C_\sigma \frac{\sigma}{1+\sigma} f(\lambda) \quad (17)$$

$$F_y = C_\alpha \frac{\tan\alpha}{1+\sigma} f(\lambda) \quad (18)$$

where C_α is the cornering stiffness, C_σ is the longitudinal tire stiffness and λ is given by

$$\lambda = \frac{\mu F_z(1+\sigma)}{2\{(C_\sigma\sigma)^2 + (C_\sigma\tan\sigma)^2\}^{1/2}} \quad (19)$$

and

$$f(\lambda) = (2-\lambda)\lambda, \quad \text{If } \lambda < 1 \quad (20)$$

$$f(\lambda) = 1, \quad \text{If } \lambda \geq 1 \quad (21)$$

F_z is the vertical force on the tire and μ is the tire-road friction coefficient. The rotational dynamics of the 4 wheels are then given by the following equations

$$J_w\dot{\omega}_{fl} = T_{dfl} - T_{bfl} - r_{eff}F_{xfl} \quad (22)$$

$$J_w\dot{\omega}_{fr} = T_{dfr} - T_{bfr} - r_{eff}F_{xfr} \quad (23)$$

$$J_w\dot{\omega}_{rl} = T_{drl} - T_{brl} - r_{eff}F_{xrl} \quad (24)$$

$$J_w\dot{\omega}_{rr} = T_{drr} - T_{brr} - r_{eff}F_{xrr} \quad (25)$$

$T_{dfl}, T_{dfr}, T_{drl}, T_{drr}$ are the drive torque transmitted to the front left, front right, rear left and rear right wheels respectively and $T_{bfl}, T_{bfr}, T_{brl}, T_{brr}$ are the brake torque on the front left, front right, rear left and rear right wheels respectively. In our model, we are assuming that we have direct control over these drive and brake torques and T_i is used to represent the engine/brake torque for each wheel. Hence, the rotational dynamics of the 4 wheels takes the following form:

$$J_w\dot{\omega}_{fl} = T_{fl} - r_{eff}F_{xfl} \quad (26)$$

$$J_w\dot{\omega}_{fr} = T_{fr} - r_{eff}F_{xfr} \quad (27)$$

$$J_w\dot{\omega}_{rl} = T_{rl} - r_{eff}F_{xrl} \quad (28)$$

$$J_w\dot{\omega}_{rr} = T_{rr} - r_{eff}F_{xrr} \quad (29)$$

Section II: Environment Modeling

A 2D model for road sections typically observed on a highway consisting of straight and horizontal curve segments is used. The curve segments are designed such that there is a gradual and smooth transition between the two tangent strips of the highway. The design of a simple highway curve segment is governed by 4 variables; radius, design speed, side friction factor and superelevation [17], where the minimum allowable radius for a simple horizontal curve can be determined using the following equation,

$$R_{min} = \frac{v_{design}^2}{127(e_{max} + f_{max})} \quad (30)$$

where R_{min} is the minimum curve radius (m),v_{design} is the design speed (km/s), e_{max} is the maximum superelevation and f_{max} is the maximum allowable side friction factor.

In our highway model, superelevation or tilting of the horizontal curve is ignored. A design speed of 65 mph (0.029 km/s) is chosen. The typical side friction factor can range from less than 0.1 to 0.5 depending on the tire conditions, vehicle operating parameters and speed, environment condition as well as on the road surface condition. A nominal maximum allowable side friction factor of 0.15 is chosen. Stationary and dynamic vehicles with varying speed are inserted into the environment to mimic a typical highway condition.

Section III: System Specification

A typical sedan's vehicle parameters are used for the vehicle model as shown in table 2.

Table 2: Vehicle Parameters

Vehicle Parameters		Value
m	Total mass of vehicle	2205
l_{car}	Length of vehicle body	4.8
w_{car}	Width of vehicle body	1.8
I_z	Yaw moment of inertia of vehicle	2873
l_w	Distance between left and right wheels (track length)	1.75
l_f	Longitudinal distance from c.g. to front tires	1.318
l_r	Longitudinal distance from c.g. to rear tires	1.542
r_{eff}	Effective tire radius	0.3543
C_α	Cornering stiffness of tire	80000
C_σ	Longitudinal stiffness of tire	26000
μ	Tire-road friction coefficient	0.8
J_w	Rotational moment of inertia of each wheel	0.8

Meanwhile, the parameters used to generate the highway is as shown in table 3 below.

Table 3: Highway Parameters

Highway Parameters		Value
v_{limit}	Speed limit	25 m/s
v_{design}	Design speed	29 m/s
e_{max}	Maximum superelevation	0
f_{max}	Maximum allowable side friction factor	0.15
w	Highway width	3.7

Section IV: Control System Design

This section is divided into five subsections, which covers the problem statement for the controller design, a high-level overview of the traditional linearized LQR formulation, the LQR-RRT* algorithm with pseudocodes, the cost functions used and the assumptions made to simplify the linearization of the system.

Problem Description

Given the start pose, desired goal region, geometric description of the vehicle and the world (map), and the trajectory of obstacles in the operating environment, find a path that moves the vehicle from the start to goal without touching any obstacle while minimizing a desired cost function that tradeoff between safety, ride comfort and travel time.

Linear Quadratic Regulator (LQR) Algorithm

The LQR is a feedback controller that generates an optimal control sequence and an optimal trajectory that is optimal in the sense of minimizing a quadratic cost function. However, the traditional LQR is only applicable to a linear system. In our application, the vehicle dynamics is nonlinear and coupled. Hence, a first order Taylor approximation is first used to linearized the system about the operating conditions such that the traditional LQR is still applicable. Consider a continuous time non-linear system given by

$$\dot{x} = f(x, u) \tag{31}$$

Taking the first order Taylor approximation,

$$\dot{x} \approx f(x_0, u_0) + \frac{\partial f(x_0, u_0)}{\partial x}(x - x_0) + \frac{\partial f(x_0, u_0)}{\partial u}(u - u_0) \tag{32}$$

Letting $\bar{x} = x - x_0$ and $\bar{u} = u - u_0$, we will arrive at the following linearized system equation,

$$\dot{\bar{x}} \approx \frac{\partial f(x_0, u_0)}{\partial x}\bar{x} + \frac{\partial f(x_0, u_0)}{\partial u}\bar{u} \tag{33}$$

For a quadratic cost function, the optimum gain and cost to go can then be obtained by solving the following algebraic Riccati equation,

$$A^T S + SA - SBR^{-1}B^T S + Q = 0 \tag{34}$$

where

$$A = \frac{\partial f(x_0, u_0)}{\partial x}, \qquad B = \frac{\partial f(x_0, u_0)}{\partial u}$$

and Q and R are obtained from the quadratic cost function. The optimal state feedback gain is then given by

$$K_{opt} = R^{-1}B^T S \tag{35}$$

and the cost to go for this trajectory is given by

$$V(\bar{x}) = \bar{x}^T S \bar{x} \tag{36}$$

LQR-RRT* Algorithm

In [18], a variant of the RRT* algorithm called the LQR-RRT* algorithm was proposed. The LQR-RRT* algorithm allows the generation of smoother trajectory with lower cost. This project uses a modified version of the LQR-RRT* algorithm and the pseudo-code for the modified LQR-RRT* is presented in algorithm 1-3 below. V contains the list of all vertices of the tree, while E contains the list of all parent-to-child connection for the vertices and $CostList$ contains the list of the cost to get to and arrive at a particular vertex of the tree.

Algorithm 1 The LQR - RRT* Algorithm

while $i < N$ **do**
  $x_{rand} \leftarrow Sample$
  $x_{nearest} \leftarrow LQRNearest(V, x_{rand})$
  $x_{new} \leftarrow LQRSteer(x_{nearest}, x_{rand})$
  $X_{near} \leftarrow LQRNear(V, x_{new})$
  $(x_{min}, \sigma_{min}, minCost) \leftarrow ChooseParent(X_{near}, x_{new}, CostList)$
  if $ObstacleFree(\sigma)$ **then**
    $V \leftarrow V \cup x_{new}$
    $E \leftarrow E \cup \{x_{min}, x_{new}\}$
    $CostList \leftarrow CostList \cup minCost$
    $(V, E, CostList) \leftarrow Rewire((V, E, CostList), X_{near}, x_{new})$

Algorithm 2 ChooseParent$(X_{near}, x_{new}, CostList)$

$minCost \leftarrow \inf; x_{min} \leftarrow \emptyset; \sigma_{min} \leftarrow \emptyset$
for $x \in X_{near}$ **do**
  $\sigma \leftarrow LQRSteer(x, x_{new})$
  if $ObstacleFree(\sigma)$ **then**
    if $CostList(x) + Cost(\sigma) < minCost$ **then**
      $minCost \leftarrow CostList(x) + Cost(\sigma)$
      $x_{min} \leftarrow x; \sigma_{min} \leftarrow \sigma$
return $(x_{min}, \sigma_{min}, minCost)$

Algorithm 3 Rewire$((V, E, CostList), X_{near}, x_{new}))$

for $x \in X_{near}$ **do**
  $\sigma \leftarrow LQRSteer(x_{new}, x)$
  if $ObstacleFree(\sigma)$ **then**
    if $CostList(x_{new}) + Cost(\sigma) < CostList(x)$ **then**
      $CostList(x) \leftarrow CostList(x_{new}) + Cost(\sigma)$
      $x_{parent} \leftarrow Parent(x)$
      $E \leftarrow E \backslash \{x_{parent}, x\}$
      $E \leftarrow E \cup \{x_{new}, x\}$
return (V,E,CostList)

The rest of the algorithm primitives are as described below:

- ***Sample***: The full state space of the vehicle model consists of 10 variables $X, \dot{X}, Y, \dot{Y}, \psi, \dot{\psi}$, $\omega_{fl}, \omega_{fr}, \omega_{rl}$, and ω_{rr}. Instead of sampling over the full state space, random sampling is done over the lower dimension task space, which is made up of $X, Y, \psi, and\ V$. The position parameters (X and Y) are randomly sampled from the free space, while the yaw angle and velocity are sampled within ranges $\psi \in [\psi_{min}, \psi_{max}]$ and $V \in [V_{min}, V_{max}]$.

ψ_{min} and ψ_{max} are set based on the tangential of the road such that $\psi_{min} = \psi_{road} - \Delta$ and $\psi_{max} = \psi_{road} + \Delta$ to ensure the generation of smoother and more feasible trajectories. Based on data on lane change maneuver from [19], Δ is set to be $0.1\ rad$.

- ***LQRNearest**:* From the set of vertices in the current tree, V, and given a state, x, this function returns the nearest vertex to the state x in terms of the LQR cost to go function. The LQR cost to go function is obtained by linearizing the system about the current consider vertex, v with zero action ($u = 0$) and evaluating the LQR gain matrix, $K(v)$ and cost matrix, $S(v)$. The nearest vertex is then found by evaluating the cost matrix and finding the vertex with the minimum cost.
- ***LQRSteer***: This function calculates the optimum path connecting the two given states using the local LQR policy calculated at the parent vertex.
- ***LQRNear***: This function returns a set of vertices that are located near in the sense of "LQR cost to go" to a state, x, and is found using the equation below,

$$(v - x)^T S(v - x) \leq \gamma \left(\frac{\log n}{n}\right)^{\frac{1}{d}} \quad (37)$$

Where n is the number of vertices in the tree, d is the dimension of the space and γ is a user specified constant.

- ***Cost***: This function returns the LQR cost to go for travelling from the parent vertex to the current vertex as well as the cost incurred at the current vertex.
- ***ObstacleFree***: This function checks for collision in the generated trajectory and any trajectories that result in a collision is eliminated. Furthermore, for safety (roll-over prevention) and driver comfort, an upper bound on the yaw rate, $\dot{\psi}_{max}$ and vehicle acceleration, $\dot{V}_{max}$ is used to eliminate generated trajectories that does not confine to these upper bounds. The upper bound on the yaw rate is given by the following equation adapted from [16]:

$$\dot{\psi}_{max} = 0.85\frac{\mu g}{V} \quad (38)$$

Cost Function

Instead of only using a quadratic cost to represent the cost of a trajectory, the cost of a trajectory is calculated as a summation of cost to go (travel between vertices) and a vertex cost (cost incur for stopping at a vertex). The vertex cost is introduced to penalize the use of vertices that are too close to the obstacles or are located too far laterally from the closest lane's center line. The cost to go function takes the following form,

$$J = \int_0^\infty x^T Q x + u^T R u \quad (39)$$

where Q and R are selected such that the target is penalized for being away from the target state ($X, Y, \psi, V(\dot{X}, \dot{Y})$), passenger comfort ($\dot{\psi}, \dot{X}\dot{\psi}$) and input energy ($\delta, T_{fl}, T_{fr}, T_{rl}, T_{rr}$). Whereas the vertex cost is shown in table 4 below.

Table 4: Vertex Cost

Cost	*Formula*	*Physical Interpretation*	*Impact*
c_o	$\sum_{i=0}^{n} \lvert o_i \rvert$	o_i is lateral offset with the closest center line	behavior
c_{obs}^{s}	$\sum_{i=0}^{n} s_i$	s_i is the transformed distance to static obstacles	safety
c_{obs}^{d}	$\sum_{i=0}^{n} d_i$	d_i is the transformed distance to dynamic obstacles	safety

System Linearization and Assumptions

The full state space of the vehicle model that consists of 10 variables $X, \dot{X}, Y, \dot{Y}, \psi, \dot{\psi}$, $\omega_{fl}, \omega_{fr}, \omega_{rl}$, and ω_{rr} is first linearized in order to arrive at a continuous time linear system. Before linearizing the state space model, some assumptions are made to further simplify the non-linearity within the model. Fig. 5 below shows a plot of the lambda value against longitudinal slip ratio using our vehicle parameters. It can be observed that for low longitudinal slip ratio of less than 0.07, the lambda value takes the value of 1. Hence, by setting the maximum velocity difference between the wheel and the vehicle speed to be less than 0.07 of the nominal speed, we can assume that the lambda value takes the value of 1.

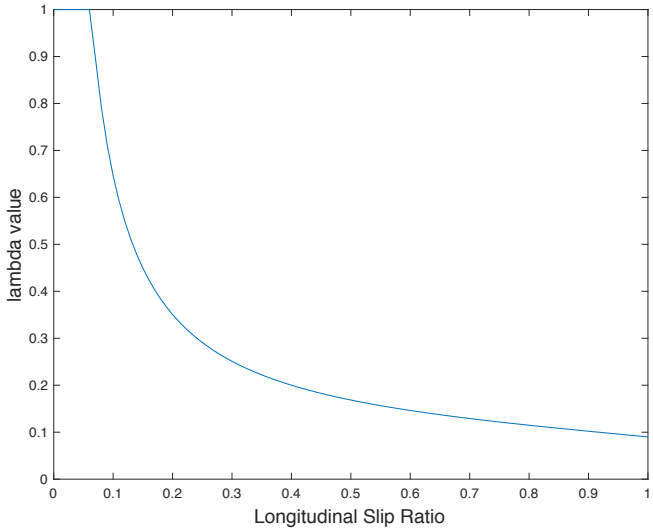


Fig. 5: Lambda Value vs Longitudinal Slip Ratio

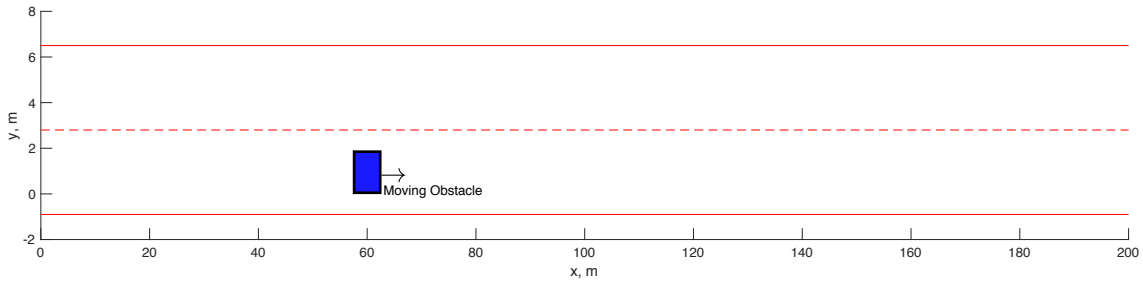
Matlab® Symbolic toolbox is then used to linearize the system and to generate functions that auto output the linearize state space system at any given state.

Section V: Simulation Results and Discussions

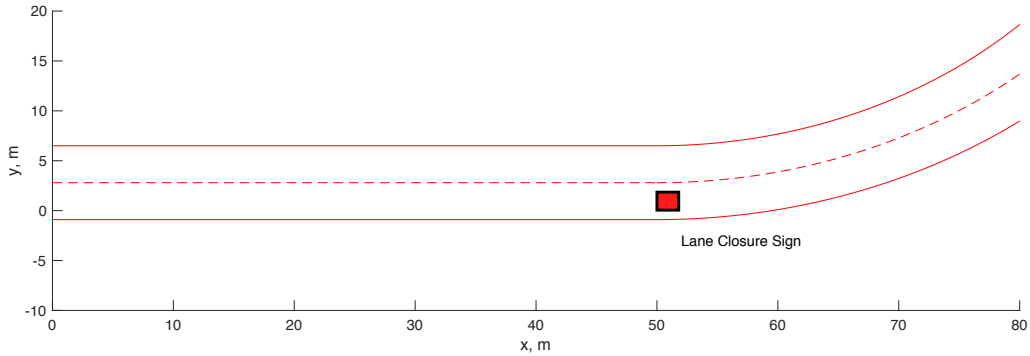
Due to the nonlinearity of the system and the limited control input, the randomly sampled state can result in an un-stabilizable linearized system, hence a stable-unstable decomposition is first conducted on the linearized system. The controllability of the unstable part of the linearized system is then examined. The randomly sampled states with uncontrollable unstable parts are discarded and a new sample is drawn.

The following two test scenarios are simulated:

1. A straight segment of the freeway with a moving vehicle (obstacle) occupying the right most lane and assumed to be driving in a conservative manner in low speed as shown in Fig. 6(a) below.
2. A curved segment of the freeway with the right most lane being closed off for construction when entering the curved segment and is shown in Fig. 6(b) below.



(a) Straight segment of the freeway used for scenario 1



(b) 90° curved segment of the freeway used for scenario 2

Fig. 6: Environment Models with Obstacles

For the first test scenarios (straight segment), three different vertex cost functions are examined. The different vertex cost functions used are as follows:

1. Only penalizing for deviation from typical human driving behavior (lateral offset from the closest center line).
2. Penalizing for both deviation from typical human driving behavior (lateral offset from the closest center line) and distance from goal region.
3. Penalizing for deviation from typical human driving behavior (lateral offset from the closest center line), distance from goal region and distance to obstacle.

Whereas for the second test scenario (curved segment), a vertex cost function that penalizes for both deviation from typical human driving behavior (lateral offset from the closest center line) and distance from goal region is used.

Scenario 1a: Straight Segment with Vertex Cost Function (1)

The random tree generated after 29000 iterations is as shown in Fig. 7 below, where the red line is the current optimum path with the lowest cost. An initial feasible route was found at iteration

7000, which was then iteratively improved with subsequent iterations. The vehicle was able to safely avoid the obstacle by performing a lane change and continuing its path along the highway.

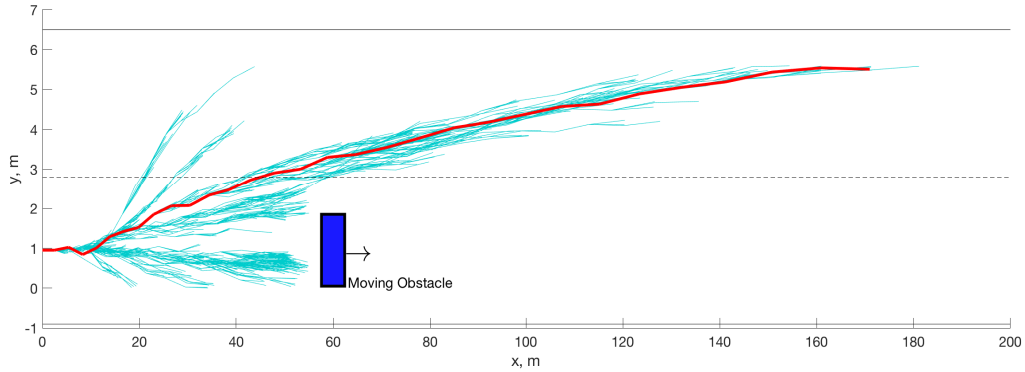


Fig. 7: Simulation Results for Scenario 1a

Scenario 1b: Straight Segment with Vertex Cost Function (2)

The random tree generated with vertex cost function (2) after 27000 iterations is as shown in Fig. 8 below, where the red line is the current optimum path with the lowest cost. The vehicle was able to safely avoid the obstacle by performing a lane change and continuing its path along the highway. The trajectories are penalized for the distance to goal and this causes the optimum path to be more tangential to the road profile compared to scenario 1a.

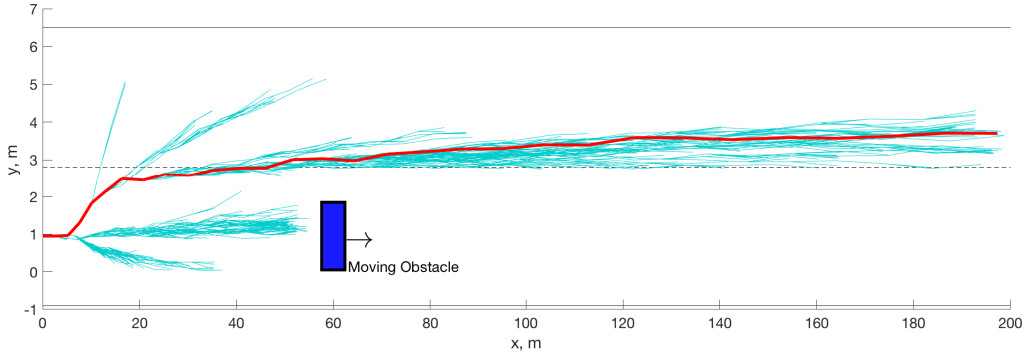


Fig. 8: Simulation Results for Scenario 1b

Scenario 1c: Straight Segment with Vertex Cost Function (3)

The random tree generated with vertex cost function (3) after 22000 iterations is as shown in Fig. 9 below, where the red line is the current optimum path with the lowest cost. The vehicle was able to safely avoid the obstacle by performing a lane change and continuing its path along the highway. The optimum trajectory is slightly shifted upward compared to scenario 1b to compensate for the penalty due to distance to object.

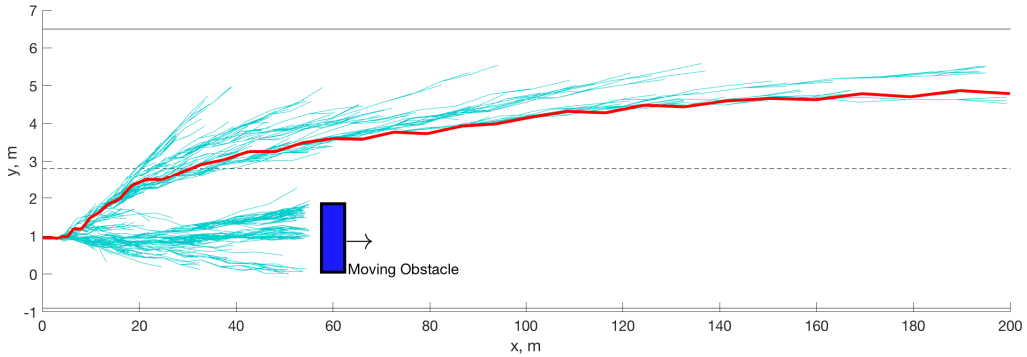


Fig. 9: Simulation Results for Scenario 1c

Fig. 10 below shows the optimum path generated using each of the 3 vertex cost functions. Using a different vertex cost function can results in a slightly different optimum trajectory and the optimum trajectory is a function of the tradeoff between safety, travel time and conformity to typical human driving behaviors.

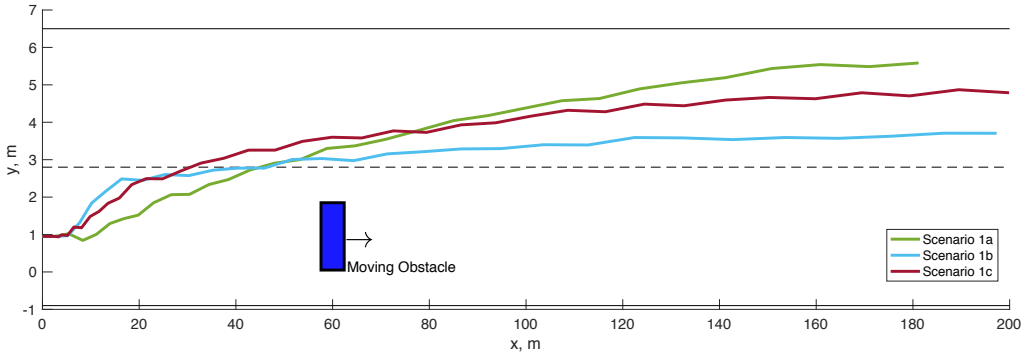


Fig. 10: Simulation Results for Scenario 1

Scenario 2: Curved Segment

The random tree generated for the curved segment after 22000 iterations is as shown in Fig. 11 below. It can be observed that more iterations are needed to generate a feasible path that allows it to completely clear the curve. A better sampling strategy is required in order to obtain a more efficient tree generation algorithm as it can be noticed that most of the proposed vertices result in an infeasible path.

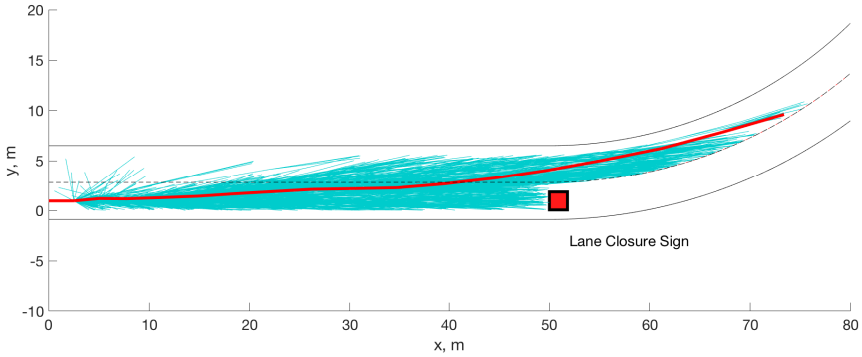


Fig. 11: Simulation Results for Scenario 2

Section VI: Conclusions and Future Works

The modified LQR-RRT* algorithm was successfully implemented on a 10 DOF vehicle model and allows the vehicle to successfully overtake a moving obstacle by conducting a lane change maneuver. Depending on the cost function used based on the tradeoff between safety, travel time and conformity to typical human driving behavior, the resulting optimum trajectories generated by the modified LQR-RRT* algorithm is different from each other.

However, the current algorithm did not perform as well on a curved segment on the highway. This can be attributed to the inefficient sampling strategy used to propose the tree vertex. More efficient sampling strategy need to be examined.

The computational time and cost increases with the number of tree vertex, a branch and bound method can be introduced in future version to prune off infeasible path and path with high cost. Implementation of the branch and bound methods can improve the performance and convergence rate of the current algorithm. Furthermore, in this project, it is assumed that the other vehicle in the highway segment operates in a conservative manner with a constant speed and heading, the feasibility of this approach on aggressive drivers with unpredictable driving behavior need to be examined in future works. Besides that, in future works, we hope to introduce roll over prevention into the LQR-RRT* algorithm.

References

1. 2016 Quick Facts on Crash Statistics, National Center for Statistics and Analysis (NCSA), National Highway Traffic Safety Administration (NHTSA), 2017
2. Khatib, Oussama. "Real-time obstacle avoidance for manipulators and mobile robots." *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*. Vol. 2. IEEE, 1985.
3. Lozano-Pérez, Tomás, and Michael A. Wesley. "An algorithm for planning collision-free paths among polyhedral obstacles." *Communications of the ACM* 22.10 (1979): 560-570.
4. Takahashi, Osamu, and Robert J. Schilling. "Motion planning in a plane using generalized Voronoi diagrams." *IEEE Transactions on robotics and automation* 5.2 (1989): 143-150.
5. Preparata, Franco P., and Michael I. Shamos. *Computational geometry: an introduction*. Springer Science & Business Media, 2012.
6. Moravec, Hans, and Alberto Elfes. "High resolution maps from wide angle sonar." *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*. Vol. 2. IEEE, 1985.
7. LaValle, Steven M. Planning algorithms. Cambridge university press, 2006.
8. Kavraki, Lydia E., et al. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces." IEEE transactions on Robotics and Automation 12.4 (1996): 566-580.
9. LaValle, Steven M. "Rapidly-exploring random trees: A new tool for path planning." (1998).
10. Karaman, Sertac, and Emilio Frazzoli. "Incremental sampling-based algorithms for optimal motion planning." *Robotics Science and Systems VI* 104 (2010): 2.
11. Du, Mingbo, et al. "Drivers' visual behavior-guided RRT motion planner for autonomous on-road driving." *Sensors* 16.1 (2016): 102.

12. Karaman, Sertac, et al. "Anytime motion planning using the RRT." *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011.
13. Naderi, Kourosh, Joose Rajamäki, and Perttu Hämäläinen. "RT-RRT*: a real-time path planning algorithm based on RRT." *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*. ACM, 2015.
14. Adiyatov, Olzhas, and Huseyin Atakan Varol. "A novel RRT*-based algorithm for motion planning in Dynamic environments." *Mechatronics and Automation (ICMA), 2017 IEEE International Conference on*. IEEE, 2017.
15. Xu, Wenda, et al. "A real-time motion planner with trajectory optimization for autonomous vehicles." *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012.
16. Rajamani, Rajesh. Vehicle dynamics and control. Springer Science & Business Media, 2011.
17. Hancock, Michael W., and Bud Wright. "A Policy on Geometric Design of Highways and Streets." (2013).
18. Perez, Alejandro, et al. "LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics." *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012.
19. Ghaffari, A., et al. "Lane change trajectory model considering the driver effects based on MANFIS." (2012): 261-275.